

*IBM SPSS Modeler 18.1 - Guía de
automatización y scripts Python*

IBM

Nota

Antes de utilizar esta información y el producto al que da soporte, lea la información del apartado "Avisos" en la página 367.

Información de producto

Esta edición se aplica a la versión 18, release 0, modificación 0 de IBM SPSS Modeler y a todos los releases y las modificaciones subsiguientes hasta que se indique lo contrario en nuevas ediciones.

Contenido

Capítulo 1. Scripts y lenguaje de scripts 1

| | |
|---|----|
| Visión general de scripts | 1 |
| Tipos de scripts | 1 |
| Scripts de ruta | 1 |
| Ejemplo de script de ruta: entrenamiento de una red neuronal | 3 |
| Límites de tamaño de código Jython | 4 |
| Scripts autónomos | 4 |
| Ejemplo de script autónomo: guardar y cargar un modelo | 4 |
| Ejemplo de script autónomo: Generación de un modelo de selección de características | 5 |
| Scripts de Supernodo | 5 |
| Script de supernodo de ejemplo | 6 |
| Creación de bucles y ejecución condicional en rutas | 6 |
| Bucles en rutas | 7 |
| Ejecución condicional en rutas | 11 |
| Ejecutar e interrumpir scripts | 12 |
| Buscar y reemplazar | 12 |

Capítulo 2. Language de scripts 15

| | |
|--|----|
| Visión general de lenguaje de script | 15 |
| Python y Jython | 15 |
| Scripts de Python | 16 |
| Operaciones | 16 |
| Listas | 16 |
| Cadenas | 17 |
| Observaciones | 18 |
| Sintaxis de las sentencias | 19 |
| Identificadores | 19 |
| Bloques de código | 19 |
| Pasar argumentos a un script | 20 |
| Ejemplos | 20 |
| Métodos matemáticos | 21 |
| Utilización de caracteres no ASCII | 22 |
| Programación orientada a objetos | 23 |
| Definición de una clase | 24 |
| Creación de una instancia de clase | 24 |
| Añadir atributos a una instancia de clase | 24 |
| Definición de atributos de clase y métodos | 24 |
| Variables ocultas | 25 |
| Heredado | 25 |

Capítulo 3. Scripts de IBM SPSS Modeler 27

| | |
|---|----|
| Tipos de scripts | 27 |
| Rutas, rutas de supernodo y diagramas | 27 |
| Rutas | 27 |
| Rutas de Supernodo | 27 |
| Diagramas | 27 |
| Ejecución de una ruta | 27 |
| El contexto de los scripts | 28 |
| Referencia a nodos existentes | 29 |
| Buscar nodos | 29 |
| Establecimiento de propiedades | 30 |

| | |
|---|----|
| Creación de nodos y modificación de rutas | 31 |
| Creación de nodos | 31 |
| Enlazar y desenlazar nodos | 31 |
| Importar, sustituir y eliminar nodos | 33 |
| Atravesar los nodos de una ruta | 33 |
| Borrado o eliminación de elementos | 34 |
| Obtener información sobre los nodos | 34 |

Capítulo 4. API de scripts. 37

| | |
|---|----|
| Introducción a la API de scripts | 37 |
| Ejemplo 1: buscar nodos utilizando un filtro personalizado | 37 |
| Ejemplo 2: permitir a los usuarios obtener información de directorio o archivo basándose en sus privilegios | 37 |
| Metadatos: información sobre datos | 38 |
| Acceso a objetos generados | 41 |
| Manejo de errores | 42 |
| Parámetros de ruta, sesión y Supernodo | 43 |
| Valores globales | 46 |
| Trabajar con varias rutas: Scripts autónomos | 47 |

Capítulo 5. Sugerencias para scripts 49

| | |
|--|----|
| Modificación de ejecución de rutas | 49 |
| Nodos de recorrido en bucle | 49 |
| Acceso a objetos en el Repositorio de IBM SPSS Collaboration and Deployment Services | 50 |
| Generación de una contraseña codificada | 52 |
| Comprobación de script | 52 |
| Scripts desde la línea de comandos | 52 |
| Compatibilidad con releases anteriores | 53 |
| Acceder a resultados de ejecución de la ruta | 53 |
| Modelo de contenido de tabla | 54 |
| Modelo de contenido XML | 55 |
| Modelo de contenido JSON | 57 |
| Modelo de contenido de estadísticas de columna y modelo de contenido de estadísticas por pares | 58 |

Capítulo 6. Argumentos de la línea de comandos 63

| | |
|--|----|
| Invocación del software | 63 |
| Utilización de argumentos de la línea de comandos | 63 |
| Argumentos del sistema | 64 |
| Argumentos de parámetros | 65 |
| Argumentos de conexión del servidor | 66 |
| Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión | 67 |
| Argumentos de conexión de IBM SPSS Analytic Server | 67 |
| Combinación de varios argumentos | 68 |

Capítulo 7. Referencia de propiedades 69

| | |
|---|----|
| Visión general de referencia de propiedades | 69 |
| Sintaxis para propiedades | 69 |

| | |
|---|----|
| Ejemplos de las propiedades node y stream | 71 |
| Visión general de propiedades de nodo | 71 |
| Propiedades de nodos comunes | 71 |

Capítulo 8. Propiedades de ruta 73

Capítulo 9. Propiedades de nodos de origen 77

| | |
|--|-----|
| Propiedades comunes de nodos de origen | 77 |
| Propiedades de asimport | 81 |
| Propiedades del nodo cognosimport | 82 |
| propiedades de databasenode | 84 |
| Propiedades de datacollectionimportnode | 86 |
| Propiedades de excelimportnode | 88 |
| Propiedades de extensionimportnode | 89 |
| Propiedades de fixedfilenode | 92 |
| Propiedades del nodo gsdata_import | 94 |
| Propiedades de sasimportnode | 94 |
| propiedades de simgenode | 95 |
| Propiedades de statisticsimportnode | 97 |
| Propiedades del nodo tm1odataimport | 97 |
| Propiedades del nodo tm1import (en desuso) | 98 |
| Propiedades del nodo twcimport | 99 |
| propiedades de userinputnode | 100 |
| Propiedades de variablefilenode | 100 |
| Propiedades de xmlimportnode | 104 |
| Propiedades de dataviewimport | 104 |

Capítulo 10. Propiedades de nodos de operaciones con registros 107

| | |
|---|-----|
| propiedades de appendnode | 107 |
| propiedades de aggregatenode | 107 |
| propiedades de balancenode | 108 |
| Propiedades derive_stbnode | 109 |
| propiedades de distinctnode | 111 |
| Propiedades de extensionprocessnode | 112 |
| propiedades de mergenode | 114 |
| propiedades rfmaggregatenode | 115 |
| propiedades de samplenode | 118 |
| propiedades de selectnode | 119 |
| propiedades de sortnode | 120 |
| Propiedades de spacetimeboxes | 120 |
| Propiedades streamingtimeseries | 122 |
| Propiedades de cplexnode | 129 |

Capítulo 11. Propiedades de nodos de operaciones con campos 131

| | |
|--|-----|
| propiedades de anonymizenode | 131 |
| properties autodatapreprenode | 132 |
| propiedades de astimeintervalsnode | 135 |
| propiedades de binningnode | 135 |
| propiedades de derivenode | 138 |
| propiedades de ensamblenode | 140 |
| propiedades de fillernode | 141 |
| propiedades de filternode | 142 |
| propiedades de historynode | 143 |
| propiedades de partitionnode | 144 |
| propiedades de reclassifynode | 145 |
| propiedades de reordernode | 146 |

| | |
|---|-----|
| propiedades de reprojectnode | 146 |
| propiedades de restructurenode | 147 |
| propiedades de rfmanalysisnode | 147 |
| propiedades de settoflagnode | 149 |
| propiedades de statisticstransformnode | 149 |
| Propiedades timeintervalsnode (en desuso) | 150 |
| Propiedades de transposenode | 154 |
| propiedades de typenode | 155 |

Capítulo 12. Propiedades de nodos Gráfico 161

| | |
|--|-----|
| Propiedades comunes del nodo Gráfico | 161 |
| Propiedades de collectionnode | 162 |
| Propiedades de distributionnode | 163 |
| Propiedades de evaluationnode | 164 |
| Propiedades de graphboardnode | 166 |
| Propiedades de histogramnode | 168 |
| Propiedades de mapvisualizaton | 169 |
| Propiedades de multiplotnode | 173 |
| Propiedades de plotnode | 174 |
| Propiedades de timeplotnode | 176 |
| Propiedades de webnode | 177 |

Capítulo 13. Propiedades de nodos de modelado 179

| | |
|---|-----|
| Propiedades comunes de nodos de modelado | 179 |
| propiedades de anomalydetectionnode | 179 |
| propiedades de apriorinode | 181 |
| propiedades associationrulesnode | 182 |
| propiedades de autotransformnode | 185 |
| Propiedades de ajustes de algoritmo | 186 |
| propiedades de nodo de agrupación en clústeres automática | 187 |
| propiedades de autonumericnode | 188 |
| Propiedades de bayesnetnode | 190 |
| propiedades de c50node | 192 |
| propiedades de carmanode | 193 |
| propiedades de cartnode | 194 |
| propiedades de chaidnode | 197 |
| propiedades de coxregnode | 199 |
| Propiedades de decisionlistnode | 200 |
| propiedades de discriminantnode | 202 |
| Propiedades de extensionmodelnode | 203 |
| propiedades de factornode | 206 |
| propiedades de featureselectionnode | 207 |
| propiedades de genlinnode | 209 |
| Propiedades de glmnode | 212 |
| Propiedades de gle | 216 |
| propiedades de kmeansnode | 221 |
| propiedades de knnnode | 222 |
| propiedades de kohonenode | 223 |
| Propiedades de linearnode | 225 |
| Propiedades de linearasnode | 226 |
| Propiedades de logregnode | 227 |
| propiedades de lsvmnode | 231 |
| propiedades de neuralnetnode | 232 |
| Propiedades de neuralnetwork | 235 |
| propiedades de questnode | 236 |
| propiedades randomtrees | 238 |
| Propiedades de regressionnode | 240 |

| | |
|--|-----|
| propiedades de sequencenode | 242 |
| propiedades de slrmnode | 243 |
| propiedades de statisticsmodelnode | 244 |
| propiedades de stpnode | 244 |
| propiedades de svmnode | 248 |
| Propiedades de tcmlnode | 249 |
| Propiedades ts | 253 |
| propiedades de treeas | 259 |
| Propiedades de twostepnode | 261 |
| Propiedades de twostepAS | 262 |

Capítulo 14. Propiedades del nodo de nugget de modelo 265

| | |
|---|-----|
| Propiedades de applyanomalydetectionnode | 265 |
| Propiedades de applyapriorinode | 265 |
| Propiedades de applyassociationrulesnode | 266 |
| Propiedades de applyautoclassifiernode | 266 |
| Propiedades de applyautoclusternode | 267 |
| Propiedades de applyautonumericnode | 267 |
| Propiedades de applybayesnetnode | 267 |
| Propiedades de applyc50node | 267 |
| Propiedades de applycarmanode | 268 |
| Propiedades de applycartnode | 268 |
| Propiedades de applychaidnode | 268 |
| Propiedades de applycoxregnode | 269 |
| Propiedades de applydecisionlistnode | 269 |
| Propiedades de applydiscriminantnode | 269 |
| Propiedades de applyextension | 270 |
| Propiedades de applyfactornode | 271 |
| Propiedades de applyfeatureselectionnode | 272 |
| Propiedades de applygeneralizedlinearnode | 272 |
| Propiedades de applyglmnode | 272 |
| Propiedades de applygle | 273 |
| Propiedades de applykmeansnode | 273 |
| Propiedades de applyknnnode | 274 |
| Propiedades de applykohonenode | 274 |
| Propiedades de applylinearnode | 274 |
| Propiedades de applylinearasnode | 274 |
| Propiedades de applylogregnode | 275 |
| Propiedades de applysvmnode | 275 |
| Propiedades de applyneuralnetnode | 275 |
| propiedades de applyneuralnetworknode | 276 |
| Propiedades de applyocsvmnode | 276 |
| Propiedades de applyquestnode | 276 |
| Propiedades applyrandomtrees | 277 |
| Propiedades de applyregressionnode | 278 |
| propiedades de applyselflearningnode | 278 |
| Propiedades de applysequencenode | 278 |
| Propiedades de applysvmnode | 278 |
| Propiedades de applystpnode | 278 |
| Propiedades de applytcmlnode | 279 |
| Propiedades applyts | 279 |
| Propiedades applytimeseriesnode (en desuso) | 279 |
| Propiedades de applytreeas | 280 |
| Propiedades de applytwostepnode | 280 |
| Propiedades de applytwostepAS | 280 |
| Propiedades de applyxgboosttreenode | 281 |
| Propiedades de applyxgboostlinearnode | 281 |

Capítulo 15. Propiedades de nodos de modelado de bases de datos 283

| | |
|---|-----|
| Propiedades de nodos de modelado de Microsoft | 283 |
| Propiedades de nodos de modelado de Microsoft | 283 |
| Propiedades de nugget de modelo de Microsoft | 285 |
| Propiedades de nodos de modelado de Oracle | 287 |
| Propiedades de nodos de modelado de Oracle | 287 |
| Propiedades de nugget de modelo de Oracle | 293 |
| Propiedades de nodos de modelado de IBM | |
| Netezza Analytics | 294 |
| Propiedades de nodos de modelado de Netezza | 294 |
| Propiedades de nugget de modelo de Netezza | 305 |

Capítulo 16. Propiedades del nodo de resultados. 307

| | |
|---|-----|
| propiedades de analysisnode | 307 |
| propiedades de dataauditnode | 308 |
| Propiedades de extensionoutputnode | 310 |
| propiedades de matrixnode | 311 |
| propiedades de meansnode | 313 |
| propiedades de reportnode | 314 |
| propiedades de setglobalsnode | 316 |
| propiedades de simevalnode | 316 |
| propiedades de simfitnode | 317 |
| propiedades de statisticsnode | 318 |
| Propiedades de statisticsoutputnode | 319 |
| propiedades de tablenode | 319 |
| propiedades de transformnode | 322 |

Capítulo 17. Propiedades de nodos Exportar 325

| | |
|--|-----|
| Propiedades de nodos Exportar comunes | 325 |
| Propiedades de asexport | 325 |
| Propiedades del nodo de exportación Cognos | 326 |
| propiedades de databaseexportnode | 328 |
| Propiedades de datacollectionexportnode | 332 |
| Propiedades de excelexportnode | 332 |
| Propiedades de extensionexportnode | 333 |
| Propiedades de outputfilenode | 334 |
| Propiedades de sasexportnode | 335 |
| Propiedades de statisticsexportnode | 336 |
| Propiedades del nodo tm1odataexport | 336 |
| Propiedades del nodo tm1export (en desuso) | 337 |
| Propiedades de xmlexportnode | 339 |

Capítulo 18. Propiedades de nodos de IBM SPSS Statistics 341

| | |
|--|-----|
| Propiedades de statisticsimportnode | 341 |
| propiedades de statisticstransformnode | 341 |
| propiedades de statisticsmodelnode | 342 |
| Propiedades de statisticsoutputnode | 342 |
| Propiedades de statisticsexportnode | 343 |

Capítulo 19. Propiedades de nodo Python 345

| | |
|---|-----|
| Propiedades de smotnode | 345 |
| Propiedades de xgboosttreenode | 345 |
| Propiedades de xboostlinearnode | 347 |

Propiedades de ocsvmnode. 348

Capítulo 20. Propiedades de supernodo 351

Apéndice A. Referencia de nombres de nodo. 353

Nombres de nugget de modelo 353

Evitar nombres duplicados del modelo 355

Nombres de tipo de resultados 355

Apéndice B. Migración desde scripts de herencia a scripts Python. 357

Visión general de la migración de scripts de herencia 357

Diferencias generales 357

El contexto de los scripts 357

Comparativa de comandos y funciones. 357

Literales y comentarios 358

Operadores 358

Comandos condicionales y de bucle 359

Variables 360

Tipos modelo, resultado y nodo 360

Nombres de propiedades 360

Referencias de nodos 360

Obtener y establecer propiedades. 361

Edición de rutas 361

Operaciones de nodo. 362

Bucle 363

Ejecución de rutas. 363

Acceso a objetos mediante el sistema de archivos y el repositorio 364

Operaciones de ruta 365

Operaciones de modelo 365

Operaciones de resultado de documento 365

Otras diferencias entre scripts heredados y scripts Python 366

Python 366

Avisos 367

Marcas comerciales 368

Términos y condiciones para la documentación del producto 369

productos 369

documentación 369

del producto 369

documentación del producto 369

Capítulo 1. Scripts y lenguaje de scripts

Visión general de scripts

Los scripts en IBM® SPSS Modeler son una herramienta potente para automatizar procesos en la interfaz de usuario. Los scripts pueden realizar los mismos tipos de acciones que se realizan con el ratón o el teclado y se utilizan para automatizar tareas que resultarían extremadamente repetitivas o llevarían mucho tiempo si se realizaran manualmente.

Puede utilizar los scripts para:

- Imponer un orden concreto para la ejecución de nodos en una ruta.
- Establecer propiedades de los nodos y realizar derivaciones usando un subconjunto de CLEM (Control Language for Expression Manipulation).
- Especificar una secuencia automática de acciones que normalmente implican la interacción del usuario (por ejemplo, puede generar un modelo y comprobarlo a continuación).
- Configurar procesos complejos que requieren una interacción sustancial del usuario, como los procedimientos de validación cruzada que requieren una repetitiva generación y comprobación de modelo.
- Configurar procesos que manipulen rutas; por ejemplo, puede tomar una ruta de entrenamiento de modelo, ejecutarla y producir la ruta de comprobación del modelo automáticamente.

Este capítulo proporciona descripciones de alto nivel y ejemplos de scripts de nivel de ruta, scripts autónomos y scripts en Supernodos en la interfaz de IBM SPSS Modeler. Para obtener más información sobre el lenguaje, la sintaxis y los comandos consulte los capítulos siguientes.

Nota:

No puede importar y ejecutar scripts creados en IBM SPSS Statistics dentro de IBM SPSS Modeler.

Tipos de scripts

IBM SPSS Modeler utiliza tres tipos de scripts:

- Los **scripts de la ruta** se guardan como una propiedad de ruta y se guardan y se cargan con una ruta específica. Por ejemplo, puede escribir un script de ruta que automatice el proceso de entrenamiento y aplicación de un nugget de modelo. También puede especificar que cuando se ejecute una ruta particular, se ejecute el script, en lugar del contenido del lienzo de la ruta.
- Los **scripts autónomos** no están asociados a ninguna ruta en particular y se guardan en archivos de texto externos. Puede utilizar un script autónomo, por ejemplo, para manipular varias rutas a la vez.
- Los **scripts Supernodos** se guardan como una propiedad de ruta de supernodo. Los scripts Supernodos sólo están disponibles en supernodos terminales. Puede utilizar un script de supernodo para controlar la secuencia de ejecución del contenido del supernodo. En supernodos no terminales (origen o proceso), puede definir propiedades del supernodo o los nodos que contiene en su script de ruta directamente.

Scripts de ruta

Los scripts se pueden utilizar para personalizar operaciones dentro de una ruta particular y se guardan con esa ruta. Los scripts de la ruta se pueden utilizar para especificar un orden de ejecución particular para los nodos terminales de una ruta. El cuadro de diálogo del script de ruta se utiliza para editar el script que está guardado con la ruta actual.

Para acceder a la pestaña de scripts de ruta en el cuadro de diálogo Propiedades de ruta:

1. Desde el menú **Herramientas**, elija:

Propiedades de ruta > Ejecución

2. Pulse en la pestaña **Ejecución** para trabajar con scripts en la ruta actual.

Utilice los iconos de barra de herramientas en la parte superior del recuadro de diálogo del script de ruta para las operaciones siguientes:

- Importar el contenido de un script autónomo preexistente en la ventana.
- Guardar un script como archivo de texto.
- Imprimir un script.
- Añadir script predeterminado.
- Editar un script (deshacer, cortar, copiar, pegar y otras funciones de edición comunes).
- Ejecutar el script completo actual.
- Ejecutar líneas concretas de un script.
- Detener un script durante la ejecución. (Este icono sólo está habilitado cuando un script se está ejecutando).
- Comprobar la sintaxis del script y, si se encuentra algún error, mostrarlos para la revisión en el panel inferior del recuadro de diálogo.

Nota: A partir de la versión 16.0, SPSS Modeler utiliza el lenguaje de scripts Python. Todas las versiones anteriores a la 16.0 utilizaban un lenguaje de script exclusivo para SPSS Modeler, al que ahora se denomina como script de legado. Según el tipo de script con el que trabaje, en la pestaña **Ejecución** seleccione la modalidad de ejecución **Predeterminada (script opcional)** y, a continuación, seleccione **Python** o **Legacy**.

Puede especificar si un script se va a ejecutar o no cuando se ejecuta la ruta. Para ejecutar el script cada vez que se ejecuta la ruta, respetando el orden de ejecución del script, seleccione **Ejecutar este script**. De este modo se proporciona una automatización a nivel de ruta para acelerar la generación del modelo. Sin embargo, la configuración predeterminada es omitir el script durante la ejecución de la ruta. Incluso si selecciona la opción **Omitir este script**, siempre puede ejecutar la ruta directamente desde este cuadro de diálogo.

El editor de scripts incluye las siguientes características que ayudan a crear scripts:

- Resaltado de sintaxis. Se resaltan las palabras claves, los valores literales (tales como cadenas y números) y los comentarios.
- Numeración de líneas.
- Coincidencia de bloques. Cuando se coloca el cursor al inicio de un bloque de programa, también se resalta el bloque final correspondiente.
- Finalización automática sugerida.

Los colores y los estilos de texto que utiliza la función de resaltado de la sintaxis se pueden personalizar utilizando las preferencias de visualización de IBM SPSS Modeler. Para acceder a las preferencias de visualización, elija **Herramientas > Opciones > Opciones de usuario** y seleccione la pestaña **Sintaxis**.

Se puede acceder a una lista de finalizaciones de sintaxis sugeridas seleccionando la **Sugerencia automática** en el menú de contexto o pulsando Ctrl más espacio. Utilice las teclas de cursor para desplazarse hacia arriba y hacia abajo por la lista y, a continuación, pulse Intro para insertar el texto seleccionado. Para salir de la modalidad de sugerencia automática sin modificar el texto existente, pulse Esc.

La pestaña **Depurar** muestra mensajes de depuración y se puede utilizar para evaluar el estado del script una vez que se haya ejecutado. La pestaña **Depurar** está formada por un área de texto de solo lectura y

un campo de texto de entrada de una sola línea. El área de texto muestra el texto que se envía a la salida estándar o un error estándar mediante los scripts, por ejemplo, a través del texto del mensaje de error. El campo de texto de entrada toma la entrada del usuario. Esta entrada se evalúa dentro del contexto del script que se ha ejecutado más recientemente en el diálogo (conocido como el *contexto de los scripts*). El área de texto contiene el comando y la salida resultante, de modo que el usuario puede ver un rastreo de los comandos. El campo de entrada de texto siempre contiene el indicador de comandos (--> para el script de legado).

Un contexto de script nuevo se crea en las circunstancias siguientes:

- Se ejecuta un script utilizando **Ejecutar este script** o **Ejecutar líneas seleccionadas**.
- Se modifica el lenguaje de script.

Si se crea un nuevo contexto de script, el área de texto se borra.

Nota: La ejecución de una ruta fuera del panel del script no modifica el contexto del script del panel del script. Los valores de las variables que se han creado como parte de la ejecución no son visibles dentro del recuadro de diálogo de script.

Ejemplo de script de ruta: entrenamiento de una red neuronal

Una ruta se puede usar para entrenar un modelo de red neuronal cuando se ejecute. Normalmente, para comprobar el modelo, se inserta el nodo de modelado para agregar el modelo a la ruta, realizar las conexiones adecuadas y ejecutar el nodo Análisis.

Mediante un script de IBM SPSS Modeler se puede automatizar el proceso de comprobar el nugget de modelo tras crearlo. Por ejemplo, el siguiente script de ruta para la ruta de demostración `druglearn.str` (disponible en la carpeta `/Demos/streams/` de su instalación de IBM SPSS Modeler) se puede ejecutar desde el cuadro de diálogo de propiedades de ruta (**Herramientas > Propiedades de ruta > Ruta**):

```
stream = modeler.script.stream()
neuralnetnode = stream.findByType("neuralnetwork", None)
results = []
neuralnetnode.run(results)
appliernode = stream.createModelApplierAt(results[0], "Drug", 594, 187)
analysisnode = stream.createAt("analysis", "Drug", 688, 187)
typenode = stream.findByType("type", None)
stream.linkBetween(appliernode, typenode, analysisnode)
analysisnode.run([])
```

Los puntos siguientes describen cada línea de este ejemplo de script.

- La primera línea define una variable que apunta a la ruta actual.
- En la línea 2, el script busca el nodo generador Red neuronal.
- En la línea 3, el script crea una lista donde los resultados de la ejecución se pueden almacenar.
- En la línea 4, se crea el nugget de modelo Red neuronal. Se almacena en la lista definida en línea 3.
- En la línea 5, se crea un nodo de aplicación de modelo para el nugget de modelo y se coloca en el lienzo de rutas.
- En la línea 6, se crea un nodo de análisis denominado Drug.
- En la línea 7, el script busca el nodo Type.
- En la línea 8, el script conecta el nodo de aplicación de modelo creado en la línea 5 entre el nodo Type y el nodo Analysis.
- Finalmente, el nodo Análisis se ejecuta para producir el informe Análisis.

Es posible utilizar un script para crear y ejecutar una ruta desde cero, comenzando con un lienzo vacío. Para obtener más información sobre el lenguaje de script en general, consulte *Conceptos básicos del lenguaje de scripts*.

Límites de tamaño de código Jython

Jython compila cada script para el código de bytes Java que, a continuación, es ejecutado por la máquina virtual Java (JVM). Sin embargo, Java impone un límite en el tamaño de un único archivo de código de bytes. Así, cuando Jython intenta cargar el código de bytes, puede hacer que la JVM se bloquee. IBM SPSS Modeler no puede evitar que esto suceda.

Asegúrese de escribir los scripts Jython utilizando buenas prácticas de codificación (por ejemplo minimizando el código duplicado utilizando variables o funciones para calcular valores intermedios comunes). Si es necesario, puede que sea necesario dividir el código en varios archivos de origen o definirlo utilizando módulos porque éstos se compilan en archivos de código de bytes independientes.

Scripts autónomos

El cuadro de diálogo script autónomo se usa para crear o editar un script que se ha guardado como archivo de texto. En él se muestra el nombre del archivo y se proporcionan recursos para la carga, almacenamiento, importación y ejecución de scripts.

Para acceder al cuadro de diálogo del script autónomo:

En el menú principal, elija:

Herramientas > Script autónomo

Los scripts autónomos y los de ruta comparten las mismas opciones de comprobación de sintaxis de scripts y barra de herramientas. Consulte el tema “Scripts de ruta” en la página 1 para obtener más información.

Ejemplo de script autónomo: guardar y cargar un modelo

Los scripts autónomos son útiles para la manipulación de rutas. Presuponga que tiene dos rutas, una que crea un modelo y otra que usa diagramas para examinar el conjunto de reglas generado a partir de la primera ruta con campos de datos existentes. Un script autónomo para este escenario tendría un aspecto similar a éste:

```
taskrunner = modeler.script.session().getTaskRunner()

# Modify this to the correct Modeler installation Demos folder.
# Note use of forward slash and trailing slash.
installation = "C:/Program Files/IBM/SPSS/Modeler/18.1/Demos/"

# First load the model builder stream from file and build a model
druglearn_stream = taskrunner.openStreamFromFile(installation + "streams/druglearn.str", True)
results = []
druglearn_stream.findByType("c50", None).run(results)

# Save the model to file
taskrunner.saveModelToFile(results[0], "rule.gm")

# Now load the plot stream, read the model from file and insert it into the stream
drugplot_stream = taskrunner.openStreamFromFile(installation + "streams/drugplot.str", True)
model = taskrunner.openModelFromFile("rule.gm", True)
modelapplier = drugplot_stream.createModelApplier(model, "Drug")

# Now find the plot node, disconnect it and connect the
# model applier node between the derive node and the plot node
derivenode = drugplot_stream.findByType("derive", None)
plotnode = drugplot_stream.findByType("plot", None)
drugplot_stream.disconnect(plotnode)
```

```

modelapplier.setPositionBetween(derivinode, plotnode)
drugplot_stream.linkBetween(modelapplier, derivinode, plotnode)
plotnode.setPropertyValue("color_field", "%C-Drug")
plotnode.run([])

```

Nota: Para obtener más información sobre el lenguaje de scripts en general, consulte Conceptos básicos del lenguaje de scripts.

Ejemplo de script autónomo: Generación de un modelo de selección de características

Comenzando con un lienzo vacío, este ejemplo crea una ruta que genera un modelo de selección de características, aplica el modelo y crea una tabla que indica los 15 campos más importantes respecto al objetivo especificado.

```

stream = modeler.script.session().createProcessorStream("featureselection", True)

statisticsimportnode = stream.createAt("statisticsimport", "Statistics File", 150, 97)
statisticsimportnode.setPropertyValue("full_filename", "$CLEO_DEMOS/customer_dbase.sav")

typenode = stream.createAt("type", "Type", 258, 97)
typenode.setKeyedPropertyValue("direction", "response_01", "Target")

featureselectionnode = stream.createAt("featureselection", "Feature Selection", 366, 97)
featureselectionnode.setPropertyValue("top_n", 15)
featureselectionnode.setPropertyValue("max_missing_values", 80.0)
featureselectionnode.setPropertyValue("selection_mode", "TopN")
featureselectionnode.setPropertyValue("important_label", "Check Me Out!")
featureselectionnode.setPropertyValue("criteria", "Likelihood")

stream.link(statisticsimportnode, typenode)
stream.link(typenode, featureselectionnode)
models = []
featureselectionnode.run(models)

# Assumes the stream automatically places model apply nodes in the stream
applynode = stream.findByType("applyfeatureselection", None)
tablenode = stream.createAt("table", "Table", applynode.getXPosition() + 96,
applynode.getYPosition())
stream.link(applynode, tablenode)
tablenode.run([])

```

El script crea un nodo de origen para leer en los datos, utiliza un nodo Tipo para definir el rol del campo response_01 hacia Destino y, a continuación, crea y ejecuta un nodo Selección de características. Este script también conecta cada nodo y posiciones en el lienzo de la ruta para producir un diseño legible. El nugget de modelo resultante se conecta al nodo Tabla, que indica los 15 campos más importantes, tal y como determinan las propiedades selection_mode y top_n. Consulte el tema “propiedades de featureselectionnode” en la página 207 para obtener más información.

Scripts de Supernodo

Puede crear y guardar scripts en cualquier supernodo de terminal utilizando el lenguaje de script de IBM SPSS Modeler. Estos scripts sólo están disponibles para supernodos terminales y se suelen utilizar cuando crea rutas de plantilla o para imponer un orden de ejecución especial del contenido del supernodo. Los scripts de supernodo también permiten ejecutar más de un script en una ruta.

Por ejemplo, supongamos que necesita especificar el orden de ejecución de una ruta compleja y su supernodo contiene varios nodos, incluyendo un nodo Val. globales, que se debe ejecutar antes de derivar un nuevo campo utilizado en un nodo Gráfico. En este caso, puede crear un script de supernodo que

ejecute el nodo Val. globales en primer lugar. Los valores calculados por este nodo, como la media o la desviación estándar, se pueden usar posteriormente cuando se ejecute el nodo Gráfico.

En un script de Supernodo, puede especificar las propiedades del nodo de la misma manera que otros scripts. También puede cambiar y definir las propiedades de cualquier supernodo o sus nodos encapsulados directamente desde un script de ruta. Consulte el tema Capítulo 20, “Propiedades de supernodo”, en la página 351 para obtener más información. Este método funciona para supernodos de origen y proceso y supernodos terminales.

Nota: Puesto que solo los supernodos de terminal pueden ejecutar sus propios scripts, la pestaña Scripts del recuadro de diálogo Supernodo solo está disponible para los supernodos de terminal.

Para abrir el cuadro de diálogo de script de supernodo desde el lienzo principal:

Seleccione un supernodo terminal en el lienzo de rutas y, en el menú de supernodo, seleccione:

Script de Supernodo...

Para abrir el cuadro de diálogo de script de supernodo desde el lienzo de supernodo aumentado:

Pulse con el botón derecho del ratón en el lienzo del supernodo y en el menú contextual elija:

Script de Supernodo...

Script de supernodo de ejemplo

El siguiente script de supernodo establece el orden en que se ejecutarán los nodos terminales del supernodo. Este orden garantiza que el nodo Val. globales se ejecuta primero para que los valores que calcula este nodo se puedan utilizar cuando se ejecute otro nodo.

```
execute 'Val. globales'  
execute 'gains'  
execute 'profit'  
execute 'age v. $CC-pep'  
execute 'Table'
```

Bloquear y desbloquear supernodos

El ejemplo siguiente ilustra cómo puede bloquear y desbloquear un supernodo:

```
stream = modeler.script.stream()  
superNode=stream.findByID('id854RNTSD5MB')  
# desbloquear un supernodo  
print 'unlock the super node with password abcd'  
if superNode.unlock('abcd'):  
    print 'unlocked.'  
else:  
    print 'invalid password.'  
# bloquear un supernodo  
print 'lock the super node with password abcd'  
superNode.lock('abcd')
```

Creación de bucles y ejecución condicional en rutas

A partir de la versión 16.0, SPSS Modeler permite crear scripts básicos desde dentro de una ruta seleccionando valores en varios cuadros de diálogo en lugar de tener que escribir instrucciones directamente en lenguaje de script. Los dos principales tipos de script que pueden crearse de este modo son los bucles sencillos y un modo de ejecutar nodos si se cumple una condición.

En una misma ruta pueden combinarse bucles y reglas de ejecución condicional. Por ejemplo, suponga que tiene datos relativos a ventas de vehículos de fabricantes de todo el mundo. Podría crearse un bucle para procesar los datos en una ruta, identificando los detalles por país del fabricante, y sacar los datos a distintas gráficas para mostrar detalles tales como volumen de ventas por modelo, niveles de emisión por fabricante y cilindrada, etc. Si solo le interesara analizar la información procedente de Europa, también podría añadir condiciones al bucle que impidieran la creación de gráficas de fabricantes procedentes de América y Asia.

Nota: Puesto que tanto un bucle como una ejecución condicional están basados en scripts de segundo plano, solo se aplican a una ruta entera cuando se ejecuta.

- **Bucles** Los bucles pueden utilizarse para automatizar tareas repetitivas. Por ejemplo, esto podría suponer añadir un determinado número de nodos a una ruta y modificar un parámetro del nodo cada vez. De forma opcional, podría controlarse la ejecución de una ruta o rama varias veces, como en los ejemplos siguientes:
 - Ejecutar la ruta un determinado número de veces y cambiar el origen cada vez.
 - Ejecutar la ruta un determinado número de veces cambiando el valor de una variable cada vez.
 - Ejecutar la ruta un determinado número de veces especificando un campo adicional en cada ejecución.
 - Construir un modelo un determinado número de veces y cambiar la configuración del modelo cada vez.
- **Ejecución condicional** Puede utilizarse para controlar cómo ejecutan los nodos en función de condiciones definidas previamente como, por ejemplo:
 - Dependiendo de si un determinado valor es verdadero o falso, se controla la ejecución de un nodo.
 - Definir si la iteración de nodos se ejecutará en paralelo o de forma secuencial.

Tanto bucles ejecuciones condicionales se configuran en la pestaña Ejecución dentro del cuadro de diálogo Propiedades de ruta. Los nodos que se utilicen en bucles o de forma condicional aparecerán con un símbolo adicional en el lienzo de rutas para indicar que forman parte de una ejecución por bucles o condicional.

Puede accederse a la pestaña Ejecución de tres maneras:

- Mediante los menús de la parte superior del cuadro de diálogo principal:
 1. En el menú Herramientas, seleccione:
Propiedades de la ruta > Ejecución.
 2. Pulse en la pestaña Ejecución para trabajar con los scripts de la ruta actual.
- Dentro de una ruta:
 1. Pulse con el botón derecho en un nodo y seleccione **Bucles/Ejecución condicional.**
 2. Seleccione la opción de submenú que corresponda.
- En la barra de herramientas gráfica de la parte superior del cuadro de diálogo principal, pulse en el icono de propiedades de ruta.

Si es la primera vez que configura los detalles de un bucle o de una ejecución condicional, en la pestaña Ejecución seleccione el modo de ejecución **Ejecución de bucles/condicional** y después seleccione la subpestaña **Condicional** o **Bucles**.

Bucles en rutas

Con la creación de bucles puede automatizar las tareas repetitivas en las rutas, por ejemplo:

- Ejecutar la ruta un determinado número de veces y cambiar el origen cada vez.
- Ejecutar la ruta un determinado número de veces cambiando el valor de una variable cada vez.
- Ejecutar la ruta un determinado número de veces especificando un campo adicional en cada ejecución.

- Construir un modelo un determinado número de veces y cambiar la configuración del modelo cada vez.

Configurar las condiciones que deben cumplirse en la subpestaña **Bucle** de la pestaña Ejecución de la ruta. Para visualizar la subpestaña, seleccione el modo de ejecución **Ejecución en bucle/condicional**.

Los requisitos de bucle que defina entrarán en vigor cuando se ejecute la ruta, si se ha establecido la modalidad de ejecución **Ejecución en bucle/condicional**. De forma opcional, puede generar el código de script para los requisitos de bucle y pegarlo en el editor de scripts pulsando **Pegar...** en el ángulo inferior derecho de la subpestaña Bucle y la visualización de la pestaña Ejecución principal cambiará para mostrar la modalidad de ejecución **Predeterminada (script opcional)** con el script en la parte superior de la pestaña. Esto significa que puede definir bucles utilizando las diferentes opciones del cuadro de diálogo de bucle antes de generar un script que puede personalizar adicionalmente en el editor de scripts. Tenga en cuenta que cuando pulsa **Pegar...** los requisitos de bucle que ha definido también se mostrarán en el script generado.

Importante: Las variables de bucle que establezca en una ruta de SPSS Modeler pueden sustituirse si se ejecuta la ruta en un trabajo IBM SPSS Collaboration and Deployment Services. Esto se debe a que la entrada del editor del trabajo IBM SPSS Collaboration and Deployment Services sobrescribe la entrada de SPSS Modeler. Por ejemplo, si se establece una variable de bucle en la ruta para crear un nombre de archivo de salida diferente para cada bucle, los archivos se especifican correctamente en SPSS Modeler, pero son sustituidos por la entrada fija especificada en la pestaña Resultado del Gestor de despliegue de IBM SPSS Collaboration and Deployment Services.

Para configurar un bucle

1. Cree una clave de iteración para definir la estructura principal del bucle principal que se creará en una ruta. Consulte el tema Crear una clave de iteración para obtener más información.
2. Si es necesario, defina una o varias variables de iteración. Consulte el tema Crear una variable de iteración para obtener más información.
3. Las iteraciones y las variables que cree se muestran en el cuerpo principal de la subpestaña. De forma predeterminada, las iteraciones se ejecutan en el orden en que aparecen. Para subir o bajar una iteración en la lista, pulse la iteración para seleccionarla y, a continuación, utilice la flecha arriba o la flecha abajo de la columna de la derecha de la subpestaña para cambiar el orden.

Creación de una clave de iteración para bucles de rutas

Utilice una clave de iteración para definir la estructura principal del bucle principal que se creará en una ruta. Por ejemplo, si está analizando las ventas de automóviles, puede crear un parámetro de ruta *País de fabricación* y utilizarlo como la clave de la iteración. Cuando se ejecute la ruta, esta clave se establece en cada valor de país diferente de sus datos durante cada iteración. Utilice el cuadro de diálogo Definir clave de iteración para configurar la clave.

Para abrir el cuadro de diálogo, seleccione el botón **Clave de iteración...** en el ángulo inferior izquierdo de la subpestaña Bucle o pulse con el derecho cualquier nodo de la ruta y seleccione **Ejecución en bucle/condicional > Definir clave de iteración (campos)** o **Ejecución en bucle/condicional > Definir clave de iteración (valores)**. Si abre el cuadro de diálogo desde la ruta, algunos campos se completan automáticamente, tales como el nombre del nodo.

Para configurar una clave de iteración, complete los campos siguientes:

Iterar en. Puede seleccionar entre una de las opciones siguientes:

- **Parámetro de ruta - Campos.** Utilice esta opción para crear un bucle que establezca el valor de un parámetro de ruta existente en cada campo especificado de forma ordenada.
- **Parámetro de ruta - Valores.** Utilice esta opción para crear un bucle que establezca el valor de un parámetro de ruta existente en cada valor especificado de forma ordenada.

- **Propiedad del nodo - Campos.** Utilice esta opción para crear un bucle que establezca el valor de una propiedad de nodo en cada campo especificado de forma ordenada.
- **Propiedad del nodo - Valores.** Utilice esta opción para crear un bucle que establezca el valor de una propiedad de nodo en cada valor especificado de forma ordenada.

Qué se ha de establecer. Elija el elemento cuyo valor se establecerá cada vez que se ejecute el bucle. Puede seleccionar entre una de las opciones siguientes:

- **Parámetro.** Solo está disponible si se selecciona **Parámetro de ruta - Campos** o **Parámetro de ruta - Valores**. Seleccione el parámetro necesario en la lista disponible.
- **Nodo.** Solo está disponible si se selecciona **Propiedad del nodo - Campos** o **Propiedad del nodo - Valores**. Seleccione el nodo para el que desee configurar un bucle. Pulse el botón Examinar para abrir el diálogo Seleccionar nodo y elija el nodo que desee. Si hay demasiados nodos en la lista, puede filtrar la visualización para que únicamente se muestren los nodos de una de las siguientes categorías: Origen, Proceso, Gráfico, Modelado, Resultados o Aplicar modelo.
- **Propiedad.** Solo está disponible si se selecciona **Propiedad del nodo - Campos** o **Propiedad del nodo - Valores**. Seleccione la propiedad del nodo en la lista disponible.

Campos de uso. Solo está disponible si se selecciona **Parámetro de ruta - Campos** o **Propiedad del nodo - Campos**. Seleccione el campo o los campos de un nodo que se utilizarán para proporcionar los valores de iteración. Puede seleccionar entre una de las opciones siguientes:

- **Nodo.** Solo está disponible si se selecciona **Parámetro de ruta - Campos**. Seleccione el nodo que contiene los detalles para los que desee configurar un bucle. Pulse el botón Examinar para abrir el diálogo Seleccionar nodo y elija el nodo que desee. Si hay demasiados nodos en la lista, puede filtrar la visualización para que únicamente se muestren los nodos de una de las siguientes categorías: Origen, Proceso, Gráfico, Modelado, Resultados o Aplicar modelo.
- **Lista de campos.** Pulse el botón de lista de la columna derecha para visualizar el cuadro de diálogo Seleccionar campos, donde puede seleccionar los campos del nodo para proporcionar los datos de iteración. Para obtener más información, consulte “Selección de campos en iteraciones” en la página 10.

Valores de uso. Solo está disponible si se selecciona **Parámetro de ruta - Valores** o **Propiedad del nodo - Valores**. En el campo seleccionado, seleccione el o los valores que se utilizarán como valores de iteración. Puede seleccionar entre una de las opciones siguientes:

- **Nodo.** Solo está disponible si se selecciona **Parámetro de ruta - Valores**. Seleccione el nodo que contiene los detalles para los que desee configurar un bucle. Pulse el botón Examinar para abrir el diálogo Seleccionar nodo y elija el nodo que desee. Si hay demasiados nodos en la lista, puede filtrar la visualización para que únicamente se muestren los nodos de una de las siguientes categorías: Origen, Proceso, Gráfico, Modelado, Resultados o Aplicar modelo.
- **Lista de campos.** Seleccione el campo del nodo para proporcionar los datos de iteración.
- **Lista de valores.** Pulse el botón de lista de la columna derecha para visualizar el cuadro de diálogo Seleccionar valores, donde puede seleccionar los valores del campo para proporcionar los datos de iteración.

Creación de una variable de iteración para bucles de rutas

Puede utilizar variables de iteración para cambiar los valores de los parámetros o las propiedades de ruta de los nodos seleccionados en una ruta, cada vez que se ejecute un bucle. Por ejemplo, si el bucle de ruta está analizando los datos de ventas de automóviles y utiliza *País de fabricación* como clave de iteración, puede tener un gráfico de resultados que muestre las ventas por modelo y otro gráfico de resultados que muestre información sobre emisiones contaminantes. En estos casos puede crear variables de iteración que creen nuevos títulos para los gráficos resultantes, tales como *Emisiones de vehículos suecos* y *Ventas de automóviles japoneses por modelo*. Utilice el cuadro de diálogo Definir variable de iteración para configurar las variables que necesite.

Para abrir el cuadro de diálogo, seleccione el botón **Añadir variable...** en el ángulo superior izquierdo de la subpestaña Bucle, o pulse con el botón derecho cualquier nodo de la ruta y seleccione **Ejecución en bucle/condicional > Definir variable de iteración**.

Para configurar una variable de iteración, complete los campos siguientes:

Cambiar. Seleccione el tipo de atributo que desea enmendar. Puede elegir **Parámetro de ruta** o **Propiedad del nodo**.

- Si selecciona **Parámetro de ruta**, elija el parámetro necesario y, a continuación, utilizando una de las opciones siguientes, si están disponibles en su ruta, defina cómo se debe establecer el valor de dicho parámetro con cada iteración del bucle:
 - **Variable global.** Seleccione la variable global en la que se debe establecer el parámetro de ruta.
 - **Casilla de resultados de tabla.** Para que un parámetro de ruta sea el valor de una casilla de resultados de tabla, seleccione la tabla en la lista y especifique la **Fila** y la **Columna** que se han de utilizar.
 - **Especificar manualmente.** Seleccione esta opción si desea especificar manualmente el valor que tomará este parámetro en cada iteración. Cuando regrese a la subpestaña Bucle se habrá creado una columna nueva en la que puede especificar el texto necesario.
- Si selecciona **Propiedad del nodo**, elija el nodo necesario y una de sus propiedades, a continuación, establezca el valor que desea que se utilice para dicha propiedad. Establezca el nuevo valor de la propiedad utilizando una de las opciones siguientes:
 - **Solo.** El valor de la propiedad utilizará el valor de la clave de iteración. Para obtener más información, consulte “Creación de una clave de iteración para bucles de rutas” en la página 8.
 - **Como prefijo de tallo.** Utiliza el valor de la clave de iteración como un prefijo para lo que especifique en el campo **Tallo**.
 - **Como sufijo de tallo.** Utiliza el valor de la clave de iteración como un sufijo para lo que especifique en el campo **Tallo**.

Si selecciona la opción de prefijo o de sufijo se le solicitará que añada el texto adicional en el campo **Tallo**. Por ejemplo, si el valor de la clave de iteración es *País de fabricación* y selecciona **Como prefijo de tallo**, puede entrar - *ventas por modelo* en este campo.

Selección de campos en iteraciones

Cuando se crean iteraciones pueden seleccionarse uno o más campos mediante el cuadro de diálogo Seleccionar campos.

Ordenar por: puede ordenar campos disponibles para su visualización eligiendo una de las siguientes opciones:

- **Natural:** el orden de los campos es aquél en que pasaron desde la parte anterior de la ruta de datos al nodo actual.
- **Nombre:** ordena los campos siguiendo un orden alfabético para su visualización.
- **Tipo:** ordena los campos en función de su nivel de medición. Esta opción es útil cuando se seleccionan campos con un nivel de medición en particular.

Seleccione los campos de la lista de uno en uno o mantenga pulsada la tecla Mayús o Ctrl mientras selecciona otros campos para seleccionar varios campos. También puede utilizar los botones que se muestran bajo la lista para seleccionar grupos de campos en función de su nivel de medición o seleccionar y anular la selección de todos los campos de la tabla.

Tenga en cuenta que los campos disponibles para su selección se filtran para mostrar solo los campos que son adecuados para el parámetro de ruta o la propiedad de nodo que se están utilizando. Por ejemplo, si se está utilizando un parámetro de ruta que tiene un tipo de almacenamiento de cadena, solo se mostrarán los campos que tengan un tipo de almacenamiento de cadena.

Ejecución condicional en rutas

Con la ejecución condicional puede controlar cómo se ejecutan los nodos terminales, en función de las condiciones de coincidencia de contenido de ruta que defina. Ejemplos de ello pueden ser los siguientes:

- Dependiendo de si un determinado valor es verdadero o falso, se controla la ejecución de un nodo.
- Definir si la iteración de nodos se ejecutará en paralelo o de forma secuencial.

Configurar las condiciones que deben cumplirse en la subpestaña **Condicional** de la pestaña Ejecución de la ruta. Para visualizar la subpestaña, seleccione el modo de ejecución **Ejecución en bucle/condicional**.

Los requisitos de ejecución condicional que defina entrarán en vigor cuando se ejecute la ruta, si se ha establecido la modalidad de ejecución **Ejecución en bucle/condicional**. De forma opcional, puede generar el código de script para los requisitos de ejecución condicional y pegarlo en el editor de scripts pulsando **Pegar...** en el ángulo inferior derecho de la subpestaña Condicional; la visualización de la pestaña Ejecución principal cambiará para mostrar la modalidad de ejecución **Predeterminada (script opcional)** con el script en la parte superior de la pestaña. Esto significa que puede definir condiciones utilizando las diferentes opciones del cuadro de diálogo de bucle antes de generar un script que puede personalizar adicionalmente en el editor de scripts. Tenga en cuenta que cuando pulsa **Pegar...** los requisitos de bucle que ha definido también se mostrarán en el script generado.

Para configurar una condición:

1. En la columna de la derecha de la subpestaña Condicional, pulse el botón Añadir condición nueva



para abrir el cuadro de diálogo Añadir sentencia de ejecución condicional. En este diálogo especifica la condición que se debe cumplir para que se ejecute el nodo.

2. En el cuadro de diálogo Sentencia de ejecución condicional, especifique lo siguiente:
 - a. **Nodo.** Seleccione el nodo para el que desee configurar una ejecución condicional. Pulse el botón Examinar para abrir el diálogo Seleccionar nodo y elija el nodo que desee. Si hay demasiados nodos en la lista, puede filtrar la visualización para que únicamente se muestren los nodos de una de las siguientes categorías: Exportar, Gráfico, Modelado o Resultados.
 - b. **Condición basada en.** Especifique la condición que se debe cumplir para que se ejecute el nodo. Puede elegir una de estas cuatro opciones: **Parámetro de ruta**, **Variable global**, **Casilla de resultados de tabla** o **Siempre verdadero**. Los detalles que especifique en la mitad inferior del cuadro de diálogo están controlados por la condición que elija.
 - **Parámetro de ruta.** Seleccione el parámetro de la lista disponible y, a continuación, seleccione el **Operador** para ese parámetro; por ejemplo, el operador puede ser Más, Igual, Menor que, Entre, etc. A continuación especifique el **Valor**, o los valores mínimos o máximos, dependiendo del operador seleccionado.
 - **Variable global.** Seleccione la variable de la lista disponible; por ejemplo, esto podría incluir: Media, Suma, Valor mínimo, Valor máximo o Desviación estándar. A continuación, seleccione **Operador** y los valores necesarios.
 - **Casilla de resultados de tabla.** Seleccione el nodo de tabla de la lista disponible y, a continuación, seleccione la **Fila** y la **Columna** en la tabla. A continuación, seleccione **Operador** y los valores necesarios.
 - **Siempre verdadero.** Seleccione esta opción si siempre se ha de ejecutar el nodo. Si selecciona esta opción, no hay parámetros adicionales que seleccionar.
3. Repita los pasos 1 y 2 tantas veces como sea necesario hasta que haya configurado todas las condiciones que requiere. El nodo que ha seleccionado y la condición que se debe cumplir antes de que se ejecute el nodo se muestran en el cuerpo principal de la subpestaña de las columnas **Ejecutar nodo** y **Si esta condición es verdadera** respectivamente.
4. De forma predeterminada, los nodos y las condiciones se ejecutan en el orden en que aparecen. Para subir o bajar un nodo y condición en la lista, pulse el nodo para seleccionarlo y, a continuación, utilice la flecha arriba o la flecha abajo en la columna de la derecha de la subpestaña para cambiar el orden.

Además, puede establecer las siguientes opciones en la parte inferior de la subpestaña Condicional:

- **Evaluar todo en orden.** Seleccione esta opción para evaluar cada condición en el orden en que se muestra en la subpestaña. Los nodos para los que se han encontrado condiciones que son "True" se ejecutarán una vez evaluadas todas las condiciones.
- **Ejecutar uno por uno.** Sólo está disponible si se selecciona **Evaluar todo en orden**. Si se selecciona significa que si la condición se evalúa como "True", el nodo asociado con esa condición se ejecutará antes de que se evalúe la siguiente condición.
- **Evaluar hasta primer acierto.** Si se selecciona, significa que solo se ejecutará el primer nodo cuya evaluación de las condiciones devuelva el valor "True".

Ejecutar e interrumpir scripts

Existen diversas formas de ejecutar scripts. Por ejemplo, en el script de ruta o en el cuadro de diálogo del script, el botón "Ejecutar este script" ejecuta el script completo:



Figura 1. Botón Ejecutar este script

El botón "Ejecutar líneas seleccionadas" ejecuta una única línea, o un bloque de líneas adyacentes, que ha seleccionado en el script:



Figura 2. Botón Ejecutar líneas seleccionadas

Un script se puede ejecutar mediante cualquiera de los siguientes métodos:

- Pulse en el botón "Ejecutar este script" o "Ejecutar líneas seleccionadas" dentro de un script de ruta o un cuadro de diálogo de script.
- Ejecutando una ruta donde **Ejecutar este script** esté establecido como el método de ejecución predeterminado.
- Utilizando la marca `-execute` al inicio en modo interactivo. Consulte el tema "Utilización de argumentos de la línea de comandos" en la página 63 para obtener más información.

Nota: Un script de supernodo se ejecuta cuando el supernodo se ejecuta, siempre que haya seleccionado **Ejecutar este script** en el recuadro de diálogo Script de supernodo.

Interrumpir ejecución de script

En el cuadro de diálogo de scripts de ruta, se activará el botón rojo de detención durante la ejecución de scripts. Pulsando este botón, puede abandonar la ejecución del script y de cualquier ruta actual.

Buscar y reemplazar

El cuadro de diálogo Buscar/reemplazar está disponible en lugares donde edita texto de script o de expresión, incluido el editor de scripts, el generador de expresiones CLEM o cuando define una plantilla en el nodo Informe. Cuando edite texto en cualquiera de estas áreas, pulse **Ctrl** para acceder al cuadro de diálogo, asegurándose de que el cursor está centrado en un área de texto. Por ejemplo, si trabaja en un nodo Rellenar, puede acceder al cuadro de diálogo desde cualquiera de las áreas de texto de la pestaña Configuración o desde el campo de texto del generador de expresiones.

1. Con el cursor en un área de texto, pulse **Ctrl+F** para acceder al cuadro de diálogo Buscar/reemplazar.

2. Introduzca el texto que desee buscar o selecciónelo de la lista desplegable de elementos buscados recientemente.
3. Introduzca el texto de reemplazo, si lo tiene.
4. Pulse en **Buscar siguiente** para iniciar la búsqueda.
5. Pulse en **Reemplazar** para reemplazar la sección actual o en **Reemplazar todos** para actualizar todas las instancias o sólo las seleccionadas.
6. El cuadro de diálogo se cierra después de cada operación. Pulse F3 desde cualquier área de texto para repetir la operación de búsqueda más reciente o pulse Ctrl+F para volver a acceder al cuadro de diálogo.

Opciones de búsqueda

Coincidir mayúsculas y minúsculas. Especifica si la operación de búsqueda hace distinción entre mayúsculas y minúsculas; por ejemplo, si *miVar* es igual que *miVar*. El texto de reemplazo siempre se introduce exactamente como se ha introducido, independientemente de este ajuste.

Sólo palabras completas. Especifica si la operación de búsqueda tiene en cuenta el texto incluido dentro de las palabras. Por ejemplo, si se selecciona, la búsqueda de *fuego* no será igual que *cortafuegos* ni que *corta-fuegos*.

Expresiones regulares. Especifica si se utiliza la sintaxis de expresiones regulares (consulte la sección siguiente). Si está seleccionado, la opción **Sólo palabras completas** está desactivada y su valor se ignora.

Sólo texto seleccionado. Controla el ámbito de la búsqueda al utilizar la opción **Reemplazar todos**.

Sintaxis de expresiones regulares

Las expresiones regulares le permiten buscar caracteres especiales como caracteres de tabulador o de nueva línea, clases o rangos de caracteres como de la *a* a la *d*, cualquier dígito o no dígito y límites como el principio o el final de una línea. Se admiten los siguientes tipos de expresiones.

Tabla 1. Coincidencias de caracteres.

| Caracteres | Coincidencias |
|------------|---|
| x | El carácter x |
| \\ | El carácter de barra inclinada invertida |
| \\0n | El carácter con valor octal 0n (0 <= n <= 7) |
| \\0nn | El carácter con valor octal 0nn (0 <= n <= 7) |
| \\0mnn | El carácter con valor octal 0mnn (0 <= m <= 3, 0 <= n <= 7) |
| \\xhh | El carácter con valor hexadecimal 0xhh |
| \\uhhhh | El carácter con valor hexadecimal 0xhhhh |
| \\t | El carácter de tabulador ('\\u0009') |
| \\n | El carácter de nueva línea (avance de línea) ('\\u000A') |
| \\r | El carácter de retorno de carro ('\\u000D') |
| \\f | El carácter de avance de página ('\\u000C') |
| \\a | El carácter de alerta (campana) ('\\u0007') |
| \\e | El carácter de escape ('\\u001B') |
| \\cx | El carácter de control correspondiente a x |

Tabla 2. Clases de caracteres coincidentes.

| Clases de caracteres | Coincidencias |
|----------------------|---|
| [abc] | a, b o c (clase simple) |
| [^abc] | Cualquier carácter excepto a, b o c (resta) |
| [a-zA-Z] | De la a a la z o de la A a la Z, ambas inclusive (rango) |
| [a-d[m-p]] | De la a a la d o de la m a la p (unión). Esto también puede especificarse como [a-dm-p]. |
| [a-z&&[def]] | De la a a la z y d, e o f (intersección) |
| [a-z&&[^bc]] | De la a a la z, excepto b y c (resta). Esto también puede especificarse como [ad-z]. |
| [a-z&&[^m-p]] | De la a a la z y no de la m a la p (resta). Esto también puede especificarse como [a-lq-z]. |

Tabla 3. Clases de caracteres predefinidas.

| Clases de caracteres predefinidas | Coincidencias |
|-----------------------------------|---|
| . | Cualquier carácter (puede o no coincidir con los terminadores de línea) |
| \d | Cualquier dígito: [0-9] |
| \D | Un no dígito: [^0-9] |
| \s | Un carácter de espacio en blanco: [\t\n\r] |
| \S | Un carácter de espacio en blanco: [^\s] |
| \w | Un carácter de palabra: [a-zA-Z_0-9] |
| \W | Un carácter que no sea de palabra: [^\w] |

Tabla 4. Coincidencias de límite.

| Reconocedores de límite | Coincidencias |
|-------------------------|---|
| ^ | El comienzo de una línea |
| \$ | El final de una línea |
| \b | Un límite alfabético |
| \B | Un límite no alfabético |
| \A | El comienzo de la entrada |
| \Z | El final de la entrada pero para el terminador final, si lo hay |
| \z | El final de la entrada |

Capítulo 2. Language de scripts

Visión general de lenguaje de script

El recurso de scripts para IBM SPSS Modeler le permite crear scripts que funcionen en la interfaz de usuario de SPSS Modeler, manipular los objetos de salida y ejecutar la sintaxis de comandos. Puede ejecutar scripts directamente desde SPSS Modeler.

Los scripts de IBM SPSS Modeler están escritos en el lenguaje de script Python. La implementación Java de Python que IBM SPSS Modeler se denomina Jython. El lenguaje de script consta de las siguientes características:

- Un formato para hacer referencia a nodos, rutas, proyectos, resultados y otros objetos de IBM SPSS Modeler.
- Un conjunto de instrucciones o comandos de scripts que se puede utilizar para manipular tales objetos.
- Un lenguaje de expresión de script para establecer los valores de las variables, los parámetros y otros objetos.
- Compatibilidad con comentarios, continuaciones y bloques de texto literal.

Las secciones siguientes describen el lenguaje de scripts Python, la implementación de Jython por parte de Python y la sintaxis básica para empezar a crear scripts en IBM SPSS Modeler. Las secciones siguientes recogen información sobre comandos y propiedades específicas.

Python y Jython

Jython es una implementación del lenguaje de scripts Python, escrito en el lenguaje Java e integrado con la plataforma Java. Python es un potente lenguaje de script orientado a objetos. Jython es útil porque proporciona las características de productividad de un lenguaje de script maduro y, a diferencia de Python, se ejecuta en cualquier entorno que soporte una máquina virtual Java (JVM). Esto significa que las bibliotecas Java de la máquina virtual Java están disponibles para utilizarlas cuando se escriben programas. Con Jython, puede beneficiarse de esta diferencia y utilizar la sintaxis y la mayoría de las características del lenguaje Python.

Como lenguaje de script, Python (y su implementación Jython) es fácil de aprender y ofrece una codificación eficaz con la estructura mínima necesaria para crear un programa de ejecución. El código se puede entrar de forma interactiva, es decir, una línea cada vez. Python es un lenguaje de script interpretado; no hay ningún paso de precompilación, tal como existe en Java. Los programas de Python simplemente son archivos de texto que se interpretan a medida que se entran (después de analizar los errores de sintaxis). Las expresiones simples, tales como los valores definidos, y también las acciones más complejas, tales como las definiciones de función, se ejecutan y están disponibles para su uso de forma inmediata. Los cambios realizados en el código se pueden probar fácilmente. Sin embargo, la interpretación del script tiene algunas desventajas. Por ejemplo, utilizar una variable no definida no es un error del compilador, por lo tanto solo se detecta si (y cuando) se ejecuta la sentencia en la que se utiliza la variable. En este caso, se puede editar y ejecutar el programa para depurar el error.

Python lo ve todo como un objeto, incluidos todos los datos y el código. Por lo tanto, puede manipular estos objetos con líneas de código. Algunos tipos de selección, tales como los números y cadenas, se consideran valores y no objetos, lo cual resulta más práctico, y Python da soporte a todo ello. Se da soporte a un valor nulo. Este valor nulo tiene el nombre reservado de None.

Para obtener una introducción más detallada del lenguaje de script Python y Jython y algunos scripts de ejemplo, consulte el tema <http://www.ibm.com/developerworks/java/tutorials/j-jython1/j-jython1.html> y el tema <http://www.ibm.com/developerworks/java/tutorials/j-jython2/j-jython2.html>.

Scripts de Python

Esta guía del lenguaje de script de Python es una introducción a los componentes que tienen más probabilidad de ser utilizados cuando se ejecutan scripts en IBM SPSS Modeler, incluidos conceptos y principios básicos de programación. Le proporcionará los conocimientos suficientes para comenzar a desarrollar sus propios scripts Python y utilizarlos en IBM SPSS Modeler.

Operaciones

a asignación se realiza mediante un signo de igual (=). Por ejemplo, para asignar el valor "3" a una variable llamada "x" debe utilizar la siguiente sentencia:

```
x = 3
```

El signo igual también se utiliza para asignar datos de tipo de cadena a una variable. Por ejemplo, para asignar el valor "a string value" a la variable "y" utilice la sentencia siguiente:

```
y = "a string value"
```

La tabla siguiente enumera algunas de las operaciones numéricas y de comparación utilizadas con más frecuencia y sus descripciones.

Tabla 5. Operaciones numéricas y de comparación comunes

| Operación | Descripción |
|-------------|------------------------------|
| $x < y$ | ¿Es x menor que y? |
| $x > y$ | ¿Es x mayor que y? |
| $x \leq y$ | ¿Es x menor que o igual a y? |
| $x \geq y$ | ¿Es x mayor que o igual a y? |
| $x == y$ | ¿Es x igual a y? |
| $x != y$ | ¿Es x no igual a y? |
| $x \lt;> y$ | ¿Es x no igual a y? |
| $x + y$ | Sumar y a x |
| $x - y$ | Restar y de x |
| $x * y$ | Multiplicar x por y |
| x / y | Dividir x por y |
| $x ** y$ | Elevar x a la potencia de y |

Listas

Las listas son secuencias de elementos. Una lista puede contener cualquier número de elementos, y los elementos de la lista pueden ser cualquier tipo de objeto. Las listas también se pueden considerar como matrices. El número de elementos de una lista puede aumentar o disminuir a medida que se añaden, eliminan o sustituyen elementos.

Ejemplos

| | |
|--------------------------------------|--|
| <code>[]</code> | Cualquier lista vacía. |
| <code>[1]</code> | Una lista con un solo elemento, un entero. |
| <code>["Mike", 10, "Don", 20]</code> | Una lista con cuatro elementos, dos elementos de cadena y dos elementos de entero. |
| <code>[[], [7], [8, 9]]</code> | Una lista de listas. Cada sublista es una lista vacía o una lista de elementos de enteros. |

```
x = 7; y = 2; z = 3;
[1, x, y, x + y]
```

Una lista de enteros. Este ejemplo muestra el uso de variables y expresiones.

Puede asignar una lista a una variable, por ejemplo:

```
mylist1 = ["one", "two", "three"]
```

A continuación, puede acceder a los elementos específicos de la lista, por ejemplo:

```
mylist[0]
```

Esto genera el resultado siguiente:

```
one
```

El número entre corchetes ([]) se considera un *index* y hace referencia a un elemento concreto de la lista. Los elementos de una lista se indexan a partir de 0.

También puede seleccionar un rango de elementos de una lista; esto se denomina *porciones*. Por ejemplo, `x[1:3]` selecciona el segundo y el tercer elemento de `x`. El índice final es uno más allá de la selección.

Cadenas

Una *cadena* es una secuencia inmutable de caracteres que se trata como un valor. Las cadenas dan soporte a todas las funciones de secuencias inmutables y operadores que generan como resultado una nueva serie. Por ejemplo, `"abcdef"[1:4]` da como resultado la salida `"bcd"`.

En Python, los caracteres se representan mediante cadenas de caracteres de longitud uno.

Los literales de cadenas se definen mediante comillas simples o triples. Las cadenas definidas mediante comillas simples no pueden abarcar líneas, mientras que las series definidas mediante comillas triples sí que pueden. Una cadena puede estar entre comillas simples (') o entre comillas dobles ("). Un carácter entrecomillado puede contener el otro carácter entrecomillado o el carácter entrecomillado de escape, que es el carácter de barra invertida (\).

Ejemplos

```
"Esta es una cadena"
'Esta también es una cadena'
"Es una cadena"
'Este manual se titula "Guía de scripts y automatización Python".'
"Estas son comillas de escape (\") en una cadena entrecomillada"
```

El analizador de Python automáticamente concatena varias cadenas separadas por un espacio en blanco. Esto facilita la entrada de cadenas largas y la combinación de tipos de comillas en una sola cadena, por ejemplo:

```
"Esta cadena utiliza ' y " 'esta cadena utiliza ".'
```

Esto resulta en la siguiente salida:

```
Esta cadena utiliza ' y esa cadena utiliza ".
```

Las cadenas dan soporte a varios métodos útiles. Algunos de estos métodos se proporcionan en la tabla siguiente.

Tabla 6. Métodos de serie

| Método | Uso |
|--|---|
| <code>s.capitalize()</code> | Mayúscula inicial <code>s</code> |
| <code>s.count(ss {,start {,end}})</code> | Recuento de apariciones de <code>ss</code> en <code>s[start:end]</code> |

Tabla 6. Métodos de serie (continuación)

| Método | Uso |
|--|---|
| s.startswith(str {, start {, end}}) s.endswith(str {, start {, end}}) | Probar si s comienza por str Probar si s acaba en str |
| s.expandtabs({size}) | Sustituir tabulaciones por espacios, el valor predeterminado de size es 8 |
| s.find(str {, start {, end}}) s.rfind(str {, start {, end}}) | Busca el primer índice de str en s; si no se encuentra, el resultado es -1. rfind busca de derecha a izquierda. |
| s.index(str {, start {, end}}) s.rindex(str {, start {, end}}) | Busca el primer índice de str en s; si no se encuentra: se genera ValueError. rindex busca de derecha a izquierda. |
| s.isalnum | Probar si la cadena es alfanumérica |
| s.isalpha | Probar si la cadena es alfabética |
| s.isnum | Probar si la cadena es numérica |
| s.isupper | Probar si la cadena está toda en mayúsculas |
| s.islower | Probar si la cadena está toda en minúsculas |
| s.isspace | Probar si la cadena está toda en espacios en blanco |
| s.istitle | Probar si la cadena es una secuencia de cadenas alfanuméricas con mayúscula inicial |
| s.lower() s.upper() s.swapcase() s.title() | Convertir todo a minúsculas Convertir todo a mayúsculas Convertir de mayúsculas a minúsculas o viceversa Convertir todo a mayúsculas o minúsculas del título |
| s.join(seq) | Unir las cadenas de seq con s como separador |
| s.splitlines({keep}) | Dividir s en líneas, si keep es true, mantener las nuevas líneas |
| s.split({sep {, max}}) | Dividir s en "palabras" utilizando sep (el valor predeterminado de sep es un espacio en blanco) para un máximo de max veces |
| s.ljust(width) s.rjust(width) s.center(width) s.zfill(width) | Justificar cadena a izquierda ancho de campo width Justificar cadena a derecha ancho de campo width Justificar cadena al centro ancho de campo width Rellenar con 0. |
| s.lstrip() s.rstrip() s.strip() | Eliminar espacios en blanco iniciales Eliminar espacios en blanco de cola Eliminar espacios en blanco iniciales y de cola |
| s.translate(str {,delc}) | Traducir s utilizando la tabla, después eliminar cualquier carácter de delc. str debe ser una cadena con una longitud de == 256. |
| s.replace(old, new {, max}) | Sustituye todas las apariciones de max de la cadena old por la cadena new |

Observaciones

Los comentarios se introducen con el signo de almohadilla (#) o hash. Todo el texto que sigue al signo de almohadilla en la misma línea se considera parte del comentario y se omite. Un comentario puede comenzar en cualquier columna. El ejemplo siguiente muestra el uso de los comentarios:

```
#The HelloWorld application is one of the most simple
print 'Hello World' # print the Hello World line
```

Sintaxis de las sentencias

La sintaxis de las sentencias para Python es muy sencilla. En general, cada línea de origen es una sola sentencia. A excepción de las sentencias `expression` y `assignment`, cada sentencia se introduce mediante un nombre de palabra clave, tal como `if` o `for`. Las líneas en blanco o las líneas de comentarios se pueden insertar en cualquier lugar entre cualquier sentencia del código. Si existe más de una sentencia en una línea, cada sentencia debe estar separada por un signo de punto y coma (;).

Las sentencias muy largas pueden continuar en más de una línea. En este caso, la sentencia que ha de continuar en la línea siguiente debe acabar con una barra invertida (\), por ejemplo:

```
x = "A loooooooooooooooooooooooooong string" + \
    "another loooooooooooooooooooooooooong string"
```

Cuando una estructura está encerrada entre paréntesis (), corchetes [] o llaves {}, la sentencia puede continuar en una línea nueva después de cualquier coma, sin tener que insertar una barra invertida, por ejemplo:

```
x = (1, 2, 3, "hello",
    "goodbye", 4, 5, 6)
```

Identificadores

Los identificadores se utilizan para el nombre de las variables, funciones, clases y palabras clave. Los identificadores pueden tener cualquier longitud, pero debe empezar con un carácter alfabético en mayúsculas o minúsculas o el carácter de subrayado (_). Los nombres que empiezan con un carácter de subrayado están generalmente reservados para los nombres internos o privados. Después del primer carácter, el identificador puede contener cualquier número y combinación de caracteres alfabéticos, los números del 0-9, y el carácter de subrayado.

Existen algunas palabras reservadas en Python que no se pueden utilizar para el nombre de variables, funciones o clases. Estas palabras entran en las siguientes categorías:

- **Introducciones de sentencias:** `assert`, `break`, `class`, `continue`, `def`, `del`, `elif`, `else`, `except`, `exec`, `finally`, `for`, `from`, `global`, `if`, `import`, `pass`, `print`, `raise`, `return`, `try` y `while`
- **Introducciones de parámetros:** `as`, `import` y `in`
- **Operadores:** `and`, `in`, `is`, `lambda`, `not` y `or`

El uso incorrecto de palabras claves suele generar `SyntaxError`.

Bloques de código

Bloques de código son grupos de sentencias que se utilizan donde se esperan sentencias individuales. Los bloques de código pueden seguir a cualquiera de las sentencias siguientes: `if`, `elif`, `else`, `for`, `while`, `try`, `except`, `def` y `class`. Estas sentencias introducen el bloque de código con el carácter de dos puntos (:), por ejemplo:

```
if x == 1:
    y = 2
    z = 3
elif:
    y = 4
    z = 5
```

Se utiliza la indentación para delimitar los bloques de código (en lugar de las llaves que se utilizan en Java). Todas las líneas de un bloque han de indentarse en la misma posición. Esto es debido a que un cambio en la indentación indica el final de un bloque de código. Normalmente la indentación es de cuatro espacios por nivel. Se recomienda utilizar espacios para la indentación, en lugar de tabulaciones. No se deben combinar espacios y tabulaciones. Las líneas del bloque de un módulo situado más al extremo deben comenzar en la columna uno, de lo contrario, se genera el error `SyntaxError`.

Las sentencias que componen un bloque de código (y siguen el signo de dos puntos) también deben estar en una sola línea, separadas por signos de punto y coma, por ejemplo:

```
if x == 1: y = 2; z = 3;
```

Pasar argumentos a un script

Pasar argumentos a un script puede resultar útil para poder utilizar un script reiteradamente sin modificarlo. Los argumentos se pasan en la línea de comandos como valores de la lista `sys.argv`. El número de valores que se pasan se puede obtener mediante el comando `len(sys.argv)`. Por ejemplo:

```
import sys
print "test1"
print sys.argv[0]
print sys.argv[1]
print len(sys.argv)
```

En este ejemplo, el comando `import` importa toda la clase `sys`, por lo que se pueden utilizar los métodos existentes para esta clase, tales como `argv`.

El script de este ejemplo se puede invocar utilizando la línea siguiente:

```
/u/mjloos/test1 mike don
```

Esto genera el resultado siguiente:

```
/u/mjloos/test1 mike don
test1
mike
don
3
```

Ejemplos

La palabra clave `print` imprime los argumentos situados inmediatamente después de la misma. Si la sentencia va seguida de una coma, no se incluye una línea nueva en los resultados. Por ejemplo:

```
print "Esto muestra el uso de una",
print " coma al final de una sentencia de impresión."
```

Esto genera el resultado siguiente:

```
Esto muestra el uso de una coma al final de una sentencia de impresión.
```

La sentencia `for` se utiliza para la iteración por un bloque de código. Por ejemplo:

```
mylist1 = ["one", "two", "three"]
for lv in mylist1:
    print lv
    continue
```

En este ejemplo, se asignan tres cadenas a la lista `mylist1`. Los elementos de la lista se imprimen a continuación, con un elemento de cada línea. Esto genera el resultado siguiente:

```
uno
two
tres
```

En este ejemplo, el iterador `lv` toma el valor de cada elemento de la lista `mylist1` por orden, mientras el bucle `for` implementa el bloque de código de cada elemento. Un iterador puede ser cualquier identificador válido de cualquier longitud.

La sentencia `if` es una sentencia condicional. Evalúa la condición y devuelve `true` o `false`, en función del resultado de la evaluación. Por ejemplo:

```

mylist1 = ["one", "two", "three"]
for lv in mylist1:
    if lv == "two"
        print "The value of lv is ", lv
    else
        print "The value of lv is not two, but ", lv
        continue

```

En este ejemplo, se evalúa el valor del iterador lv. Si el valor de lv es two se devuelve una cadena diferente a la cadena que se devuelve si el valor de lv no es two. Esto resulta en la siguiente salida:

```

The value of lv is not two, but one
The value of lv is two
The value of lv is not two, but three

```

Métodos matemáticos

Desde el módulo matemáticas puede acceder a métodos matemáticos útiles. Algunos de estos métodos se proporcionan en la tabla siguiente. A menos que se especifique lo contrario, todos los valores se devuelven como valores flotantes.

Tabla 7. Métodos matemáticos

| Método | Uso |
|---------------------|---|
| math.ceil(x) | Devuelve el punto más alto de x como un valor flotante, que es el entero más pequeño mayor o igual a x |
| math.copysign(x, y) | Devuelve x con el signo de y. copysign(1, -0.0) devuelve -1 |
| math.fabs(x) | Devuelve el valor absoluto de x |
| math.factorial(x) | Devuelve el factor de x. Si x es negativo o no es un entero, se genera ValueError. |
| math.floor(x) | Devuelve el punto más bajo de x como un valor flotante, que es el entero más alto menor o igual a x |
| math.frexp(x) | Devuelve la mantisa (m) y el exponente (e) de x como el par (m, e). m es un valor flotante y e es un entero, tal como $x == m * 2^{**}e$ exactamente. Si x es cero, devuelve (0,0, 0), de lo contrario $0,5 \leq \text{abs}(m) < 1$. |
| math.fsum(iterable) | Devuelve una suma de coma flotante precisa de los valores de iterable |
| math.isinf(x) | Comprueba si el valor flotante x es positivo o negativo infinito |
| math.isnan(x) | Comprueba si el valor flotante x es NaN (no es un número) |
| math.ldexp(x, i) | Devuelve $x * (2^{**}i)$. Esencialmente es la función inversa de frexp. |
| math.modf(x) | Devuelve las partes de fracción y entero de x. Los dos resultados llevan el signo de x y son flotantes. |
| math.trunc(x) | Devuelve el valor Real de x, que se ha truncado en un Integral. |
| math.exp(x) | Devuelve $e^{**}x$ |
| math.log(x[, base]) | Devuelve el logaritmo de x para el valor dado de base. Si no se especifica base, se devuelve el logaritmo natural de x. |
| math.log1p(x) | Devuelve el logaritmo natural de 1+x (base e) |
| math.log10(x) | Devuelve el logaritmo de base-10 de x |

Tabla 7. Métodos matemáticos (continuación)

| Método | Uso |
|-----------------------------|---|
| <code>math.pow(x, y)</code> | Devuelve x elevado a la potencia de y . <code>pow(1.0, x)</code> y <code>pow(x, 0.0)</code> siempre devuelve 1, incluso si x es cero o NaN. |
| <code>math.sqrt(x)</code> | Devuelve la raíz cuadrada de x |

Además de las funciones matemáticas, hay algunos métodos trigonométricos útiles. Estos métodos se muestran en la siguiente tabla.

Tabla 8. Métodos trigonométricos

| Método | Uso |
|-------------------------------|---|
| <code>math.acos(x)</code> | Devuelve el arco coseno de x en radianes |
| <code>math.asin(x)</code> | Devuelve el arcoseno de x en radianes |
| <code>math.atan(x)</code> | Devuelve el arco tangente de x en radianes |
| <code>math.atan2(y, x)</code> | Devuelve <code>atan(y / x)</code> en radianes. |
| <code>math.cos(x)</code> | Devuelve el coseno de x en radianes. |
| <code>math.hypot(x, y)</code> | Devuelve la norma euclidiana de <code>sqrt(x*x + y*y)</code> . Esta es la longitud del vector desde el origen al punto (x, y) . |
| <code>math.sin(x)</code> | Devuelve el seno de x en radianes |
| <code>math.tan(x)</code> | Devuelve la tangente de x en radianes |
| <code>math.degrees(x)</code> | Convierte el ángulo x de radianes a grados |
| <code>math.radians(x)</code> | Convierte el ángulo x de grados a radianes |
| <code>math.acosh(x)</code> | Devuelve el coseno hiperbólico inverso de x |
| <code>math.asinh(x)</code> | Devuelve el seno hiperbólico inverso de x |
| <code>math.atanh(x)</code> | Devuelve la tangente hiperbólica inversa de x |
| <code>math.cosh(x)</code> | Devuelve el coseno hiperbólico de x |
| <code>math.sinh(x)</code> | Devuelve el seno hiperbólico de x |
| <code>math.tanh(x)</code> | Devuelve la tangente hiperbólica de x |

También hay constantes matemáticas. El valor de `math.pi` es la constante matemática pi. El valor de `math.e` es la constante matemática e.

Utilización de caracteres no ASCII

Para utilizar caracteres no ASCII, Python requiere la codificación y decodificación explícitas de las cadenas en Unicode. En IBM SPSS Modeler, se presupone que los scripts Python están codificados UTF-8, la cual es una codificación Unicode estándar que da soporte a caracteres no ASCII. El script siguiente se compilará porque SPSS Modeler ha establecido el compilador Python en UTF-8.

```
stream = modeler.script.stream()
filenode = stream.createAt("variablefile", "テストノード", 96, 64)
```

Sin embargo, el nodo resultante tendrá una etiqueta incorrecta.



ãfã, 'ãf^ãf ãf%ãf%

Figura 3. Etiqueta del nodo que contiene caracteres no ASCII, visualiza incorrectamente

La etiqueta es incorrecta porque Python ha convertido el propio literal de serie en una cadena ASCII.

Python permite que los literales de cadenas Unicode se especifiquen añadiendo un prefijo con el carácter u antes del literal de cadena:

```
stream = modeler.script.stream()
filenode = stream.createAt("variablefile", u"テストノード", 96, 64)
```

Esto crear una cadena Unicode y la etiqueta aparecerá correctamente.



テストノード

Figura 4. Etiqueta de nodo que contiene caracteres no ASCII, visualizados correctamente

La utilización de Python y Unicode es un tema de gran volumen que no entra dentro del ámbito de este documento. Existen muchas publicaciones y recursos en línea disponibles que describen detalladamente este tema.

Programación orientada a objetos

La programación orientada a objetos se basa en el concepto de crear un modelo del problema de destino en sus programas. La programación orientada a objetos disminuye los errores y promociona la reutilización del código. Python es un lenguaje orientado a objetos. Los objetos definidos en Python tienen las características siguientes:

- **Identidad.** Cada objeto debe ser distinguido y ello debe poder demostrarse mediante pruebas. Las pruebas `is` e `is not` existen para este fin.
- **Estado** Cada objeto debe ser capaz de almacenar el estado. Para este fin, existen atributos, tales como variables de instancias y campos.
- **Comportamiento.** Cada objeto debe ser capaz de manipular su estado. Para este fin existen métodos.

Python incluye las características siguientes para dar soporte a la programación orientada a objetos:

- **Creación de objetos basada en clases.** Las clases son plantillas para la creación de objetos. Los objetos son estructuras de datos con el comportamiento asociado.
- **Herencia con polimorfismo.** Python da soporte a la herencia individual y múltiple. Todos los métodos de instancias de Python son polimórficos y se pueden alterar temporalmente mediante subclases.
- **Encapsulación con ocultación de datos.** Python permite ocultar los atributos. Cuando se ocultan los atributos, se puede acceder a los mismos desde fuera de la clase únicamente mediante los métodos de la clase. Las clases implementan métodos para modificar los datos.

Definición de una clase

En una clase Python, se pueden definir tanto variables como métodos. A diferencia de Java, en Python puede definir cualquier número de clases públicas por archivo de origen (o *module*). Por lo tanto, un módulo en Python puede considerarse similar a un paquete en Java.

En Python, las clases se definen utilizando la sentencia `class`. La sentencia `class` tiene el formato siguiente:

```
class name (superclasses): statement
```

o

```
class name (superclasses):  
    assignment  
    .  
    .  
    function  
    .  
    .
```

Cuando define una clase tiene la opción de proporcionar cero o más sentencias *assignment*. Estos crean atributos de clase que comparten todas las instancias de la clase. Puede proporcionar cero o más definiciones de *function*. Estas definiciones de función crean métodos. La lista de superclases es opcional.

El nombre de clase debe ser exclusivo en el mismo ámbito, esto es, dentro de un módulo, función o clase. Puede definir varias variables para que hagan referencia a la misma clase.

Creación de una instancia de clase

Las clases se utilizan para contener (o compartir) los atributos de clase o para crear instancias de clase. Para crear una instancia de una clase, debe llamar a la clase como si fuera una función. Por ejemplo, considere las clases siguientes:

```
class MyClass:  
    pass
```

Aquí, se utiliza la sentencia `pass` por que se requiere una sentencia para completar la clase, pero no se requiere ninguna acción de programación.

La sentencia siguiente crea una instancia de la clase `MyClass`:

```
x = MyClass()
```

Añadir atributos a una instancia de clase

A diferencia de Java, en Python los clientes pueden añadir atributos a una instancia de una clase. Solo se cambia la instancia. Por ejemplo, para añadir atributos a una instancia de `x`, establezca valores nuevos en dicha instancia:

```
x.attr1 = 1  
x.attr2 = 2  
:  
:  
x.attrN = n
```

Definición de atributos de clase y métodos

Cualquier variable enlazada a una clase es un *atributo de clase*. Cualquier función definida en una clase es un *método*. Los métodos reciben como primer argumento una instancia de la clase, que convencionalmente se denomina `self`. Por ejemplo, para definir algunos atributos de clase y métodos, puede entrar el siguiente código:

```

class MyClass
    attr1 = 10          #class attributes
    attr2 = "hello"

    def method1(self):
        print MyClass.attr1  #reference the class attribute

    def method2(self):
        print MyClass.attr2  #reference the class attribute

    def method3(self, text):
        self.text = text      #instance attribute
        print text, self.text  #print my argument and my attribute

    method4 = method3  #make an alias for method3

```

Dentro de una clase, debe cualificar todas las referencias a los atributos de clase con el nombre de clase; por ejemplo, `MyClass.attr1`. Todas las referencias a los atributos de la instancia deben cualificarse con la variable `self`, por ejemplo, `self.text`. Fuera de la clase, debe cualificar todas las referencias a los atributos de clase con el nombre de clase (por ejemplo, `MyClass.attr1`) o con una instancia de la clase (por ejemplo, `x.attr1`, donde `x` es una instancia de la clase). Fuera de la clase, todas las referencias a las variables de la instancia deben cualificarse con una instancia de la clase, por ejemplo, `x.text`.

Variables ocultas

Los datos se pueden ocultar creando variables *privadas*. Solo la propia clase puede acceder a las variables privadas. Si declara nombres con el formato `__xxx` o `__xxx_yyy`, estos es, con dos signos de subrayado antes de los nombres, el analizador Python automáticamente añadirá el nombre de clase al nombre declarado y creará las variables ocultas, por ejemplo:

```

class MyClass:
    __attr = 10  #private class attribute

    def method1(self):
        pass

    def method2(self, p1, p2):
        pass

    def __privateMethod(self, text):
        self.__text = text  #private attribute

```

A diferencia de Java, en Python todas las referencias a variables de instancia deben estar calificadas con `self`; no existe un uso implícito de `this`.

Heredado

La posibilidad de herencia de las clases es fundamental en la programación orientada a objetos. Python da soporte a la herencia individual y múltiple. *Herencia individual* significa que solo puede haber una superclase. *Herencia múltiple* significa que puede haber más de una superclase.

La herencia se implementa generando subclases de otras clases. Cualquier número de clases Python pueden ser superclases. En la implementación de Jython en Python, solo se puede heredar directa o indirectamente de una clase Java. No es necesario suministrar una superclase.

Cualquier atributo o método de una superclase también está en cualquier subclase y lo puede utilizar la propia clase o cualquier cliente, siempre que el atributo o método no esté oculto. Se puede utilizar cualquier instancia de una subclase; esto se denomina *polimorfismo*. Estas características permiten la reutilización y facilitan la extensión.

Ejemplo

```
class Class1: pass    #no inheritance
class Class2: pass
class Class3(Class1): pass    #single inheritance
class Class4(Class3, Class2): pass    #multiple inheritance
```

Capítulo 3. Scripts de IBM SPSS Modeler

Tipos de scripts

En IBM SPSS Modeler existen tres tipos de scripts:

- Los *scripts de ruta* se utilizan para controlar la ejecución de una sola ruta y se almacenan dentro de la ruta.
- Los *scripts Supernodo* se utilizan para controlar el comportamiento de los supernodos.
- Los *scripts autónomos o de sesión* se pueden utilizar para coordinar la ejecución entre un número de rutas diferentes.

Existen diferentes métodos disponibles que puede utilizar en scripts en IBM SPSS Modeler lo que le permite acceder a una amplia gama de funciones de SPSS Modeler. Estos métodos se utilizan también en Capítulo 4, “API de scripts”, en la página 37 para crear funciones más avanzadas.

Rutas, rutas de supernodo y diagramas

La mayoría de las veces, el término *ruta* significa lo mismo independientemente de que se trate de una ruta cargada de un archivo o utilizada dentro de un supernodo. En general significa una colección de nodos conectados entre sí que puede ejecutarse. Sin embargo, en la creación de scripts no todas las operaciones se soportan en todos los sitios, lo que significa que el autor de un script deberá tener en cuenta qué variante de ruta está utilizando.

Rutas

Una ruta es el principal tipo de documento de IBM SPSS Modeler. Se puede guardar, cargar, editar y ejecutar. Las rutas también pueden tener parámetros, valores globales, un script y otra información asociada a ellos.

Rutas de Supernodo

Una *ruta de Supernodo* es el tipo de ruta que se utiliza en un Supernodo. Al igual que una ruta normal, contiene nodos enlazados entre sí. Las rutas de Supernodo tienen una serie de diferencias respecto de una ruta normal.

- Los parámetros y scripts están asociados al Supernodo propietario de la ruta de Supernodo en lugar de a la propia ruta de Supernodo.
- Las rutas de Supernodo tienen nodos de conector de entrada y salida adicionales dependiendo del tipo de Supernodo. Estos nodos de conector se utilizan en los flujos de información entrantes y salientes de la ruta de Supernodo y se crean automáticamente cuando se crea el Supernodo.

Diagramas

El término *diagrama* abarca las funciones soportadas en rutas normales y en rutas de supernodo como, por ejemplo, la adición y eliminación de nodos y la modificación de conexiones entre nodos.

Ejecución de una ruta

El ejemplo siguiente ejecuta todos los nodos ejecutables en la ruta y es el tipo de script de ruta más sencillo:

```
modeler.script.stream().runAll(None)
```

El ejemplo siguiente también se ejecuta todos los nodos ejecutables de la ruta:

```
stream = modeler.script.stream()
stream.runAll(None)
```

En este ejemplo, la ruta se almacena en una variable denominada `stream`. Almacenar la ruta en una variable resulta útil ya que un script se utiliza generalmente para modificar la ruta o los nodos contenidos en una ruta. Si se crea una variable que almacena los resultados de la ruta, el script resultará más conciso.

El contexto de los scripts

El módulo `modeler.script` proporciona el contexto en el que se ejecuta un script. El módulo se importa automáticamente a un script de SPSS Modeler durante la ejecución. El módulo define cuatro funciones que proporcionan un script con acceso a su entorno de ejecución:

- La función `session()` devuelve la sesión para el script. La sesión define información, tal como el entorno local y el proceso de fondo de SPSS Modeler (ya sea un proceso local o un proceso de SPSS Modeler Server conectado a la red) que se está utilizando para ejecutar rutas.
- La función `stream()` se puede utilizar con la ruta y los scripts Supernodo. Esta función devuelve la ruta que es propietaria del script de ruta o el script Supernodo que se está ejecutando.
- La función `diagram()` se puede utilizar con los scripts Supernodo. Esta función devuelve el diagrama dentro del Supernodo. Para otros tipos de script, esta función devuelve el mismo que la función `stream()`.
- La función `supernode()` se puede utilizar con los scripts Supernodo. Esta función devuelve el Supernodo propietario del script que se está ejecutando.

En la tabla siguiente se resumen las cuatro funciones y sus resultados.

Tabla 9. Resumen de las funciones de `modeler.script`

| Tipo de script | <code>session()</code> | <code>stream()</code> | <code>diagram()</code> | <code>supernode()</code> |
|----------------|------------------------|---|--------------------------------------|--------------------------|
| Autónomo | Devuelve una sesión | Devuelve la ruta gestionada actual en el momento en que se invoca el script (por ejemplo, la ruta se pasa con la opción <code>-stream</code> de modalidad de proceso por lotes) o <code>None</code> . | Igual que para <code>stream()</code> | No es aplicable |
| Ruta | Devuelve una sesión | Devuelve una ruta | Igual que para <code>stream()</code> | No es aplicable |
| Supernodo | Devuelve una sesión | Devuelve una ruta | Devuelve una ruta Supernodo | Devuelve un Supernodo |

El módulo `modeler.script` también define un modo de finalizar el script con un código de salida. La función `exit(exit-code)` detiene la ejecución del script y devuelve el código de salida de entero suministrado.

Uno de los métodos que se define para una ruta es `runAll(List)`. Este método ejecuta todos los nodos ejecutables. Los modelos o resultados que se generan mediante la ejecución de los nodos se añaden a la lista suministrada.

Es común que la ejecución de ruta genere resultados, tales como modelos, gráficos y otros. Para capturar este resultado, un script puede proporcionar una variable que se inicializa en una lista, por ejemplo:

```
stream = modeler.script.stream()
results = []
stream.runAll(results)
```

Cuando se completa la ejecución, se puede acceder a todos los objetos generados por la ejecución en la lista `results`.

Referencia a nodos existentes

Una ruta suele estar construida previamente con algunos parámetros que se deben modificar antes de ejecutar la ruta. Para modificar estos parámetros se han de realizar las tareas siguientes:

1. Localizar los nodos en la ruta relevante.
2. Cambiar los valores de los nodos o de la ruta (o de ambas cosas).

Buscar nodos

Las rutas proporcionan varios modos de localizar un nodo existente. Estos métodos se resumen en la siguiente tabla.

Tabla 10. Métodos para localizar un nodo existente

| Método | Tipo devuelto | Descripción |
|---|---------------|---|
| <code>s.findAll(type, label)</code> | Colección | Devuelve una lista de todos los nodos con el tipo y la etiqueta. El tipo o la etiqueta pueden ser <code>None</code> , en cuyo caso se utiliza el otro parámetro. |
| <code>s.findAll(filter, recursive)</code> | Colección | Devuelve una colección de todos los nodos que están aceptados por el filtro especificado. Si el distintivo recursivo es <code>True</code> , también se buscan los supernodos contenidos en la ruta especificada. |
| <code>s.findById(id)</code> | Nodo | Devuelve el nodo con el ID proporcionado o <code>None</code> si no existe dicho nodo. La búsqueda se limita a la ruta actual. |
| <code>s.findByName(type, label)</code> | Nodo | Devuelve el nodo con el tipo, etiqueta o ambas cosas. El tipo o el nombre pueden ser <code>None</code> , en cuyo caso se utiliza el otro parámetro. Si varios nodos dan como resultado una coincidencia, se elige uno arbitrario y se devuelve. Si ningún nodo da como resultado una coincidencia, se devuelve el valor <code>None</code> . |
| <code>s.findDownstream(fromNodes)</code> | Colección | Busca en la lista de nodos suministrada y devuelve el conjunto de nodos en sentido descendente de los nodos suministrados. La lista devuelta incluye los nodos proporcionados originalmente. |

Tabla 10. Métodos para localizar un nodo existente (continuación)

| Método | Tipo devuelto | Descripción |
|---------------------------|---------------|---|
| s.findUpstream(fromNodes) | Colección | Busca en la lista de nodos suministrada y devuelve el conjunto de nodos en sentido ascendente de los nodos suministrados. La lista devuelta incluye los nodos proporcionados originalmente. |

Por ejemplo, si una ruta contiene un nodo Filtro único que el script necesita para acceso, el nodo Filtro se puede encontrar utilizando el siguiente script:

```
stream = modeler.script.stream()
node = stream.findByType("filter", None)
...
```

Como alternativa, si se conoce el ID del nodo (tal como se muestra en la pestaña Anotaciones del cuadro de diálogo del nodo) se puede utilizar el ID para buscar el nodo, por ejemplo:

```
stream = modeler.script.stream()
node = stream.findById("id32FJT71G2") # the filter node ID
...
```

Establecimiento de propiedades

Los nodos, rutas, modelos y resultados tienen propiedades a las que se puede acceder y que, en la mayor parte de los casos, se pueden establecer. Las propiedades suelen utilizarse para modificar el aspecto o el comportamiento del objeto. En la tabla siguiente se resumen los métodos disponibles para establecer y acceder a las propiedades de los objetos.

Tabla 11. Métodos para establecer y acceder a las propiedades de los objetos

| Método | Tipo devuelto | Descripción |
|---|-----------------|---|
| p.getPropertyValue(propertyName) | Object | Devuelve el valor de la propiedad con nombre, o None si no existe tal propiedad. |
| p.setPropertyValue(propertyName, value) | No es aplicable | Establece el valor de la propiedad con nombre. |
| p.setPropertyValues(properties) | No es aplicable | Establece los valores de la propiedad con nombre. Cada entrada de la correlación de propiedades consta de una clave que representa el nombre de la propiedad y del valor que debe asignarse a la propiedad. |
| p.getKeyedPropertyValue(propertyName, keyName) | Object | Devuelve el valor de la propiedad con nombre, o None si no existe dicha propiedad o clave. |
| p.setKeyedPropertyValue(propertyName, keyName, value) | No es aplicable | Establece el valor de la propiedad con nombre y de la clave. |

Por ejemplo, si desea establecer el valor de un nodo Archivo variable al comienzo de una ruta, puede utilizar el siguiente script:

```
stream = modeler.script.stream()
node = stream.findByType("variablefile", None)
node.setPropertyValue("full_filename", "$CLEO/DEMOS/DRUG1n")
...
```

Como alternativa, puede que desee filtrar un campo desde un nodo Filtrar. En este caso, el valor es con clave en el nombre de campo, por ejemplo:

```
stream = modeler.script.stream()
# Locate the filter node ...
node = stream.findByType("filter", None)
# ... and filter out the "Na" field
node.setKeyedPropertyValue("include", "Na", False)
```

Creación de nodos y modificación de rutas

En algunas situaciones, es posible que desee añadir nuevos nodos a rutas existentes. Para añadir nodos a rutas existentes suele ser necesario realizar las tareas siguientes:

1. Crear los nodos.
2. Enlazar los nodos con el flujo de ruta existente.

Creación de nodos

Las rutas proporcionan varios modos de crear nodos. Estos métodos se resumen en la siguiente tabla.

Tabla 12. Métodos para crear nodos

| Método | Tipo devuelto | Descripción |
|--|---------------|---|
| <code>s.create(nodeType, name)</code> | Nodo | Crea un nodo del tipo especificado y lo añade a la ruta especificada. |
| <code>s.createAt(nodeType, name, x, y)</code> | Nodo | Crea un nodo del tipo especificado y lo añade a la ruta especificada en la ubicación especificada. Si $x < 0$ o $y < 0$, no se establece la ubicación. |
| <code>s.createModelApplier(modelOutput, name)</code> | Nodo | Crea un nodo aplicador de modelos que se deriva del objeto de resultados del modelo proporcionado. |

Por ejemplo, para crear un tipo de nodo nuevo en una ruta puede utilizar el siguiente script:

```
stream = modeler.script.stream()
# Create a new type node
node = stream.create("type", "My Type")
```

Enlazar y desenlazar nodos

Cuando un nodo nuevo se crea dentro de una ruta, debe estar conectado a una ruta de nodos para poder utilizarlo. Las rutas proporciona varios métodos para enlazar y desenlazar nodos. Estos métodos se resumen en la siguiente tabla.

Tabla 13. Métodos para enlazar y desenlazar nodos

| Método | Tipo devuelto | Descripción |
|--------------------------------------|-----------------|--|
| <code>s.link(source, target)</code> | No es aplicable | Crea un nuevo enlace entre los nodos de origen y destino. |
| <code>s.link(source, targets)</code> | No es aplicable | Crea nuevos enlaces entre el nodo de origen y cada nodo de destino de la lista suministrada. |

Tabla 13. Métodos para enlazar y desenlazar nodos (continuación)

| Método | Tipo devuelto | Descripción |
|---|-----------------|---|
| s.linkBetween(inserted, source, target) | No es aplicable | Conecta un nodo entre dos instancias de otro nodo (los nodos de origen y de destino) y establece la posición del nodo insertado de modo que quede entre ellos. Cualquier enlace directo entre los nodos de origen y de destino se elimina en primer lugar. |
| s.linkPath(path) | No es aplicable | Crea una nueva ruta entre instancias de nodo. El primer nodo se enlaza con el segundo, el segundo nodo se enlaza con el tercero y así sucesivamente. |
| s.unlink(source, target) | No es aplicable | Elimina cualquier enlace directo entre los nodos de origen y de destino. |
| s.unlink(source, targets) | No es aplicable | Elimina los enlaces directos entre el nodo de origen y cada objeto de la lista de destinos. |
| s.unlinkPath(path) | No es aplicable | Elimina cualquier ruta que existe entre las instancias del nodo. |
| s.disconnect(node) | No es aplicable | Elimina los enlaces entre el nodo suministrado y todos los demás nodos de la ruta especificada. |
| s.isValidLink(source, target) | booleano | Devuelve True si es válido crear un enlace entre los nodos de origen y de destino especificados. Este método comprueba que ambos objetos pertenezcan a la ruta especificada, que el nodo de origen puede proporcionar un enlace y que el nodo de destino puede recibir un enlace, y que la creación de un enlace de este tipo no creará un circularidad en la ruta. |

El script de ejemplo siguiente realiza estas cinco tareas:

1. Crea un nodo de entrada Archivo de variables, un nodo Filtro y un nodo de salida Tabla.
2. Conecta los nodos entre sí.
3. Establece el nombre de archivo del nodo de entrada Archivo de variables.
4. Filtra el campo "Drug" de la salida resultante.
5. Ejecute el nodo Tabla.

```
stream = modeler.script.stream()
filenode = stream.createAt("variablefile", "My File Input ", 96, 64)
filternode = stream.createAt("filter", "Filter", 192, 64)
tablenode = stream.createAt("table", "Table", 288, 64)
stream.link(filenode, filternode)
stream.link(filternode, tablenode)
filenode.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
filternode.setKeyedPropertyValue("include", "Drug", False)
results = []
tablenode.run(results)
```

Importar, sustituir y eliminar nodos

Además de crear y conectar nodos, a menudo es necesario sustituir y suprimir nodos de la ruta. Los métodos que están disponibles para importar, sustituir y suprimir nodos se resumen en la tabla siguiente.

Tabla 14. Métodos para importar, sustituir y suprimir nodos

| Método | Tipo devuelto | Descripción |
|--|-----------------|--|
| <code>s.replace(originalNode, replacementNode, discardOriginal)</code> | No es aplicable | Sustituye el nodo especificado de la ruta actual. Tanto el nodo original como el nodo de sustitución deben ser propiedad de la ruta especificada. |
| <code>s.insert(source, nodes, newIDs)</code> | Lista | Inserta copias de los nodos en la lista suministrada. Se presupone que todos los nodos de la lista suministrada se encuentran dentro de la ruta especificada. El distintivo <code>newIDs</code> indica si se deben generar nuevos ID para cada nodo o si se debe copiar y utilizar el ID existente. Se presupone que todos los nodos de una ruta tienen ID exclusivos, por lo tanto este distintivo se debe establecer en <code>True</code> si la ruta de origen es la misma que la ruta especificada. El método devuelve la lista de nodos recién insertados, en la que el orden de los nodos está sin definir (es decir, el orden no es necesariamente el mismo que el orden de los nodos de la lista de entrada). |
| <code>s.delete(node)</code> | No es aplicable | Elimina el nodo especificado de la ruta especificada. El nodo debe ser propiedad de la ruta especificada. |
| <code>s.deleteAll(nodes)</code> | No es aplicable | Elimina todos los nodos especificados de la ruta especificada. Todos los nodos de la colección debe pertenecer a la ruta especificada. |
| <code>s.clear()</code> | No es aplicable | Elimina todos los nodos de la ruta especificada. |

Atravesar los nodos de una ruta

Un requisito común es identificar los nodos que están en un punto de la ruta anterior o posterior a un determinado nodo. La ruta proporciona una serie de métodos que pueden utilizarse para identificar estos nodos. Estos métodos se resumen en la siguiente tabla.

Tabla 15. Métodos para identificar los nodos en sentido ascendente y descendente

| Método | Tipo devuelto | Descripción |
|---------------------------|---------------|---|
| <code>s.iterator()</code> | Iterator | Devuelve un iterador de los objetos de nodo que están contenidos en la ruta especificada. Si la ruta se modifica entre llamadas a la función <code>next()</code> , el comportamiento del iterador no está definido. |

Tabla 15. Métodos para identificar los nodos en sentido ascendente y descendente (continuación)

| Método | Tipo devuelto | Descripción |
|------------------------------|---------------|--|
| s.predecessorAt(node, index) | Nodo | Devuelve el predecesor inmediato especificado del nodo suministrado o None si el índice está fuera de los límites. |
| s.predecessorCount(node) | int | Devuelve el número de predecesores inmediatos del nodo suministrado. |
| s.predecessors(node) | Lista | Devuelve los predecesores inmediatos del nodo suministrado. |
| s.successorAt(node, index) | Nodo | Devuelve el sucesor inmediato especificado del nodo suministrado o None si el índice está fuera de los límites. |
| s.successorCount(node) | int | Devuelve el número de sucesores inmediatos del nodo suministrado. |
| s.successors(node) | Lista | Devuelve los sucesores inmediatos del nodo suministrado. |

Borrado o eliminación de elementos

Los scripts de herencia admiten varios usos del comando `clear`, por ejemplo:

- `clear outputs` Para suprimir todos los elementos de salida de la paleta del gestor.
- `clear generated palette` Para borrar todos los nugget de modelos de la paleta de modelos.
- `clear stream` Para eliminar el contenido de una ruta.

Los scripts Python admiten un conjunto de funciones parecido; el comando `removeAll()` se utiliza para borrar las rutas, las salidas y los gestores de modelos. Por ejemplo:

- Para borrar el gestor de rutas:


```
session = modeler.script.session()
session.getStreamManager.removeAll()
```
- Para borrar el gestor de salidas:


```
session = modeler.script.session()
session.getDocumentOutputManager().removeAll()
```
- Para borrar el gestor de modelos:


```
session = modeler.script.session()
session.getModelOutputManager().removeAll()
```

Obtener información sobre los nodos

Los nodos entran en diferentes categorías, tales como nodos de importación y exportación de datos, nodos de construcción de modelos y otros tipos de nodos. Cada nodo proporciona una serie de métodos que pueden utilizarse para obtener información sobre el nodo.

Los métodos que se pueden utilizar para obtener el ID, el nombre y la etiqueta de un nodo se resumen en la tabla siguiente.

Tabla 16. Métodos para obtener el ID, el nombre y la etiqueta de un nodo

| Método | Tipo devuelto | Descripción |
|-------------------|-----------------|--|
| n.getLabel() | cadena | Devuelve la etiqueta de visualización del nodo especificado. La etiqueta es el valor de la propiedad custom_name sólo si la propiedad es una serie no vacía y la propiedad use_custom_name no está establecida; de lo contrario, la etiqueta es el valor de getName(). |
| m.setLabel(label) | No es aplicable | Establece la etiqueta de visualización del nodo especificado. Si la nueva etiqueta es una cadena no vacía se asigna a la propiedad custom_name, y False se asigna a la propiedad use_custom_name, de tal modo que la etiqueta especificada tiene prioridad; de lo contrario, se asigna una serie vacía a la propiedad custom_name y se asigna True a la propiedad use_custom_name. |
| n.getName() | cadena | Devuelve el nombre del nodo especificado. |
| n.getID() | cadena | Devuelve el ID del nodo especificado. Se crea un ID nuevo cada vez que se crea un nuevo nodo. El ID se conserva con el nodo cuando se guarda como parte de una ruta de modo que, cuando se abre la ruta, los ID de nodo se conservan. Sin embargo, si un nodo guardado se inserta en una ruta, el nodo insertado se considera un nuevo objeto y se le asigna un nuevo ID. |

Los métodos que se pueden utilizar para obtener información acerca de un nodo se resumen en la tabla siguiente.

Tabla 17. Métodos para obtener información acerca de un nodo

| Método | Tipo devuelto | Descripción |
|------------------|---------------|--|
| n.getTypeName() | cadena | Devuelve el nombre de script de este nodo. Este es el mismo nombre que puede utilizarse para crear una nueva instancia de este nodo. |
| n.isInitial() | Booleana | Devuelve True si es un nodo <i>initial</i> , esto es, un nodo que aparece al inicio de una ruta. |
| n.isInline() | Booleana | Devuelve True si es un nodo <i>in-line</i> , esto es, un nodo que aparece a mitad de una ruta. |
| n.isTerminal() | Booleana | Devuelve True si es un nodo <i>terminal</i> , esto es, un nodo que aparece al final de una ruta. |
| n.getXPosition() | int | Devuelve el desplazamiento de la posición x del nodo en la ruta. |

Tabla 17. Métodos para obtener información acerca de un nodo (continuación)

| Método | Tipo devuelto | Descripción |
|--------------------------------------|-----------------|--|
| n.getYPosition() | <i>int</i> | Devuelve el desplazamiento de la posición y del nodo en la ruta. |
| n.setXYPosition(x, y) | No es aplicable | Devuelve la posición del nodo en la ruta. |
| n.setPositionBetween(source, target) | No es aplicable | Establece la posición del nodo en la ruta, de modo que esté posicionado entre los nodos suministrados. |
| n.isCacheEnabled() | <i>Booleana</i> | Devuelve True si la memoria caché está habilitada; devuelve False de lo contrario. |
| n.setCacheEnabled(val) | No es aplicable | Habilita o inhabilita la memoria caché para este objeto. Si la memoria caché está llena y la memoria caché pasa a estar inhabilitada, la memoria caché se vacía. |
| n.isCacheFull() | <i>Booleana</i> | Devuelve True si la memoria caché está llena; devuelve False de lo contrario. |
| n.flushCache() | No es aplicable | Vacía la memoria caché de este nodo. No tiene efecto si la memoria caché no está activada o no está llena. |

Capítulo 4. API de scripts

Introducción a la API de scripts

La API de scripts proporciona acceso a una amplia gama de funciones de SPSS Modeler. Todos los métodos descritos hasta ahora forman parte de la API y se puede acceder a los mismos de forma implícita en el script sin importaciones adicionales. Sin embargo, si desea hacer referencia a las clases de la API, debe importar la API explícitamente con la sentencia siguiente:

```
import modeler.api
```

Esta sentencia import es necesaria para muchos de los ejemplos de la API de scripts.

Se puede encontrar una guía completa de las clases, los métodos y parámetros que están disponibles a través de la API de scripts en el documento *Guía de referencia de la API de scripts Python de IBM SPSS Modeler*.

Ejemplo 1: buscar nodos utilizando un filtro personalizado

La sección “Buscar nodos” en la página 29 se incluye un ejemplo de cómo buscar un nodo en una ruta utilizando el nombre de tipo del nodo como criterio de búsqueda. En algunas situaciones, se requiere una búsqueda más genérica y ésta se puede implementar utilizando la clase `NodeFilter` y el método `findAll()` de la ruta. Este tipo de búsqueda requiere los pasos siguientes:

1. Crear una clase nueva que amplíe `NodeFilter` e implemente una versión personalizada del método `accept()`.
2. Llamar al método `findAll()` de la ruta con una instancia de esta clase nueva. Esto devuelve todos los nodos que cumplen el criterio definido en el método `accept()`.

El ejemplo siguiente muestra cómo buscar nodos de una ruta que tienen habilitada la memoria caché de nodo. La lista de nodos devuelta se puede utilizar para vaciar o inhabilitar las memorias caché de estos nodos.

```
import modeler.api

class CacheFilter(modeler.api.NodeFilter):
    """A node filter for nodes with caching enabled"""
    def accept(this, node):
        return node.isCacheEnabled()

cachingnodes = modeler.script.stream().findAll(CacheFilter(), False)
```

Ejemplo 2: permitir a los usuarios obtener información de directorio o archivo basándose en sus privilegios

Para evitar que la PSAPI se abra a los usuarios, se puede utilizar un método denominado `session.getServerFileSystem()` mediante la llamada de la función PSAPI para crear un objeto de sistema de archivos.

El ejemplo siguiente muestra cómo permitir que un usuario obtenga información de directorio o archivo basándose en los privilegios del usuario que se conecta a IBM SPSS Modeler Server.

```
import modeler.api
stream = modeler.script.stream()
sourceNode = stream.findByID('')
session = modeler.script.session()
fileSystem = session.getServerFileSystem()
```

```

parameter = stream.getParameterValue('VPATH')
serverDirectory = fileSystem.getServerFile(parameter)
files = fileSystem.GetFiles(serverDirectory)
for f in files:
    if f.isDirectory():
        print 'Directory:'
    else:
        print 'File:'
        sourceNode.setPropertyValue('full_filename',f.getPath())
        break
    print f.getName(),f.getPath()
stream.execute()

```

Metadatos: información sobre datos

Puesto que en una ruta los nodos se conectan entre sí, está disponible la información relativa a las columnas o los campos disponibles en cada nodo. Por ejemplo, en la interfaz de usuario de Modeler esto permite seleccionar por qué campos hay que ordenar o agregar. Esta información se llama modelo de datos.

Los scripts también pueden acceder al modelo de datos inspeccionando los campos que entran en un nodo o que salen de él. En algunos nodos coinciden los modelos de datos de entrada y salida; por ejemplo, un nodo Ordenar se limita a cambiar el orden de los registros, sin alterar el modelo de datos. Otros, como el nodo Derivar, pueden añadir nuevos campos. Otros, como el nodo Filtrar, pueden renombrar o eliminar campos.

En el ejemplo siguiente, el script toma la ruta estándar IBM SPSS Modeler druglearn.str y, para cada campo, construye un modelo descartando uno de los campos de entrada. Lo hace de la siguiente manera:

1. Accede al modelo de datos de salida del nodo Tipo.
2. Itera cada campo del modelo de datos de salida.
3. Modifica el nodo Filtro de cada campo de entrada.
4. Cambia el nombre del modelo que se construye.
5. Ejecuta el nodo de construcción de modelos.

Nota: Antes de ejecutar el script en la ruta druglearn.str, no olvide establecer el lenguaje de script a Python (la ruta se creó en una versión anterior de IBM SPSS Modeler, de modo que el lenguaje de script de la ruta está establecido a Herencia).

```

import modeler.api

stream = modeler.script.stream()
filternode = stream.findByType("filter", None)
typenode = stream.findByType("type", None)
c50node = stream.findByType("c50", None)
# Usar siempre un nombre de modelo personalizado
c50node.setPropertyValue("use_model_name", True)

lastRemoved = None
fields = typenode.getOutputDataModel()
for field in fields:
    # Si este es el campo de destino, se hace caso omiso del mismo
    if field.getModelingRole() == modeler.api.ModelingRole.OUT:
        continue

    # Se rehabilita el campo eliminado más recientemente
    if lastRemoved != None:
        filternode.setKeyedPropertyValue("include", lastRemoved, True)

    # Se elimina el campo
    lastRemoved = field.getColumnName()
    filternode.setKeyedPropertyValue("include", lastRemoved, False)

```

```
# Se establece el nombre del nuevo modelo y se ejecuta la construcción
c50node.setPropertyValue("model_name", "Exclude " + lastRemoved)
c50node.run([])
```

El objeto `DataModel` (modelo de datos) proporciona una serie de métodos de acceso a la información relativa a los campos y columnas del modelo de datos. Estos métodos se resumen en la siguiente tabla.

Tabla 18. Métodos del objeto `DataModel` de acceso a la información relativa a campos o columnas

| Método | Tipo devuelto | Descripción |
|---------------------------------------|--------------------------|---|
| <code>d.getColumnCount()</code> | <i>int</i> | Devuelve el número de columnas del modelo de datos. |
| <code>d.columnIterator()</code> | Iterator | Devuelve un iterador que devuelve cada columna en el orden "natural" de inserción. El iterador devuelve instancias de <code>Column</code> . |
| <code>d.nameIterator()</code> | Iterator | Devuelve un iterador que devuelve el nombre de cada columna en el orden "natural" de inserción. |
| <code>d.contains(nombre)</code> | <i>Booleana</i> | Devuelve <code>True</code> si en este <code>DataModel</code> existe una columna con el nombre proporcionado y <code>False</code> en caso contrario. |
| <code>d.getColumn(nombre)</code> | <code>Column</code> | Devuelve la columna cuyo nombre es el especificado. |
| <code>d.getColumnGroup(nombre)</code> | <code>ColumnGroup</code> | Devuelve el grupo de columnas nombrado o <code>None</code> si no existe dicho grupo de columnas. |
| <code>d.getColumnGroupCount()</code> | <i>int</i> | Devuelve el número de grupos de columnas de este modelo de datos. |
| <code>d.columnGroupIterator()</code> | Iterator | Devuelve un iterador que devuelve a su vez cada grupo de columnas. |
| <code>d.toArray()</code> | <code>Column[]</code> | Devuelve el modelo de datos como un vector de columnas. Las columnas van ordenadas según su orden "natural" de inserción. |

Cada campo (objeto `Column`) incluye una serie de métodos de acceso a la información de la columna. La tabla que se muestra a continuación muestra una selección de los mismos.

Tabla 19. Métodos del objeto `Column` de acceso a la información de una columna

| Método | Tipo devuelto | Descripción |
|---------------------------------|--------------------------|---|
| <code>c.getColumnName()</code> | <i>cadena</i> | Devuelve el nombre de la columna. |
| <code>c.getColumnLabel()</code> | <i>cadena</i> | Devuelve la etiqueta de la columna o una cadena vacía si no hay ninguna etiqueta asociada a la columna. |
| <code>c.getMeasureType()</code> | <code>MeasureType</code> | Devuelve el tipo de medición de la columna. |
| <code>c.getStorageType()</code> | <code>StorageType</code> | Devuelve el tipo de almacenamiento de la columna. |

Tabla 19. Métodos del objeto Column de acceso a la información de una columna (continuación)

| Método | Tipo devuelto | Descripción |
|-------------------------|---------------|---|
| c.isMeasureDiscrete() | Booleana | Devuelve True si la columna es discreta. Se consideran discretas las columnas que son un conjunto o un distintivo. |
| c.isModelOutputColumn() | Booleana | Devuelve True si la columna es una columna de resultado del modelo. |
| c.isStorageDatetime() | Booleana | Devuelve True si el almacenamiento de la columna es un valor de hora, fecha o indicación de fecha y hora. |
| c.isStorageNumeric() | Booleana | Devuelve True si el almacenamiento de la columna es un entero o un número real. |
| c.isValidValue(value) | Booleana | Devuelve True si el valor especificado es válido para este almacenamiento y valid cuando se conocen los valores de columna válidos. |
| c.getModelingRole() | ModelingRole | Devuelve el rol de modelado de la columna. |
| c.getSetValues() | Object[] | Devuelve un vector de valores válidos para la columna o None si se desconocen los valores o si la columna no es un conjunto. |
| c.getValueLabel(valor) | cadena | Devuelve la etiqueta del valor de la columna o una cadena vacía si no hay ninguna etiqueta asociada al valor. |
| c.getFalseFlag() | Object | Devuelve el valor indicador de "falso" de la columna o None si se desconoce el valor o la columna no es un indicador. |
| c.getTrueFlag() | Object | Devuelve el valor indicador de "verdadero" de la columna o None si se desconoce el valor o la columna no es un indicador. |
| c.getLowerBound() | Object | Devuelve el valor del límite inferior de los valores de la columna o None si se desconoce el valor o si la columna no es continua. |
| c.getUpperBound() | Object | Devuelve el valor del límite superior de los valores de la columna o None si se desconoce el valor o si la columna no es continua. |

Observe que la mayoría de los métodos de acceso a la información de una columna tienen métodos equivalentes definidos en el propio objeto DataModel. Por ejemplo, las dos sentencias siguientes son equivalentes:

```
dataModel.getColumn("unNombre").getModelingRole()
dataModel.getModelingRole("unNombre")
```

Acceso a objetos generados

Ejecución de una ruta normalmente implica producir objetos de salida adicionales. Estos objetos adicionales pueden ser un nuevo modelo o un fragmento de la salida que proporciona información para utilizarla en las ejecuciones posteriores.

En el ejemplo siguiente, la ruta `druglearn.str` se utiliza de nuevo como punto de partida para la ruta. En este ejemplo, se ejecutan todos los nodos de la ruta y los resultados se almacenan en una lista. A continuación, el script crea un bucle por los resultados y se guarda cualquier salida del modo resultante de la ejecución como un archivo de modelo de IBM SPSS Modeler (.gm) y se exporta como PMML.

```
import modeler.api

stream = modeler.script.stream()

# Set this to an existing folder on your system.
# Include a trailing directory separator
modelFolder = "C:/temp/models/"

# Execute the stream
models = []
stream.runAll(models)

# Save any models that were created
taskrunner = modeler.script.session().getTaskRunner()
for model in models:
    # If the stream execution built other outputs then ignore them
    if not(isinstance(model, modeler.api.ModelOutput)):
        continue

    label = model.getLabel()
    algorithm = model.getModelDetail().getAlgorithmName()

    # save each model...
    modelFile = modelFolder + label + algorithm + ".gm"
    taskrunner.saveModelToFile(model, modelFile)

    # ...and export each model PMML...
    modelFile = modelFolder + label + algorithm + ".xml"
    taskrunner.exportModelToFile(model, modelFile, modeler.api.FileFormat.XML)
```

La clase `taskrunner` proporciona un modo práctico de ejecutar diferentes tareas comunes. Los métodos que están disponibles en esta clase se resumen en la tabla siguiente.

Tabla 20. Métodos de la clase `taskrunner` para realizar tareas comunes

| Método | Tipo devuelto | Descripción |
|---|-----------------|--|
| <code>t.createStream(name, autoConnect, autoManage)</code> | Ruta | Crea y devuelve una nueva ruta. Tenga en cuenta que el código que debe crear las rutas de forma privada sin que las vea el usuario debe establecer el distintivo <code>autoManage</code> en <code>False</code> . |
| <code>t.exportDocumentToFile(documentOutput, filename, fileFormat)</code> | No es aplicable | Exporta la descripción de la ruta a un archivo utilizando el formato de archivo especificado. |
| <code>t.exportModelToFile(modelOutput, filename, fileFormat)</code> | No es aplicable | Exporta el modelo a un archivo utilizando el formato de archivo especificado. |
| <code>t.exportStreamToFile(stream, filename, fileFormat)</code> | No es aplicable | Exporta la ruta a un archivo utilizando el formato de archivo especificado. |

Tabla 20. Métodos de la clase taskrunner para realizar tareas comunes (continuación)

| Método | Tipo devuelto | Descripción |
|--|-----------------|---|
| t.insertNodeFromFile(filename, diagram) | Nodo | Lee un nodo en el archivo especificado y lo devuelve insertándolo en el diagrama suministrado. Tenga en cuenta que lo pueden utilizar los objetos Nodo y Supernodo. |
| t.openDocumentFromFile(filename, autoManage) | DocumentOutput | Lee un nodo en el archivo especificado y lo devuelve. |
| t.openModelFromFile(filename, autoManage) | ModelOutput | Lee un modelo en el archivo especificado y lo devuelve. |
| t.openStreamFromFile(filename, autoManage) | Ruta | Lee una ruta en el archivo especificado y la devuelve. |
| t.saveDocumentToFile(documentOutput, filename) | No es aplicable | Guarda el documento en la ubicación de archivo especificada. |
| t.saveModelToFile(modelOutput, filename) | No es aplicable | Guarda el modelo en la ubicación de archivo especificada. |
| t.saveStreamToFile(stream, filename) | No es aplicable | Guarda la ruta en la ubicación de archivo especificada. |

Manejo de errores

El lenguaje Python proporciona manejo de errores mediante el bloque de código `try...except`. Se puede utilizar en los scripts para capturar excepciones y manejar los problemas que podrían ocasionar la finalización del script.

En el script de ejemplo siguiente, se realiza un intento para recuperar un modelo desde el Repositorio de IBM SPSS Collaboration and Deployment Services. Esta operación puede hacer que se genere una excepción, por ejemplo, es posible que las credenciales de inicio de sesión en el repositorio no se hayan configurado correctamente o que la ruta del repositorio sea errónea. En el script, esto puede generar una excepción `ModelerException` (todas las excepciones que genera IBM SPSS Modeler se derivan de `modeler.api.ModelerException`).

```
import modeler.api

session = modeler.script.session()
try:
    repo = session.getRepository()
    m = repo.retrieveModel("/some-non-existent-path", None, None, True)
    # print goes to the Modeler UI script panel Debug tab
    print "Everything OK"
except modeler.api.ModelerException, e:
    print "An error occurred:", e.getMessage()
```

Nota: Algunas operaciones de scripts pueden generar excepciones Java estándar, estas excepciones no se derivan de `ModelerException`. Para capturar estas excepciones, se puede utilizar un bloque `except` adicional que capture todas las excepciones Java, por ejemplo:

```
import modeler.api

session = modeler.script.session()
try:
    repo = session.getRepository()
    m = repo.retrieveModel("/some-non-existent-path", None, None, True)
    # print goes to the Modeler UI script panel Debug tab
    print "Everything OK"
except modeler.api.ModelerException, e:
```

```

    print "An error occurred:", e.getMessage()
except java.lang.Exception, e:
    print "A Java exception occurred:", e.getMessage()

```

Parámetros de ruta, sesión y Supernodo

Los parámetros proporcionan una forma útil de pasar valores en el momento de la ejecución, en lugar de codificarlos directamente en un script. Los parámetros y sus valores se definen de la misma forma que las rutas, es decir, como entradas de la tabla de los parámetros de una ruta o Supernodo o como parámetros de la línea de comandos. Las clases de Ruta y Supernodo implementan un conjunto de funciones definidas por el objeto `ParameterProvider`, como se muestra en la tabla siguiente. La sesión proporciona una llamada `getParameters()` que devuelve un objeto que define dichas funciones.

Tabla 21. Funciones definidas por el objeto `ParameterProvider`

| Método | Tipo devuelto | Descripción |
|--|---------------------|--|
| <code>p.parameterIterator()</code> | Iterator | Devuelve un iterador de nombres de parámetro para este objeto. |
| <code>p.getParameterDefinition(parameterName)</code> | ParameterDefinition | Devuelve la definición de parámetro para el parámetro con el nombre especificado, o <code>None</code> si no existe tal parámetro en este proveedor. El resultado puede ser una instantánea de la definición en el momento en que el método se ha llamado y que no necesariamente refleja las modificaciones posteriores realizadas en el parámetro a través de este proveedor. |
| <code>p.getParameterLabel(parameterName)</code> | <i>cadena</i> | Devuelve la etiqueta del parámetro con nombre, o <code>None</code> si no existe tal parámetro. |
| <code>p.setParameterLabel(parameterName, label)</code> | No es aplicable | Establece la etiqueta del parámetro con nombre. |
| <code>p.getParameterStorage(parameterName)</code> | ParameterStorage | Devuelve el almacenamiento del parámetro con nombre, o <code>None</code> si no existe tal parámetro. |
| <code>p.setParameterStorage(parameterName, storage)</code> | No es aplicable | Establece el almacenamiento del parámetro con nombre. |
| <code>p.getParameterType(parameterName)</code> | ParameterType | Devuelve el tipo del parámetro con nombre, o <code>None</code> si no existe tal parámetro. |
| <code>p.setParameterType(parameterName, type)</code> | No es aplicable | Establece el tipo del parámetro con nombre. |
| <code>p.getParameterValue(parameterName)</code> | Object | Devuelve el valor del parámetro con nombre, o <code>None</code> si no existe tal parámetro. |
| <code>p.setParameterValue(parameterName, value)</code> | No es aplicable | Establece el valor del parámetro con nombre. |

En el ejemplo siguiente, el script agrega algunos datos Telco para averiguar qué región tiene los datos de promedio de ingresos más bajos. A continuación, se establece un parámetro de ruta con esta región. Este parámetro de ruta se utiliza en un nodo Seleccionar para excluir dicha región de los datos, antes de que se cree un modelo de abandono en el resto.

El ejemplo es artificial porque el script genera el propio nodo Seleccionar y, por lo tanto, podría haber generado el valor correcto directamente en la expresión del nodo Seleccionar. Sin embargo, las rutas se suelen construir previamente, de modo que establecer los parámetros de este modo proporciona un ejemplo útil.

La primera parte del script de ejemplo crea el parámetro de ruta que contendrá la región con el promedio de ingresos más bajo. El script también crea los nodos de la rama de agregación y la rama de creación de modelos y los conecta.

```
import modeler.api

stream = modeler.script.stream()

# Inicializar un parámetro de ruta
stream.setParameterStorage("LowestRegion", modeler.api.ParameterStorage.INTEGER)

# Crear primero la rama de agregación para calcular el promedio de ingresos por región
statisticsimportnode = stream.createAt("statisticsimport", "SPSS File", 114, 142)
statisticsimportnode.setPropertyValue("full_filename", "$CLEO_DEMOS/telco.sav")
statisticsimportnode.setPropertyValue("use_field_format_for_storage", True)

aggregatenode = modeler.script.stream().createAt("aggregate", "Aggregate", 294, 142)
aggregatenode.setPropertyValue("keys", ["region"])
aggregatenode.setKeyedPropertyValue("aggregates", "income", ["Mean"])

tablenode = modeler.script.stream().createAt("table", "Table", 462, 142)

stream.link(statisticsimportnode, aggregatenode)
stream.link(aggregatenode, tablenode)

selectnode = stream.createAt("select", "Select", 210, 232)
selectnode.setPropertyValue("mode", "Discard")
# Hacer referencia al parámetro de ruta en la selección
selectnode.setPropertyValue("condition", "'region' = '$P-LowestRegion'")

typenode = stream.createAt("type", "Type", 366, 232)
typenode.setKeyedPropertyValue("direction", "churn", "Target")

c50node = stream.createAt("c50", "C5.0", 534, 232)

stream.link(statisticsimportnode, selectnode)
stream.link(selectnode, typenode)
stream.link(typenode, c50node)
```

El script de ejemplo crea la ruta siguiente.

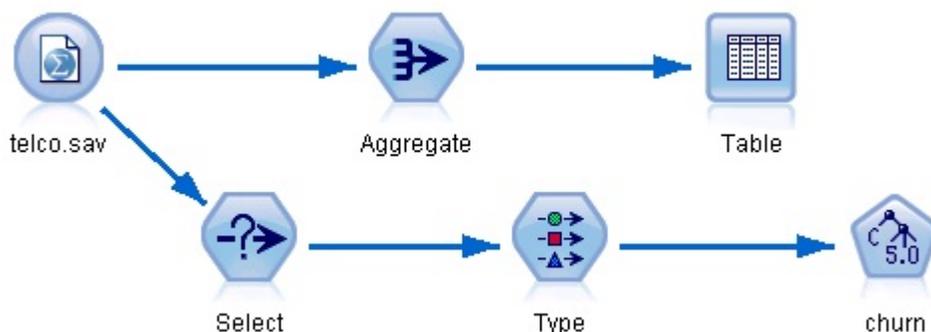


Figura 5. Ruta resultante del script de ejemplo

La parte siguiente del script de ejemplo ejecuta el nodo Tabla al final de la rama de agregación.

```
# Ejecutar primero el nodo Tabla
results = []
tablenode.run(results)
```

La parte siguiente del script de ejemplo accede a la salida de la tabla que ha generado la ejecución del nodo Tabla. A continuación, el script itera por las filas de la tabla, buscando la región con el promedio de ingresos más bajo.

```
# Ejecutar el nodo tabla para generar una sola tabla como salida
table = results[0]

# la salida de la tabla contiene un RowSet que permite acceder a valores como filas y columnas
rowset = table.getRowSet()
min_income = 1000000.0
min_region = None

# Del modo que se ha definido el nodo, la primera columna
# contiene la región y la segunda contiene el promedio de ingresos
row = 0
rowcount = rowset.getRowCount()
while row < rowcount:
    if rowset.getValueAt(row, 1) < min_income:
        min_income = rowset.getValueAt(row, 1)
        min_region = rowset.getValueAt(row, 0)
    row += 1
```

La parte siguiente del script utiliza la región con el promedio de ingresos más bajo para establecer el parámetro de ruta "LowestRegion" creado anteriormente. El script ejecuta el constructor de modelos excluyendo la región especificada de los datos de formación.

```
# Comprobar que se ha asignado un valor
if min_region != None:
    stream.setParameterValue("LowestRegion", min_region)
else:
    stream.setParameterValue("LowestRegion", -1)

# Finalmente, ejecutar el constructor de modelos con el criterio de selección
c50node.run([])
```

A continuación, se muestra el script de ejemplo.

```
import modeler.api

stream = modeler.script.stream()

# Crear un parámetro de ruta
stream.setParameterStorage("LowestRegion", modeler.api.ParameterStorage.INTEGER)

# Crear primero la rama de agregación para calcular el promedio de ingresos por región
statisticsimportnode = stream.createAt("statisticsimport", "SPSS File", 114, 142)
statisticsimportnode.setPropertyValue("full_filename", "$CLEO_DEMOS/telco.sav")
statisticsimportnode.setPropertyValue("use_field_format_for_storage", True)

aggregatenode = modeler.script.stream().createAt("aggregate", "Aggregate", 294, 142)
aggregatenode.setPropertyValue("keys", ["region"])
aggregatenode.setKeyedPropertyValue("aggregates", "income", ["Mean"])

tablenode = modeler.script.stream().createAt("table", "Table", 462, 142)

stream.link(statisticsimportnode, aggregatenode)
stream.link(aggregatenode, tablenode)

selectnode = stream.createAt("select", "Select", 210, 232)
selectnode.setPropertyValue("mode", "Discard")
# Hacer referencia al parámetro de ruta en la selección
selectnode.setPropertyValue("condition", "'region' = '$P-LowestRegion'")
```

```

typenode = stream.createAt("type", "Type", 366, 232)
typenode.setKeyedPropertyValue("direction", "churn", "Target")

c50node = stream.createAt("c50", "C5.0", 534, 232)

stream.link(statisticsimportnode, selectnode)
stream.link(selectnode, typenode)
stream.link(typenode, c50node)

# Ejecutar primero el nodo Tabla
results = []
tablenode.run(results)

# Ejecutar el nodo tabla para generar una sola tabla como salida
table = results[0]

# la salida de la tabla contiene un RowSet que permite acceder a valores como filas y columnas
rowset = table.getRowSet()
min_income = 1000000.0
min_region = None

# Del modo que se ha definido el nodo, la primera columna
# contiene la región y la segunda contiene el promedio de ingresos
row = 0
rowcount = rowset.getRowCount()
while row < rowcount:
    if rowset.getValueAt(row, 1) < min_income:
        min_income = rowset.getValueAt(row, 1)
        min_region = rowset.getValueAt(row, 0)
    row += 1

# Comprobar que se ha asignado un valor
if min_region != None:
    stream.setParameterValue("LowestRegion", min_region)
else:
    stream.setParameterValue("LowestRegion", -1)

# Finalmente, ejecutar el constructor de modelos con el criterio de selección
c50node.run([])

```

Valores globales

Los valores globales se utilizan para calcular diferentes estadísticas de resumen para los campos especificados. Se puede acceder a estos valores de resumen desde cualquier lugar de la ruta. Los valores globales son similares a los parámetros de ruta, ya que se puede acceder a los mismos por nombre a través de la ruta. Se diferencian de los parámetros de ruta en que los valores asociados se actualizan automáticamente cuando se ejecuta uno nodo Establecer globales, en lugar de asignarlos mediante script o desde la línea de comandos. Se accede a los valores globales de una ruta invocando el método `getGlobalValues()` de la ruta.

El objeto `GlobalValues` define las funciones que se muestran en la tabla siguiente.

Tabla 22. Funciones definidas por el objeto `GlobalValues`

| Método | Tipo devuelto | Descripción |
|------------------------------------|---------------|--|
| <code>g.fieldNameIterator()</code> | Iterator | Devuelve un iterador para cada nombre de campo con al menos un valor global. |

Tabla 22. Funciones definidas por el objeto GlobalValues (continuación)

| Método | Tipo devuelto | Descripción |
|-----------------------------|---------------|---|
| g.getValue(type, fieldName) | Object | Devuelve el valor global para el tipo especificado y nombre de campo, o None si no se puede localizar ningún valor. Generalmente se espera que el valor devuelto sea un número, aunque en las funciones futuras se pueden devolver tipos de valores diferentes. |
| g.getValues(fieldName) | Mapa | Devuelve un mapa que contiene las entradas conocidas para el nombre de campo especificado o None si no hay entradas para el campo. |

GlobalValues.Type define el tipo de estadísticas de resumen disponibles. Están disponibles las siguientes estadísticas de resumen:

- MAX: el valor máximo del campo.
- MEAN: el valor medio del campo.
- MIN: el valor mínimo del campo.
- STDDEV: la desviación estándar del campo.
- SUM: la suma de los valores del campo.

Por ejemplo, el script siguiente accede el valor medio del campo "income" que calcula un nodo Val. globales:

```
import modeler.api

globals = modeler.script.stream().getGlobalValues()
mean_income = globals.getValue(modeler.api.GlobalValues.Type.MEAN, "income")
```

Trabajar con varias rutas: Scripts autónomos

Para trabajar con varias rutas se debe utilizar un script autónomo. El script autónomo se puede editar y ejecutar dentro de la interfaz de usuario de IBM SPSS Modeler o se puede pasar como un parámetro de línea de comandos en modalidad de proceso por lotes.

El siguiente script autónomo abre dos rutas. Una de estas rutas crea un modelo y la segunda ruta traza la distribución de los valores predichos.

```
# Change to the appropriate location for your system
demosDir = "C:/Program Files/IBM/SPSS/Modeler/18.1/DEMOS/streams/"

session = modeler.script.session()
tasks = session.getTaskRunner()

# Open the model build stream, locate the C5.0 node and run it
buildstream = tasks.openStreamFromFile(demosDir + "druglearn.str", True)
c50node = buildstream.findByType("c50", None)
results = []
c50node.run(results)

# Now open the plot stream, find the Na_to_K derive and the histogram
plotstream = tasks.openStreamFromFile(demosDir + "drugplot.str", True)
derivenode = plotstream.findByType("derive", None)
histogramnode = plotstream.findByType("histogram", None)

# Create a model applier node, insert it between the derive and histogram nodes
# then run the histogram
```

```
applyc50 = plotstream.createModelApplier(results[0], results[0].getName())
applyc50.setPositionBetween(derivenode, histogramnode)
plotstream.linkBetween(applyc50, derivenode, histogramnode)
histogramnode.setPropertyValue("color_field", "$C-Drug")
histogramnode.run([])

# Finally, tidy up the streams
buildstream.close()
plotstream.close()
```

El ejemplo siguiente muestra cómo también puede iterar en las rutas abiertas (todas las rutas abiertas en la pestaña Rutas). Tenga en cuenta que esto solo está soportado en scripts autónomos.

```
for stream in modeler.script.streams():
    print stream.getName()
```

Capítulo 5. Sugerencias para scripts

Esta sección proporciona una visión general de las técnicas y sugerencias para utilizar scripts, incluida la modificación de la ejecución de la ruta, la utilización de una contraseña codificada en un script y el acceso a objetos en el Repositorio de IBM SPSS Collaboration and Deployment Services.

Modificación de ejecución de rutas

Cuando se ejecuta una ruta, sus nodos terminales se ejecutan en un orden optimizado para la situación predeterminada. En algunos casos, es posible que prefiera un orden de ejecución diferente. Para modificar el orden de ejecución de una ruta, lleve a cabo los siguientes pasos desde la pestaña Ejecución del cuadro de diálogo Propiedades de ruta:

1. Comience con un script vacío.
2. Pulse en el botón **Añadir script predeterminado** de la barra de herramientas y añada el script de ruta predeterminado.
3. Cambie el orden de las instrucciones del script de ruta predeterminado por el orden en que desee que se ejecuten las instrucciones.

Nodos de recorrido en bucle

Puede utilizar un bucle for para recorrer en bucle todos los nodos de una ruta. Por ejemplo, los siguientes dos ejemplos de script recorren en bucle todos los nodos y cambia los nombres de campos de cualquier nodo Filtrar a mayúsculas.

Estos scripts pueden emplearse en cualquier ruta que tenga un nodo Filtrar, incluso si no hay campos filtrados. Simplemente, añada un nodo Filtrar que recorra todos los campos para cambiar todos los nombres de campo a mayúsculas.

```
# Alternative 1: using the data model nameIterator() function
stream = modeler.script.stream()
for node in stream.iterator():
    if (node.getTypeName() == "filter"):
        # nameIterator() returns the field names
        for field in node.getInputDataModel().nameIterator():
            newname = field.upper()
            node.setKeyedPropertyValue("new_name", field, newname)

# Alternative 2: using the data model iterator() function
stream = modeler.script.stream()
for node in stream.iterator():
    if (node.getTypeName() == "filter"):
        # iterator() returns the field objects so we need
        # to call getColumnName() to get the name
        for field in node.getInputDataModel().iterator():
            newname = field.getColumnName().upper()
            node.setKeyedPropertyValue("new_name", field.getColumnName(), newname)
```

El script recorre en bucle los nodos de la ruta actual y comprueba cada nodo para ver si es un Filtro. Si es así, el script recorre en bucle cada campo del nodo y utiliza la función `field.upper()` o `field.getColumnName().upper()` para cambiar el nombre a mayúsculas.

Acceso a objetos en el Repositorio de IBM SPSS Collaboration and Deployment Services

Si tiene una licencia para Repositorio de IBM SPSS Collaboration and Deployment Services, puede almacenar y recuperar objetos del repositorio utilizando comandos de script. Utilice el repositorio para gestionar el ciclo de vida de los modelos de minería de datos y los objetos predictivos relacionados en el contexto de aplicaciones empresariales, herramientas y soluciones.

Conexión con Repositorio de IBM SPSS Collaboration and Deployment Services

Para acceder al repositorio, en primer lugar, debe configurar una conexión válida con el mismo, a través del menú **Herramientas** de la interfaz de usuario de SPSS Modeler o a través de la línea de comandos. Si desea obtener más información, consulte "Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión" en la página 67.

Obtener acceso al repositorio

Se puede acceder al repositorio desde la sesión, por ejemplo:

```
repo = modeler.script.session().getRepository()
```

Recuperación de objetos del repositorio

En un script, utilice las funciones `retrieve*` para acceder a distintos objetos, incluyendo rutas, modelos, salidas y nodos. Se muestra un resumen de las funciones de recuperación en la tabla siguiente.

Tabla 23. Recuperar funciones de script

| Tipo de objeto | Función de repositorio |
|----------------|--|
| Ruta | <code>repo.retrieveStream(String path, String version, String label, Boolean autoManage)</code> |
| Modelo | <code>repo.retrieveModel(String path, String version, String label, Boolean autoManage)</code> |
| Resultado | <code>repo.retrieveDocument(String path, String version, String label, Boolean autoManage)</code> |
| Nodo | <code>repo.retrieveProcessor(String path, String version, String label, ProcessorDiagram diagram)</code> |

Por ejemplo, puede recuperar una ruta del repositorio con la función siguiente:

```
stream = repo.retrieveStream("/projects/retention/risk_score.str", None, "production", True)
```

Este ejemplo recupera la ruta `risk_score.str` de la carpeta especificada. La etiqueta `production` identifica qué versión de la ruta recuperar, y el último parámetro especifica que SPSS Modeler va a gestionar la ruta (por ejemplo, por lo que la ruta aparece en la pestaña **Rutas** si la interfaz de usuario de SPSS Modeler es visible). Como alternativa, para utilizar una versión específica sin etiquetar:

```
stream = repo.retrieveStream("/projects/retention/risk_score.str", "0:2015-10-12 14:15:41.281",  
None, True)
```

Nota: Si ambos parámetros, el de versión y el de etiqueta, son `None`, se devuelve la última versión.

Almacenar objetos en el repositorio

Para utilizar scripts para almacenar objetos en el repositorio, utilice las funciones `store*`. Se muestra un resumen de las funciones de almacenamiento en la tabla siguiente.

Tabla 24. Funciones de script de almacenamiento

| Tipo de objeto | Función de repositorio |
|----------------|--|
| Ruta | repo.storeStream(ProcessorStream stream, String path, String label) |
| Modelo | repo.storeModel(ModelOutput modelOutput, String path, String label) |
| Resultado | repo.storeDocument(DocumentOutput documentOutput, String path, String label) |
| Nodo | repo.storeProcessor(Processor node, String path, String label) |

Por ejemplo, puede almacenar una nueva versión de la ruta `risk_score.str` con la función siguiente:

```
versionId = repo.storeStream(stream,
"/projects/retention/risk_score.str", "test")
```

Este ejemplo almacena una nueva versión de la ruta, asocia la etiqueta "test" a la misma y devuelve el marcador de la versión para la versión recién creada.

Nota: Si no desea asociar una etiqueta a la nueva versión, pase None para la etiqueta.

Gestión de carpetas de repositorio

Al utilizar las carpetas dentro del repositorio, puede organizar objetos en grupos lógicos y hacer que sea más fácil ver qué objetos están relacionados. Cree carpetas utilizando la función `createFolder()`, como en el ejemplo siguiente:

```
newpath = repo.createFolder("/projects", "cross-sell")
```

Este ejemplo crea una carpeta nueva que se llama "cross-sell" en la carpeta "/projects". La función devuelve la vía de acceso completa de la carpeta nueva.

Para cambiar el nombre de una carpeta, utilice la función `renameFolder()`:

```
repo.renameFolder("/projects/cross-sell", "cross-sell-Q1")
```

El primer parámetro es la vía de acceso completa que se va a renombrar, y el segundo es el nombre nuevo que se va a proporcionar a dicha carpeta.

Para suprimir una carpeta vacía, utilice la función `deleteFolder()`:

```
repo.deleteFolder("/projects/cross-sell")
```

Bloquear y desbloquear objetos

Puede bloquear un objeto desde un script para evitar que otros usuarios actualicen cualquiera de las versiones existentes o creen nuevas versiones. También puede desbloquear un objeto que haya bloqueado.

La sintaxis para bloquear y desbloquear un objeto es:

```
repo.lockFile(REPOSITORY_PATH)
repo.lockFile(URI)
```

```
repo.unlockFile(REPOSITORY_PATH)
repo.unlockFile(URI)
```

Como cuando se almacenan y se recuperan objetos, `RUTA_REPOSITORIO` le ofrece la ubicación del objeto en el repositorio. La ruta debe estar entre comillas y utilizar barras inclinadas como delimitadores. No distingue entre mayúsculas y minúsculas.

```
repo.lockFile("/myfolder/Stream1.str")
repo.unlockFile("/myfolder/Stream1.str")
```

Si lo prefiere, puede utilizar un identificador de recursos uniforme (URI) en lugar de una ruta de repositorio para proporcionar la ubicación del proyecto. El URI debe incluir el prefijo `spsscr:` y debe estar entre comillas. Sólo se puede utilizar barras inclinadas como delimitadores, y los espacios deben estar codificados. Es decir, utilizar `%20` en lugar de un espacio en la ruta. El URI no distingue entre mayúsculas y minúsculas. A continuación aparecen algunos ejemplos:

```
repo.lockFile("spsscr:///myfolder/Stream1.str")
repo.unlockFile("spsscr:///myfolder/Stream1.str")
```

Tenga en cuenta que el bloqueo de objetos se aplica a todas las versiones de un objeto: no puede bloquear o desbloquear versiones por separado.

Generación de una contraseña codificada

En algunos casos, puede que necesite incluir una contraseña en un script. Por ejemplo, es posible que desee acceder a un origen de datos protegido con contraseña. Las contraseñas codificadas pueden utilizarse en:

- Propiedades de los nodos para un origen de base de datos y nodos de resultado
- Argumentos de línea de comando para conectarse al servidor
- Propiedades de conexión con la base de datos almacenadas en un archivo `.par` (archivo de parámetro generado desde la pestaña Publicar de un nodo de exportación)

A través de la interfaz de usuario, está disponible una herramienta para generar contraseñas codificadas basándose en el algoritmo Blowfish (consulte <http://www.schneier.com/blowfish.html> si desea más información). Una vez codificada, puede copiar y almacenar la contraseña en archivos de script y argumentos de líneas de comando. La propiedad de nodo `epassword` utilizada para `database` y `databaseexport` almacena la contraseña codificada.

1. Para generar una contraseña codificada, en el menú Herramientas seleccione:

Codificar contraseña...

2. Especifique una contraseña en el cuadro de texto Contraseña.
3. Pulse en **Codificar** para generar una codificación aleatoria de la contraseña.
4. Pulse en el botón Copiar para copiar la contraseña codificada al Portapapeles.
5. Pegue la contraseña en el script o parámetro deseado.

Comprobación de script

Puede comprobar rápidamente la sintaxis de todos los tipos de scripts pulsando en el botón de comprobación de la barra de herramientas del cuadro de diálogo Script de ruta.



Figura 6. Iconos de barra de herramientas del script de ruta

En la comprobación por script se avisa de cualquier error que se detecte en el código y se sugieren recomendaciones de mejora. Para ver la línea con errores, pulse en los comentarios, en la mitad inferior del cuadro de diálogo. Los errores se señalan en rojo.

Scripts desde la línea de comandos

El uso scripts permite ejecutar operaciones típicamente desarrolladas en la interfaz de usuario. Simplemente especifique y ejecute una ruta independiente en la línea de comandos cuando ejecute IBM SPSS Modeler. Por ejemplo:

```
client -script scores.txt -execute
```

La marca `-script` carga el script especificado, mientras que la marca `-execute` ejecuta todos los comandos del archivo de script.

Compatibilidad con releases anteriores

Los scripts creados en versiones anteriores de IBM SPSS Modeler deberían funcionar normalmente sin cambios en la versión actual. Sin embargo, los nuggets de modelos podrán ahora insertarse en la ruta automáticamente (es el comportamiento predeterminado) y podrán sustituir o complementar un nugget existente del tipo en la ruta. El que esto ocurra depende de la configuración de las opciones **Añadir modelo a ruta** y **Sustituir modelo anterior** (**Herramientas > Opciones > Opciones de usuario > Notificaciones**). Por ejemplo, es posible que tenga que modificar un script de una versión anterior en el que la sustitución del nugget se trate borrando el nugget existente e insertando uno nuevo.

Es posible que los scripts creados en esta versión no funcionen en versiones anteriores.

Si un script creó una liberación antigua utiliza un comando que se ha sustituido desde entonces (o desaprobado), la forma antigua se seguirá admitiendo, pero aparecerá un mensaje de advertencia. Por ejemplo, la palabra clave antigua `generated` se ha sustituido por `model`, y `clear generated` se ha sustituido por `clear generated palette`. Los scripts que utilizan las formas antiguas se seguirán ejecutando, pero se mostrará una advertencia.

Acceder a resultados de ejecución de la ruta

Muchos nodos de IBM SPSS Modeler producen datos de salida tales como modelos, diagramas y datos tabulares. Muchos de estos datos de salida contienen valores útiles que pueden ser utilizados por scripts para guiar la ejecución subsiguiente. Estos valores se agrupan en contenedores de contenido (denominados simplemente contenedores) a los que se puede acceder utilizando etiquetas o identificadores que identifican cada contenedor. La forma en que se accede a estos valores depende del formato o "modelo de contenido" utilizado por el contenedor.

Por ejemplo, muchos resultados de modelo predictivo utilizan una variante de XML llamada PMML para representar información sobre el modelo, tal como qué campos utiliza un árbol de decisiones en cada bifurcación o cómo están conectadas las neuronas de una red neuronal y con qué intensidades. Los resultados del modelo que utilizan PMML proporcionan un modelo de contenido XML que se puede utilizar para acceder a esa información. Por ejemplo:

```
stream = modeler.script.stream()
# Suponga que la ruta contiene un nodo generador de modelos C5.0
# y que el origen de datos, los predictores y los destinos ya se han
# configurado
modelbuilder = stream.findByType("c50", None)
results = []
modelbuilder.run(results)
modeloutput = results[0]

# Ahora que tenemos el objeto de salida del modelo C5.0, acceda al modelo
# de contenido pertinente
cm = modeloutput.getContentModel("PMML")

# El modelo de contenido PMML es un modelo de contenido genérico basado en XML
# que utiliza sintaxis del lenguaje XPath. Utilice ese modelo para encontrar
# los nombres de los campos de datos.
# La llamada devuelve una lista de series de caracteres correspondientes a los valores XPath
dataFieldNames = cm.getStringValues("/PMML/DataDictionary/DataField", "name")
```

IBM SPSS Modeler es compatible con los modelos de contenido siguientes en los scripts:

- **Modelo de contenido de tabla:** proporciona acceso a datos tabulares simples representados como filas y columnas

- **Modelo de contenido XML:** proporciona acceso a contenido almacenado en formato XML
- **Modelo de contenido JSON:** proporciona acceso a contenido almacenado en formato JSON
- **Modelo de contenido de estadísticas de columna:** proporciona acceso a estadísticas de resumen sobre un campo determinado
- **Modelo de contenido de estadísticas de columna por pares:** proporciona acceso a estadísticas de resumen entre dos campos o valores entre dos campos separados

Modelo de contenido de tabla

El modelo de contenido de tabla proporciona un modelo sencillo para acceder a los datos simples de fila y columna. Los valores en una columna determinada deben tener todos el mismo tipo de almacenamiento (por ejemplo, series o enteros).

API

Tabla 25. API

| Devolver | Método | Descripción |
|-------------|---|--|
| int | getRowCount() | Devuelve el número de filas en esta tabla. |
| int | getColumnCount() | Devuelve el número de columnas en esta tabla. |
| Cadena | getColumnName(int columnIndex) | Devuelve el nombre de la columna en el índice de columna especificado. El índice de columna comienza en el 0. |
| StorageType | getStorageType(int columnIndex) | Devuelve el tipo de almacenamiento de la columna en el índice especificado. El índice de columna comienza en el 0. |
| Object | getValueAt(int rowIndex, int columnIndex) | Devuelve el valor en los índices de fila y columna especificados. Los índices de fila y columna comienzan en el 0. |
| void | reset() | Desecha cualquier almacenamiento interno asociado con este modelo de contenido. |

Nodos y salidas

Esta tabla lista los nodos que crean salidas que incluyen este tipo de modelo de contenido.

Tabla 26. Nodos y salidas

| Nombre de nodo | Nombre de resultado | ID de contenedor |
|----------------|---------------------|------------------|
| tabla | tabla | "tabla" |

Script de ejemplo

```
stream = modeler.script.stream()
from modeler.api import StorageType

# Establecer el nodo de importación de archivo de variable
varfilenode = stream.createAt("variablefile", "DRUG Data", 96, 96)
varfilenode.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUGIn")
```

```

# Crear el nodo Agregar y conectarlo al nodo de archivo de variable
aggregatenode = stream.createAt("aggregate", "Aggregate", 192, 96)
stream.link(varfilenode, aggregatenode)

# Configurar el nodo Agregar
aggregatenode.setPropertyValue("keys", ["Drug"])
aggregatenode.setKeyedPropertyValue("aggregates", "Age", ["Min", "Max"])
aggregatenode.setKeyedPropertyValue("aggregates", "Na", ["Mean", "SDev"])

# Crear el nodo de salida de tabla y conectarlo al nodo Agregar
tablenode = stream.createAt("table", "Table", 288, 96)
stream.link(aggregatenode, tablenode)

# Ejecutar el nodo de tabla y capturar el objeto de salida de resultado de la tabla
results = []
tablenode.run(results)
tableoutput = results[0]

# Acceder al modelo de contenido del resultado de la tabla
tablecontent = tableoutput.getContentModel("table")

# Para cada columna, imprimir el nombre de columna, tipo y la primera fila
# de valores del contenido de la tabla
col = 0
while col < tablecontent.getColumnCount():
    print tablecontent.getColumnName(col), \
          tablecontent.getStorageType(col), \
          tablecontent.getValueAt(0, col)
    col = col + 1

```

La salida en la pestaña Depuración de scripts tendrá un aspecto similar a este:

```

Age_Min Integer 15
Age_Max Integer 74
Na_Mean Real 0.730851098901
Na_SDev Real 0.116669731242
Drug String drugY
Record_Count Integer 91

```

Modelo de contenido XML

El modelo de contenido XML proporciona acceso a contenido basado en XML.

El modelo de contenido XML permite acceder a componentes utilizando expresiones XPath. Las expresiones XPath son series que definen qué elementos o atributos son necesarios para el solicitante. El modelo de contenido XML hace que sea transparente la creación de diversos objetos y expresiones de compilación que normalmente son necesarios para el soporte de XPath. Esto hace que sea más sencillo hacer llamadas desde scripts Python.

El modelo de contenido XML incluye una función que devuelve el documento XML como serie de caracteres. Esto permite que los usuarios del script Python utilicen su biblioteca preferida de Python para analizar XML.

API

Tabla 27. API

| Devolver | Método | Descripción |
|----------|------------------|--|
| Cadena | getXMLAsString() | Devuelve el XML en forma de serie de caracteres. |

Tabla 27. API (continuación)

| Devolver | Método | Descripción |
|---|--|--|
| Número | <code>getNumericValue(String xpath)</code> | Devuelve un resultado de tipo numérico al analizar la vía de acceso (por ejemplo, contar el número de elementos que coinciden con la expresión de vía de acceso). |
| Booleano | <code>getBooleanValue(String xpath)</code> | Devuelve un resultado de tipo booleano al evaluar la expresión de vía de acceso especificada. |
| Cadena | <code>getStringValue(String xpath, String attribute)</code> | Devuelve el valor de atributo o valor de nodo XML que coincide con la vía de acceso especificada. |
| Lista de series de caracteres | <code>getStringValues(String xpath, String attribute)</code> | Devuelve una lista de todos los valores de atributo o valores de nodo XML que coinciden con la vía de acceso especificada. |
| Lista de series de caracteres | <code>getValuesList(String xpath, <Lista de series de caracteres> attributes, boolean includeValue)</code> | Devuelve una lista de todos los valores de atributo que coinciden con la vía de acceso especificada junto con el valor de nodo XML si es necesario. |
| Hash table (key:string, value:list of string) | <code>getValuesMap(String xpath, String keyAttribute, <Lista de series de caracteres> attributes, boolean includeValue)</code> | Devuelve una tabla hash que utiliza el atributo de clave o valor de nodo XML como clave y la lista de valores de atributo especificados como valores de la tabla. |
| Booleano | <code>isNamespaceAware()</code> | Indica si los analizadores XML deben tener en cuenta los espacios de nombres. El valor predeterminado es <code>False</code> . |
| void | <code>setNamespaceAware(boolean value)</code> | Establece si los analizadores XML deben tener en cuenta los espacios de nombres. Esto también invoca a <code>reset()</code> para asegurarse de que los cambios sean captados por llamadas posteriores. |
| void | <code>reset()</code> | Desecha cualquier almacenamiento interno asociado con el modelo de contenido (por ejemplo, un objeto DOM almacenado en memoria caché). |

Nodos y salidas

Esta tabla lista los nodos que crean salidas que incluyen este tipo de modelo de contenido.

Tabla 28. Nodos y salidas

| Nombre de nodo | Nombre de resultado | ID de contenedor |
|--|---------------------------------|------------------|
| La mayoría de constructores de modelos | La mayoría de modelos generados | "PMML" |
| "autodataprep" | n/d | "PMML" |

Script de ejemplo

El código del script Python para acceder al contenido puede tener este aspecto:

```
results = []
modelbuilder.run(results)
modeloutput = results[0]
cm = modeloutput.getContentModel("PMML")

dataFieldNames = cm.getStringValues("/PMML/DataDictionary/DataField", "name")
predictedNames = cm.getStringValues("//MiningSchema/MiningField[@usageType='predicted']",
    "name")
```

Modelo de contenido JSON

El modelo de contenido JSON se utiliza para proporcionar soporte para contenido con formato JSON. Esto proporciona una API básica para permitir que los solicitantes extraigan valores bajo la asunción de que saben qué valores se deben acceder.

API

Tabla 29. API

| Devolver | Método | Descripción |
|---------------------------------------|---|--|
| Cadena | <code>getJSONAsString()</code> | Devuelve el contenido JSON como serie de caracteres. |
| Object | <code>getObjectAt(<Lista de objetos> path, JSONArtifact artifact)</code> throws Exception | Devuelve el objeto situado en la vía de acceso especificada. El artefacto raíz proporcionado puede ser nulo, en cuyo caso se utiliza la raíz del contenido. El valor devuelto puede ser una serie literal, un entero, un número real, un valor booleano o un artefacto JSON (ya sea un objeto JSON o una matriz JSON). |
| Tabla hash (key:object, value:object) | <code>getChildValuesAt(<Lista de objetos> path, JSONArtifact artifact)</code> throws Exception | Devuelve los valores hijo de la vía de acceso especificada si la vía conduce a un objeto JSON, o devuelve nulo en caso contrario. Las claves de la tabla son series, mientras que el valor asociado puede ser una serie literal, un entero, un número real, un valor booleano o un artefacto JSON (ya sea un objeto JSON o una matriz JSON). |
| Lista de objetos | <code>getChildrenAt(<Lista de objetos> path path, JSONArtifact artifact)</code> throws Exception | Devuelve la lista de objetos situados en la vía de acceso especificada si la vía conduce a una matriz JSON, o devuelve nulo en caso contrario. Los valores devueltos pueden ser una serie literal, un entero, un número real, un valor booleano o un artefacto JSON (ya sea un objeto JSON o una matriz JSON). |
| void | <code>reset()</code> | Desecha cualquier almacenamiento interno asociado con el modelo de contenido (por ejemplo, un objeto DOM almacenado en memoria caché). |

Script de ejemplo

Si existe un nodo generador de salida que crea salida en formato JSON, se puede utilizar lo siguiente para acceder a información sobre un conjunto de libros:

```
results = []
outputbuilder.run(results)
output = results[0]
cm = output.getContentModel("jsonContent")

bookTitle = cm.getObjectAt(["books", "ISIN123456", "title"], None)

# Como alternativa, obtenga el objeto de libro y utilícelo como raíz
# para entradas subsiguientes
book = cm.getObjectAt(["books", "ISIN123456"], None)
bookTitle = cm.getObjectAt(["title"], book)

# Obtener todos los valores hijos para un libro determinado
bookInfo = cm.getChildValuesAt(["books", "ISIN123456"], None)

# Obtener la tercera entrada de libro. Se supone el valor de nivel superior "books"
# contiene una matriz JSON que se puede indexar
bookInfo = cm.getObjectAt(["books", 2], None)

# Obtener una lista de todas las entradas hijas
allBooks = cm.getChildrenAt(["books"], None)
```

Modelo de contenido de estadísticas de columna y modelo de contenido de estadísticas por pares

El modelo de contenido de estadísticas de columna proporciona acceso a estadísticas que se pueden calcular para cada campo (estadísticas univariadas). El modelo de contenido de estadísticas por pares proporciona acceso a estadísticas que se pueden calcular para pares de campos o pares de valores de un campo.

Las medidas estadísticas posibles son:

- Count
- UniqueCount
- ValidCount
- Mean
- Sum
- Min
- Max
- Range
- Variance
- StandardDeviation
- StandardErrorOfMean
- Skewness
- SkewnessStandardError
- Kurtosis
- KurtosisStandardError
- Mediana
- Mode
- Pearson

- Covariance
- TTest
- FTest

Algunos valores sólo son adecuados para estadísticas de una sola columna, mientras que otros sólo son adecuados para estadísticas por pares.

Los nodos que generan estadísticas son los siguientes:

- El **nodo Estadísticas** produce estadísticas de columna y puede producir estadísticas por pares cuando se especifican campos de correlación
- El **nodo Auditoría de datos** produce estadísticas de columna y puede producir estadísticas por pares cuando se especifica un campo de preformato.
- El **nodo Medias** produce estadísticas por pares cuando compara pares de campos o cuando compara valores de un campo con otros resúmenes de campo.

Qué modelos de contenido y estadísticas se pueden utilizar depende de las prestaciones del nodo en cuestión y de los valores contenidos en ese nodo.

La API ColumnStatsContentModel

Tabla 30. La API ColumnStatsContentModel.

| Devolver | Método | Descripción |
|---------------------|--|---|
| List<StatisticType> | getAvailableStatistics() | Devuelve las estadísticas disponibles en este modelo. No todos los campos tendrán necesariamente valores para todas las estadísticas. |
| List<String> | getAvailableColumns() | Devuelve los nombres de columna para los que se han calculado estadísticas. |
| Number | getStatistic(String column, StatisticType statistic) | Devuelve los valores estadísticos asociados a la columna. |
| void | reset() | Desecha cualquier almacenamiento interno asociado con el modelo de contenido. |

La API PairwiseStatsContentModel

Tabla 31. La API PairwiseStatsContentModel.

| Devolver | Método | Descripción |
|---------------------|--------------------------------|---|
| List<StatisticType> | getAvailableStatistics() | Devuelve las estadísticas disponibles en este modelo. No todos los campos tendrán necesariamente valores para todas las estadísticas. |
| List<String> | getAvailablePrimaryColumns() | Devuelve los nombres de columna primaria para los que se han calculado estadísticas. |
| List<Object> | getAvailablePrimaryValues() | Devuelve los valores de la columna primaria para la que se han calculado estadísticas. |
| List<String> | getAvailableSecondaryColumns() | Devuelve los nombres de columna secundaria para los que se han calculado estadísticas. |

Tabla 31. La API PairwiseStatsContentModel (continuación).

| Devolver | Método | Descripción |
|----------|--|--|
| Number | getStatistic(String primaryColumn, String secondaryColumn, StatisticType statistic) | Devuelve los valores estadísticos asociados a las columnas. |
| Number | getStatistic(String primaryColumn, Object primaryValue, String secondaryColumn, StatisticType statistic) | Devuelve los valores estadísticos asociados al valor de la columna primaria y la columna secundaria. |
| void | reset() | Desecha cualquier almacenamiento interno asociado con el modelo de contenido. |

Nodos y salidas

Esta tabla lista los nodos que generan datos de salida y que incluyen este tipo de modelo de contenido.

Tabla 32. Nodos y salidas.

| Nombre de nodo | Nombre de resultado | ID de contenedor | Notas |
|--|---------------------|----------------------|--|
| "means" (nodo Medias) | "means" | "columnStatistics" | |
| "means" (nodo Medias) | "means" | "pairwiseStatistics" | |
| "dataaudit" (nodo Auditoría de datos) | "means" | "columnStatistics" | |
| "statistics" (nodo Estadísticas) | "statistics" | "columnStatistics" | Sólo se genera cuando se examinan campos determinados. |
| "statistics" (nodo Estadísticas) | "statistics" | "pairwiseStatistics" | Sólo se genera cuando se correlacionan campos. |

Script de ejemplo

```

from modeler.api import StatisticType
stream = modeler.script.stream()

# Definir los datos de entrada
varfile = stream.createAt("variablefile", "File", 96, 96)
varfile.setPropertyValue("full_filename", "$CLE0/DEMOS/DRUG1n")

# Crear el nodo de estadísticas. Este nodo puede producir
# estadísticas de columna y estadísticas por pares
statisticsnode = stream.createAt("statistics", "Stats", 192, 96)
statisticsnode.setPropertyValue("examine", ["Age", "Na", "K"])
statisticsnode.setPropertyValue("correlate", ["Age", "Na", "K"])
stream.link(varfile, statisticsnode)

results = []
statisticsnode.run(results)
statsoutput = results[0]
statscm = statsoutput.getContentModel("columnStatistics")
if (statscm != None):
    cols = statscm.getAvailableColumns()
    stats = statscm.getAvailableStatistics()
    print "Column stats:", cols[0], str(stats[0]), " = ", statscm.getStatistic(cols[0], stats[0])

```

```
statscm = statsoutput.getContentModel("pairwiseStatistics")
if (statscm != None):
    pcols = statscm.getAvailablePrimaryColumns()
    scols = statscm.getAvailableSecondaryColumns()
    stats = statscm.getAvailableStatistics()
    corr = statscm.getStatistic(pcols[0], scols[0], StatisticType.Pearson)
    print "Pairwise stats:", pcols[0], scols[0], " Pearson = ", corr
```

Capítulo 6. Argumentos de la línea de comandos

Invocación del software

Puede utilizar la línea de comandos del sistema operativo para ejecutar IBM SPSS Modeler de la siguiente manera:

1. En un ordenador en el que se haya instalado IBM SPSS Modeler, abra una ventana de DOS o del indicador de comandos.
2. Para iniciar la interfaz de IBM SPSS Modeler en modo interactivo, escriba el comando `clementine` seguido de los argumentos necesarios; por ejemplo:

```
modelerclient -stream report.str -execute
```

Los argumentos disponibles (modificadores) permiten conectar con un servidor, cargar rutas, ejecutar scripts o especificar otros parámetros, según sea necesario.

Utilización de argumentos de la línea de comandos

Se pueden añadir argumentos de línea de comandos (también denominados *marcas*) al comando inicial `modelerclient` o para alterar la invocación de IBM SPSS Modeler.

Hay varios tipos de argumentos de línea de comandos disponibles que se describen más adelante en esta sección.

Tabla 33. Tipos de argumentos de línea de comandos.

| Tipo de argumento | Dónde se describe |
|---|---|
| Argumentos del sistema | Consulte el tema “Argumentos del sistema” en la página 64 para obtener más información. |
| Argumentos de parámetros | Consulte el tema “Argumentos de parámetros” en la página 65 para obtener más información. |
| Argumentos de conexión del servidor | Consulte el tema “Argumentos de conexión del servidor” en la página 66 para obtener más información. |
| Argumentos de conexión de Repositorio de IBM SPSS Collaboration and Deployment Services | Consulte el tema “Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión” en la página 67 para obtener más información. |
| Argumentos de conexión de IBM SPSS Analytic Server | Consulte el tema “Argumentos de conexión de IBM SPSS Analytic Server” en la página 67 para obtener más información. |

Por ejemplo, se pueden utilizar las marcas `-server`, `-stream` y `-execute` para conectar con un servidor y, a continuación, cargar y ejecutar una ruta, de la siguiente forma:

```
modelerclient -server -hostname myserver -port 80 -username dminer  
-password 1234 -stream mystream.str -execute
```

Tenga en cuenta que al ejecutarse en una instalación cliente local, no se necesitan argumentos de conexión con el servidor.

Los valores de parámetros que contienen espacios se pueden poner entre comillas dobles, por ejemplo:

```
modelerclient -stream mystream.str -Pusername="Joe User" -execute
```

También puede ejecutar scripts y estados de IBM SPSS Modeler de esta forma, utilizando las marcas `-state` y `-script` respectivamente.

Nota: Si utiliza un parámetro estructurado en un comando, delante de las comillas debe poner barras invertidas. Así evitará que se quiten las comillas durante la interpretación de la serie.

Depuración de argumentos de línea de comandos

Para depurar una línea de comandos, utilice el comando `modelercli` para iniciar IBM SPSS Modeler con los argumentos deseados. Esto permite comprobar que los comandos se ejecutarán como se espera. También puede confirmar los valores de cualquier parámetro pasado desde la línea de comandos en el cuadro de diálogo Parámetros de sesión (menú Herramientas, Definir parámetros de sesión).

Argumentos del sistema

En la siguiente tabla se describen los argumentos del sistema disponibles para la invocación de la línea de comandos de la interfaz de usuario.

Tabla 34. Argumentos del sistema

| Argumento | Comportamiento/Descripción |
|-------------------------|--|
| @ <archivo de comandos> | El carácter @ seguido de un nombre de archivo especifica una lista de comandos. Cuando <code>modelercli</code> encuentra un argumento que comienza por @, opera en los comandos de este archivo como si hubieran estado en la línea de comandos. Consulte el tema “Combinación de varios argumentos” en la página 68 para obtener más información. |
| -directory <dir> | Define el directorio de trabajo predeterminado. En el modo local, este directorio se utiliza tanto para datos como para resultados. Ejemplo: <code>-directory c:/</code> o <code>-directory c:\\</code> |
| -server_directory <dir> | Define el directorio de servidor predeterminado para datos. El directorio de trabajo, especificado con la marca <code>-directory</code> , se utiliza para resultados. |
| -execute | Después del inicio, ejecuta cualquier ruta, estado o script que se haya cargado en el inicio. Si se carga un script además de una ruta o un estado, el script se ejecutará solo. |
| -stream <ruta> | Carga en el inicio la ruta especificada. Se pueden especificar varias rutas, pero la última se definirá como la actual. |
| -script <script> | Carga en el inicio el script autónomo especificado. Se puede especificar además de una ruta o un estado, tal y como se describe a continuación, pero sólo se puede cargar un único script en el inicio. |
| -model <modelo> | En el inicio, carga el modelo generado (archivo de formato <code>.gm</code>) especificado. |
| -state <estado> | Carga en el inicio el estado especificado guardado. |
| -project <proyecto> | Carga el proyecto especificado. Sólo se puede cargar un único proyecto en el inicio. |
| -output <resultado> | Carga en el inicio el objeto de resultados guardado (archivo de formato <code>.cou</code>). |
| -help | Muestra una lista de argumentos de la línea de comandos. Cuando se especifica esta opción, todos los demás argumentos se ignoran y se muestra la pantalla Ayuda. |
| -P <nombre>=<valor> | Se utiliza para definir un parámetro de inicio. También se puede utilizar para definir propiedades de nodos (parámetros de intervalo). |

Nota: Los directorios predeterminados también se pueden establecer en la interfaz de usuario. Para acceder a las opciones en el menú Archivo, seleccione **Definir directorio** o **Definir directorio de servidor**.

Carga de varios archivos

Desde la línea de comandos puede cargar varias rutas, estados y resultados en el inicio repitiendo el argumento relevante para cada objeto cargado. Por ejemplo, para cargar y ejecutar dos rutas llamadas `report.str` y `train.str`, utilizaría el siguiente comando:

```
modelerclient -stream report.str -stream train.str -execute
```

Carga de objetos desde el Repositorio de IBM SPSS Collaboration and Deployment Services

Dado que puede cargar determinados objetos de un archivo o desde el Repositorio de IBM SPSS Collaboration and Deployment Services (si dispone de licencia), el prefijo de nombre de archivo `spsscr:` y, si lo desea, `file:` (para objetos en disco) indica a IBM SPSS Modeler donde buscar el objeto. El prefijo funciona con las siguientes marcas:

- `-stream`
- `-script`
- `-output`
- `-model`
- `-project`

Puede utilizar el prefijo para crear un URI que especifique la ubicación del objeto, por ejemplo, `-stream "spsscr:///folder_1/scoring_stream.str"`. La presencia del prefijo `spsscr:` requiere que se especifique una conexión válida a Repositorio de IBM SPSS Collaboration and Deployment Services en el mismo comando. Así, por ejemplo, el comando completo sería:

```
modelerclient -spsscr_hostname myhost -spsscr_port 8080  
-spsscr_username myusername -spsscr_password mypassword  
-stream "spsscr:///folder_1/scoring_stream.str" -execute
```

Recuerde que en la línea de comandos *debe* utilizar un URI. El `REPOSITORY_PATH` más simple no se admite. (Sólo funciona en scripts.) Para obtener más detalles sobre los URI para objetos en el Repositorio de IBM SPSS Collaboration and Deployment Services, consulte el tema “Acceso a objetos en el Repositorio de IBM SPSS Collaboration and Deployment Services” en la página 50.

Argumentos de parámetros

Los parámetros se pueden utilizar como marcas durante la ejecución de la línea de comandos de IBM SPSS Modeler. En los argumentos de la línea de comandos, la marca `-P` se utiliza para denotar un parámetro del tipo `-P <nombre>=<valor>`.

Los parámetros pueden ser:

- **Parámetros simples** (o parámetros utilizados directamente en expresiones CLEM).
- **Parámetros de intervalo**, también denominados **propiedades de nodos**. Estos parámetros se utilizan para modificar la configuración de los nodos en la ruta. Consulte el tema “Visión general de propiedades de nodo” en la página 71 para obtener más información.
- **Parámetros de la línea de comandos**, que son parámetros utilizados para alterar la invocación de IBM SPSS Modeler.

Por ejemplo, puede proporcionar contraseñas y nombres de usuario de orígenes de datos como una marca de la línea de comandos, por ejemplo:

```
modelerclient -stream response.str -P:databasenode.datasource="{\"ORA 10gR2\", user1, mypsw,  
true}"
```

El formato es el mismo que el del parámetro `datasource` de la propiedad de nodo `databasenode`. Para obtener más información, consulte: “propiedades de databasenode” en la página 84.

Nota: Si el nodo tiene nombre, debe encerrar el nombre del nodo entre comillas dobles y utilizar una barra inclinada invertida como carácter de escape antes de las comillas. Por ejemplo, si el nodo de origen de datos del ejemplo anterior tiene el nombre *Source_ABC*, la entrada sería la siguiente:

```
modelerclient -stream response.str -P:databasenode.\"Source_ABC\".datasource="{\"ORA 10gR2\",
  user1, mypsw, true}"
```

También se requiere una barra inclinada invertida delante de las comillas que identifican un parámetro estructurado, como en el siguiente ejemplo de origen de datos de TM1:

```
clemb -server -hostname 9.115.21.169 -port 28053 -username administrator
  -execute -stream C:\Share\TM1_Script.str -P:tmlimport.pm_host="http://9.115.21.163:9510/pmhub/pm"
  -P:tmlimport.tml_connection={\"SData\", \"\", \"admin\", \"apple\"}
  -P:tmlimport.selected_view={\"SalesPriorCube\", \"salesmargin%\"}
```

Argumentos de conexión del servidor

La marca `-server` indica a IBM SPSS Modeler que debe conectar con un servidor público, y las marcas `-hostname`, `-use_ssl`, `-port`, `-username`, `-password` y `-domain` se utilizan para indicar a IBM SPSS Modeler cómo conectar con el servidor público. Si no se especifica ningún argumento `-server`, se utilizará el servidor predeterminado o local.

Ejemplos

Para conectarse con un servidor público:

```
modelerclient -server -hostname myserver -port 80 -username dminer
  -password 1234 -stream mystream.str -execute
```

Para conectarse con un clúster de servidores:

```
modelerclient -server -cluster "QA Machines" \
  -spsscr_hostname pes_host -spsscr_port 8080 \
  -spsscr_username asmith -spsscr_epassword xyz
```

Tenga en cuenta que para conectarse a un clúster de servidores necesita Coordinator of Processes a través de IBM SPSS Collaboration and Deployment Services, de modo que debe utilizar el argumento `-cluster` junto con las opciones de conexión de repositorio (`spsscr_*`). Consulte el tema “Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión” en la página 67 para obtener más información.

Tabla 35. Argumentos de conexión del servidor.

| Argumento | Comportamiento/Descripción |
|---------------------------------------|---|
| <code>-server</code> | Ejecuta IBM SPSS Modeler en el modo servidor, conectando con un servidor público utilizando las marcas <code>-hostname</code> , <code>-port</code> , <code>-username</code> , <code>-password</code> y <code>-domain</code> . |
| <code>-hostname <nombre></code> | Nombre de host del equipo servidor. Disponible en el modo servidor solamente. |
| <code>-use_ssl</code> | Especifica que la conexión debería utilizar SSL (secure socket layer). La marca es opcional, el parámetro predeterminado <i>no</i> utiliza SSL. |
| <code>-port <número></code> | Número de puerto del servidor especificado. Disponible en el modo servidor solamente. |
| <code>-cluster <nombre></code> | Especifica una conexión a un clúster de servidores en lugar de un servidor especificado; este argumento es una alternativa a los argumentos <code>hostname</code> , <code>port</code> y <code>use_ssl</code> . El nombre es el del clúster o un URI exclusivo que identifica el clúster en el Repositorio de IBM SPSS Collaboration and Deployment Services. Coordinator of Processes gestiona el clúster de servidores a través de IBM SPSS Collaboration and Deployment Services. Consulte el tema “Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión” en la página 67 para obtener más información. |

Tabla 35. Argumentos de conexión del servidor (continuación).

| Argumento | Comportamiento/Descripción |
|--|--|
| -username <nombre> | Nombre de usuario con el que iniciar sesión en el servidor. Disponible en el modo servidor solamente. |
| -password <contraseña> | Contraseña con la que iniciar sesión en el servidor. Disponible en el modo servidor solamente. Nota: Si el argumento -password no se utiliza, se le pediría una contraseña. |
| -epassword <cadena de contraseña codificada> | Contraseña codificada con la que iniciar sesión en el servidor. Disponible en el modo servidor solamente. Nota: Se puede generar una contraseña codificada desde el menú Herramientas de la aplicación IBM SPSS Modeler. |
| -domain <nombre> | Dominio utilizado para iniciar sesión en el servidor. Disponible en el modo servidor solamente. |
| -P <nombre>=<valor> | Se utiliza para definir un parámetro de inicio. También se puede utilizar para definir propiedades de nodos (parámetros de intervalo). |

Repositorio de IBM SPSS Collaboration and Deployment Services Argumentos de conexión

Si desea almacenar o recuperar objetos de IBM SPSS Collaboration and Deployment Services a través de la línea de comandos, debe especificar una conexión válida con Repositorio de IBM SPSS Collaboration and Deployment Services. Por ejemplo:

```
modelerclient -spsscr_hostname myhost -spsscr_port 8080
-spsscr_username myusername -spsscr_password mypassword
-stream "spsscr:///folder_1/scoring_stream.str" -execute
```

La siguiente tabla enumera los argumentos que pueden utilizarse para configurar la conexión.

Tabla 36. Argumentos de conexión de Repositorio de IBM SPSS Collaboration and Deployment Services

| Argumento | Comportamiento/Descripción |
|---|--|
| -spsscr_hostname <nombre del host o dirección IP> | El nombre del host o la dirección IP del servidor en que se ha instalado Repositorio de IBM SPSS Collaboration and Deployment Services. |
| -spsscr_port <número> | Número de puerto en el que el Repositorio de IBM SPSS Collaboration and Deployment Services acepta las conexiones (normalmente 8080 de forma predeterminada). |
| -spsscr_use_ssl | Especifica que la conexión debería utilizar SSL (secure socket layer). La marca es opcional, el parámetro predeterminado <i>no</i> utiliza SSL. |
| -spsscr_username <nombre> | Nombre de usuario con el que iniciar sesión en el Repositorio de IBM SPSS Collaboration and Deployment Services. |
| -spsscr_password <contraseña> | Contraseña con la que iniciar sesión en el Repositorio de IBM SPSS Collaboration and Deployment Services. |
| -spsscr_epassword <contraseña codificada> | Contraseña cifrada con la que iniciar sesión en el Repositorio de IBM SPSS Collaboration and Deployment Services. |
| -spsscr_providename <nombre> | El proveedor de autenticación utilizado para iniciar la sesión en el Repositorio de IBM SPSS Collaboration and Deployment Services (Active Directory o LDAP). Esto no resulta necesario si se utiliza el proveedor nativo (repositorio local). |

Argumentos de conexión de IBM SPSS Analytic Server

Si desea almacenar o recuperar objetos de IBM SPSS Analytic Server mediante la línea de comandos, debe indicar una conexión válida con IBM SPSS Analytic Server.

Nota: La ubicación predeterminada de Analytic Server se obtiene de SPSS Modeler Server. Los usuarios también pueden definir sus propias conexiones de Analytic Server a través de **Herramientas > Conexiones de Analytic Server**.

La siguiente tabla enumera los argumentos que pueden utilizarse para configurar la conexión.

Tabla 37. Argumentos de conexión de IBM SPSS Analytic Server

| Argumento | Comportamiento/Descripción |
|-----------------------------|---|
| -analytic_server_username | Nombre de usuario con el que iniciar sesión en IBM SPSS Analytic Server. |
| -analytic_server_password | La contraseña con la que se inicia sesión en IBM SPSS Analytic Server. |
| -analytic_server_epassword | La contraseña codificada con la que se inicia sesión en IBM SPSS Analytic Server. |
| -analytic_server_credential | Las credenciales utilizadas para iniciar sesión en IBM SPSS Analytic Server. |

Combinación de varios argumentos

Es posible combinar varios argumentos en un único archivo de comandos especificado en la invocación utilizando el símbolo @ seguido del nombre de archivo. De este modo podrá acortar la invocación de la línea de comandos y superar cualquier limitación del sistema operativo en la longitud del comando. Por ejemplo, el siguiente comando de inicio utiliza todos los argumentos especificados en el archivo de referencia <nombre de archivo de comandos>.

```
modelerclient @<commandFileName>
```

Ponga el nombre del archivo y la ruta del archivo de comandos entre comillas si hay que incluir espacios, de la siguiente forma:

```
modelerclient @ "C:\Program Files\IBM\SPSS\Modeler\mn\scripts\my_command_file.txt"
```

El archivo de comandos puede contener todos los argumentos especificados previamente a nivel individual en el inicio. Por ejemplo:

```
-stream report.str
-Porder.full_filename=APR_orders.dat
-Preport.filename=APR_report.txt
-execute
```

Cuando escriba y referencie archivos de comandos, asegúrese de cumplir estas restricciones:

- Utilice sólo un comando por línea.
- No incruste un argumento @archivo de comandos en un archivo de comandos.

Capítulo 7. Referencia de propiedades

Visión general de referencia de propiedades

Puede especificar una serie de propiedades distintas para nodos, rutas, proyectos y supernodos. Algunas propiedades son comunes a todos los nodos, como el nombre, la anotación y la información sobre herramientas, mientras que otras son específicas para determinados tipos de nodos. Otras propiedades hacen referencia a operaciones de rutas de alto nivel, como el comportamiento del Supernodo o el almacenamiento en caché. Se puede acceder a las propiedades a través de la interfaz de usuario estándar (por ejemplo, al abrir un cuadro de diálogo para editar opciones para un nodo) y se pueden utilizar también de varias otras formas.

- Las propiedades se pueden modificar a través de los scripts, como se describe en esta sección. Si desea obtener más información, consulte “Sintaxis para propiedades”.
- Las propiedades de los nodos se pueden utilizar en los parámetros de Supernodo.
- Asimismo, las propiedades de los nodos se pueden utilizar como parte de una opción de línea de comandos (mediante la marca -P) al iniciar IBM SPSS Modeler.

En el contexto de los scripts de IBM SPSS Modeler, las propiedades de nodos y rutas se suelen llamar **parámetros de intervalo**. En esta guía, se denominan propiedades de nodos y rutas.

Si desea obtener más información sobre el lenguaje de scripts, consulte Lenguaje de scripts.

Sintaxis para propiedades

Las propiedades se pueden establecer con la sintaxis siguiente

```
OBJECT.setPropertyValue(PROPERTY, VALUE)
```

o:

```
OBJECT.setKeyedPropertyValue(PROPERTY, KEY, VALUE)
```

El valor de propiedades se puede recuperar usando la sintaxis siguiente:

```
VARIABLE = OBJECT.getPropertyValue(PROPERTY)
```

o:

```
VARIABLE = OBJECT.getKeyedPropertyValue(PROPERTY, KEY)
```

donde OBJECT es un nodo o salida, PROPERTY es el nombre de la propiedad de nodo al que la expresión se refiere, y KEY es el valor de la clave para las propiedades clave. Por ejemplo, la siguiente sintaxis se utiliza para buscar el nodo de filtro y, a continuación, establecer el valor predeterminado para incluir todos los campos y filtrar el campo Age en los datos en sentido descendente:

```
filternode = modeler.script.stream().findByType("filter", None)
filternode.setPropertyValue("default_include", True)
filternode.setKeyedPropertyValue("include", "Age", False)
```

Todos los nodos utilizados en IBM SPSS Modeler pueden encontrarse utilizando la función `findByType(TYPE, LABEL)` de la ruta. Al menos debe especificarse TYPE o LABEL.

Propiedades estructuradas

Hay dos formas en las que los scripts utilizan propiedades estructuradas para mejorar la claridad durante el análisis:

- Otorgando estructura a los nombres de las propiedades para los nodos complejos, como Tipo, Filtro o Equilibrar.

- Proporcionando un formato para especificar varias propiedades a la vez.

Estructuración para las interfaces complejas

Los scripts para los nodos con tablas y otras interfaces complejas, como, por ejemplo, los nodos Tipo, Filtro o Equilibrar, deben seguir una estructura determinada para realizar el análisis correctamente. Estas propiedades necesitan un nombre más complejo que el de un solo identificador. Este nombre se denomina clave. Por ejemplo, en un nodo Filtrar, cada campo disponible (en la parte superior) se activa o desactiva. Para poder consultar esta información, el nodo Filtrar almacena un elemento de información por campo (independientemente de que el campo sea verdadero o falso). Esta propiedad debe tener (o se le ha dado) el valor True o False. Supongamos que un nodo Filtrar denominado `minodo` tiene (en la parte superior) un campo denominado Edad. Para desactivar esto, establezca la propiedad `include`, con la clave `Age`, en el valor `False`, del modo siguiente:

```
mynode.setKeyedPropertyValue("include", "Age", False)
```

Estructuración para definir varias propiedades

Si hay muchos nodos, puede asignar más de una propiedad de nodo o ruta al mismo tiempo. Esto se denomina **comando de conjunto múltiple** o **bloque de conjuntos**.

En algunos casos, una propiedad estructurada puede ser bastante compleja. A continuación se muestra un ejemplo:

```
sortnode.setPropertyValue("keys", [{"K", "Descending"}, {"Age", "Ascending"}, {"Na", "Descending"}])
```

Otra ventaja de las propiedades estructuradas es la capacidad de definir varias propiedades en un nodo antes de que éste sea estable. De forma predeterminada, un conjunto múltiple define todas las propiedades del bloque antes de realizar una acción basada en una configuración de propiedades individuales. Por ejemplo, al definir un nodo Archivo fijo, el uso de dos pasos para definir las propiedades del campo daría lugar a errores porque el nodo no será constante hasta que las dos configuraciones sean válidas. La definición de las propiedades como un conjunto múltiple salva este problema al definir ambas propiedades antes de actualizar el modelo de datos.

Abreviaturas

Las abreviaturas estándar se utilizan en la sintaxis para las propiedades de nodos. El aprendizaje de las abreviaturas le ayudará en la creación de scripts.

Tabla 38. Abreviaturas estándar utilizadas en toda la sintaxis

| Abreviatura | Significado |
|-------------|--|
| abs | Valor absoluto |
| lon | Longitud |
| min | Mínimo |
| máx | Máximo |
| correl | Correlation |
| covar | Covariance |
| núm | Número o numérico |
| pct | Porcentaje |
| transp | Transparencia |
| xval | Validación cruzada |
| var | Varianza o variable (en nodos de origen) |

Ejemplos de las propiedades node y stream

Las propiedades de nodos y rutas se pueden utilizar de varias formas con IBM SPSS Modeler. Normalmente se utilizan como parte de un script, bien un **script autónomo**, utilizado para automatizar rutas u operaciones o un **script de ruta**, utilizado para automatizar procesos en una sola ruta. Los parámetros de nodo se pueden especificar también utilizando las propiedades para los nodos del Supernodo. En el nivel más básico, las propiedades se pueden utilizar también como una opción de línea de comandos para iniciar IBM SPSS Modeler. Si utiliza el argumento -p como parte de la invocación de la línea de comandos, podrá utilizar una propiedad de ruta para cambiar una configuración de la ruta.

Tabla 39. Ejemplos de las propiedades node y stream

| Propiedad | Significado |
|-----------------------|---|
| s.max_size | Hace referencia a la propiedad max_size del nodo denominado s. |
| s:samplenode.max_size | Hace referencia a la propiedad max_size del nodo denominado s que debe ser un nodo Muestrear. |
| :samplenode.max_size | Hace referencia a la propiedad max_size del nodo Muestrear de la ruta actual (debe haber sólo un nodo Muestrear). |
| s:sample.max_size | Hace referencia a la propiedad max_size del nodo denominado s que debe ser un nodo Muestrear. |
| t.direction.Age | Hace referencia al rol del campo <i>Edad</i> del nodo Tipo t. |
| :.max_size | *** NO ES LEGAL *** Debe especificar el nombre o el tipo de nodo. |

El ejemplo s:sample.max_size muestra que no es necesario deletrear los tipos de nodos al completo.

El ejemplo t.direction.Age muestra que algunos nombres de intervalo se pueden estructurar por sí mismos, en aquellos casos en que los atributos de un nodo sean más complejos que los intervalos individuales con valores individuales. Dichos intervalos se denominan **estructurados** o **complejos**.

Visión general de propiedades de nodo

Cada tipo de nodo tiene su propio conjunto de propiedades legales y cada propiedad tiene un tipo. Este tipo puede ser un tipo general, número, marca o cadena, en cuyo caso, las configuraciones de la propiedad se forzarán en el tipo correcto. Surgirá un error en caso de que no se puedan forzar. También se puede dar el caso de que la referencia de la propiedad pueda especificar el rango de valores legales como Discard, PairAndDiscard e IncludeAsText, en cuyo caso se producirá un error si se utiliza otro valor. Las propiedades de marcas se deben leer o definir mediante los valores True y False. (Las variaciones que contengan Off, OFF, off, No, NO, no, n, N, f, F, false, False, FALSE o 0 también se reconocen al configurar los valores, pero pueden provocar errores al leer los valores de propiedad en algunos casos. El resto de valores se consideran verdaderos. El uso de verdadero y falso de forma consistente evitará confusiones). En las tablas de referencia de esta guía, las propiedades estructuradas se indican como tales en la columna **Descripción de la propiedad** y se proporcionan los formatos de uso.

Propiedades de nodos comunes

Existen varias propiedades que son comunes a todos los nodos (incluidos los Supernodos) en IBM SPSS Modeler.

Tabla 40. Propiedades comunes de nodos.

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|---------------|-----------------------------|
| use_custom_name | tag | |

Tabla 40. Propiedades comunes de nodos (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|---|---|
| name | <i>cadena</i> | Propiedad de sólo lectura que lee el nombre (automático o personalizado) para un nodo del lienzo. |
| custom_name | <i>cadena</i> | Especifica un nombre personalizado para el nodo. |
| tooltip | <i>cadena</i> | |
| annotation | <i>cadena</i> | |
| palabras clave | <i>cadena</i> | Intervalo estructurado que especifica una lista de palabras clave asociadas al objeto (por ejemplo, ["PalabraClave1" "PalabraClave2"]). |
| cache_enabled | <i>tag</i> | |
| tipo_nodo | source_supernode process_supernode terminal_supernode todos los nombres de nodos tal como se especifican para script | Propiedad de sólo lectura utilizada para hacer referencia a un nodo por tipo. Por ejemplo, en lugar de hacer referencia al nodo sólo por el nombre, como ingresos_reales, puede también especificar el tipo, como userInputnode o filternode. |

Las propiedades específicas del Supernodo se tratan aparte como con los demás nodos. Consulte el tema Capítulo 20, "Propiedades de supernodo", en la página 351 para obtener más información.

Capítulo 8. Propiedades de ruta

Los scripts pueden controlar una serie de propiedades de la ruta. Para hacer referencia a propiedades de ruta, debe establecer el método de ejecución para que utilice scripts:

```
stream = modeler.script.stream()
stream.setPropertyValue("execute_method", "Script")
```

Ejemplo

La propiedad de nodo se utiliza para hacer referencia a los nodos en la ruta actual. El siguiente script de ruta muestra un ejemplo:

```
stream = modeler.script.stream()
annotation = stream.getPropertyValue("annotation")

annotation = annotation + "\n\nEsta ruta se llama \"" + stream.getLabel() + "\" y
contiene los nodos siguientes:\n"

for node in stream.iterator():
    annotation = annotation + "\n" + node.getTypeName() + " nodo denominado \"" + node.getLabel()
    + "\"

stream.setPropertyValue("annotation", annotation)
```

El ejemplo anterior utiliza la propiedad node para crear una lista con todos los nodos de la ruta y escribir dicha lista en las anotaciones. La anotación generada tendrá el siguiente aspecto:

Esta ruta se llama "druglearn" y contiene los siguientes nodos:

```
type node called "Define Types"
derive node called "Na_to_K"
variablefile node called "DRUG1n"
neuralnetwork node called "Drug"
c50 node called "Drug"
filter node called "Discard Fields"
```

Las propiedades de la ruta se describen en la tabla siguiente.

Tabla 41. Propiedades de ruta.

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|------------------|-----------------------------|
| execute_method | Normal Script | |

Tabla 41. Propiedades de ruta (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|---------------------------|---|-----------------------------|
| date_format | "DDMAA" "MMDDYY" "AAMDD" "YYMMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-AAAA" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.AAAA" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/AAAA" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | |
| date_baseline | number | |
| date_2digit_baseline | number | |
| time_format | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | |
| time_rollover | tag | |
| import_datetime_as_string | tag | |
| decimal_places | number | |
| decimal_symbol | Predeterminado Period Comma | |
| angles_in_radians | tag | |
| use_max_set_size | tag | |
| max_set_size | number | |
| ruleset_evaluation | Voting FirstHit | |

Tabla 41. Propiedades de ruta (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|--|--|--|
| refresh_source_nodes | tag | Se utiliza para actualizar los nodos de origen de forma automática al realizar la ejecución de la ruta. |
| script | cadena | |
| annotation | cadena | |
| name | cadena | Nota: Esta propiedad es de sólo lectura. Si desea cambiar el nombre de una ruta, debe guardarla con un nombre diferente. |
| parameters | | Utilice esta propiedad para actualizar los parámetros de ruta desde dentro de un script autónomo. |
| nodos | | Consulte la información detallada que se muestra a continuación. |
| codificación | SystemDefault "UTF-8" | |
| stream_rewriting | booleano | |
| stream_rewriting_maximise_sql | booleano | |
| stream_rewriting_optimise_clem_ejecución | booleano | |
| stream_rewriting_optimise_syntax_ejecución | booleano | |
| enable_parallelism | booleano | |
| sql_generation | booleano | |
| database_caching | booleano | |
| sql_logging | booleano | |
| sql_generation_logging | booleano | |
| sql_log_native | booleano | |
| sql_log_prettyprint | booleano | |
| record_count_suppress_input | booleano | |
| record_count_feedback_interval | entero | |
| use_stream_auto_create_node_valores | booleano | Si es true, se utilizan los valores específicos de la ruta, de lo contrario, se utilizan las preferencias de usuario. |
| create_model_applier_for_new_modelos | booleano | Si es true, cuando un constructor de modelos crea un modelo nuevo y no tiene enlaces de actualización activos, se añade un nuevo aplicador de modelos. Nota: Si utiliza IBM SPSS Modeler Batch versión 15, debe añadir de forma explícita el aplicador de modelos dentro de su script. |
| create_model_applier_update_links | createEnabled createDisabled doNotCreate | Define el tipo de enlace creado cuando se añade automáticamente un nodo aplicador de modelos. |

Tabla 41. Propiedades de ruta (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|----------------------------------|--|---|
| create_source_node_from_builders | booleano | Si es true, cuando un constructor de modelos crea un resultado de origen nuevo y no tiene enlaces de actualización activos, se añade un nuevo nodo de origen. |
| create_source_node_update_links | createEnabled createDisabled doNotCreate | Define el tipo de enlace creado cuando se añade automáticamente un nodo de origen. |
| has_coordinate_system | booleano | Si es verdadero, aplica un sistema de coordenadas a la ruta completa. |
| coordinate_system | cadena | El nombre del sistema de coordenadas proyectadas seleccionado. |
| deployment_area | ModelRefresh Puntuación Ninguno | Selecciona cómo desea desplegar la ruta. Si este valor está establecido en Ninguno, no se utiliza ninguna otra entrada de despliegue. |
| scoring_terminal_node_id | cadena | Elija la rama de puntuación en la ruta. Puede ser cualquier terminal de la ruta. |
| scoring_node_id | cadena | Elija el nugget de la rama de puntuación. |
| model_build_node_id | cadena | Elija el nodo de modelado de la ruta. |

Capítulo 9. Propiedades de nodos de origen

Propiedades comunes de nodos de origen

Las propiedades comunes a todos los nodos de origen se enumeran a continuación, con información sobre nodos específicos en los temas siguientes.

Ejemplo 1

```
varfilenode = modeler.script.stream().create("variablefile", "Var. File")
varfilenode.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
varfilenode.setKeyedPropertyValue("check", "Age", "None")
varfilenode.setKeyedPropertyValue("values", "Age", [1, 100])
varfilenode.setKeyedPropertyValue("type", "Age", "Range")
varfilenode.setKeyedPropertyValue("direction", "Age", "Input")
```

Ejemplo 2

Este script presupone que el archivo de datos especificado contiene un campo Region que representa una cadena de varias líneas.

```
from modeler.api import StorageType
from modeler.api import MeasureType

# Cree un nodo Archivo de variables que lee el conjunto de datos que contiene
# el campo "Region"
varfilenode = modeler.script.stream().create("variablefile", "My Geo Data")
varfilenode.setPropertyValue("full_filename", "C:/mydata/mygeodata.csv")
varfilenode.setPropertyValue("treat_square_brackets_as_lists", True)

# Sustituya el tipo de almacenamiento para que sea una lista...
varfilenode.setKeyedPropertyValue("custom_storage_type", "Region", StorageType.LIST)
# ...y especifique el tipo de los valores de la lista y la profundidad de lista
varfilenode.setKeyedPropertyValue("custom_list_storage_type", "Region", StorageType.INTEGER)
varfilenode.setKeyedPropertyValue("custom_list_depth", "Region", 2)

# Cambie ahora la medición para identificar el campo como un valor geoespacial...
varfilenode.setKeyedPropertyValue("measure_type", "Region", MeasureType.GEOSPATIAL)
# ...y, finalmente, especifique la información necesaria sobre el tipo
# específico de objeto geoespacial
varfilenode.setKeyedPropertyValue("geo_type", "Region", "MultiLineString")
varfilenode.setKeyedPropertyValue("geo_coordinates", "Region", "2D")
varfilenode.setKeyedPropertyValue("has_coordinate_system", "Region", True)
varfilenode.setKeyedPropertyValue("coordinate_system", "Region",
    "ETRS_1989_EPSG_Arctic_zone_5-47")
```

Tabla 42. Propiedades comunes de nodos de origen.

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|--|---|
| dirección | Input Destino Both Ninguno Partition Split Frequency RecordID | Propiedad con clave para los roles de los campos. Formato de uso: NODO.direction.NOMBRECAMPO Nota: Los valores In y Out han quedado en desuso. Es posible que ya no puedan utilizarse en futuras versiones. |

Tabla 42. Propiedades comunes de nodos de origen (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|---|---|
| type | Range Flag Set Sin tipo Discrete Conjunto ordenado Predeterminado | Tipo de campo. Si se establece esta propiedad como <i>Default</i> , se borrará cualquier configuración del parámetro <i>values</i> y si <i>value_mode</i> tiene el valor <i>Specify</i> , se restablecerá a <i>Read</i> . Si <i>value_mode</i> se establece en <i>Pass</i> o <i>Read</i> , la configuración de <i>type</i> no le afectará. Formato de uso: NODO.type.NOMBRECAMPO |
| storage | Desconocido Cadena Entero Real Hora Fecha Marca de tiempo | Propiedad con clave de solamente lectura para el tipo de almacenamiento de campos. Formato de uso: NODO.storage.NOMBRECAMPO |
| check | Ninguno Nullify Coerce Descartar Warn Abort | Propiedad con clave para la comprobación del rango y el tipo de campo. Formato de uso: NODO.check.NOMBRECAMPO |
| values | [value value] | Para un campo continuo (rango), el primer valor es el mínimo y el último valor es el máximo. Para campos nominales (conjunto), especifique todos los valores. Para los campos marca, el primer valor representa <i>falso</i> y el último, <i>verdadero</i> . La configuración de esta propiedad establece de forma automática la propiedad <i>value_mode</i> en <i>Specify</i> . El almacenamiento se determina en función del primer valor de la lista, por ejemplo, si el primer valor es una <i>cadena</i> , el almacenamiento se establece en <i>Cadena</i> . Formato de uso: NODO.values.NOMBRECAMPO |
| value_mode | Leer Pasar Leer+ Actual Especifica | Determina la forma en que se han establecido los valores para un campo en la siguiente lectura de datos. Formato de uso: NODO.value_mode.NOMBRECAMPO Tenga en cuenta que no puede establecer esta propiedad directamente en <i>Specify</i> . Para utilizar valores específicos, establezca la propiedad <i>values</i> . |
| default_value_mode | Leer Pasar | Especifica el método predeterminado para configurar los valores de todos los campos. Formato de uso: NODO.default_value_mode Esta configuración puede anularse para determinados campos mediante la propiedad <i>value_mode</i> . |

Tabla 42. Propiedades comunes de nodos de origen (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|-------------------|---|
| extend_values | tag | Se aplica cuando value_mode se establece en <i>Read</i> . Establézcala en <i>T</i> para añadir nuevos valores de lectura a los valores existentes del campo. Establézcala en <i>F</i> para descartar los valores existentes y favorecer a los nuevos valores de lectura. Formato de uso: NODO.extend_values.NOMBRECAMPO |
| value_labels | cadena | Se utiliza para especificar una etiqueta de valor. Tenga en cuenta que estos valores se deben especificar primero. |
| enable_missing | tag | Cuando está definida como <i>T</i> , activa el seguimiento de los valores perdidos para el campo. Formato de uso: NODO.enable_missing.NOMBRECAMPO |
| missing_values | [value value ...] | Especifica los valores de datos que denotan los datos perdidos. Formato de uso: NODO.missing_values.NOMBRECAMPO |
| range_missing | tag | Cuando esta propiedad se establece como <i>T</i> , especifica si se define un rango de valores perdidos (en blanco) para un campo. Formato de uso: NODO.range_missing.NOMBRECAMPO |
| missing_lower | cadena | Si range_missing es verdadero, especifica el límite inferior del rango de valores perdidos. Formato de uso: NODO.missing_lower.NOMBRECAMPO |
| missing_upper | cadena | Si range_missing es verdadero, especifica el límite superior del rango de valores perdidos. Formato de uso: NODO.missing_upper.NOMBRECAMPO |
| null_missing | tag | Cuando esta propiedad se establece en <i>T</i> , los valores nulos (valores no definidos que se muestran como \$null\$ en el software) se consideran valores perdidos. Formato de uso: NODO.null_missing.NOMBRECAMPO |
| whitespace_missing | tag | Cuando esta propiedad está definida como <i>T</i> , los valores que solamente contienen un espacio en blanco (espacios, tabulaciones y líneas nuevas) se consideran valores perdidos. Formato de uso: NODO.whitespace_missing.NOMBRECAMPO |
| description | cadena | Se utiliza para especificar la descripción o etiqueta de un campo. |

Tabla 42. Propiedades comunes de nodos de origen (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|------------------------|---|--|
| default_include | tag | Propiedad con clave para especificar si el comportamiento predeterminado es para pasar o filtrar los campos: NODO.default_include Ejemplo: set minodo:filternode.default_include = false |
| include | tag | Propiedad con clave que se utiliza para determinar si los campos individuales se han incluido o se han filtrado: NODO.include.NOMBRECAMPO. |
| new_name | cadena | |
| measure_type | Range / MeasureType.RANGE Discrete / MeasureType.DISCRETE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS Collection / MeasureType.COLLECTION Geospatial / MeasureType.GEOSPATIAL | Esta propiedad con clave es similar a type en cuanto a que puede utilizarse para definir la medición asociada al campo. La diferencia es que, en los scripts Python, la función de establecimiento puede pasar también uno de los valores MeasureType, mientras que la función de obtención siempre devolverá los valores MeasureType. |
| collection_measure | Range / MeasureType.RANGE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS | Para campos de recopilación (listas con profundidad 0), esta propiedad con clave define el tipo de medición asociado con los valores subyacentes. |
| geo_type | Point Multipunto Cadena lineal Cadena multilínea Polígono Multipolígono | En campos geoespaciales, esta propiedad con clave define el tipo del objeto geoespacial representado por este campo. Debería ser coherente con la profundidad de lista de los valores. |
| has_coordinate_system | booleano | En campos geoespaciales, esta propiedad define si este campo tiene un sistema de coordenadas |
| coordinate_system | cadena | En campos geoespaciales, esta propiedad con clave define el sistema de coordenadas para este campo. |

Tabla 42. Propiedades comunes de nodos de origen (continuación).

| Nombre de la propiedad | Tipo de datos | Descripción de la propiedad |
|--------------------------|--|---|
| custom_storage_type | Unknown / MeasureType.UNKNOWN String / MeasureType.STRING Integer / MeasureType.INTEGER Real / MeasureType.REAL Time / MeasureType.TIME Date / MeasureType.DATE Timestamp / MeasureType.TIMESTAMP List / MeasureType.LIST | Esta propiedad con clave es similar a custom_storage en cuanto a que puede utilizarse para definir el almacenamiento de alteración temporal para el campo. La diferencia es que, en los scripts Python, la función de establecimiento puede pasar también uno de los valores StorageType, mientras que la función de obtención siempre devolverá los valores StorageType. |
| custom_list_storage_type | String / MeasureType.STRING Integer / MeasureType.INTEGER Real / MeasureType.REAL Time / MeasureType.TIME Date / MeasureType.DATE Timestamp / MeasureType.TIMESTAMP | Para campos de lista, esta propiedad con clave especifica el tipo de almacenamiento de los valores subyacentes. |
| custom_list_depth | entero | Para campos de lista, esta propiedad con clave especifica la profundidad del campo. |
| max_list_length | entero | Solo está disponible para datos con un nivel de medición de <i>Geoespacial</i> o <i>Colección</i> . Establezca la longitud máxima de la lista especificando el número de elementos que puede contener la lista. |
| max_string_length | entero | Solo está disponible para datos <i>sin tipos</i> y se utiliza al generar SQL para crear una tabla. Entre el valor de la serie más larga en los datos; esto genera una columna en la tabla que es lo suficientemente grande para contener la serie. |

Propiedades de asimport

El origen de Analytic Server permite ejecutar una ruta en un sistema de archivos distribuido de Hadoop (HDFS en sus siglas inglesas).

Ejemplo

```
node.setPropertyValue("use_default_as", False)
node.setPropertyValue("connection",
["false","9.119.141.141","9080","analyticserver","ibm","admin","admin","false","","",""])
```

Tabla 43. Propiedades de asimport.

| Propiedades de asimport | Tipo de datos | Descripción de la propiedad |
|-------------------------|---------------|---|
| data_source | cadena | Nombre del origen de datos. |
| use_default_as | booleano | Si se establece en True, utiliza la conexión de Analytic Server predeterminada configurada en el archivo options.cfg de servidor. Si se establece en False, utiliza la conexión de este nodo. |

Tabla 43. Propiedades de asimport (continuación).

| Propiedades de asimport | Tipo de datos | Descripción de la propiedad |
|-------------------------|---|---|
| connection | ["string","string","string", "string","string","string","string", "string","string","string", "string", ,"string"] | Una propiedad de la lista que contiene los detalles de conexión de Analytic Server. El formato es: ["is_secure_connect", "server_url", "server_port", "context_root", "consumer", "user_name", "password", "use-kerberos-auth", "kerberos-krb5-config-file-path", "kerberos-jaas-config-file-path", "kerberos-krb5-service-principal-name", "enable-kerberos-debug"] Donde: is_secure_connect: indica si se utiliza la conexión segura y es true o false. use-kerberos-auth: indica si se utiliza autenticación Kerberos y es true o false. enable-kerberos-debug: indica si se utiliza el modo de depuración de la autenticación Kerberos y es true o false. |

Propiedades del nodo cognosimport



El nodo de origen de IBM Cognos importa datos desde las bases de datos de Cognos Analytics.

Ejemplo

```
node = stream.create("cognosimport", "My node")
node.setPropertyValue("cognos_connection", ["http://mycogsrv1:9300/p2pd/servlet/dispatch",
True, "", "", ""])
node.setPropertyValue("cognos_package_name", "/Public Folders/GOSALES")
node.setPropertyValue("cognos_items", ["[GreatOutdoors].[BRANCH].[BRANCH_CODE]",
"[GreatOutdoors].[BRANCH].[COUNTRY_CODE]"])
```

Tabla 44. propiedades del nodo cognosimport.

| Propiedades del nodo cognosimport | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|------------------|--|
| mode | Datos Informe | Especifica si se importarán los datos de Cognos (valor predeterminado) o informes. |

Tabla 44. propiedades del nodo cognosimport (continuación).

| Propiedades del nodo cognosimport | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|---|--|
| cognos_connection | ["string",flag,"string", "string", "string"] | <p>Una propiedad de la lista que contiene los detalles de conexión para el servidor de Cognos. El formato es: ["Cognos_server_URL", login_mode, "namespace", "username", "password"]</p> <p>donde: Cognos_server_URL es la URL del servidor de Cognos que contiene el origen. login_mode indica si se utiliza el inicio de sesión anónimo, y es true o false; si se establece en true, los campos siguientes deben establecerse en "" . namespace especifica el proveedor de autenticación de seguridad utilizado para registrarse en el servidor. username y password son los utilizados para registrarse en el servidor de Cognos.</p> <p>En lugar de login_mode, también hay disponibles las modalidades siguientes:</p> <ul style="list-style-type: none"> • anonymousMode. Por ejemplo: ['Cognos_server_url', 'anonymousMode', "namespace", "username", "password"] • credentialMode. Por ejemplo: ['Cognos_server_url', 'credentialMode', "namespace", "username", "password"] • storedCredentialMode. Por ejemplo: ['Cognos_server_url', 'storedCredentialMode', "stored_credential_name"] <p>Donde stored_credential_name es el nombre de una credencial de Cognos del repositorio.</p> |
| cognos_package_name | cadena | <p>La ruta y el nombre del paquete de Cognos desde el que está importando objetos de datos, por ejemplo: /Public Folders/GOSALES Nota: solo es válida la barra diagonal.</p> |
| cognos_items | ["campo", "campo", ... , "campo"] | <p>El nombre de uno o más objetos de datos que van a importarse. El formato de <i>campo</i> es [espacio de nombre].[asunto de consulta].[elemento de consulta]</p> |
| cognos_filters | campo | <p>El nombre de uno o más filtros que van a aplicarse antes de importar datos.</p> |

Tabla 44. propiedades del nodo cognosimport (continuación).

| Propiedades del nodo cognosimport | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|---------------|--|
| cognos_data_parameters | lista | Valores de parámetros de solicitud de datos. Los pares nombre-valor van encerrados entre corchetes, los pares múltiples están separados por comas y toda la cadena está encerrada entre corchetes. Formato: [["param1", "valor"],...["paramN", "valor"]] |
| cognos_report_directory | campo | La ruta de Cognos de una carpeta o paquete de la que importar informes, por ejemplo: /Public Folders/GOSALES Nota: solo es válida la barra diagonal. |
| cognos_report_name | campo | La ruta y nombre dentro de la ubicación del informe de un informe que se ha de importar. |
| cognos_report_parameters | lista | Valores de parámetros de informe. Los pares nombre-valor van encerrados entre corchetes, los pares múltiples están separados por comas y toda la cadena está encerrada entre corchetes. Formato: [["param1", "valor"],...["paramN", "valor"]] |

propiedades de databasenode



El nodo Base de datos se puede utilizar para importar datos desde otros paquetes mediante ODBC (del inglés, Open Database Connectivity), incluidos Microsoft SQL Server, DB2, Oracle, etc.

Ejemplo

```
import modeler.api
stream = modeler.script.stream()
nnode = stream.create("database", "My node")
node.setPropertyValue("mode", "Table")
node.setPropertyValue("query", "SELECT * FROM drug1n")
node.setPropertyValue("datasource", "Drug1n_db")
node.setPropertyValue("username", "spss")
node.setPropertyValue("password", "spss")
node.setPropertyValue("tablename", ".Drug1n")
```

Tabla 45. propiedades de databasenode.

| Propiedades de databasenode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|----------------|---|
| mode | Tabla Query | Especifique <i>Table</i> para conectarse a una tabla de base de datos mediante los controles del cuadro de diálogo, o <i>Query</i> para realizar una consulta a la base de datos seleccionada mediante SQL. |

Tabla 45. propiedades de databasenode (continuación).

| Propiedades de databasenode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|----------------------------------|--|
| datasource | <i>cadena</i> | Nombre de la base de datos (consulte la siguiente nota). |
| nombre de usuario | <i>cadena</i> | Detalles de conexión de la base de datos (consulte la siguiente nota). |
| contraseña | <i>cadena</i> | |
| credencial | <i>cadena</i> | Nombre de la credencial almacenada en IBM SPSS Collaboration and Deployment Services. Se puede utilizar en lugar de las propiedades username y password. El nombre de usuario y la contraseña de la credencial deben coincidir con el nombre de usuario y la contraseña necesarios para acceder a la base de datos |
| use_credential | | Se establece en True o False. |
| epassword | <i>cadena</i> | Especifica una contraseña codificada como una alternativa a codificar una contraseña en un script. Consulte el tema "Generación de una contraseña codificada" en la página 52 para obtener más información. Esta propiedad es de sólo lectura durante la ejecución. |
| tablename | <i>cadena</i> | Nombre de la tabla a la que se desea tener acceso. |
| strip_spaces | Ninguno Left Right Both | Opciones para descartar los espacios iniciales y finales en las cadenas. |
| use_quotes | AsNeeded Always Nunca | Especifica si los nombres de columna y tabla aparecen entre comillas cuando las consultas se envían a la base de datos (en el caso, por ejemplo, de que contengan espacios o signos de puntuación). |
| consulta | <i>cadena</i> | Especifica el código SQL para la consulta que desea enviar. |

Nota: Si el nombre de la base de datos (en la propiedad datasource) contiene uno o más espacios, puntos (llamados también "punto y aparte") o subrayados, puede utilizar el formato de "barra inclinada invertida y comillas dobles" para tratarlo como una cadena. Por ejemplo: "{\db2v9.7.6_linux\}" o "{\TDATA 131\}". Además, encierre siempre los valores de cadena de datasource entre dobles comillas y llaves, como en el ejemplo siguiente: "{\SQL Server\",spssuser,abcd1234,false}".

Nota: Si el nombre de la base de datos (en la propiedad datasource) contiene espacios, entonces en vez de las propiedades individuales para datasource, username y password, utilice un único origen de datos en el siguiente formato:

Tabla 46. Propiedades de databasenode - específicas de datasource.

| Propiedades de databasenode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|---------------|--|
| datasource | cadena | Formato: [database_name,username,password[,true false]] El último parámetro se usa con contraseñas cifradas. Si se define como true, la contraseña se cifrará antes de usarse. |

Utilice este formato también si está cambiando el origen de datos, sin embargo, si tan sólo desea cambiar el nombre de usuario o contraseña, puede usar las propiedades username o password.

Propiedades de datacollectionimportnode



El nodo Importación de datos de Data Collection importa datos de encuesta basándose en el modelo de datos de Data Collection utilizado por productos de investigación de mercado. Se debe instalar la biblioteca de datos de Data Collection para utilizar este nodo.

Ejemplo

```
node = stream.create("datacollectionimport", "My node")
node.setPropertyValue("metadata_name", "mrQvDsc")
node.setPropertyValue("metadata_file", "C:/Program Files/IBM/SPSS/DataCollection/DDL/Data/
Quanvert/Museum/museum.pkd")
node.setPropertyValue("casedata_name", "mrQvDsc")
node.setPropertyValue("casedata_source_type", "File")
node.setPropertyValue("casedata_file", "C:/Program Files/IBM/SPSS/DataCollection/DDL/Data/
Quanvert/Museum/museum.pkd")
node.setPropertyValue("import_system_variables", "Common")
node.setPropertyValue("import_multi_response", "MultipleFlags")
```

Tabla 47. propiedades de datacollectionimportnode.

| propiedades de datacollectionimportnode | Tipo de datos | Descripción de la propiedad |
|---|---------------|--|
| metadata_name | cadena | El nombre del MDSC. El valor especial DimensionsMDD indica que se debería utilizar el documento de metadatos de Data Collection estándar. Otro posibles valores podrían ser: mrAD0Dsc mrI2dDsc mrLogDsc mrQdiDrsDsc mrQvDsc mrSampleReportingMDSC mrSavDsc mrSCDsc mrScriptMDSC El valor especial none indica que no existe ningún MDSC. |
| metadata_file | cadena | Nombre del archivo en el que se almacenan los metadatos. |

Tabla 47. propiedades de datacollectionimportnode (continuación).

| propiedades de datacollectionimportnode | Tipo de datos | Descripción de la propiedad |
|---|---|---|
| casedata_name | <i>cadena</i> | El nombre del CDSC. Entre los posibles valores se encuentran: mrADODsc mrI2dDsc mrLogDsc mrPunchDSC mrQdiDrsDsc mrQvDsc mrRdbDsc2 mrSavDsc mrScDSC mrXm1Dsc El valor especial none indica que no existe ningún CDSC. |
| casedata_source_type | Desconocido File Folder UDL DSN | Indica el tipo de origen del CDSC. |
| casedata_file | <i>cadena</i> | Cuando casedata_source_type es <i>File</i> , especifica el archivo que contiene los datos de casos. |
| casedata_folder | <i>cadena</i> | Cuando casedata_source_type es <i>Folder</i> , especifica la carpeta que contiene los datos de casos. |
| casedata_udl_string | <i>cadena</i> | Cuando casedata_source_type es <i>UDL</i> , especifica la cadena de conexión OLD-DB del origen de datos que contiene los datos de casos. |
| casedata_dsn_string | <i>cadena</i> | Cuando casedata_source_type es <i>DSN</i> , especifica la cadena de conexión ODBC del origen de datos. |
| casedata_project | <i>cadena</i> | Al leer datos de casos de una base de datos de Data Collection, puede escribir el nombre del proyecto. Para el resto de tipos de datos de casos, esta configuración se deberá dejar en blanco. |
| version_import_mode | Todos Latest Especifica | Define el modo en que deben tratarse las versiones. |
| specific_version | <i>cadena</i> | Cuando version_import_mode es <i>Specify</i> , define la versión de los datos de casos que se van a importar. |
| use_language | <i>cadena</i> | Determina si deben usarse las etiquetas de un idioma concreto. |
| language | <i>cadena</i> | Si use_language es verdadero, define el código de idioma que se va a usar en la importación. Este código de idioma debe incluirse entre aquellos disponibles en los datos de casos. |

Tabla 47. propiedades de datacollectionimportnode (continuación).

| propiedades de datacollectionimportnode | Tipo de datos | Descripción de la propiedad |
|---|---------------------------|--|
| use_context | cadena | Determina si se debe importar un contexto específico. Los contextos se utilizan para modificar la descripción asociada con las respuestas. |
| context | cadena | Cuando use_context es verdadero, define el contexto de la importación. Este contexto debe encontrarse entre aquellos disponibles en los datos de casos. |
| use_label_type | cadena | Determina si se debe importar un tipo de etiqueta específico. |
| label_type | cadena | Cuando use_label_type es verdadero, define el tipo de etiqueta de la importación. Este tipo de etiqueta debe encontrarse entre aquellos disponibles en los datos de casos. |
| id_usuario | cadena | En el caso de las bases de datos que requieren un inicio de sesión explícito, puede proporcionar un ID de usuario y una contraseña para acceder al origen de datos. |
| contraseña | cadena | |
| import_system_variables | Común Ninguno Todos | Especifica las variables del sistema que se importan. |
| import_codes_variables | tag | |
| import_sourcefile_variables | tag | |
| import_multi_response | MultipleFlags Single | |

Propiedades de excelimportnode



El nodo de importación Excel importa datos de Microsoft Excel en formato de archivo .xlsx. No es necesario un origen de datos ODBC.

Ejemplos

```
#Para usar un rango con nombre:
node = stream.create("excelimport", "My node")
node.setPropertyValue("excel_file_type", "Excel2007")
node.setPropertyValue("full_filename", "C:/drug.xlsx")
node.setPropertyValue("use_named_range", True)
node.setPropertyValue("named_range", "DRUG")
node.setPropertyValue("read_field_names", True)
```

```
#Para usar un rango explícito:
node = stream.create("excelimport", "My node")
node.setPropertyValue("excel_file_type", "Excel2007")
node.setPropertyValue("full_filename", "C:/drug.xlsx")
node.setPropertyValue("worksheet_mode", "Name")
```

```
node.setPropertyValue("worksheet_name", "Drug")
node.setPropertyValue("explicit_range_start", "A1")
node.setPropertyValue("explicit_range_end", "F300")
```

Tabla 48. propiedades de excelimportnode.

| Propiedad de excelimportnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|--------------------------------|---|
| excel_file_type | Excel2007 | |
| full_filename | <i>cadena</i> | El nombre completo del archivo, incluyendo la ruta. |
| use_named_range | <i>Booleana</i> | Si usar un rango con nombre. Si es verdadero, la propiedad named_range se utiliza para especificar el rango de lectura y se ignoran el resto de configuraciones de rango de datos y hojas de trabajo. |
| named_range | <i>cadena</i> | |
| worksheet_mode | Índice Name | Determina si la hoja de trabajo se define por el índice o por el nombre. |
| worksheet_index | <i>entero</i> | Índice de la hoja de trabajo que se va a leer, siendo 0 la primera hoja de trabajo, 1 la segunda, etc. |
| worksheet_name | <i>cadena</i> | Nombre de la hoja de trabajo que se va a leer. |
| data_range_mode | FirstNonBlank ExplicitRange | Especifica cómo debe establecerse el rango. |
| blank_rows | StopReading ReturnBlankRows | Cuando data_range_mode es <i>FirstNonBlank</i> , especifica cómo deben tratarse las filas en blanco. |
| explicit_range_start | <i>cadena</i> | Cuando data_range_mode es <i>ExplicitRange</i> , especifica el punto de partida del rango de lectura. |
| explicit_range_end | <i>cadena</i> | |
| read_field_names | <i>Booleana</i> | Determina si la primera fila del rango concreto debería usarse como nombres de campo (columna). |

Propiedades de extensionimportnode



Con el nodo de importación de extensión, puede ejecutar scripts R o Python para Spark para importar datos.

Ejemplo de Python para Spark

```
##### Ejemplo de script de Python para Spark
import modeler.api
stream = modeler.script.stream()
node = stream.create("extension_importer", "extension_importer")
node.setPropertyValue("syntax_type", "Python")

python_script = ""
```

```

import spss.pyspark
from pyspark.sql.types import *

cxt = spss.pyspark.runtime.getContext()

_schema = StructType([StructField('id', LongType(), nullable=False), \
StructField('age', LongType(), nullable=True), \
StructField('Sex', StringType(), nullable=True), \
StructField('BP', StringType(), nullable=True), \
StructField('Cholesterol', StringType(), nullable=True), \
StructField('K', DoubleType(), nullable=True), \
StructField('Na', DoubleType(), nullable=True), \
StructField('Drug', StringType(), nullable=True)])

if cxt.isComputeDataModelOnly():
    cxt.setSparkOutputSchema(_schema)
else:
    df = cxt.getSparkInputData()
    if df is None:
        drugList=[(1,23,'F','HIGH','HIGH',0.792535,0.031258,'drugY'), \
(2,47,'M','LOW','HIGH',0.739309,0.056468,'drugC'),\
(3,47,'M','LOW','HIGH',0.697269,0.068944,'drugC'),\
(4,28,'F','NORMAL','HIGH',0.563682,0.072289,'drugX'),\
(5,61,'F','LOW','HIGH',0.559294,0.030998,'drugY'),\
(6,22,'F','NORMAL','HIGH',0.676901,0.078647,'drugX'),\
(7,49,'F','NORMAL','HIGH',0.789637,0.048518,'drugY'),\
(8,41,'M','LOW','HIGH',0.766635,0.069461,'drugC'),\
(9,60,'M','NORMAL','HIGH',0.777205,0.05123,'drugY'),\
(10,43,'M','LOW','NORMAL',0.526102,0.027164,'drugY')]
        sqlcxt = cxt.getSparkSQLContext()
        rdd = cxt.getSparkContext().parallelize(drugList)
        print 'pyspark read data count = '+str(rdd.count())
        df = sqlcxt.createDataFrame(rdd, _schema)

    cxt.setSparkOutputData(df)
    """

node.setPropertyValue("python_syntax", python_script)

```

Ejemplo de R

```

#### Ejemplo de script de R
node.setPropertyValue("syntax_type", "R")

R_script = """# 'JSON Import' Node v1.0 for IBM SPSS Modeler
# Paquete 'RJSONIO' creado por Duncan Temple Lang - http://cran.r-project.org/web/packages/RJSONIO
# Paquete 'plyr' creado por Hadley Wickham http://cran.r-project.org/web/packages/plyr
# Desarrollador de nodo: Danil Savine - IBM Extreme Blue 2014
# Descripción: Este nodo le permite importar a SPSS datos de una tabla de un JSON.
# Instalar función para paquetes
packages <- function(x){
  x <- as.character(match.call()[[2]])
  if (!require(x,character.only=TRUE)){
    install.packages(pkgs=x,repos="http://cran.r-project.org")
    require(x,character.only=TRUE)
  }
}
# paquetes
packages(RJSONIO)
packages(plyr)
### Esta función se utiliza para generar automáticamente el dataModel
getMetaData <- function(data) {
  if (dim(data)[1]<=0) {

    print("Warning : modelerData has no line, all fieldStorage fields set to strings")
    getStorage <- function(x){return("string")}
  }
}

```

```

} else {

  getStorage <- function(x) {
    res <- NULL
    #si x es un factor, typeof devolverá un entero para que tratemos el caso en el lado
    if(is.factor(x)) {
      res <- "string"
    } else {
      res <- switch(typeof(unlist(x)),
                    integer = "integer",
                    double = "real",
                    character = "string",
                    "string")
    }
    return (res)
  }
}

col = vector("list", dim(data)[2])
for (i in 1:dim(data)[2]) {
  col[[i]] <- c(fieldName=names(data[i]),
               fieldLabel="",
               fieldStorage=getStorage(data[i]),
               fieldMeasure="",
               fieldFormat="",
               fieldRole="")
}
mdm<-do.call(cbind,col)
mdm<-data.frame(mdm)
return(mdm)
}

# De JSON a una lista
txt <- readLines('C:/test.json')
formattedtxt <- paste(txt, collapse = '')
json.list <- fromJSON(formattedtxt)
# Aplicar ruta a json.list
if(strsplit(x='true', split='
', fixed=TRUE)[[1]][1]) {
  path.list <- unlist(strsplit(x='id_array', split=','))
  i = 1
  while(i<length(path.list)+1){
    if(is.null(getElement(json.list, path.list[i]))){
      json.list <- json.list[[1]]
    }else{
      json.list <- getElement(json.list, path.list[i])
      i <- i+1
    }
  }
}

# De lista a marco de datos a través de json no listado
i <-1
filled <- data.frame()
while(i < length(json.list)+ 1){
  unlisted.json <- unlist(json.list[[i]])
  to.fill <- data.frame(t(as.data.frame(unlisted.json, row.names = names(unlisted.json))),
                       stringsAsFactors=FALSE)
  filled <- rbind.fill(filled,to.fill)
  i <- 1 + i
}

# Exportar a datos de SPSS Modeler
modelerData <- filled
print(modelerData)
modelerDataModel <- getMetaData(modelerData)
print(modelerDataModel)

```

"""

```
node.setPropertyValue("r_syntax", R_script)
```

Tabla 49. Propiedades de *extensionimportnode*

| Propiedades de <i>extensionimportnode</i> | Tipo de datos | Descripción de la propiedad |
|---|---------------------------|--|
| <code>syntax_type</code> | <i>R</i> <i>Python</i> | Especifica qué script se ejecuta: R o Python (R es el valor predeterminado). |
| <code>r_syntax</code> | <i>string</i> | Sintaxis de scripts R a ejecutar. |
| <code>python_syntax</code> | <i>string</i> | Sintaxis de scripts Python a ejecutar. |

Propiedades de *fixedfilenode*



El nodo Archivo fijo importa datos desde archivos de texto de campo fijo; esto es, archivos cuyos campos no están delimitados pero empiezan en la misma posición y tienen una longitud fija. Los datos heredados o generados por la máquina se suelen almacenar en formato de campo fijo.

Ejemplo

```
node = stream.create("fixedfile", "My node")
node.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
node.setPropertyValue("record_len", 32)
node.setPropertyValue("skip_header", 1)
node.setPropertyValue("fields", [{"Age", 1, 3}, {"Sex", 5, 7}, {"BP", 9, 10}, {"Cholesterol", 12, 22}, {"Na", 24, 25}, {"K", 27, 27}, {"Drug", 29, 32}])
node.setPropertyValue("decimal_symbol", "Period")
node.setPropertyValue("lines_to_scan", 30)
```

Tabla 50. propiedades de *fixedfilenode*.

| Propiedad de <i>fixedfilenode</i> | Tipo de datos | Descripción de la propiedad |
|--------------------------------------|-----------------------------------|--|
| <code>record_len</code> | <i>number</i> | Especifica el número de caracteres de cada registro. |
| <code>line_oriented</code> | <i>tag</i> | Omite el carácter de nueva línea al final de cada registro. |
| <code>decimal_symbol</code> | Predeterminado Comma Period | Tipo de separador decimal utilizado en el origen de datos. |
| <code>skip_header</code> | <i>number</i> | Especifica el número de líneas que se ignorarán al principio del primer registro. Esto resulta útil para ignorar las cabeceras de columna. |
| <code>auto_recognize_datetime</code> | <i>tag</i> | Especifica si las fechas o las horas se identifican automáticamente en los datos de origen. |
| <code>lines_to_scan</code> | <i>number</i> | |
| <code>campos</code> | <i>list</i> | Propiedad estructurada. |
| <code>full_filename</code> | <i>cadena</i> | Nombre completo del archivo que se va a leer, incluido el directorio. |

Tabla 50. propiedades de fixedfilenode (continuación).

| Propiedad de fixedfilenode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---|---|
| strip_spaces | Ninguno Left Right Both | Descarta los espacios iniciales y finales en las cadenas de importación. |
| invalid_char_mode | Descartar Replace | Elimina los caracteres no válidos (nulo, 0 o cualquier carácter que no exista en la codificación actual) de la entrada de datos o sustituye los caracteres no válidos con el símbolo especificado de un carácter. |
| invalid_char_replacement | <i>cadena</i> | |
| use_custom_values | <i>tag</i> | |
| custom_storage | Desconocido Cadena Entero Real Hora Fecha Marca de tiempo | |
| custom_date_format | "DDMMYY" "MMDDYY" "YYMMDD" "YYMMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-YYYY" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.YYYY" "MM.DD.YY" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/YYYY" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | Aplicable solamente si ha especificado un almacenamiento personalizado. |

Tabla 50. propiedades de fixedfilenode (continuación).

| Propiedad de fixedfilenode | Tipo de datos | Descripción de la propiedad |
|----------------------------|--|---|
| custom_time_format | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | Aplicable solamente si ha especificado un almacenamiento personalizado. |
| custom_decimal_symbol | campo | Aplicable solamente si ha especificado un almacenamiento personalizado. |
| codificación | StreamDefault SystemDefault "UTF-8" | Especifica el método de codificación de textos. |

Propiedades del nodo gsdata_import



Utilice el nodo de origen Geoespacial para llevar datos de mapa o espaciales en su sesión de minería de datos.

Tabla 51. propiedades del nodo gsdata_import

| Propiedades del nodo gsdata_import | Tipo de datos | Descripción de la propiedad |
|------------------------------------|---------------|--|
| full_filename | cadena | Entre la vía de acceso del archivo .shp que desea cargar. |
| map_service_URL | cadena | Escriba el URL de servicio de mapas al que conectarse. |
| map_name | cadena | Sólo si se utiliza map_service_URL; esto contiene la estructura de carpeta de nivel superior del servicio de mapa. |

Propiedades de sasimportnode



El nodo importar SAS importa datos SAS a IBM SPSS Modeler.

Ejemplo

```

node = stream.create("sasimport", "My node")
node.setPropertyValue("format", "Windows")
node.setPropertyValue("full_filename", "C:/data/retail.sas7bdat")
node.setPropertyValue("member_name", "Test")
node.setPropertyValue("read_formats", False)
node.setPropertyValue("full_format_filename", "Test")
node.setPropertyValue("import_names", True)

```

Tabla 52. propiedades de sasimportnode.

| Propiedad de sasimportnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|--|---|
| format | Windows UNIX Transport SAS7 SAS8 SAS9 | Formato del archivo que se va a importar. |
| full_filename | <i>cadena</i> | El nombre completo del archivo introducido, incluyendo su ruta. |
| member_name | <i>cadena</i> | Especifique el miembro para importar desde el archivo de transporte SAS especificado. |
| read_formats | <i>tag</i> | Lee formatos de datos (como etiquetas de variables) del archivo de formatos especificado. |
| full_format_filename | <i>cadena</i> | |
| import_names | NamesAndLabels LabelsAsNames | Especifica el método para la correlación de nombres y etiquetas de variables en la importación. |

propiedades de simgennode



El nodo de generación de simulación proporciona una manera fácil de generar datos, ya sea desde cero utilizando las distribuciones o estadísticas especificada por el usuario o automáticamente utilizando las distribuciones obtenidas de la ejecución de un nodo de simulación de ajuste sobre datos históricos. Esto es útil cuando desea evaluar el resultado de un modelo predictivo en caso de dudas sobre las entradas del modelo.

Tabla 53. propiedades de simgennode.

| propiedades de simgennode | Tipo de datos | Descripción de la propiedad |
|---------------------------|------------------------|--|
| campos | Propiedad estructurada | Ver ejemplo |
| correlaciones | Propiedad estructurada | Ver ejemplo |
| keep_min_max_setting | <i>booleano</i> | |
| refit_correlations | <i>booleano</i> | |
| max_cases | <i>entero</i> | El valor mínimo es 1000, el valor máximo es 2,147,483,647. |
| create_iteration_field | <i>booleano</i> | |
| iteration_field_name | <i>cadena</i> | |
| replicate_results | <i>booleano</i> | |
| random_seed | <i>entero</i> | |
| parameter_xml | <i>cadena</i> | Devuelve el XML del parámetro como una cadena. |

Ejemplo de fields

Este es un parámetro de ranura estructurado con la sintaxis siguiente:

```
simgennode.setPropertyValue("fields", [
    [field1, storage, locked, [distribution1], min, max],
    [field2, storage, locked, [distribution2], min, max],
    [field3, storage, locked, [distribution3], min, max]
])
```

distribution es una declaración de nombre de distribución seguido por una lista con parejas de nombres de atributo y valores. Cada distribución se define de la siguiente forma:

```
[distributionname, [[par1], [par2], [par3]]]
```

```
simgennode = modeler.script.stream().createAt("simgen", u"Sim Gen", 726, 322)
simgennode.setPropertyValue("fields",
[["Age", "integer", False, ["Uniform",["min","1"],["max","2"]]], "", ""])
```

Por ejemplo, para crear un nodo que genere un solo campo con una distribución binomial puede utilizar el siguiente script:

```
simgen_node1 = modeler.script.stream().createAt("simgen", u"Sim Gen", 200, 200)
simgen_node1.setPropertyValue("fields", [["Education", "Real", False, ["Binomial", [{"n", 32},
["prob", 0.7] ]]], "", ""])
```

La distribución binomial tiene 2 parámetros: n y prob. Puesto que binomial no admite los valores mínimo y máximo, éstos se suministran como una serie vacía.

Nota: No se puede establecer distribution directamente; utilícelo en combinación con la propiedad fields.

Los ejemplos siguientes muestran todos los tipos de distribución posibles. Tenga en cuenta que el umbral se especifica como thresh en NegativeBinomialFailures y en NegativeBinomialTrial.

```
stream = modeler.script.stream()

simgennode = stream.createAt("simgen", u"Sim Gen", 200, 200)

beta_dist = ["Field1", "Real", False, ["Beta", [{"shape1", "1"}, {"shape2", "2"} ]]], "", ""]
binomial_dist = ["Field2", "Real", False, ["Binomial", [{"n", "1"}, {"prob", "1"} ]]], "", ""]
categorical_dist = ["Field3", "String", False, ["Categorical", [{"A", 0.3}, {"B", 0.5}, {"C", 0.2} ]]], "", ""]
dice_dist = ["Field4", "Real", False, ["Dice", [{"1", "0.5"}, {"2", "0.5"} ]]], "", ""]
exponential_dist = ["Field5", "Real", False, ["Exponential", [{"scale", "1"} ]]], "", ""]
fixed_dist = ["Field6", "Real", False, ["Fixed", [{"value", "1"} ]]], "", ""]
gamma_dist = ["Field7", "Real", False, ["Gamma", [{"scale", "1"}, {"shape", "1"} ]]], "", ""]
lognormal_dist = ["Field8", "Real", False, ["Lognormal", [{"a", "1"}, {"b", "1"} ]]], "", ""]
negbinomialfailures_dist = ["Field9", "Real", False, ["NegativeBinomialFailures", [{"prob", "0.5"}, {"thresh", "1"} ]]], "", ""]
negbinomialtrial_dist = ["Field10", "Real", False, ["NegativeBinomialTrials", [{"prob", "0.2"}, {"thresh", "1"} ]]], "", ""]
normal_dist = ["Field11", "Real", False, ["Normal", [{"mean", "1"}, {"stddev", "2"} ]]], "", ""]
poisson_dist = ["Field12", "Real", False, ["Poisson", [{"mean", "1"} ]]], "", ""]
range_dist = ["Field13", "Real", False, ["Range", [{"BEGIN", "1,3"}, {"END", "2,4"} ], [{"PROB", "[[0.5],[0.5]"} ]]], "", ""]
triangular_dist = ["Field14", "Real", False, ["Triangular", [{"min", "0"}, {"max", "1"}, {"mode", "1"} ]]], "", ""]
uniform_dist = ["Field15", "Real", False, ["Uniform", [{"min", "1"}, {"max", "2"} ]]], "", ""]
weibull_dist = ["Field16", "Real", False, ["Weibull", [{"a", "0"}, {"b", "1"}, {"c", "1"} ]]], "", ""]

simgennode.setPropertyValue("fields", [
beta_dist, \
binomial_dist, \
categorical_dist, \
dice_dist, \
exponential_dist, \
fixed_dist, \
gamma_dist, \
lognormal_dist, \
negbinomialfailures_dist, \
negbinomialtrial_dist, \
normal_dist, \
poisson_dist, \
range_dist, \
triangular_dist, \
uniform_dist, \
weibull_dist
])
```

Ejemplo de correlations

Este es un parámetro de ranura estructurado con la sintaxis siguiente:

```
simgennode.setPropertyValue("correlations", [  
    [field1, field2, correlation],  
    [field1, field3, correlation],  
    [field2, field3, correlation]  
])
```

La correlación puede ser cualquier número entre +1 y -1. Puede especificar tantas correlaciones como desee. Las correlaciones no especificadas se establecen en cero. Si alguno de los campos se desconocen, el valor de la correlación debe establecerse en la matriz de correlación (o tabla) y se muestra en texto rojo. Cuando hay campos desconocidos, no es posible ejecutar el nodo.

Propiedades de statisticsimportnode



El nodo IBM SPSS StatisticsArchivo lee los datos desde un formato de archivo *.sav* que utiliza IBM SPSS Statistics y archivos caché guardados en IBM SPSS Modeler, que también puede utilizar el mismo formato.

Las propiedades de este nodo están descritas en “Propiedades de statisticsimportnode” en la página 341.

Propiedades del nodo tm1odataimport



El nodo de origen de IBM Cognos TM1 importa datos desde las bases de datos de Cognos TM1.

Tabla 54. propiedades del nodo tm1odataimport

| Propiedades del nodo tm1odataimport | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|--|--|
| admin_host | <i>cadena</i> | El URL del nombre de host de la API REST. |
| server_name | <i>cadena</i> | El nombre del servidor TM1 seleccionado de admin_host. |
| credential_type | <i>inputCredential</i> o <i>storedCredential</i> | Se utiliza para indicar el tipo de credencial. |
| input_credential | <i>lista</i> | Cuando credential_type es inputCredential; especifique el dominio, el nombre de usuario y la contraseña. |
| stored_credential_name | <i>cadena</i> | Cuando credential_type es storedCredential; especifique el nombre de la credencial en el servidor C&DS. |
| selected_view | <i>["campo" "campo"]</i> | Una propiedad de la lista que contiene los detalles del cubo TM1 seleccionado y el nombre de la vista de cubo donde los datos se importarán en SPSS. Por ejemplo: TM1_import.setPropertyValue("selected_view", ['plan_BudgetPlan', 'Goal Input']) |

Tabla 54. propiedades del nodo tm1odataimport (continuación)

| Propiedades del nodo tm1odataimport | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|---------------------|---|
| is_private_view | flag | Especifica si selected_view es una vista privada. El valor predeterminado es false. |
| selected_columns | ["field"] | Especifique la columna seleccionada; solo se puede especificar un elemento. Por ejemplo: setProperty("selected_columns", ["Measures"]) |
| selected_rows | ["campo" "campo"] | Especifique las filas seleccionadas: Por ejemplo: setProperty("selected_rows", ["Dimension_1_1", "Dimension_2_1", "Dimension_3_1", "Periods"]) |

Propiedades del nodo tm1import (en desuso)



El nodo de origen de IBM Cognos TM1 importa datos desde las bases de datos de Cognos TM1.

Nota: Este nodo estaba en desuso en Modeler 18.0. El nombre del script del nodo de sustitución es *tm1odataimport*.

Tabla 55. propiedades del nodo tm1import.

| Propiedades del nodo tm1import | Tipo de datos | Descripción de la propiedad |
|--------------------------------|-----------------------------------|--|
| pm_host | cadena | Nota: sólo para la versión 16.0 y 17.0 Nombre del host. Por ejemplo: TM1_import.setProperty("pm_host", 'http://9.191.86.82:9510/pmhub/pm') |
| tm1_connection | ["campo", "campo", ... "campo"] | Nota: sólo para la versión 16.0 y 17.0 Una propiedad de la lista que contiene los detalles de conexión para el servidor de TM1. El formato es: ["TM1_Server_Name", "tm1_username", "tm1_password"] Por ejemplo: TM1_import.setProperty("tm1_connection", ['Planning Sample', "admin", "apple"]) |
| selected_view | ["campo" "campo"] | Una propiedad de la lista que contiene los detalles del cubo TM1 seleccionado y el nombre de la vista de cubo donde los datos se importarán en SPSS. Por ejemplo: TM1_import.setProperty("selected_view", ['plan_BudgetPlan', 'Goal Input']) |

Tabla 55. propiedades del nodo *tm1import* (continuación).

| Propiedades del nodo <i>tm1import</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|--------------------------------|--|
| <code>selected_column</code> | <code>["field"]</code> | Especifique la columna seleccionada; solo se puede especificar un elemento. Por ejemplo: <code>setProperty("selected_columns", ["Measures"])</code> |
| <code>selected_rows</code> | <code>["campo" "campo"]</code> | Especifique las filas seleccionadas: Por ejemplo: <code>setProperty("selected_rows", ["Dimension_1_1", "Dimension_2_1", "Dimension_3_1", "Periods"])</code> |

Propiedades del nodo *twcimport*



El nodo de origen de TWC importa datos meteorológicos de The Weather Company, una empresa de IBM. Puede utilizarlo para obtener datos meteorológicos históricos o predicción para una ubicación. Esto puede ayudarle a desarrollar soluciones empresariales dirigidas a la meteorología para una mejor toma de decisiones utilizando los datos meteorológicos más exactos y precisos disponibles.

Tabla 56. Propiedades del nodo *twcimport*

| Propiedades del nodo <i>twcimport</i> | Tipo de datos | Descripción de la propiedad |
|---|-----------------------------|---|
| <code>TWCDataImport.latitude</code> | <i>Real</i> | Especifica un valor de latitud con el formato [-90.0090.0] |
| <code>TWCDataImport.longitude</code> | <i>Real</i> | Especifica un valor de longitud con el formato [-180.00180.0]. |
| <code>TWCDataImport.licenseKey</code> | <i>string</i> | Especifica la clave de licencia obtenida de The Weather Company. |
| <code>TWCDataImport.measurmentUnit</code> | English Metric Hybrid | Especifica la unidad de medida. Los valores posibles son English, Metric o Hybrid. Metric es el valor predeterminado. |
| <code>TWCDataImport.dataType</code> | Historical Forecast | Especifica el tipo de datos meteorológicos para la entrada. Los valores posibles son Historical o Prediction. Histórico es el valor predeterminado. |
| <code>TWCDataImport.startDate</code> | <i>Entero</i> | Si se especifica Histórico para <code>TWCDataImport.dataType</code> , especifique una fecha de inicio con el formato <code>aaaaMMdd</code> . |
| <code>TWCDataImport.endDate</code> | <i>Entero</i> | Si se especifica Histórico para <code>TWCDataImport.dataType</code> , especifique una fecha de finalización con el formato <code>aaaaMMdd</code> . |
| <code>TWCDataImport.forecastHour</code> | 6 12 24 48 | Si se especifica Predicción para <code>TWCDataImport.dataType</code> , especifique 6, 12, 24 o 48 para la hora. |

propiedades de userinputnode



El nodo Datos de usuario proporciona una manera fácil de crear datos sintéticos, ya sea partiendo de cero o modificando los datos existentes. Esto resulta útil, por ejemplo, cuando desee crear un conjunto de datos de comprobación para el modelado.

Ejemplo

```
node = stream.create("userinput", "My node")
node.setPropertyValue("names", ["test1", "test2"])
node.setKeyedPropertyValue("data", "test1", "2, 4, 8")
node.setKeyedPropertyValue("custom_storage", "test1", "Integer")
node.setPropertyValue("data_mode", "Ordered")
```

Tabla 57. propiedades de userinputnode.

| Propiedad de userinputnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---|--|
| datos | | |
| nombres | | Intervalo estructurado que establece o devuelve una lista de nombres de campos generados por el nodo. |
| custom_storage | Desconocido Cadena Entero Real Hora Fecha Marca de tiempo | Intervalo con clave que establece o devuelve el almacenamiento para un campo. |
| data_mode | Combined Ordered | Si se especifica Combined, los registros se generarán para cada combinación de valores del conjunto y valores mínimos y máximos. El número de registros generados será igual al producto del número de valores de cada campo. Si se especifica Ordered, se tomará un valor de cada columna para cada registro con el fin de generar una fila de datos. El número de registros generados será igual al número más grande de valores asociados a un campo. Los campos que tengan menos valores de datos se rellenarán con valores nulos. |
| values | | Nota: Esta propiedad ya no se utiliza, y no debe usarse; en su lugar, se usa <code>userinputnode.data</code> . |

Propiedades de variablefilenode



El nodo Archivo variable lee datos desde los archivos de texto de campo libre, esto es, campos cuyos registros contienen un número constante de campos pero un número variado de caracteres. Este nodo resulta también útil para los archivos con texto de cabecera de longitud fija y determinados tipos de anotaciones.

Ejemplo

```

node = stream.create("variablefile", "My node")
node.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
node.setPropertyValue("read_field_names", True)
node.setPropertyValue("delimit_other", True)
node.setPropertyValue("other", ",")
node.setPropertyValue("quotes_1", "Discard")
node.setPropertyValue("decimal_symbol", "Comma")
node.setPropertyValue("invalid_char_mode", "Replace")
node.setPropertyValue("invalid_char_replacement", "|")
node.setKeyedPropertyValue("use_custom_values", "Age", True)
node.setKeyedPropertyValue("direction", "Age", "Input")
node.setKeyedPropertyValue("type", "Age", "Range")
node.setKeyedPropertyValue("values", "Age", [1, 100])

```

Tabla 58. propiedades de variablefilenode.

| Propiedad de variablefilenode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|-----------------------------------|--|
| skip_header | <i>number</i> | Especifica el número de caracteres que se ignorarán al principio del primer registro. |
| num_fields_auto | <i>tag</i> | Determina el número de campos de cada registro de forma automática. Los registros deben terminar con un carácter de nueva línea. |
| num_fields | <i>number</i> | Especifica manualmente el número de campos de cada registro. |
| delimit_space | <i>tag</i> | Especifica el carácter utilizado para delimitar los límites de los campos del archivo. |
| delimit_tab | <i>tag</i> | |
| delimit_new_line | <i>tag</i> | |
| delimit_non_printing | <i>tag</i> | |
| delimit_comma | <i>tag</i> | En aquellos casos en los que la coma sea el delimitador del campo y el separador decimal para rutas, establezca <i>delimit_other</i> en <i>True</i> y especifique una coma como delimitador mediante la propiedad <i>other</i> . |
| delimit_other | <i>tag</i> | Permite especificar un delimitador personalizado mediante la propiedad <i>other</i> . |
| other | <i>cadena</i> | Especifica el delimitador utilizado cuando <i>delimit_other</i> es <i>True</i> . |
| decimal_symbol | Predeterminado Comma Period | Especifica el separador decimal utilizado en el origen de datos. |
| multi_blank | <i>tag</i> | Trata varios caracteres delimitadores vacíos adyacentes como un único delimitador. |
| read_field_names | <i>tag</i> | Trata la primera fila del archivo de datos como etiquetas para la columna. |
| strip_spaces | Ninguno Left Right Both | Descarta los espacios iniciales y finales en las cadenas de importación. |

Tabla 58. propiedades de variablefilenode (continuación).

| Propiedad de variablefilenode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|---|
| invalid_char_mode | Descartar Replace | Elimina los caracteres no válidos (nulo, 0 o cualquier carácter que no exista en la codificación actual) de la entrada de datos o sustituye los caracteres no válidos con el símbolo especificado de un carácter. |
| invalid_char_replacement | <i>cadena</i> | |
| break_case_by_newline | <i>flag</i> | Especifica que el delimitador de línea es el carácter de nueva línea. |
| lines_to_scan | <i>number</i> | Especifica cuántas líneas se van a explorar para los tipos de datos especificados. |
| auto_recognize_datetime | <i>tag</i> | Especifica si las fechas o las horas se identifican automáticamente en los datos de origen. |
| quotes_1 | Descartar PairAndDiscard IncludeAsText | Especifica cómo se tratarán las comillas simples en la importación. |
| quotes_2 | Descartar PairAndDiscard IncludeAsText | Especifica cómo se tratarán las comillas dobles en la importación. |
| full_filename | <i>cadena</i> | Nombre completo del archivo que se va a leer, incluido el directorio. |
| use_custom_values | <i>tag</i> | |
| custom_storage | Desconocido Cadena Entero Real Hora Fecha Marca de tiempo | |

Tabla 58. propiedades de variablefilenode (continuación).

| Propiedad de variablefilenode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|---|
| custom_date_format | "DDMMYY" "MMDDYY" "YYMMDD" "YYMMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-YYYY" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.YYYY" "MM.DD.YY" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/YYYY" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | Aplicable solamente si ha especificado un almacenamiento personalizado. |
| custom_time_format | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | Aplicable solamente si ha especificado un almacenamiento personalizado. |
| custom_decimal_symbol | campo | Aplicable solamente si ha especificado un almacenamiento personalizado. |
| codificación | StreamDefault SystemDefault "UTF-8" | Especifica el método de codificación de textos. |

Propiedades de xmlimportnode



El nodo de origen XML importa datos en formato XML en la ruta. Puede importar un único archivo o todos los archivos en un directorio. Puede especificar opcionalmente un archivo de esquema para leer la estructura XML.

Ejemplo

```
node = stream.create("xmlimport", "My node")
node.setPropertyValue("full_filename", "c:/import/ebooks.xml")
node.setPropertyValue("records", "/author/name")
```

Tabla 59. propiedades de xmlimportnode.

| propiedades de xmlimportnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|-------------------|--|
| read | single directorio | Lee un único archivo de datos (valor predeterminado) o todos los archivos XML de un directorio. |
| recurse | tag | Especifica si leer además archivos XML de todos los subdirectorios del directorio especificado. |
| full_filename | cadena | (obligatorio) Ruta completa y nombre de archivo del archivo XML a importar (si read = single). |
| directory_name | cadena | (obligatorio) Ruta completa y nombre del directorio desde el que importar los archivos XML (si read = directory). |
| full_schema_filename | cadena | Ruta completa y nombre de archivo del archivo XSD o DTD desde el que leer la estructura XML. Si omite este parámetro, se leerá la estructura desde el archivo de origen XML. |
| records | cadena | Expresión XPath (p.ej. /author/name) para definir el límite del registro. Cada vez que este elemento se encuentra en el archivo de origen se crea un nuevo registro. |
| mode | read specify | Lee todos los datos (valor predeterminado) o especifica qué elementos leer. |
| campos | | Lista de elementos (elementos y atributos) para importar. Cada elemento de la lista es una expresión XPath. |

Propiedades de dataviewimport



El nodo Vista de datos importa datos de Vista de datos en IBM SPSS Modeler.

Ejemplo

```

stream = modeler.script.stream()

dnode = stream.createAt("dataviewimport", "Data View", 96, 96)
dnode.setPropertyValue("analytic_data_source",
["", "/folder/adv", "LATEST"])
dnode.setPropertyValue("table_name", ["", "com.ibm.spss.Table"])
dnode.setPropertyValue("data_access_plan",
["", "DataAccessPlan"])
dnode.setPropertyValue("optional_attributes",
[["", "NewDerivedAttribute"]])
dnode.setPropertyValue("include_xml", True)
dnode.setPropertyValue("include_xml_field", "xml_data")

```

Tabla 60. propiedades de dataviewimport

| Propiedades de dataviewimport | Tipo de datos | Descripción de la propiedad |
|-------------------------------|-----------------|---|
| analytic_data_source | <i>cadena</i> | La objeto Vista de datos analíticos almacenado en IBM SPSS Collaboration and Deployment Services. El nombre de vía de acceso y la etiqueta de la versión que se va a utilizar. ["Object ID", "Full path", "Version"] |
| table_name | <i>cadena</i> | La tabla de vista de datos utilizada en la Vista de datos analíticos. El nombre de la tabla debe estar cualificada para paquete. Puede obtener el paquete exportando la lista de materiales del cliente de gestor de despliegue de IBM SPSS Collaboration and Deployment Services y buscando en el archivo default.bom en el archivo ZIP exportado. El nombre del paquete siempre debe ser el mismo a menos que la lista de materiales se haya importado de IBM Operational Decision Management (iLOG). ["Object ID", "Name"] |
| data_access_plan | <i>cadena</i> | El plan de acceso de datos que se utiliza para proporcionar los datos para la Vista de datos analíticos. ["Object ID", "Name"] |
| optional_attributes | <i>cadena</i> | Una lista de los atributos derivados a incluir. [{"ID1", "Name1"}, {"ID2", "Name2"}] |
| include_xml | <i>booleano</i> | True si se va a incluir un campo con datos de instancia XOM. A menos que se utilicen nodos de IBM Analytical Decision Management iLOG, el valor recomendado es false. La activación de esta opción puede añadir una gran cantidad de proceso adicional. |
| include_xml_field | <i>cadena</i> | El nombre del campo a añadir cuando include_xml se establece en true. |

Capítulo 10. Propiedades de nodos de operaciones con registros

propiedades de appendnode



El nodo Añadir concatena conjuntos de registros. Es útil para combinar conjuntos de datos con estructuras parecidas, pero con datos diferentes.

Ejemplo

```
node = stream.create("append", "My node")
node.setPropertyValue("match_by", "Name")
node.setPropertyValue("match_case", True)
node.setPropertyValue("include_fields_from", "All")
node.setPropertyValue("create_tag_field", True)
node.setPropertyValue("tag_field_name", "Append_Flag")
```

Tabla 61. propiedades de appendnode.

| Propiedad de appendnode | Tipo de datos | Descripción de la propiedad |
|-------------------------|------------------|--|
| match_by | Position Name | Se pueden añadir conjuntos de datos basándose en la posición que tienen los campos en el origen de datos principal o el nombre de los campos en los conjuntos de datos de entrada. |
| match_case | <i>tag</i> | Activa la coincidencia de mayúsculas y minúsculas al hacer coincidir nombres de campos. |
| include_fields_from | Main Todos | |
| create_tag_field | <i>tag</i> | |
| tag_field_name | <i>cadena</i> | |

propiedades de agregatenode



El nodo Agregar reemplaza una secuencia de registros de entrada con registros de salida agregados y resumidos.

Ejemplo

```
node = stream.create("aggregate", "My node")
# dbnode es un nodo de importación de base de datos configurado
stream.link(dbnode, node)
node.setPropertyValue("contiguous", True)
node.setPropertyValue("keys", ["Drug"])
node.setKeyedPropertyValue("aggregates", "Age", ["Sum", "Mean"])
```

```

node.setPropertyValue("inc_record_count", True)
node.setPropertyValue("count_field", "index")
node.setPropertyValue("extension", "Aggregated_")
node.setPropertyValue("add_as", "Prefix")

```

Tabla 62. propiedades de *aggregatenode*.

| Propiedad de <i>aggregatenode</i> | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|------------------|---|
| keys | <i>lista</i> | Enumera los campos que se pueden usar como claves en la agregación. Por ejemplo, si Sexo y Región son los campos clave, cada combinación exclusiva de V y M con las regiones N y S (cuatro combinaciones exclusivas) tendrá un registro agregado. |
| contiguous | <i>tag</i> | Seleccione esta opción si sabe que todos los registros con los mismos valores clave se agrupan en la entrada (por ejemplo, si la entrada se clasifica en los campos clave). Con ello puede mejorar el rendimiento. |
| aggregates | | Enumera los campos numéricos cuyos valores se añadirán, así como los modos de agregación elegidos. |
| aggregate_exprs | | Propiedad con clave que aplica claves al nombre de campo derivado con la expresión agregada utilizada para calcularla. Por ejemplo: <code>aggregatenode.setKeyedPropertyValue("aggregate_exprs", "Na_MAX", "MAX('Na')")</code> |
| extensión | <i>cadena</i> | Especifica un prefijo o sufijo para campos agregados duplicados (consulte un ejemplo a continuación). |
| add_as | Suffix Prefix | |
| inc_record_count | <i>tag</i> | Crea un campo adicional que indica la cantidad de registros de entrada agregados para conformar cada registro agregado. |
| count_field | <i>cadena</i> | Especifica el nombre del campo de recuento de registros. |
| allow_approximation | <i>Booleana</i> | Permite la aproximación de estadísticas de ordenación cuando se realiza la agregación en Analytic Server |
| bin_count | <i>entero</i> | Especifica el número de intervalos a utilizar en la aproximación |

propiedades de *balancenode*



El nodo Equilibrar corrige los desequilibrios de un conjunto de datos para que cumpla una condición determinada. La directiva de equilibrado ajusta la proporción de registros si una condición es verdadera por el factor determinado.

Ejemplo

```
node = stream.create("balance", "My node")
node.setPropertyValue("training_data_only", True)
node.setPropertyValue("directives", [[1.3, "Age > 60"], [1.5, "Na > 0.5"]])
```

Tabla 63. propiedades de balancenode.

| Propiedad de balancenode | Tipo de datos | Descripción de la propiedad |
|--------------------------|---------------|--|
| directives | | Propiedad estructurada para equilibrar la proporción de los valores de campos basados en un número determinado (consulte el ejemplo a continuación). |
| training_data_only | tag | Especifica que sólo se deben equilibrar los datos de entrenamiento. Si no se incluye ningún campo de partición en la ruta, se ignorará esta opción. |

Esta propiedad de nodo utiliza el siguiente formato:

```
[[ número, cadena ] \ [ número, cadena] \ ... [número, cadena ]].
```

Nota: Si las cadenas (que utilizan comillas dobles) están incrustadas en la expresión, han de estar precedidas del carácter de escape " \ ". El carácter " \ " es también el carácter de continuación de línea, que puede utilizar para alinear los argumentos para mayor claridad.

Propiedades derive_stbnode



El nodo Cajas-espacio-tiempo deriva Cajas-espacio-tiempo de los campos latitud, longitud e indicación de fecha y hora. Las Cajas-espacio-tiempo también pueden identificarse como lugares comunes.

Ejemplo

```
node = modeler.script.stream().createAt("derive_stb", "My node", 96, 96)

# Modalidad Individual Records (registros individuales)
node.setPropertyValue("mode", "IndividualRecords")
node.setPropertyValue("latitude_field", "Latitude")
node.setPropertyValue("longitude_field", "Longitude")
node.setPropertyValue("timestamp_field", "OccurredAt")
node.setPropertyValue("densities", ["STB_GH7_1HOUR", "STB_GH7_30MINS"])
node.setPropertyValue("add_extension_as", "Prefix")
node.setPropertyValue("name_extension", "stb_")

# Modalidad Hangouts
node.setPropertyValue("mode", "Hangouts")
node.setPropertyValue("hangout_density", "STB_GH7_30MINS")
node.setPropertyValue("id_field", "Event")
node.setPropertyValue("qualifying_duration", "30MINUTES")
node.setPropertyValue("min_events", 4)
node.setPropertyValue("qualifying_pct", 65)
```

Tabla 64. propiedades del nodo Cajas-Espacio-Tiempo

| Propiedades de derive_stbnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|-------------------------------|-----------------------------|
| mode | IndividualRecords Hangouts | |
| latitude_field | campo | |

Tabla 64. propiedades del nodo Cajas-Espacio-Tiempo (continuación)

| Propiedades de derive_stbnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|--|---|
| longitude_field | campo | |
| timestamp_field | campo | |
| hangout_density | densidad | Una sola densidad. Consulte en densities los valores de densidad válidos. |
| densities | [densidad,densidad,..., densidad] | Cada densidad es una cadena, por ejemplo, STB_GH8_1DAY. Nota: Existen límites para que las densidades sean válidas. En geohash, se pueden utilizar los valores de GH1 a GH15. Para la parte temporal, se pueden utilizar los valores siguientes: EVER 1YEAR 1MONTH 1DAY 12HOURS 8HOURS 6HOURS 4HOURS 3HOURS 2HOURS 1HOUR 30MINS 15MINS 10MINS 5MINS 2MINS 1MIN 30SECS 15SECS 10SECS 5SECS 2SECS 1SEC |
| id_field | campo | |
| qualifying_duration | 1DAY 12HOURS 8HOURS 6HOURS 4HOURS 3HOURS 2Hours 1HOUR 30MIN 15MIN 10MIN 5MIN 2MIN 1MIN 30SECS 15SECS 10SECS 5SECS 2SECS 1SECS | Debe ser una cadena. |
| min_events | entero | El valor de entero válido mínimo es de 2. |
| qualifying_pct | entero | Debe estar en el rango de 1 a 100. |

Tabla 64. propiedades del nodo Cajas-Espacio-Tiempo (continuación)

| Propiedades de derive_stbnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|------------------|-----------------------------|
| add_extension_as | Prefix Suffix | |
| name_extension | cadena | |

propiedades de distinctnode



El nodo Distinguir se puede usar para eliminar registros duplicados pasando el primero de los registros distintos a la ruta de datos o descartando el primer registro y pasando cualquier duplicado a la ruta de datos en su lugar.

Ejemplo

```
node = stream.create("distinct", "My node")
node.setPropertyValue("mode", "Include")
node.setPropertyValue("fields", ["Age" "Sex"])
node.setPropertyValue("keys_pre_sorted", True)
```

Tabla 65. propiedades de distinctnode.

| Propiedad de distinctnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|------------------------|--|
| mode | Incluir Descartar | Se puede incluir el primer nodo distinto en la ruta o descartar el primer nodo distinto y pasar en su lugar todos los registros duplicados a la ruta de datos. |
| grouping_fields | lista | Enumera los campos utilizados para determinar si los registros son idénticos. Nota: Esta propiedad está en desuso desde IBM SPSS Modeler 16 y posterior. |
| composite_value | Intervalo estructurado | Vea el ejemplo que se muestra a continuación. |
| composite_values | Intervalo estructurado | Vea el ejemplo que se muestra a continuación. |
| inc_record_count | flag | Crea un campo adicional que indica la cantidad de registros de entrada agregados para conformar cada registro agregado. |
| count_field | cadena | Especifica el nombre del campo de recuento de registros. |
| sort_keys | Intervalo estructurado | Nota: Esta propiedad está en desuso desde IBM SPSS Modeler 16 y posterior. |
| default_ascending | flag | |
| low_distinct_key_count | tag | Especifica que sólo tiene un pequeño número de registros y/o un pequeño número de valores exclusivos del campo(s) clave. |
| keys_pre_sorted | tag | Especifica que todos los registros con los mismos valores clave se agrupan en la entrada. |
| disable_sql_generation | tag | |

Ejemplo para la propiedad `composite_value`

La propiedad `composite_value` tiene el formato general siguiente:

```
node.setKeyedPropertyValue("composite_value", FIELD, FILLOPTION)
```

FILLOPTION tiene el formato [FillType, Option1, Option2, ...].

Ejemplos:

```
node.setKeyedPropertyValue("composite_value", "Age", ["First"])
node.setKeyedPropertyValue("composite_value", "Age", ["last"])
node.setKeyedPropertyValue("composite_value", "Age", ["Total"])
node.setKeyedPropertyValue("composite_value", "Age", ["Average"])
node.setKeyedPropertyValue("composite_value", "Age", ["Min"])
node.setKeyedPropertyValue("composite_value", "Age", ["Max"])
node.setKeyedPropertyValue("composite_value", "Date", ["Earliest"])
node.setKeyedPropertyValue("composite_value", "Date", ["Latest"])
node.setKeyedPropertyValue("composite_value", "Code", ["FirstAlpha"])
node.setKeyedPropertyValue("composite_value", "Code", ["LastAlpha"])
```

Las opciones personalizadas requieren más de un argumento, añadidos como una lista, por ejemplo:

```
node.setKeyedPropertyValue("composite_value", "Name", ["MostFrequent", "FirstRecord"])
node.setKeyedPropertyValue("composite_value", "Date", ["LeastFrequent", "LastRecord"])
node.setKeyedPropertyValue("composite_value", "Pending", ["IncludesValue", "T", "F"])
node.setKeyedPropertyValue("composite_value", "Marital", ["FirstMatch", "Married", "Divorced",
"Separated"])
node.setKeyedPropertyValue("composite_value", "Code", ["Concatenate"])
node.setKeyedPropertyValue("composite_value", "Code", ["Concatenate", "Space"])
node.setKeyedPropertyValue("composite_value", "Code", ["Concatenate", "Comma"])
node.setKeyedPropertyValue("composite_value", "Code", ["Concatenate", "UnderScore"])
```

Ejemplo para la propiedad `composite_values`

La propiedad `composite_values` tiene el formato general siguiente:

```
node.setPropertyValue("composite_values", [
    [FIELD1, [FILLOPTION1]],
    [FIELD2, [FILLOPTION2]],
    .
    .
])
```

Ejemplo:

```
node.setPropertyValue("composite_values", [
    ["Age", ["First"]],
    ["Name", ["MostFrequent", "First"]],
    ["Pending", ["IncludesValue", "T"]],
    ["Marital", ["FirstMatch", "Married", "Divorced", "Separated"]],
    ["Code", ["Concatenate", "Comma"]]
])
```

Propiedades de `extensionprocessnode`



Con el nodo de Transformación de extensión, puede tomar datos de una ruta y aplicar transformaciones a los datos utilizando scripts R o scripts Python para Spark.

Ejemplo de Python para Spark

```
#### Ejemplo de script de Python para Spark
import modeler.api
stream = modeler.script.stream()
node = stream.create("extension_process", "extension_process")
node.setPropertyValue("syntax_type", "Python")

process_script = """
import spss.pyspark.runtime
from pyspark.sql.types import *

cxt = spss.pyspark.runtime.getContext()

if cxt.isComputeDataModelOnly():
    _schema = StructType([StructField("Age", LongType(), nullable=True), \
        StructField("Sex", StringType(), nullable=True), \
        StructField("BP", StringType(), nullable=True), \
        StructField("Na", DoubleType(), nullable=True), \
        StructField("K", DoubleType(), nullable=True), \
        StructField("Drug", StringType(), nullable=True)])
    cxt.setSparkOutputSchema(_schema)
else:
    df = cxt.getSparkInputData()
    print df.dtypes[:]
    _newDF = df.select("Age","Sex","BP","Na","K","Drug")
    print _newDF.dtypes[:]
    cxt.setSparkOutputData(_newDF)
"""

node.setPropertyValue("python_syntax", process_script)
```

Ejemplo de R

```
#### Ejemplo de script de R
node.setPropertyValue("syntax_type", "R")
node.setPropertyValue("r_syntax", ""day<-as.Date(modelerData$dob, format="%Y-%m-%d")
next_day<-day + 1
modelerData<-cbind(modelerData,next_day)
var1<-c(fieldName="Next day",fieldLabel="",fieldStorage="date",fieldMeasure="",fieldFormat="",
fieldRole="")
modelerDataModel<-data.frame(modelerDataModel,var1)""")
```

Tabla 66. Propiedades de *extensionprocessnode*

| Propiedades de <i>extensionprocessnode</i> | Tipo de datos | Descripción de la propiedad |
|--|------------------------------------|--|
| <code>syntax_type</code> | <i>R</i> <i>Python</i> | Especifica qué script se ejecuta: R o Python (R es el valor predeterminado). |
| <code>r_syntax</code> | <i>string</i> | Sintaxis de scripts R a ejecutar. |
| <code>python_syntax</code> | <i>string</i> | Sintaxis de scripts Python a ejecutar. |
| <code>use_batch_size</code> | <i>flag</i> | Habilitar uso de proceso por lotes. |
| <code>batch_size</code> | <i>integer</i> | Especifique el número de registros de datos a incluir en cada lote. |
| <code>convert_flags</code> | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| <code>convert_missing</code> | <i>flag</i> | Opción para convertir valores perdidos al valor NA de R. |
| <code>convert_datetime</code> | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |

Tabla 66. Propiedades de *extensionprocessnode* (continuación)

| Propiedades de <i>extensionprocessnode</i> | Tipo de datos | Descripción de la propiedad |
|--|--------------------|--|
| <code>convert_datetime_class</code> | POSIXct POSIXlt | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |

propiedades de *mergenode*



El nodo Fundir toma varios registros de entrada y crea un registro de salida único que contiene todos o algunos de los campos de entrada. Es útil para fusionar datos desde diferentes orígenes, como datos de clientes internos y datos demográficos adquiridos.

Ejemplo

```
node = stream.create("merge", "My node")
# supongamos que customerdata y salesdata son nodos de importación de base de datos configurados
stream.link(customerdata, node)
stream.link(salesdata, node)
node.setPropertyValue("method", "Keys")
node.setPropertyValue("key_fields", ["id"])
node.setPropertyValue("common_keys", True)
node.setPropertyValue("join", "PartialOuter")
node.setKeyedPropertyValue("outer_join_tag", "2", True)
node.setKeyedPropertyValue("outer_join_tag", "4", True)
node.setPropertyValue("single_large_input", True)
node.setPropertyValue("single_large_input_tag", "2")
node.setPropertyValue("use_existing_sort_keys", True)
node.setPropertyValue("existing_sort_keys", [{"id", "Ascending"}])
```

Tabla 67. propiedades de *mergenode*.

| Propiedad de <i>mergenode</i> | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|---|
| método | Order Claves Condition Rankedcondition | Especifique si los registros se fusionan en el orden en que aparecen en los archivos de datos, si se utilizarán uno o más campos clave para fusionar los registros que tengan el mismo valor en sus campos clave, si los registros se fusionarán si se satisface una condición especificada, o si se debe fusionar cada emparejamiento de fila de los conjuntos primario y todos los secundarios; utilizando la expresión de clasificación para ordenar coincidencias múltiples de menor a mayor. |
| condition | <i>cadena</i> | Si <i>method</i> se establece en <i>Condition</i> , especifica la condición para incluir o descartar registros. |
| key_fields | <i>lista</i> | |
| common_keys | <i>tag</i> | |
| join | Interior FullOuter PartialOuter Anti | |

Tabla 67. propiedades de mergenode (continuación).

| Propiedad de mergenode | Tipo de datos | Descripción de la propiedad |
|-------------------------|--|---|
| outer_join_tag.n | tag | En esta propiedad, <i>n</i> es el nombre de etiqueta tal y como recoge el cuadro de diálogo Seleccionar conjunto de datos. Tenga en cuenta que es posible que existan varios nombres de etiquetas especificados, ya que pueden ser varios los conjuntos de datos que aporten registros incompletos. |
| single_large_input | tag | Determina si se va a usar la optimización para tener una entrada relativamente grande en comparación con el resto de entradas. |
| single_large_input_tag | cadena | Especifica el nombre de etiqueta tal y como se muestra en el cuadro de diálogo Seleccionar conjunto de datos grande. Tenga en cuenta que el uso de esta propiedad es ligeramente distinto que el de la propiedad <code>outer_join_tag</code> (marca frente a cadena), ya que solamente se puede especificar un único conjunto de datos de entrada. |
| use_existing_sort_keys | tag | Determina si las entradas ya se han ordenado en función de uno o varios campos clave. |
| existing_sort_keys | [[<i>'string'</i> , <i>'Ascending'</i>] \ <i>'string'</i> , <i>'Descending'</i>]] | Especifica los campos que ya están ordenados y la dirección en que dicho orden se ha establecido. |
| primary_dataset | cadena | Si <code>method</code> es <code>Rankedcondition</code> , seleccione el conjunto de datos primario de la fusión. Esto se puede considerar como el lado izquierdo de una fusión de unión externa. |
| rename_duplicate_fields | Booleana | Si <code>method</code> es <code>Rankedcondition</code> , y este valor está establecido en <code>Y</code> , si el conjunto de datos fusionado resultante contiene varios campos con el mismo nombre procedentes de orígenes de datos distintos, se añaden las etiquetas respectivas de los orígenes de datos al comienzo de las cabeceras de columna de campo. |
| merge_condition | cadena | |
| ranking_expression | cadena | |
| Num_matches | entero | Número de coincidencias a devolver, en función de <code>merge_condition</code> y <code>ranking_expression</code> . Mínimo 1, máximo 100. |

propiedades rfmaggatenode



El nodo Adición de RFM (actualidad, frecuencia, monetario) permite tomar datos de transacciones históricas de clientes, deshacerse de los datos no utilizados y combinar todos los datos de transacciones restantes en una única fila que indica cuándo hizo negociaciones con los clientes por última vez, cuántas transacciones hicieron y el valor monetario total de dichas transacciones.

Ejemplo

```

node = stream.create("rfmaggregate", "My node")
node.setPropertyValue("relative_to", "Fixed")
node.setPropertyValue("reference_date", "2007-10-12")
node.setPropertyValue("id_field", "CardID")
node.setPropertyValue("date_field", "Date")
node.setPropertyValue("value_field", "Amount")
node.setPropertyValue("only_recent_transactions", True)
node.setPropertyValue("transaction_date_after", "2000-10-01")

```

Tabla 68. propiedades rfmaggregatenode.

| propiedades rfmaggregatenode | Tipo de datos | Descripción de la propiedad |
|------------------------------|------------------|---|
| relative_to | Fixed Today | Especifica la fecha a partir de la que se calculará la actualidad de las transacciones. |
| reference_date | date | Sólo está disponible si se selecciona Fixed en relative_to. |
| contiguous | tag | Si los datos se han clasificado previamente de forma que todos los registros con el mismo ID aparecen en la ruta de datos, al seleccionar esta opción acelerará el procesamiento. |
| id_field | campo | Especifica el campo que desea utilizar para identificar el cliente y sus transacciones. |
| date_field | campo | Especifica el campo de fecha que se utilizará para calcular la actualidad. |
| value_field | campo | Especifica el campo que se utilizará para calcular el valor monetario. |
| extensión | cadena | Especifica un prefijo o sufijo para campos agregados duplicados. |
| add_as | Suffix Prefix | Especifica si la extensión se debe añadir como sufijo o prefijo. |
| discard_low_value_records | tag | Permite utilizar la configuración discard_records_below. |
| discard_records_below | number | Especifique un valor mínimo por debajo del cual no se utilice ningún detalle de transacción al calcular los totales de RFM. Las unidades del valor se relacionan con el campo valor seleccionado. |
| only_recent_transactions | tag | Permite utilizar de la configuración specify_transaction_date o transaction_within_last. |
| specify_transaction_date | tag | |
| transaction_date_after | date | Sólo está disponible si selecciona specify_transaction_date. Especifica la fecha de transacción tras la que se incluirán los registros en su análisis. |
| transaction_within_last | number | Sólo está disponible si selecciona transaction_within_last. Especifica el número y tipo de períodos (días, semanas, meses o años) desde la fecha Calcular actualidad relativa a tras la cual se incluirán los registros en su análisis. |

Tabla 68. propiedades rfmaggregatenode (continuación).

| propiedades rfmaggregatenode | Tipo de datos | Descripción de la propiedad |
|------------------------------|--------------------------------|---|
| transaction_scale | Days Weeks Meses Años | Sólo está disponible si selecciona transaction_within_last. Especifica el número y tipo de períodos (días, semanas, meses o años) desde la fecha Calcular actualidad relativa a tras la cual se incluirán los registros en su análisis. |
| save_r2 | tag | Muestra la fecha de la segunda transacción más reciente para cada cliente. |
| save_r3 | tag | Sólo está disponible si selecciona save_r2. Muestra la fecha de la tercera transacción más reciente para cada cliente. |

Propiedades de Rprocessnode



El nodo Transformación R le permite tomar datos de una ruta de IBM(r) SPSS(r) Modeler y modificar los datos utilizando su propio script R personalizado. Una vez modificados los datos, se devuelven a la cadena.

Ejemplo

```
node = stream.create("rprocess", "My node")
node.setPropertyValue("custom_name", "my_node")
node.setPropertyValue("syntax", """"day<-as.Date(modelerData$dob, format="%Y-%m-%d")
next_day<-day + 1
modelerData<-cbind(modelerData,next_day)
var1<-c(fieldName="Next day",fieldLabel="",fieldStorage="date",fieldMeasure="",fieldFormat="",
fieldRole="")
modelerDataModel<-data.frame(modelerDataModel,var1)""")
node.setPropertyValue("convert_datetime", "POSIXct")
```

Tabla 69. propiedades de Rprocessnode.

| Propiedades de Rprocessnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|------------------------------------|---|
| syntax | cadena | |
| convert_flags | StringsAndDoubles LogicalValues | |
| convert_datetime | flag | |
| convert_datetime_class | POSIXct POSIXlt | |
| convert_missing | flag | |
| use_batch_size | flag | Habilite el uso del proceso por lotes |
| batch_size | entero | Especifique el número de registros de datos que se incluirán en cada lote |

propiedades de samplenode



El nodo Muestrear selecciona un subconjunto de registros. Se admite una variedad de tipos de muestras, entre las que se incluyen las muestras estratificadas, agrupadas en clústeres y no aleatorias (estructuradas). El muestreo puede ser de gran utilidad para mejorar el rendimiento y para seleccionar grupos de registros o transacciones relacionadas para un análisis.

Ejemplo

```
/* Create two Sample nodes to extract
different samples from the same data */
```

```
node = stream.create("sample", "My node")
node.setPropertyValue("method", "Simple")
node.setPropertyValue("mode", "Include")
node.setPropertyValue("sample_type", "First")
node.setPropertyValue("first_n", 500)
```

```
node = stream.create("sample", "My node")
node.setPropertyValue("method", "Complex")
node.setPropertyValue("stratify_by", ["Sex", "Cholesterol"])
node.setPropertyValue("sample_units", "Proportions")
node.setPropertyValue("sample_size_proportions", "Custom")
node.setPropertyValue("sizes_proportions", [["M", "High", "Default"], ["M", "Normal",
"Default"],
["F", "High", 0.3], ["F", "Normal", 0.3]])
```

Tabla 70. propiedades de samplenode.

| Propiedad de samplenode | Tipo de datos | Descripción de la propiedad |
|-------------------------|--------------------------------|--|
| método | Simple Complex | |
| mode | Incluir Descartar | Incluye o descarta los registros que reúnan la condición especificada. |
| sample_type | Primero OneInN RandomPct | Especifica el método de muestreo. |
| first_n | entero | Se incluirán o descartarán los registros hasta el punto de corte especificado. |
| one_in_n | number | Incluye o descarta cada <i>n</i> registros. |
| rand_pct | number | Especifica el porcentaje de registros que incluir o descartar. |
| use_max_size | tag | Activa el uso del parámetro maximum_size. |
| maximum_size | entero | Especifica la muestra más grande que se va a incluir o descartar de la ruta de datos. Esta opción es redundante, por lo que se desactiva cuando se especifican las opciones First e Include. |
| set_random_seed | tag | Activa el uso del parámetro de semillas aleatorias. |
| random_seed | entero | Especifica el valor utilizado como semilla aleatoria. |

Tabla 70. propiedades de *samplenode* (continuación).

| Propiedad de <i>samplenode</i> | Tipo de datos | Descripción de la propiedad |
|--------------------------------------|--|--|
| <code>complex_sample_type</code> | Random Systematic | |
| <code>sample_units</code> | Proportions Counts | |
| <code>sample_size_proportions</code> | Fixed Custom Variable | |
| <code>sample_size_counts</code> | Fixed Custom Variable | |
| <code>fixed_proportions</code> | <i>number</i> | |
| <code>fixed_counts</code> | <i>entero</i> | |
| <code>variable_proportions</code> | <i>campo</i> | |
| <code>variable_counts</code> | <i>campo</i> | |
| <code>use_min_stratum_size</code> | <i>tag</i> | |
| <code>minimum_stratum_size</code> | <i>entero</i> | Esta opción sólo se aplica cuando se toma una muestra compleja con <code>Sample units=Proportions</code> . |
| <code>use_max_stratum_size</code> | <i>tag</i> | |
| <code>maximum_stratum_size</code> | <i>entero</i> | Esta opción sólo se aplica cuando se toma una muestra compleja con <code>Sample units=Proportions</code> . |
| <code>clusters</code> | <i>campo</i> | |
| <code>stratify_by</code> | <i>[campo1 ... campoN]</i> | |
| <code>specify_input_weight</code> | <i>tag</i> | |
| <code>input_weight</code> | <i>campo</i> | |
| <code>new_output_weight</code> | <i>cadena</i> | |
| <code>sizes_proportions</code> | <i>[[string valor cadena][string valor cadena]...]</i> | Si <code>sample_units=proportions</code> y <code>sample_size_proportions=Custom</code> , especifica un valor para cada combinación posible de valores de campos de especificación. |
| <code>default_proportion</code> | <i>number</i> | |
| <code>sizes_counts</code> | <i>[[string valor cadena][string valor cadena]...]</i> | Especifica un valor para cada combinación de valores posible de campos de estratificación. Se utiliza de forma similar a <code>sizes_proportions</code> pero especificando un entero en lugar de una proporción. |
| <code>default_count</code> | <i>number</i> | |

propiedades de *selectnode*



El nodo Seleccionar selecciona o descarta un subconjunto de registros de la ruta de datos en función de una condición específica. Por ejemplo, podría seleccionar los registros que pertenezcan a una región de ventas determinada.

Ejemplo

```
node = stream.create("select", "My node")
node.setPropertyValue("mode", "Include")
node.setPropertyValue("condition", "Age < 18")
```

Tabla 71. propiedades de selectnode.

| Propiedad de selectnode | Tipo de datos | Descripción de la propiedad |
|-------------------------|----------------------|--|
| mode | Incluir Descartar | Especifica si incluir o descartar los registros seleccionados. |
| condition | <i>cadena</i> | Condición para incluir o descartar registros. |

propiedades de sortnode



Los nodos Ordenar organizan registros en orden ascendente o descendente atendiendo a los valores de uno o varios campos.

Ejemplo

```
node = stream.create("sort", "My node")
node.setPropertyValue("keys", [["Age", "Ascending"], ["Sex", "Descending"]])
node.setPropertyValue("default_ascending", False)
node.setPropertyValue("use_existing_keys", True)
node.setPropertyValue("existing_keys", [["Age", "Ascending"]])
```

Tabla 72. propiedades de sortnode.

| Propiedad de sortnode | Tipo de datos | Descripción de la propiedad |
|-----------------------|---------------|--|
| keys | <i>lista</i> | Especifica los campos que desea ordenar. Si no se especifica ninguna dirección, se utilizará la predeterminada. |
| default_ascending | <i>tag</i> | Especifica el orden de clasificación predeterminado. |
| use_existing_keys | <i>tag</i> | Determina si la clasificación se optimiza usando el orden de clasificación anterior de los campos que ya están ordenados. |
| existing_keys | | Especifica los campos que ya están ordenados y la dirección en que dicho orden se ha establecido. Utiliza el mismo formato que las propiedades keys. |

Propiedades de spacetimeboxes



Las cajas-espacio-tiempo (STB) son una extensión de las ubicaciones espaciales demarcadas mediante Geohash. En concreto, una STB es una cadena alfanumérica que representa una región de forma regular de espacio y tiempo.

Tabla 73. Propiedades de spacetimeboxes

| Propiedades de spacetimeboxes | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|--|
| mode | <i>IndividualRecords</i> <i>Hangouts</i> | |
| latitude_field | <i>campo</i> | |
| longitude_field | <i>campo</i> | |
| timestamp_field | <i>campo</i> | |
| densities | <i>[density, density, density...]</i> | <p>Cada densidad es una serie. Por ejemplo: STB_GH8_1DAY</p> <p>Tenga en cuenta que existen límites con respecto a qué densidades son válidas.</p> <p>Para geohash, se pueden utilizar los valores de GH1-GH15.</p> <p>Para la parte temporal, se pueden utilizar los valores siguientes:</p> <p>EVER 1YEAR 1MONTH 1DAY 12HOURS 8HOURS 6HOURS 4HOURS 3HOURS 2HOURS 1HOUR 30MINS 15MINS 10MINS 5MINS 2 MINS 1 MIN 30SECS 15SECS 10SECS 5 SECS 2 SECS 1SEC</p> |
| field_name_extension | <i>cadena</i> | |
| add_extension_as | <i>Prefix</i> <i>Suffix</i> | |
| hangout_density | <i>densidad</i> | Densidad única (ver más arriba) |
| id_field | <i>campo</i> | |

Tabla 73. Propiedades de spacetimexboxes (continuación)

| Propiedades de spacetimexboxes | Tipo de datos | Descripción de la propiedad |
|--------------------------------|--|--------------------------------------|
| qualifying_duration | 1DAY 12HOURS 8HOURS 6HOURS 4HOURS 2HOURS 1HOUR 30MIN 15MIN 10MIN 5MIN 2MIN 1MIN 30SECS 15SECS 10SECS 5SECS 2SECS 1SECS | Esto debe ser una serie. |
| min_events | entero | El valor mínimo es 2 |
| qualifying_pct | entero | Debe estar dentro del rango de 1-100 |

Propiedades streamingtimeseries



El nodo Serie temporal de modalidad continua crea y puntúa modelos de serie temporal en un solo paso.

Nota: Este nodo Serie temporal de modalidad continua sustituye el nodo Resolución de TS original que estaba en desuso en la versión 18 de SPSS Modeler.

Tabla 74. Propiedades streamingtimeseries

| Propiedades streamingtimeseries | Valores | Descripción de la propiedad |
|---------------------------------|---------------------|---|
| targets | campo | El nodo Serie temporal de modalidad continua prevé uno o más objetivos, de forma opcional, utilizando uno o más campos de entrada como predictores. Los campos de frecuencia y ponderación no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| candidate_inputs | [field1 ... fieldN] | Campos de entrada o predictor utilizados por el modelo. |
| use_period | flag | |
| date_time_field | campo | |

Tabla 74. Propiedades streamingtimeseries (continuación)

| Propiedades streamingtimeseries | Valores | Descripción de la propiedad |
|---------------------------------|---|-----------------------------|
| input_interval | Ninguno Desconocido Año Trimestre Mes Semana Día Hora Hour_nonperiod Minuto Minute_nonperiod Segundo Second_nonperiod | |
| period_field | <i>campo</i> | |
| period_start_value | <i>entero</i> | |
| num_days_per_week | <i>entero</i> | |
| start_day_of_week | Sunday Monday Martes Miércoles Jueves Viernes Saturday | |
| num_hours_per_day | <i>entero</i> | |
| start_hour_of_day | <i>entero</i> | |
| timestamp_increments | <i>entero</i> | |
| cyclic_increments | <i>entero</i> | |
| cyclic_periods | <i>lista</i> | |
| output_interval | Ninguno Año Trimestre Mes Semana Día Hora Minuto Segundo | |
| is_same_interval | <i>flag</i> | |
| cross_hour | <i>flag</i> | |
| aggregate_and_distribute | <i>lista</i> | |
| aggregate_default | Media Sum Mode Min Max | |
| distribute_default | Media Sum | |

Tabla 74. Propiedades streamingtimeseries (continuación)

| Propiedades streamingtimeseries | Valores | Descripción de la propiedad |
|----------------------------------|--|--|
| group_default | Media Sum Mode Min Max | |
| missing_imput | Linear_interp Series_mean K_mean K_median Linear_trend | |
| k_span_points | entero | |
| use_estimation_period | flag | |
| estimation_period | Observaciones Times | |
| date_estimation | lista | Solo está disponible si se utiliza date_time_field |
| period_estimation | lista | Solo está disponible si se utiliza use_period |
| observations_type | Latest Más antiguo | |
| observations_num | entero | |
| observations_exclude | entero | |
| method | ExpertModeler Exsmooth Arima | |
| expert_modeler_method | ExpertModeler Exsmooth Arima | |
| consider_seasonal | flag | |
| detect_outliers | flag | |
| expert_outlier_additive | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_innovational | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_transient | flag | |
| expert_outlier_seasonal_additive | flag | |
| expert_outlier_local_trend | flag | |
| expert_outlier_additive_patch | flag | |
| consider_newesmodels | flag | |

Tabla 74. Propiedades streamingtimeseries (continuación)

| Propiedades streamingtimeseries | Valores | Descripción de la propiedad |
|---|--|----------------------------------|
| exsmooth_model_type | Simple HoltLinearTrend BrownsLinearTrend DampedTrend SimpleSeasonal WintersAdditive WintersMultiplicative DampedTrendAdditive DampedTrendMultiplicative MultiplicativeTrendAdditive MultiplicativeSeasonal MultiplicativeTrendMultiplicative MultiplicativeTrend | |
| futureValue_type_method | Compute specify | |
| exsmooth_transformation_type | Ninguno SquareRoot NaturalLog | |
| arma.p | entero | |
| arma.d | entero | |
| arma.q | entero | |
| arma.sp | entero | |
| arma.sd | entero | |
| arma.sq | entero | |
| arma_transformation_type | Ninguno SquareRoot NaturalLog | |
| arma_include_constant | flag | |
| tf_arma.p. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.d. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.q. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.sp. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.sd. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.sq. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.delay. nombredcampo | entero | Para funciones de transferencia. |
| tf_arma.transformation_type. nombredcampo | Ninguno SquareRoot NaturalLog | Para funciones de transferencia. |
| arma_detect_outliers | flag | |
| arma_outlier_additive | flag | |
| arma_outlier_level_shift | flag | |

Tabla 74. Propiedades streamingtimeseries (continuación)

| Propiedades streamingtimeseries | Valores | Descripción de la propiedad |
|---------------------------------|---------------|-----------------------------|
| arima_outlier_innovational | <i>flag</i> | |
| arima_outlier_transient | <i>flag</i> | |
| arima_outlier_seasonal_additive | <i>flag</i> | |
| arima_outlier_local_trend | <i>flag</i> | |
| arima_outlier_additive_patch | <i>flag</i> | |
| conf_limit_pct | <i>real</i> | |
| eventos | <i>campos</i> | |
| forecastperiods | <i>entero</i> | |
| extend_records_into_future | <i>flag</i> | |
| conf_limits | <i>flag</i> | |
| noise_res | <i>flag</i> | |

Propiedades streamingts (en desuso)



Nota: Este nodo Serie temporal de modalidad continua estaba en desuso en la versión 18 de SPSS Modeler y se sustituye por el nodo Serie temporal de modalidad continua que se ha designado para aplicar la potencia de IBM SPSS Analytic Server y procesas grandes cantidades de datos (Big Data).El nodo Generación de análisis TS construye y puntúa modelos de series temporales en un paso, sin la necesidad de un nodo Intervalos de tiempo.

Ejemplo

```
node = stream.create("streamingts", "My node")
node.setPropertyValue("deployment_force_rebuild", True)
node.setPropertyValue("deployment_rebuild_mode", "Count")
node.setPropertyValue("deployment_rebuild_count", 3)
node.setPropertyValue("deployment_rebuild_pct", 11)
node.setPropertyValue("deployment_rebuild_field", "Year")
```

Tabla 75. propiedades streamingts.

| Propiedades de streamingts | Tipo de datos | Descripción de la propiedad |
|----------------------------|------------------------------------|--|
| custom_fields | <i>flag</i> | Si custom_fields=false, se utilizarán los valores de un nodo Tipo situado en un punto anterior de la ruta. Si custom_fields=true, deberán especificarse targets e inputs. |
| targets | [campo1...campoN] | |
| inputs | [campo1...campoN] | |
| method | ExpertModeler Exsmooth Arima | |
| calculate_conf | <i>flag</i> | |
| conf_limit_pct | <i>real</i> | |
| use_time_intervals_node | <i>flag</i> | Si use_time_intervals_node=true, se utilizarán los valores de un nodo Intervalos de tiempo situado en un punto anterior de la ruta. Si use_time_intervals_node=false, deberán especificarse interval_offset_position, interval_offset e interval_type. |

Tabla 75. propiedades streamings (continuación).

| Propiedades de streamings | Tipo de datos | Descripción de la propiedad |
|----------------------------------|--|--|
| interval_offset_position | LastObservation LastRecord | LastObservation se refiere a la Última observación válida . LastRecord se refiere a la Cuenta hacia atrás a partir del último registro . |
| interval_offset | número | |
| interval_type | Períodos Años Trimestres Meses WeeksNonPeriodic DaysNonPeriodic HoursNonPeriodic MinutesNonPeriodic SecondsNonPeriodic | |
| eventos | campos | |
| expert_modeler_method | AllModels Exsmooth Arima | |
| consider_seasonal | flag | |
| detect_outliers | flag | |
| expert_outlier_additive | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_innovational | flag | |
| expert_outlier_transient | flag | |
| expert_outlier_seasonal_additive | flag | |
| expert_outlier_local_trend | flag | |
| expert_outlier_additive_patch | flag | |
| exsmooth_model_type | Simple HoltsLinearTrend BrownsLinearTrend DampedTrend SimpleSeasonal WintersAdditive WintersMultiplicative | |
| exsmooth_transformation_type | Ninguno SquareRoot NaturalLog | |
| arima_p | entero | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_d | entero | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_q | entero | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_sp | entero | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_sd | entero | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_sq | entero | Misma propiedad que el nodo de modelado Serie de tiempo |

Tabla 75. propiedades streamings (continuación).

| Propiedades de streamings | Tipo de datos | Descripción de la propiedad |
|--|-------------------------------------|---|
| arima_transformation_type | Ninguno SquareRoot NaturalLog | Misma propiedad que el nodo de modelado Serie de tiempo |
| arima_include_constant | flag | Misma propiedad que el nodo de modelado Serie de tiempo |
| tf_arima_p.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_d.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_q.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_sp.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_sd.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_sq.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_delay.nombrecampo | entero | Misma propiedad que el nodo de modelado Serie de tiempo. Para funciones de transferencia. |
| tf_arima_transformation_type. nombredecampo | Ninguno SquareRoot NaturalLog | |
| arima_detect_outlier_mode | Ninguno Automatic | |
| arima_outlier_additive | flag | |
| arima_outlier_level_shift | flag | |
| arima_outlier_innovational | flag | |
| arima_outlier_transient | flag | |
| arima_outlier_seasonal_additive | flag | |
| arima_outlier_local_trend | flag | |
| arima_outlier_additive_patch | flag | |
| deployment_force_rebuild | flag | |
| deployment_rebuild_mode | Recuento Porcentaje | |
| deployment_rebuild_count | número | |
| deployment_rebuild_pct | número | |
| deployment_rebuild_field | <campo> | |

Propiedades de cplexnode



El nodo Optimización de CPLEX proporciona la capacidad de utilizar la optimización basada en CPLEX (matemática compleja) a través de un archivo de modelo OPL (Optimization Programming Language - Lenguaje de programación de optimización). Esta funcionalidad está disponible en el producto IBM Analytical Decision Management, pero ahora también puede utilizar el nodo CPLEX en SPSS Modeler sin necesitar IBM Analytical Decision Management.

Para obtener más información sobre la optimización de CPLEX y OPL, consulte la documentación de IBM Analytical Decision Management.

Tabla 76. Propiedades de cplexnode

| Propiedades de cplexnode | Tipo de datos | Descripción de la propiedad |
|------------------------------------|---------------|--|
| opl_model_text | <i>string</i> | Programa de scripts de OPL (Optimization Programming Language - Lenguaje de programación de optimización) que el nodo de Optimización de CPLEX ejecutará y, a continuación, generará el resultado de optimización. |
| opl_tuple_set_name | <i>string</i> | Nombre de conjunto de tuplas en el modelo OPL que corresponde a los datos de entrada. |
| opl_data_text | <i>string</i> | Definición de algunas variables o datos utilizados para OPL. |
| output_value_mode | <i>string</i> | Los valores posibles son raw o dvar. Si se proporciona dvar, en la pestaña Resultado el usuario debe especificar el nombre de variable de función de objeto en OPL para el resultado. Si se especifica raw, la función de objetivo se producirá directamente, independientemente del nombre. |
| objective_function_value_fieldname | <i>string</i> | Nombre de campo a utilizar en el resultado. El valor predeterminado es _OBJECTIVE. |

Capítulo 11. Propiedades de nodos de operaciones con campos

propiedades de anonymizenode



El nodo Anonimizar transforma la manera en que se representan los nombres y los valores de los campos a partir de ese punto de la ruta, lo que permite disfrazar los datos originales. Puede resultar útil si desea permitir que otros usuarios generen modelos utilizando datos confidenciales, como los nombres de los clientes u otros detalles.

Ejemplo

```
stream = modeler.script.stream()
varfilenode = stream.createAt("variablefile", "File", 96, 96)
varfilenode.setPropertyValue("full_filename", "$CLEO/DEMOS/DRUG1n")
node = stream.createAt("anonymize", "My node", 192, 96)
# Anonimizar nodo requiere los campos de entrada al establecer los valores
stream.link(varfilenode, node)
node.setKeyedPropertyValue("enable_anonymize", "Age", True)
node.setKeyedPropertyValue("transformation", "Age", "Random")
node.setKeyedPropertyValue("set_random_seed", "Age", True)
node.setKeyedPropertyValue("random_seed", "Age", 123)
node.setKeyedPropertyValue("enable_anonymize", "Drug", True)
node.setKeyedPropertyValue("use_prefix", "Drug", True)
node.setKeyedPropertyValue("prefix", "Drug", "myprefix")
```

Tabla 77. propiedades de anonymizenode

| Propiedades de anonymizenode | Tipo de datos | Descripción de la propiedad |
|------------------------------|-----------------|--|
| enable_anonymize | <i>flag</i> | Cuando se establece en True, activa la anonimización de los valores de los campos (equivale a seleccionar Sí para dicho campo en la columna Anonimizar valores). |
| use_prefix | <i>flag</i> | Cuando se establece en True, se utilizará un prefijo personalizado, si es que se ha establecido uno. Se aplica a los campos que se anonimizarán mediante el método Hash y es equivalente a elegir el botón de radio Personalizado en el cuadro de diálogo Reemplazar valores correspondiente a dicho campo. |
| prefix | <i>cadena</i> | Equivale a escribir un prefijo en el cuadro de texto del cuadro de diálogo Reemplazar valores. El prefijo predeterminado es el valor predeterminado, si no se ha especificado otra cosa. |
| transformation | Random Fixed | Determina si los parámetros de transformación que se aplican a un campo anonimizado mediante el método Transformar serán aleatorios o fijos. |
| set_random_seed | <i>flag</i> | Cuando se establece en True, se utilizará el valor inicial especificado (si transformation también se ha establecido en Random). |
| random_seed | <i>entero</i> | Cuando set_random_seed se establece en True, esta es la semilla para el número aleatorio. |
| scale | <i>número</i> | Cuando transformation se establece en Fixed, se utiliza este valor para "scale by" (escalar por). El valor máximo de la escala suele ser 10, pero puede reducirse para evitar desbordamientos. |

Tabla 77. propiedades de anonymizenode (continuación)

| Propiedades de anonymizenode | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------|--|
| translate | número | Cuando transformation se establece en Fixed, se utiliza este valor para "translate" (convertir). El valor máximo de traslación suele ser 1000, pero puede reducirse para evitar desbordamientos. |

properties autodatapreprenode



El nodo de preparación automática de datos (ADP) puede analizar sus datos e identificar los valores fijos, cribar los campos problemáticos o que no serán útiles y derivar nuevos atributos cuando sea necesario y mejorar el rendimiento mediante técnicas de cribado y muestreo inteligente. Puede utilizar el nodo de forma totalmente automática, permitiendo que el nodo seleccione y aplique valores fijos, o bien puede tener una vista previa de los cambios antes de que se apliquen y aceptarlos o rechazarlos.

Ejemplo

```
node = stream.create("autodataprep", "My node")
node.setPropertyValue("objective", "Balanced")
node.setPropertyValue("excluded_fields", "Filter")
node.setPropertyValue("prepare_dates_and_times", True)
node.setPropertyValue("compute_time_until_date", True)
node.setPropertyValue("reference_date", "Today")
node.setPropertyValue("units_for_date_durations", "Automatic")
```

Tabla 78. properties autodatapreprenode

| properties autodatapreprenode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|--|
| objective | Balanced Speed Exactitud Personalizado | |
| custom_fields | flag | Si es verdadero, le permite especificar el objetivo, la entrada y otros campos del nodo actual. Si es falso, se utiliza la configuración actual de un nodo Tipo situado en un punto anterior de la ruta. |
| target | campo | Especifica un campo de objetivo único. |
| inputs | [field1 ... fieldN] | Campos de entrada o predictor utilizados por el modelo. |
| use_frequency | flag | |
| frequency_field | campo | |
| use_weight | flag | |
| weight_field | campo | |
| excluded_fields | Filter Ninguno | |
| if_fields_do_not_match | StopExecution ClearAnalysis | |
| prepare_dates_and_times | flag | Controla el acceso a todos los campos de fecha y hora |
| compute_time_until_date | flag | |

Tabla 78. *properties autodatapreprenode* (continuación)

| properties autodatapreprenode | Tipo de datos | Descripción de la propiedad |
|--------------------------------------|-----------------------------|------------------------------------|
| reference_date | Today Fixed | |
| fixed_date | <i>fecha</i> | |
| units_for_date_durations | Automatic Fixed | |
| fixed_date_units | Años Meses Days | |
| compute_time_until_time | <i>flag</i> | |
| reference_time | CurrentTime Fixed | |
| fixed_time | <i>hora</i> | |
| units_for_time_durations | Automatic Fixed | |
| fixed_date_units | Hours Minutes Seconds | |
| extract_year_from_date | <i>flag</i> | |
| extract_month_from_date | <i>flag</i> | |
| extract_day_from_date | <i>flag</i> | |
| extract_hour_from_time | <i>flag</i> | |
| extract_minute_from_time | <i>flag</i> | |
| extract_second_from_time | <i>flag</i> | |
| exclude_low_quality_inputs | <i>flag</i> | |
| exclude_too_many_missing | <i>flag</i> | |
| maximum_percentage_missing | <i>número</i> | |
| exclude_too_many_categories | <i>flag</i> | |
| maximum_number_categories | <i>número</i> | |
| exclude_if_large_category | <i>flag</i> | |
| maximum_percentage_category | <i>número</i> | |
| prepare_inputs_and_target | <i>flag</i> | |
| adjust_type_inputs | <i>flag</i> | |
| adjust_type_target | <i>flag</i> | |
| reorder_nominal_inputs | <i>flag</i> | |
| reorder_nominal_target | <i>flag</i> | |
| replace_outliers_inputs | <i>flag</i> | |
| replace_outliers_target | <i>flag</i> | |
| replace_missing_continuous_inputs | <i>flag</i> | |
| replace_missing_continuous_target | <i>flag</i> | |
| replace_missing_nominal_inputs | <i>flag</i> | |
| replace_missing_nominal_target | <i>flag</i> | |
| replace_missing_ordinal_inputs | <i>flag</i> | |

Tabla 78. *properties autodatapreprenode* (continuación)

| properties autodatapreprenode | Tipo de datos | Descripción de la propiedad |
|--------------------------------------|----------------------|------------------------------------|
| replace_missing_ordinal_target | <i>flag</i> | |
| maximum_values_for_ordinal | <i>número</i> | |
| minimum_values_for_continuous | <i>número</i> | |
| outlier_cutoff_value | <i>número</i> | |
| outlier_method | Replace Eliminar | |
| rescale_continuous_inputs | <i>flag</i> | |
| rescaling_method | MinMax ZScore | |
| min_max_minimum | <i>número</i> | |
| min_max_maximum | <i>número</i> | |
| z_score_final_mean | <i>número</i> | |
| z_score_final_sd | <i>número</i> | |
| rescale_continuous_target | <i>flag</i> | |
| target_final_mean | <i>número</i> | |
| target_final_sd | <i>número</i> | |
| transform_select_input_fields | <i>flag</i> | |
| maximize_association_with_target | <i>flag</i> | |
| p_value_for_merging | <i>número</i> | |
| merge_ordinal_features | <i>flag</i> | |
| merge_nominal_features | <i>flag</i> | |
| minimum_cases_in_category | <i>número</i> | |
| bin_continuous_fields | <i>flag</i> | |
| p_value_for_binning | <i>número</i> | |
| perform_feature_selection | <i>flag</i> | |
| p_value_for_selection | <i>número</i> | |
| perform_feature_construction | <i>flag</i> | |
| transformed_target_name_extension | <i>cadena</i> | |
| transformed_inputs_name_extension | <i>cadena</i> | |
| constructed_features_root_name | <i>cadena</i> | |
| years_duration_name_extension | <i>cadena</i> | |
| months_duration_name_extension | <i>cadena</i> | |
| days_duration_name_extension | <i>cadena</i> | |
| hours_duration_name_extension | <i>cadena</i> | |
| minutes_duration_name_extension | <i>cadena</i> | |
| seconds_duration_name_extension | <i>cadena</i> | |
| year_cyclical_name_extension | <i>cadena</i> | |
| month_cyclical_name_extension | <i>cadena</i> | |
| day_cyclical_name_extension | <i>cadena</i> | |
| hour_cyclical_name_extension | <i>cadena</i> | |

Tabla 78. *properties autodatapreprenode* (continuación)

| properties autodatapreprenode | Tipo de datos | Descripción de la propiedad |
|--------------------------------|---------------|-----------------------------|
| minute_cyclical_name_extension | cadena | |
| second_cyclical_name_extension | cadena | |

propiedades de astimeintervalsnode



Utilice el nodo Intervalos de tiempo para especificar intervalos y derive un nuevo campo de tiempo para el cálculo y la previsión. Se admite una gama completa de intervalos de tiempo que abarca desde segundos a años.

Tabla 79. *propiedades de astimeintervalsnode*

| Propiedades de astimeintervalsnode | Tipo de datos | Descripción de la propiedad |
|------------------------------------|-------------------------------|---|
| time_field | campo | Solo puede aceptar un único campo continuo. Ese campo lo utiliza el nodo como la clave de agregación para convertir el intervalo. Si se utiliza aquí un campo de entero, se considera como un índice de tiempo. |
| dimensiones | [campo1 campo2 ... campon] | Estos campos se utilizan para crear series temporales individuales basándose en los valores de campo. |
| fields_to_aggregate | [campo1 campo2 ... campon] | Estos campos se agregan como parte del cambio del período del campo de tiempo. Los campos no incluidos en este selector se filtran y excluyen de los datos que salen del nodo. |

propiedades de binningnode



El nodo Intervalos crea automáticamente nuevos campos nominales (conjunto) en función de los valores de uno o más campos continuos (rango numérico) existentes. Por ejemplo, puede transformar un campo de ingresos continuo en un campo categórico nuevo que contenga grupos de ingresos como desviaciones desde la media. Una vez creados los intervalos para el campo nuevo, puede generar un nodo Derivar en función de los puntos de corte.

Ejemplo

```
node = stream.create("binning", "My node")
node.setPropertyValue("fields", ["Na", "K"])
node.setPropertyValue("method", "Rank")
node.setPropertyValue("fixed_width_name_extension", "_binned")
node.setPropertyValue("fixed_width_add_as", "Suffix")
node.setPropertyValue("fixed_bin_method", "Count")
node.setPropertyValue("fixed_bin_count", 10)
node.setPropertyValue("fixed_bin_width", 3.5)
node.setPropertyValue("tile10", True)
```

Tabla 80. propiedades de binningnode

| Propiedad de binningnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---|---|
| campos | [campo1 campo2 ... campon] | Los campos continuos (rango numérico) pendientes de transformación. Se pueden crear intervalos de varios campos de forma simultánea. |
| method | FixedWidth EqualCount Rank SDev Optimal | Método utilizado para determinar los puntos de corte de los intervalos de campo nuevos (categorías). |
| rcalculate_bins | Always IfNecessary | Especifica si se vuelven a calcular los intervalos y los datos se colocan en el intervalo adecuado cada vez que se ejecuta el nodo o si los datos sólo se añaden a los intervalos existentes y cualquier nuevo intervalo que se haya añadido. |
| fixed_width_name_extension | cadena | La extensión predeterminada es <i>_BIN</i> . |
| fixed_width_add_as | Suffix Prefix | Determina si la extensión se debe añadir al principio (prefijo) o al final (sufijo) del nombre de campo. La extensión predeterminada es <i>income_BIN</i> . |
| fixed_bin_method | Width Count | |
| fixed_bin_count | entero | Especifica un número entero para determinar el número de intervalos de anchura fija (categorías) para los nuevos campos. |
| fixed_bin_width | real | Valor (entero o real) para calcular el ancho del intervalo. |
| equal_count_name_extensión | cadena | La extensión predeterminada es <i>_TILE</i> . |
| equal_count_add_as | Suffix Prefix | Especifica una extensión, sufijo o prefijo, utilizada para el nombre de los campos generados con p-tiles estándar. La extensión predeterminada es <i>_TILE</i> más <i>N</i> , donde <i>N</i> es el número de cuantil. |
| tile4 | flag | Genera cuatro intervalos de cuantiles, cada uno con el 25% de los casos. |
| tile5 | flag | Genera cinco intervalos de quintiles. |
| tile10 | flag | Genera 10 intervalos de deciles. |
| tile20 | flag | Genera 20 intervalos de veintiles. |
| tile100 | flag | Genera 100 intervalos de percentiles. |
| use_custom_tile | flag | |
| custom_tile_name_extension | cadena | La extensión predeterminada es <i>_TILEN</i> . |
| custom_tile_add_as | Suffix Prefix | |
| custom_tile | entero | |

Tabla 80. propiedades de binningnode (continuación)

| Propiedad de binningnode | Tipo de datos | Descripción de la propiedad |
|--------------------------------|--------------------------|--|
| equal_count_method | RecordCount ValueSum | El método RecordCount trata de asignar el mismo número de registros a cada intervalo, mientras que ValueSum asigna registros de manera que la suma de los valores de cada intervalo sea la misma. |
| tied_values_method | Next Actual Random | Especifica en qué intervalo se van a insertar los datos de valor empatado. |
| rank_order | Ascending Descending | Esta propiedad incluye Ascending (el valor más bajo se marca con 1) o Descending (el valor más alto se marca con 1). |
| rank_add_as | Suffix Prefix | Esta opción se aplica al rango, rango fraccional y rango como porcentaje. |
| rango | <i>flag</i> | |
| rank_name_extension | <i>cadena</i> | La extensión predeterminada es <i>_RANK</i> . |
| rank_fractional | <i>flag</i> | Establece rangos de casos en los que el valor del campo nuevo es igual al rango dividido por la suma de las ponderaciones de los casos que no están perdidos. Los rangos fraccionales están dentro del rango de 0-1. |
| rank_fractional_name_extension | <i>cadena</i> | La extensión predeterminada es <i>_F_RANK</i> . |
| rank_pct | <i>flag</i> | Cada rango se divide por el número de registros con valores válidos y se multiplica por 100. Los rangos fraccionales de porcentaje están dentro del rango de 1-100. |
| rank_pct_name_extension | <i>cadena</i> | La extensión predeterminada es <i>_P_RANK</i> . |
| sdev_name_extension | <i>cadena</i> | |
| sdev_add_as | Suffix Prefix | |
| sdev_count | One Two Three | |
| optimal_name_extension | <i>cadena</i> | La extensión predeterminada es <i>_OPTIMAL</i> . |
| optimal_add_as | Suffix Prefix | |
| optimal_supervisor_field | <i>campo</i> | Campo elegido como campo supervisor, con el que se relacionan los campos seleccionados para los intervalos. |
| optimal_merge_bins | <i>flag</i> | Especifica que todos los intervalos con un número pequeño de casos se añadirán a un intervalo vecino de mayor tamaño. |
| optimal_small_bin_threshold | <i>entero</i> | |
| optimal_pre_bin | <i>flag</i> | Indica si debe agruparse previamente en intervalos el conjunto de datos. |
| optimal_max_bins | <i>entero</i> | Especifica un límite superior con el fin de evitar que se genere un número desmesurado de intervalos. |

Tabla 80. propiedades de binningnode (continuación)

| Propiedad de binningnode | Tipo de datos | Descripción de la propiedad |
|--------------------------|------------------------|-----------------------------|
| optimal_lower_end_point | Inclusive Exclusive | |
| optimal_first_bin | Unbounded Bounded | |
| optimal_last_bin | Unbounded Bounded | |

propiedades de derivenode



El nodo Derivar modifica los valores de datos o crea campos nuevos desde uno o más campos existentes. Crea campos del tipo fórmula, marca, nominal, estado, recuento y condicional.

Ejemplo 1

```
# Crear y configurar un nodo de campo Derivar marca
node = stream.create("derive", "My node")
node.setPropertyValue("new_name", "DrugX_Flag")
node.setPropertyValue("result_type", "Flag")
node.setPropertyValue("flag_true", "1")
node.setPropertyValue("flag_false", "0")
node.setPropertyValue("flag_expr", "'Drug' == \"drugX\"")

# Crear y configurar un nodo de campo Derivar condicional
node = stream.create("derive", "My node")
node.setPropertyValue("result_type", "Conditional")
node.setPropertyValue("cond_if_cond", "@OFFSET(\"Age\", 1) = \"Age\"")
node.setPropertyValue("cond_then_expr", "@OFFSET(\"Age\", 1) = \"Age\" >> @INDEX")
node.setPropertyValue("cond_else_expr", "\"Age\"")
```

Ejemplo 2

Este script presupone que existen dos columnas numéricas denominadas XPos e YPos que representan las coordenadas X e Y de un punto (por ejemplo, donde se ha producido un evento). El script crea un nodo Derivar que calcula una columna geoespacial de coordenadas X e Y que representan dicho punto en un sistema de coordenadas específico:

```
stream = modeler.script.stream()
# Otro código de configuración de ruta
node = stream.createAt("derive", "Location", 192, 96)
node.setPropertyValue("new_name", "Location")
node.setPropertyValue("formula_expr", "['XPos', 'YPos']")
node.setPropertyValue("formula_type", "Geospatial")
# Ahora que hemos definido el tipo de medición general, defina los
# detalles del objeto geoespacial
node.setPropertyValue("geo_type", "Point")
node.setPropertyValue("has_coordinate_system", True)
node.setPropertyValue("coordinate_system", "ETRS_1989_EPSG_Arctic_zone_5-47")
```

Tabla 81. propiedades de derivenode

| Propiedad de derivenode | Tipo de datos | Descripción de la propiedad |
|-------------------------|---------------|-----------------------------|
| new_name | cadena | Nombre del campo nuevo. |

Tabla 81. propiedades de derivenode (continuación)

| Propiedad de derivenode | Tipo de datos | Descripción de la propiedad |
|-------------------------|---|--|
| mode | Single Múltiples | Especifica si los campos son únicos o múltiples. |
| campos | lista | Se utiliza en modo Múltiple solamente para seleccionar varios campos. |
| name_extension | cadena | Especifica la extensión de los nombres de los nuevos campos. |
| add_as | Suffix Prefix | Añade la extensión como un prefijo (al principio) o como un sufijo (al final) del nombre de los campos. |
| result_type | Formula Flag Set State Count Conditional | Los seis tipos de campos nuevos que se pueden crear. |
| formula_expr | cadena | Expresión para calcular un nuevo valor de campo en el nodo Derivar. |
| flag_expr | cadena | |
| flag_true | cadena | |
| flag_false | cadena | |
| set_default | cadena | |
| set_value_cond | cadena | Estructurada para proporcionar la condición asociada a un valor dado. |
| state_on_val | cadena | Especifica el valor del campo nuevo cuando se cumple la condición Activado. |
| state_off_val | cadena | Especifica el valor del campo nuevo cuando se cumple la condición Desactivado. |
| state_on_expression | cadena | |
| state_off_expression | cadena | |
| state_initial | On Off | Asigna a cada registro del nuevo campo un valor inicial activado (On) o desactivado (Off). Este valor puede cambiar a medida que se cumplan las condiciones. |
| count_initial_val | cadena | |
| count_inc_condition | cadena | |
| count_inc_expression | cadena | |
| count_reset_condition | cadena | |
| cond_if_cond | cadena | |
| cond_then_expr | cadena | |
| cond_else_expr | cadena | |

Tabla 81. propiedades de derivenode (continuación)

| Propiedad de derivenode | Tipo de datos | Descripción de la propiedad |
|-------------------------|---|--|
| formula_measure_type | Range / MeasureType.RANGE Discrete / MeasureType.DISCRETE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS Collection / MeasureType.COLLECTION Geospatial / MeasureType.GEOSPATIAL | Esta propiedad se puede utilizar para definir la medición asociada con el campo derivado. La función de establecimiento se puede pasar como una cadena o uno de los valores MeasureType. El método de obtención siempre devolverá los valores MeasureType. |
| collection_measure | Range / MeasureType.RANGE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS | Para campos de recopilación (listas con profundidad 0), esta propiedad define el tipo de medición asociado con los valores subyacentes. |
| geo_type | Point Multipunto Cadena lineal Cadena multilínea Polígono Multipolígono | En campos geoespaciales, esta propiedad define el tipo del objeto geoespacial representado por este campo. Debería ser coherente con la profundidad de lista de los valores |
| has_coordinate_system | booleano | En campos geoespaciales, esta propiedad define si este campo tiene un sistema de coordenadas |
| coordinate_system | cadena | En campos geoespaciales, esta propiedad define el sistema de coordenadas para este campo |

propiedades de ensemblenode



El nodo Conjunto combina dos o más nugget de modelo para obtener predicciones más precisas que pueden conseguirse de cualquier modelo.

Ejemplo

```
# Crear y configurar un nodo Conjunto
# Utilizar este nodo con los modelos en demos\streams\pm_module\pm_binaryclassifier.str
node = stream.create("ensemble", "My node")
node.setPropertyValue("ensemble_target_field", "response")
node.setPropertyValue("filter_individual_model_output", False)
node.setPropertyValue("flag_ensemble_method", "ConfidenceWeightedVoting")
node.setPropertyValue("flag_voting_tie_selection", "HighestConfidence")
```

Tabla 82. propiedades de ensemblenode.

| propiedades de ensemblenode | Tipo de datos | Descripción de la propiedad |
|--------------------------------|---------------|---|
| ensemble_target_field | campo | Especifica el campo objetivo de todos los modelos utilizados en el conjunto. |
| filter_individual_model_output | tag | Especifica si los resultados de puntuación de los modelos individuales se deben eliminar. |

Tabla 82. propiedades de *ensembnode* (continuación).

| propiedades de <i>ensembnode</i> | Tipo de datos | Descripción de la propiedad |
|--|---|--|
| <code>flag_ensemble_method</code> | Voting ConfidenceWeightedVoting RawPropensityWeightedVoting AdjustedPropensityWeightedVoting HighestConfidence AverageRawPropensity AverageAdjustedPropensity | Especifica el método utilizado para determinar la puntuación del conjunto. Este conjunto sólo se aplica si el objetivo seleccionado es un campo de marca. |
| <code>set_ensemble_method</code> | Voting ConfidenceWeightedVoting HighestConfidence | Especifica el método utilizado para determinar la puntuación del conjunto. Este conjunto sólo se aplica si el objetivo seleccionado es un campo nominal. |
| <code>flag_voting_tie_selection</code> | Random HighestConfidence RawPropensity AdjustedPropensity | Si se selecciona un método de votación, especifica cómo se resolverán los empates. Este conjunto sólo se aplica si el objetivo seleccionado es un campo de marca. |
| <code>set_voting_tie_selection</code> | Random HighestConfidence | Si se selecciona un método de votación, especifica cómo se resolverán los empates. Este conjunto sólo se aplica si el objetivo seleccionado es un campo nominal. |
| <code>calculate_standard_error</code> | <i>tag</i> | Si el campo objetivo es continuo, se ejecuta un error estándar de forma predeterminada para calcular la diferencia entre los valores medidos o estimados y los valores true; y para mostrar si las estimaciones coinciden. |

propiedades de *fillernode*



El nodo Rellenar sustituye valores de campos y cambia el almacenamiento. Puede sustituir los valores en función de una condición CLEM, como `@BLANK(@FIELD)`. También puede sustituir todos los espacios vacíos o valores nulos por un valor específico. Un nodo Rellenar suelen utilizarse junto con un nodo Tipo para sustituir valores perdidos.

Ejemplo

```
node = stream.create("filler", "My node")
node.setPropertyValue("fields", ["Age"])
node.setPropertyValue("replace_mode", "Always")
node.setPropertyValue("condition", "(\"Age\" > 60) and (\"Sex\" = \"M\")")
node.setPropertyValue("replace_with", "\"old man\"")
```

Tabla 83. propiedades de *fillernode*

| Propiedad de <i>fillernode</i> | Tipo de datos | Descripción de la propiedad |
|--------------------------------|---------------|---|
| <code>campos</code> | <i>lista</i> | Campos del conjunto de datos cuyos valores se van a examinar y sustituir. |

Tabla 83. propiedades de fillernode (continuación)

| Propiedad de fillernode | Tipo de datos | Descripción de la propiedad |
|-------------------------|--|--|
| replace_mode | Always Conditional Vacío Null BlankAndNull | Se pueden sustituir todos los valores, valores vacíos o valores nulos, o bien reemplazar aquellos basados en una condición específica. |
| condition | cadena | |
| replace_with | cadena | |

propiedades de filternode



El nodo Filtrar filtra (descarta) campos, vuelve a nombrar campos y correlaciona campos de nodo de origen a otro.

Ejemplo

```
node = stream.create("filter", "My node")
node.setPropertyValue("default_include", True)
node.setKeyedPropertyValue("new_name", "Drug", "Chemical")
node.setKeyedPropertyValue("include", "Drug", False)
```

Usando la propiedad default_include. Tenga en cuenta que, si establece el valor de la propiedad `default_include`, no se incluirán o excluirán automáticamente todos los campos, sino que simplemente se determinará el valor predeterminado de los seleccionados actualmente. Esto equivale funcionalmente a pulsar en el botón **Incluir campos de forma predeterminada** del cuadro de diálogo del nodo Filtrar. Por ejemplo, imagine que ejecuta el siguiente script:

```
node = modeler.script.stream().create("filter", "Filter")
node.setPropertyValue("default_include", False)
# Incluir estos dos campos en la lista
for f in ["Age", "Sex"]:
    node.setKeyedPropertyValue("include", f, True)
```

Esto hará que el nodo pase los campos *Edad* y *Sexo* y descarte el resto. Ahora, imagine que ejecuta de nuevo el mismo script, pero designa dos campos diferentes:

```
node = modeler.script.stream().create("filter", "Filter")
node.setPropertyValue("default_include", False)
# Incluir estos dos campos en la lista
for f in ["BP", "Na"]:
    node.setKeyedPropertyValue("include", f, True)
```

De esta forma, se añadirán dos campos más al filtro, de manera que pasan un total de cuatro campos (*Edad*, *Sexo*, *PS* y *Na*). En otras palabras, al volver a establecer el valor de `default_include` en `False` no se restablecen automáticamente todos los campos.

Si lo desea, si ahora cambia `default_include` a `True` (ya sea usando un script o en el cuadro de diálogo del nodo Filtrar), cambiará el comportamiento de forma que los cuatro campos enumerados anteriormente no se incluirían, sino que quedarían descartados. Si no está seguro, se recomienda experimentar con los controles del cuadro de diálogo del nodo Filtrar para entender esta interacción.

Tabla 84. propiedades de filternode

| Propiedad de filternode | Tipo de datos | Descripción de la propiedad |
|-------------------------|---------------|---|
| default_include | flag | Propiedad con clave para especificar si el comportamiento predeterminado es para pasar o filtrar los campos: Tenga en cuenta que, si se establece esta propiedad, no se incluirán o excluirán automáticamente todos los campos, sino que simplemente se determinará si los campos seleccionados se incluyen o excluyen de forma predeterminada. Vea el ejemplo que se muestra a continuación para ver más comentarios. |
| include | flag | Propiedad con clave para incluir y eliminar el campo. |
| new_name | cadena | |

propiedades de historynode



El nodo Historial se utiliza para crear campos nuevos que contienen datos de los campos de registros anteriores. Los nodos Historial se suelen utilizar para los datos secuenciales, como los datos de series temporales. Antes de utilizar un nodo Historial, puede desear ordenar los datos utilizando un nodo Ordenar.

Ejemplo

```
node = stream.create("history", "My node")
node.setPropertyValue("fields", ["Drug"])
node.setPropertyValue("offset", 1)
node.setPropertyValue("span", 3)
node.setPropertyValue("unavailable", "Discard")
node.setPropertyValue("fill_with", "undef")
```

Tabla 85. propiedades de historynode

| Propiedad de historynode | Tipo de datos | Descripción de la propiedad |
|--------------------------|----------------------------|--|
| campos | lista | Campos para los que desea un historial. |
| offset | número | Especifica el último registro anterior al registro actual desde el que desea extraer valores del campo histórico. |
| span | número | Especifica el número de registros anteriores de los que desea extraer valores. |
| unavailable | Descartar Leave Fill | Para tratar registros que no tienen valores de historial, suele hacer referencia a los primeros registros (en la parte superior del conjunto de datos), de los que no hay registros previos que utilizar como historial. |
| fill_with | String Number | Especifica un valor o cadena que utilizar en el caso de los registros en los que no existen valores de historial disponibles. |

propiedades de partitionnode



El nodo Partición genera un campo de partición, que divide los datos en subconjuntos diferentes para las fases de entrenamiento, comprobación y validación en la generación del modelo.

Ejemplo

```
node = stream.create("partition", "My node")
node.setPropertyValue("create_validation", True)
node.setPropertyValue("training_size", 33)
node.setPropertyValue("testing_size", 33)
node.setPropertyValue("validation_size", 33)
node.setPropertyValue("set_random_seed", True)
node.setPropertyValue("random_seed", 123)
node.setPropertyValue("value_mode", "System")
```

Tabla 86. propiedades de partitionnode

| Propiedad de partitionnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|------------------------------------|--|
| new_name | cadena | Nombre del campo de partición generado por el nodo. |
| create_validation | flag | Especifica si se debe crear una partición de validación. |
| training_size | entero | Porcentaje de registros (0-100) que se van a asignar a la partición de entrenamiento. |
| testing_size | entero | Porcentaje de registros (0-100) que se van a asignar a la partición de comprobación. |
| validation_size | entero | Porcentaje de registros (0-100) que se van a asignar a la partición de entrenamiento. Se ignora si no se crea una partición de validación. |
| training_label | cadena | Etiqueta para la partición de entrenamiento. |
| testing_label | cadena | Etiqueta para la partición de comprobación. |
| validation_label | cadena | Etiqueta para la partición de validación. Se ignora si no se crea una partición de validación. |
| value_mode | Sistema SystemAndLabel Label | Especifica los valores utilizados para representar cada partición en los datos. Por ejemplo, el entero del sistema 1, la etiqueta Entrenamiento o una combinación de los dos, 1_Entrenamiento, pueden representar la muestra de entrenamiento. |
| set_random_seed | Booleana | Especifica si se debe utilizar una semilla aleatoria especificada por el usuario. |
| random_seed | entero | Valor de semilla aleatoria especificada por el usuario. Para que se pueda utilizar este valor, set_random_seed se debe establecer en True. |
| enable_sql_generation | Booleana | Especifica si se utiliza la retrotracción SQL para asignar registros a particiones. |

Tabla 86. propiedades de partitionnode (continuación)

| Propiedad de partitionnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---------------|---|
| unique_field | | Especifica el campo de entrada que se utiliza para garantizar que los registros se asignan a particiones de una forma aleatoria pero reproducible. Para que se pueda utilizar este valor, enable_sql_generation se debe establecer en True. |

propiedades de reclassifynode



El nodo Reclasificar transforma un conjunto de valores categóricos en otro. La reclasificación es útil para contraer categorías o reagrupar datos para su análisis.

Ejemplo

```
node = stream.create("reclassify", "My node")
node.setPropertyValue("mode", "Multiple")
node.setPropertyValue("replace_field", True)
node.setPropertyValue("field", "Drug")
node.setPropertyValue("new_name", "Chemical")
node.setPropertyValue("fields", ["Drug", "BP"])
node.setPropertyValue("name_extension", "reclassified")
node.setPropertyValue("add_as", "Prefix")
node.setKeyedPropertyValue("reclassify", "drugA", True)
node.setPropertyValue("use_default", True)
node.setPropertyValue("default", "BrandX")
node.setPropertyValue("pick_list", ["BrandX", "Placebo", "Generic"])
```

Tabla 87. propiedades de reclassifynode

| Propiedad de reclassifynode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|-------------------------------|--|
| mode | Single Múltiples | Single reclasifica las categorías de un campo. Multiple activa las opciones que permiten la transformación de varios campos al mismo tiempo. |
| replace_field | flag | |
| field | cadena | Sólo se utiliza en modo Single. |
| new_name | cadena | Sólo se utiliza en modo Single. |
| campos | [campo1 campo2 ... campon] | Sólo se utiliza en modo Multiple. |
| name_extension | cadena | Sólo se utiliza en modo Multiple. |
| add_as | Suffix Prefix | Sólo se utiliza en modo Multiple. |
| reclasificar | cadena | Propiedad estructurada para valores de campos. |
| use_default | flag | Utiliza el valor predeterminado. |
| valor predeterminado | cadena | Especifica un valor predeterminado. |

Tabla 87. propiedades de reclassifynode (continuación)

| Propiedad de reclassifynode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|----------------------------|---|
| pick_list | [string string ... string] | Permite al usuario importar una lista de valores nuevos conocidos para rellenar la lista desplegable de la tabla. |

propiedades de reordernode



El nodo Reorg. campos define el orden natural utilizado para mostrar los campos en la parte posterior de la ruta. Este orden afecta a la visualización de los campos en diversas ubicaciones, como las tablas, las listas y el selector de campos. Esta operación resulta útil al trabajar con conjuntos de datos amplios que hacen más visibles los campos de interés.

Ejemplo

```
node = stream.create("reorder", "My node")
node.setPropertyValue("mode", "Custom")
node.setPropertyValue("sort_by", "Storage")
node.setPropertyValue("ascending", False)
node.setPropertyValue("start_fields", ["Age", "Cholesterol"])
node.setPropertyValue("end_fields", ["Drug"])
```

Tabla 88. propiedades de reordernode

| Propiedad de reordernode | Tipo de datos | Descripción de la propiedad |
|--------------------------|-------------------------------|---|
| mode | Custom Automático | Se pueden ordenar los valores de forma automática o especificar un orden personalizado. |
| sort_by | Name Tipo Storage | |
| ascending | flag | |
| start_fields | [campo1 campo2 ... campon] | Los campos nuevos se han insertado después de estos campos. |
| end_fields | [campo1 campo2 ... campon] | Los campos nuevos se han insertado antes de estos campos. |

propiedades de reprojectnode



En SPSS Modeler, elementos como las funciones espaciales Creador de expresiones, el nodo de Predicción espacio-temporal (STP) y el nodo Visualización de mapas utilizan el sistema de coordenadas proyectado. Utilice el nodo Reproyectar para cambiar el sistema de coordenadas de los datos que importa y que utilizan un sistema de coordenadas geográficas.

Tabla 89. propiedades de reprojectnode

| Propiedades de reprojectnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|-------------------------------|--|
| reproject_fields | [campo1 campo2 ... campon] | Lista todos los campos que se van a reprojectar. |
| reproject_type | Streamdefault Especifique | Elija cómo reprojectar los campos. |

Tabla 89. propiedades de reprojectnode (continuación)

| Propiedades de reprojectnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------|--|
| coordinate_system | cadena | El nombre del sistema de coordenadas que se aplicará a los campos. Ejemplo: set reprojectnode.coordinate_system = "WGS_1984_World_Mercator" |

propiedades de restructurenode



El nodo Reestructurar convierte un campo nominal o marca en un grupo de campos que se puede rellenar con los valores todavía de otro campo. Por ejemplo, para un campo determinado llamado *tipo de pago*, con valores de *crédito*, *efectivo*, y *débito*, se crearían tres campos nuevos (*crédito*, *efectivo*, *débito*), que contendría cada uno el valor del pago real realizado.

Ejemplo

```
node = stream.create("restructure", "My node")
node.setKeyedPropertyValue("fields_from", "Drug", ["drugA", "drugX"])
node.setPropertyValue("include_field_name", True)
node.setPropertyValue("value_mode", "OtherFields")
node.setPropertyValue("value_fields", ["Age", "BP"])
```

Tabla 90. propiedades de restructurenode

| Propiedad de restructurenode | Tipo de datos | Descripción de la propiedad |
|------------------------------|-------------------------------------|---|
| fields_from | [category category category] all | |
| include_field_name | flag | Indica si se debe usar el nombre del campo en el nombre de campo reestructurado. |
| value_mode | OtherFields Flags | Indica el modo de definir los valores para los campos reestructurados. Con OtherFields, debe especificar los campos que se van a usar (consulte a continuación). Con Flags, los valores son marcas numéricas. |
| value_fields | lista | Es necesario en caso de que value_mode sea OtherFields. Especifica los campos que se van a usar como campos de valores. |

propiedades de rfmanalysisnode



El nodo Análisis de RFM (actualidad, frecuencia, monetario) permite determinar cuantitativamente qué clientes son los mejores examinando cuándo ha sido la compra más reciente de un cliente (actualidad), cuántas veces suele comprar (frecuencia) y cuánto gasta el cliente en todas las transacciones (valor monetario).

Ejemplo

```
node = stream.create("rfmanalysis", "My node")
node.setPropertyValue("recency", "Recency")
node.setPropertyValue("frequency", "Frequency")
node.setPropertyValue("monetary", "Monetary")
```

```
node.setPropertyValue("tied_values_method", "Next")
node.setPropertyValue("recalculate_bins", "IfNecessary")
node.setPropertyValue("recency_thresholds", [1, 500, 800, 1500, 2000, 2500])
```

Tabla 91. propiedades de rfanalysisnode

| propiedades de rfanalysisnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|----------------------------------|---|
| actualidad | campo | Especifica el campo de actualidad. Puede ser una fecha, marca de tiempo o un número simple. |
| frecuencia | campo | Especifica el campo de frecuencia. |
| monetary | campo | Especifica el campo de monetario. |
| recency_bins | entero | Especifica el número de intervalos de actividades recientes que se van a generar. |
| recency_weight | número | Especifica la ponderación que se aplicará a los datos de actividades recientes. El valor predeterminado es 100. |
| frequency_bins | entero | Especifica el número de intervalos de frecuencia que se van a generar. |
| frequency_weight | número | Especifica la ponderación que se aplicará a los datos de frecuencia. El valor por omisión es 10. |
| monetary_bins | entero | Especifica el número de intervalos de monetario que se van a generar. |
| monetary_weight | número | Especifica la ponderación que se aplicará a los datos de monetario. El valor predeterminado es 1. |
| tied_values_method | Next Actual | Especifica en qué intervalo se van a insertar los datos de valor empatado. |
| recalculate_bins | Always IfNecessary | |
| add_outliers | flag | Sólo está disponible si recalculate_bins se define como IfNecessary. Si se selecciona, los registros por debajo del intervalo más inferior se añaden al intervalo inferior y los registros por encima, se añaden al intervalo superior. |
| binned_field | Recency Frequency Monetary | |
| recency_thresholds | valor valor | Sólo está disponible si recalculate_bins se define como Siempre. Especifica los umbrales superior e inferior de los intervalos de actividades recientes. El umbral superior de un intervalo se utiliza como el umbral inferior del siguiente, por ejemplo, [10 30 60] definiría dos intervalos, el primer intervalo con los umbrales superior e inferior de 10 y 30, con los umbrales del segundo intervalo de 30 y 60. |
| frequency_thresholds | valor valor | Sólo está disponible si recalculate_bins se define como Siempre. |
| monetary_thresholds | valor valor | Sólo está disponible si recalculate_bins se define como Siempre. |

propiedades de settoflagnode



El nodo Marcas deriva varios campos de marcas en función de los valores categóricos definidos para uno o más campos nominales.

Ejemplo

```
node = stream.create("settoflag", "My node")
node.setKeyedPropertyValue("fields_from", "Drug", ["drugA", "drugX"])
node.setPropertyValue("true_value", "1")
node.setPropertyValue("false_value", "0")
node.setPropertyValue("use_extension", True)
node.setPropertyValue("extension", "Drug_Flag")
node.setPropertyValue("add_as", "Suffix")
node.setPropertyValue("aggregate", True)
node.setPropertyValue("keys", ["Cholesterol"])
```

Tabla 92. propiedades de settoflagnode

| Propiedad de settoflagnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|--|--|
| fields_from | [<i>category category</i> <i>category</i>] all | |
| true_value | <i>cadena</i> | Especifica el valor para verdadero utilizado por el nodo al configurar una marca. El valor predeterminado es T (del inglés 'True'). |
| false_value | <i>cadena</i> | Especifica el valor para falso utilizado por el nodo al configurar una marca. El valor predeterminado es F (del inglés 'False'). |
| use_extension | <i>flag</i> | Utiliza una extensión como sufijo o prefijo para el nuevo campo de marca. |
| extensión | <i>cadena</i> | |
| add_as | Suffix Prefix | Especifica si la extensión es un sufijo o un prefijo. |
| aggregate | <i>flag</i> | Agrupar registros en función de campos clave. Si algún registro se establece como verdadero, se activarán todos los campos de marca de un grupo. |
| keys | <i>lista</i> | Campos clave. |

propiedades de statistictransformnode



El nodo Transformación Statistics ejecuta una selección de comandos de sintaxis de IBM SPSS Statistics en los orígenes de datos de IBM SPSS Modeler. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Las propiedades de este nodo están descritas en "propiedades de statistictransformnode" en la página 341.

Propiedades timeintervalsnode (en desuso)



Nota: Este nodo estaba en desuso en la versión 18 de SPSS Modeler y se sustituye por el nuevo nodo Serie temporal. El nodo Intervalos de tiempo especifica intervalos y genera etiquetas (si es necesario) para modelar los datos de series temporales. Si los valores no están espaciados de manera uniforme, el nodo puede rellenar o agregar valores según sea necesario para crear un intervalo uniforme entre registros.

Ejemplo

```
node = stream.create("timeintervals", "My node")
node.setPropertyValue("interval_type", "SecondsPerDay")
node.setPropertyValue("days_per_week", 4)
node.setPropertyValue("week_begins_on", "Tuesday")
node.setPropertyValue("hours_per_day", 10)
node.setPropertyValue("day_begins_hour", 7)
node.setPropertyValue("day_begins_minute", 5)
node.setPropertyValue("day_begins_second", 17)
node.setPropertyValue("mode", "Label")
node.setPropertyValue("year_start", 2005)
node.setPropertyValue("month_start", "January")
node.setPropertyValue("day_start", 4)
node.setKeyedPropertyValue("pad", "AGE", "MeanOfRecentPoints")
node.setPropertyValue("agg_mode", "Specify")
node.setPropertyValue("agg_set_default", "Last")
```

Tabla 93. propiedades de timeintervalsnode.

| Propiedad de timeintervalsnode | Tipo de datos | Descripción de la propiedad |
|--------------------------------|--|---|
| interval_type | Ninguno Períodos CyclicPeriods Años Trimestres Meses DaysPerWeek DaysNonPeriodic HoursPerDay HoursNonPeriodic MinutesPerDay MinutesNonPeriodic SecondsPerDay SecondsNonPeriodic | |
| mode | Label Create | Determina si desea etiquetar los registros de manera consecutiva o crear la serie según una fecha, marca de tiempo o campo de tiempo concretos. |
| campo | campo | Al crear la serie a partir de los datos, especifica el campo que informa de la fecha u hora de cada registro. |
| period_start | entero | Especifica el intervalo de inicio para períodos o períodos cíclicos. |
| cycle_start | entero | Ciclo de inicio de los períodos cíclicos. |
| year_start | entero | En el caso de los tipos de intervalo que procedan, el año en que el primer intervalo tiene lugar. |

Tabla 93. propiedades de `timeintervalnode` (continuación).

| Propiedad de <code>timeintervalnode</code> | Tipo de datos | Descripción de la propiedad |
|--|---|--|
| <code>quarter_start</code> | <i>entero</i> | En el caso de los tipos de intervalo que procedan, el trimestre en que el primer intervalo tiene lugar. |
| <code>month_start</code> | Enero Febrero Marzo Abril Mayo Junio Julio Agosto Septiembre Octubre Noviembre Diciembre | |
| <code>day_start</code> | <i>entero</i> | |
| <code>hour_start</code> | <i>entero</i> | |
| <code>minute_start</code> | <i>entero</i> | |
| <code>second_start</code> | <i>entero</i> | |
| <code>periods_per_cycle</code> | <i>entero</i> | En el caso de los períodos cíclicos, número en cada ciclo. |
| <code>fiscal_year_begins</code> | Enero Febrero Marzo Abril Mayo Junio Julio Agosto Septiembre Octubre Noviembre Diciembre | En el caso de los intervalos trimestrales, especifica el mes en el que comienza el año fiscal. |
| <code>week_begins_on</code> | Sunday Monday Tuesday Wednesday Thursday Friday Saturday Sunday | En el caso de los intervalos periódicos (días a la semana, horas al día, minutos al día y segundos al día), especifica el día en el que comienza la semana. |
| <code>day_begins_hour</code> | <i>entero</i> | En el caso de los intervalos periódicos (horas al día, minutos al día y segundos al día), especifica la hora a la que comienza el día. Se puede usar junto con <code>day_begins_minute</code> y <code>day_begins_second</code> para determinar una hora exacta, como <code>8:05:01</code> . Vea el ejemplo de uso que se muestra a continuación. |
| <code>day_begins_minute</code> | <i>entero</i> | En el caso de los intervalos periódicos (horas al día, minutos al día y segundos al día), especifica el minuto en el que comienza el día (por ejemplo, 5 en <code>8:05</code>). |

Tabla 93. propiedades de *timeintervalsnode* (continuación).

| Propiedad de <i>timeintervalsnode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---|---|
| <i>day_begins_second</i> | <i>entero</i> | En el caso de los intervalos periódicos (horas al día, minutos al día y segundos al día), especifica el segundo en el que comienza el día (por ejemplo, 17 en 8:05:17). |
| <i>days_per_week</i> | <i>entero</i> | En el caso de los intervalos periódicos (días a la semana, horas al día, minutos al día y segundos al día), especifica el número de días a la semana. |
| <i>hours_per_day</i> | <i>entero</i> | En el caso de los intervalos periódicos (horas al día, minutos al día y segundos al día), especifica el número de horas al día. |
| <i>interval_increment</i> | 1 2 3 4 5 6 10 15 20 30 | En el caso de los minutos al día y de los segundos al día, especifica el número de minutos o segundos que se va a aumentar en cada registro. |
| <i>field_name_extension</i> | <i>cadena</i> | |
| <i>field_name_extension_as_prefix</i> | <i>tag</i> | |
| <i>date_format</i> | "DDMMAA" "MMDDYY" "AAMMDD" "YYMMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-AAAA" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.AAAA" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/AAAA" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | |

Tabla 93. propiedades de `timeintervalsnode` (continuación).

| Propiedad de <code>timeintervalsnode</code> | Tipo de datos | Descripción de la propiedad |
|---|--|---|
| <code>time_format</code> | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | |
| <code>aggregate</code> | Media Sum Mode Mín Máx Primero Last TrueIfAnyTrue | Especifica el método de agregación para un campo. |
| <code>pad</code> | Vacío MeanOfRecentPoints True False | Especifica el método de relleno para un campo. |
| <code>agg_mode</code> | Todos Especifica | Determina si se agregan o rellenan todos los campos con las funciones predeterminadas según sea necesario o bien si se especifican los campos y las funciones que deben usarse. |
| <code>agg_range_default</code> | Media Sum Mode Mín Máx | Especifica la función predeterminada que se va a usar al agregar campos continuos. |
| <code>agg_set_default</code> | Mode Primero Last | Especifica la función predeterminada que se va a usar al agregar campos nominales. |
| <code>agg_flag_default</code> | TrueIfAnyTrue Mode Primero Last | |
| <code>pad_range_default</code> | Vacío MeanOfRecentPoints | Especifica la función predeterminada que se va a usar al agregar campos continuos. |
| <code>pad_set_default</code> | Vacío MostRecentValue | |
| <code>pad_flag_default</code> | Vacío True False | |
| <code>max_records_to_create</code> | <i>entero</i> | Especifica el número máximo de registros que se van a crear al rellenar la serie. |
| <code>estimation_from_beginning</code> | <i>tag</i> | |

Tabla 93. propiedades de *timeintervalsnode* (continuación).

| Propiedad de <i>timeintervalsnode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---------------|-----------------------------|
| <code>estimation_to_end</code> | <i>tag</i> | |
| <code>estimation_start_offset</code> | <i>entero</i> | |
| <code>estimation_num_holdouts</code> | <i>entero</i> | |
| <code>create_future_records</code> | <i>tag</i> | |
| <code>num_future_records</code> | <i>entero</i> | |
| <code>create_future_field</code> | <i>tag</i> | |
| <code>future_field_name</code> | <i>cadena</i> | |

Propiedades de *transposenode*



El nodo Transponer intercambia los datos en filas y columnas de manera que los registros se conviertan en campos y los campos en registros.

Ejemplo

```
node = stream.create("transpose", "My node")
node.setPropertyValue("transposed_names", "Read")
node.setPropertyValue("read_from_field", "TimeLabel")
node.setPropertyValue("max_num_fields", "1000")
node.setPropertyValue("id_field_name", "ID")
```

Tabla 94. Propiedades de *transposenode*

| Propiedad de <i>transposenode</i> | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|----------------|--|
| <code>transpose_method</code> | <i>enum</i> | Especifica el método de transposición: Normal (normal), CASE a VAR (casetovar) o VAR a CASE (vartocase). |
| <code>transposed_names</code> | Prefix Leer | Propiedad para el método de transposición normal. Se pueden generar nuevos nombres de campo automáticamente a partir de un prefijo concreto o bien se pueden leer desde un campo existente en los datos. |
| <code>prefix</code> | <i>string</i> | Propiedad para el método de transposición normal. |
| <code>num_new_fields</code> | <i>integer</i> | Propiedad para el método de transposición normal. Al usar un prefijo, especifica el número máximo de campos nuevos que se van a crear. |
| <code>read_from_field</code> | <i>field</i> | Propiedad para el método de transposición normal. Campo del que se leen los nombres. Debe tratarse de un campo instanciado o, de lo contrario, se producirá un error al ejecutar el nodo. |
| <code>max_num_fields</code> | <i>integer</i> | Propiedad para el método de transposición normal. Al leer nombres de un campo, especifica un límite superior con el fin de evitar que se genere un número desmesurado de campos. |

Tabla 94. Propiedades de transposenode (continuación)

| Propiedad de transposenode | Tipo de datos | Descripción de la propiedad |
|----------------------------|-------------------------------------|---|
| transpose_type | Numérico Cadena Personalizado | Propiedad para el método de transposición normal. De forma predeterminada, solamente los campos continuos (rango numérico) se transponen, si bien se puede elegir un subconjunto personalizado de campos numéricos o, en su lugar, transponer todos los campos de cadena. |
| transpose_fields | <i>list</i> | Propiedad para el método de transposición normal. Especifica los campos que se van a transponer cuando se usa la opción Custom. |
| id_field_name | <i>field</i> | Propiedad para el método de transposición normal. |
| index | <i>field</i> | Propiedad para el método de transposición CASE a VAR (casetovar). Acepta varios campos a utilizar como campos de índice. field1 ... fieldN |
| column | <i>field</i> | Propiedad para el método de transposición CASE a VAR (casetovar). Acepta varios campos que se utilizarán como campos de columna. field1 ... fieldN |
| value | <i>field</i> | Propiedad para el método de transposición CASE a VAR (casetovar). Acepta varios campos que se utilizarán como campos de valor. field1 ... fieldN |
| id_variables | <i>field</i> | Propiedad para el método de transposición VAR a CASE (vartocase). Acepta varios campos que se deberán utilizar como campos de variable de ID. field1 ... fieldN |
| value_variables | <i>field</i> | Propiedad para el método de transposición VAR a CASE (vartocase). Acepta varios campos que se deberán utilizar como campos de variable de valor. field1 ... fieldN |

propiedades de typenode



El nodo Tipo especifica propiedades y metadatos de campo. Por ejemplo, puede especificar un nivel de medición (continuo, nominal, ordinal o marca) para cada campo, establecer las opciones para gestionar valores perdidos y nulos del sistema, establecer el rol de un campo con fines de modelado, especificar las etiquetas de valor y campo y especificar los valores de un campo.

Ejemplo

```
node = stream.createAt("type", "My node", 50, 50)
node.setKeyedPropertyValue("check", "Cholesterol", "Coerce")
node.setKeyedPropertyValue("direction", "Drug", "Input")
node.setKeyedPropertyValue("type", "K", "Range")
node.setKeyedPropertyValue("values", "Drug", ["drugA", "drugB", "drugC", "drugD", "drugX",
"drugY", "drugZ"])
node.setKeyedPropertyValue("null_missing", "BP", False)
```

```

node.setKeyedPropertyValue("whitespace_missing", "BP", False)
node.setKeyedPropertyValue("description", "BP", "Blood Pressure")
node.setKeyedPropertyValue("value_labels", "BP", [{"HIGH", "High Blood Pressure"},
["NORMAL", "normal blood pressure"]])

```

Observe que en algunos casos puede que sea necesario instanciar totalmente el nodo Tipo para que otros nodos funcionen adecuadamente, como, por ejemplo, la propiedad fields from del nodo Marcas.

Simplemente conecte un nodo Tabla y ejecútelos para instanciar los campos:

```

tablenode = stream.createAt("table", "Table node", 150, 50)
stream.link(node, tablenode)
tablenode.run(None)
stream.delete(tablenode)

```

Tabla 95. Propiedades de typenode.

| Propiedad de typenode | Tipo de datos | Descripción de la propiedad |
|-----------------------|--|--|
| dirección | Input Destino Both Ninguno Partition Split Frequency RecordID | Propiedad con clave para los roles de los campos. Nota: Los valores In y Out han quedado en desuso. Es posible que ya no puedan utilizarse en futuras versiones. |
| type | Range Flag Set Sin tipo Discrete OrderedSet Predeterminado | El nivel de medición del campo (anteriormente denominado el "tipo" de campo). Si se establece type en Default, se borrará cualquier valor del parámetro values, y si value_mode tiene el valor Specify, se restablecerá en Read. Si value_mode se establece en Pass o Read, el establecimiento de type no afectará a value_mode. Nota: Los tipos de datos utilizados internamente difieren de los que son visibles en el nodo tipo. La correspondencia es la siguiente: Range -> Continuous Set -> Nominal OrderedSet -> Ordinal Discrete -> Categorical |
| almacenamiento | Desconocido Cadena Entero Real Hora Fecha Marca de tiempo | Propiedad con clave de solamente lectura para el tipo de almacenamiento de campos. |
| check | Ninguno Nullify Coerce Descartar Warn Abort | Propiedad con clave para la comprobación del rango y el tipo de campo. |

Tabla 95. Propiedades de *typenode* (continuación).

| Propiedad de <i>typenode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------|---|--|
| <code>values</code> | <code>[value value]</code> | Para un campo continuos, el primer valor es el mínimo y el último es el máximo. Para campos nominales, especifique todos los valores. Para los campos marca, el primer valor representa <i>falso</i> y el último, <i>verdadero</i> . La configuración de esta propiedad establece de forma automática la propiedad <code>value_mode</code> en <code>Specify</code> . |
| <code>value_mode</code> | Leer Pasar Leer+ Actual Especifique | Determina la forma en la que se establecen los valores. Tenga en cuenta que no puede establecer esta propiedad directamente en <code>Specify</code> . Para utilizar valores específicos, establezca la propiedad <code>values</code> . |
| <code>extend_values</code> | <i>flag</i> | Se aplica cuando <code>value_mode</code> se establece en <code>Read</code> . Establézcala en <code>T</code> para añadir nuevos valores de lectura a los valores existentes del campo. Establézcala en <code>F</code> para descartar los valores existentes y favorecer a los nuevos valores de lectura. |
| <code>enable_missing</code> | <i>flag</i> | Cuando está definida como <code>T</code> , activa el seguimiento de los valores perdidos para el campo. |
| <code>missing_values</code> | <code>[value value ...]</code> | Especifica los valores de datos que denotan los datos perdidos. |
| <code>range_missing</code> | <i>flag</i> | Especifica si se ha definido un rango de valores perdidos (vacíos) para un campo. |
| <code>missing_lower</code> | <i>cadena</i> | Si <code>range_missing</code> es verdadero, especifica el límite inferior del rango de valores perdidos. |
| <code>missing_upper</code> | <i>cadena</i> | Si <code>range_missing</code> es verdadero, especifica el límite superior del rango de valores perdidos. |
| <code>null_missing</code> | <i>flag</i> | Cuando se establece en <code>T</code> , los valores <i>nulos</i> (valores no definidos que se muestran como <code>\$null\$</code> en el software) se consideran valores perdidos. |
| <code>whitespace_missing</code> | <i>flag</i> | Cuando se establece en <code>T</code> , los valores que sólo contienen un espacio en blanco (espacios, tabulaciones y líneas nuevas) se consideran valores perdidos. |
| <code>description</code> | <i>cadena</i> | Especifica la descripción de un campo. |
| <code>value_labels</code> | <code>[[Value LabelString] [Value LabelString] ...]</code> | Se utiliza para especificar etiquetas para los pares de valores. |
| <code>display_places</code> | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se muestra (sólo se aplica a campos con almacenamiento REAL). Un valor de 1 utilizará el valor predeterminado de la ruta. |
| <code>export_places</code> | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se exporta (sólo se aplica a campos con almacenamiento REAL). Un valor de 1 utilizará el valor predeterminado de la ruta. |

Tabla 95. Propiedades de *typenode* (continuación).

| Propiedad de <i>typenode</i> | Tipo de datos | Descripción de la propiedad |
|--------------------------------|---|--|
| <code>decimal_separator</code> | DEFAULT PERIOD COMMA | Establece el separador decimal para el campo (sólo se aplica a campos con almacenamiento REAL). |
| <code>date_format</code> | "DDMMAA" "MMDDYY" "AAMMDD" "YYMMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-AAAA" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.AAAA" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/AAAA" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | Establece el formato de fecha para el campo (sólo se aplica a campos con almacenamiento FECHA o MARCADETIEMPO). |
| <code>time_format</code> | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | Establece el formato de hora para el campo (sólo se aplica a campos con almacenamiento TIME o TIMESTAMP). |
| <code>number_format</code> | DEFAULT STANDARD SCIENTIFIC CURRENCY | Establece el formato de presentación de los números para el campo. |
| <code>standard_places</code> | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se muestra en formato estándar. Un valor de 1 utilizará el valor predeterminado de la ruta. Tenga en cuenta que el intervalo <code>display_places</code> existente también cambiará esto, pero ahora se ha desaprobadado. |

Tabla 95. Propiedades de *typenode* (continuación).

| Propiedad de <i>typenode</i> | Tipo de datos | Descripción de la propiedad |
|------------------------------------|---|---|
| <code>scientific_places</code> | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se muestra en formato científico. Un valor de 1 utilizará el valor predeterminado de la ruta. |
| <code>currency_places</code> | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se muestra en formato moneda. Un valor de 1 utilizará el valor predeterminado de la ruta. |
| <code>grouping_symbol</code> | DEFAULT NONE LOCALE PERIOD COMMA SPACE | Establece el símbolo de agrupación para el campo. |
| <code>column_width</code> | <i>entero</i> | Establece el ancho de columna para el campo. Un valor de 1 establecerá el ancho de columna en Auto. |
| <code>justify</code> | AUTO CENTER LEFT RIGHT | Establece la justificación de columna para el campo. |
| <code>measure_type</code> | Range / MeasureType.RANGE Discrete / MeasureType.DISCRETE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS Collection / MeasureType.COLLECTION Geospatial / MeasureType.GEOSPATIAL | Esta propiedad con clave es similar a <code>type</code> en cuanto a que puede utilizarse para definir la medición asociada al campo. La diferencia es que, en los scripts Python, la función de establecimiento puede pasar también uno de los valores MeasureType, mientras que la función de obtención siempre devolverá los valores MeasureType. |
| <code>collection_measure</code> | Range / MeasureType.RANGE Flag / MeasureType.FLAG Set / MeasureType.SET OrderedSet / MeasureType.ORDERED_SET Typeless / MeasureType.TYPELESS | Para campos de recopilación (listas con profundidad 0), esta propiedad con clave define el tipo de medición asociado con los valores subyacentes. |
| <code>geo_type</code> | Point Multipunto Cadena lineal Cadena multilínea Polígono Multipolígono | En campos geoespaciales, esta propiedad con clave define el tipo del objeto geoespacial representado por este campo. Debería ser coherente con la profundidad de lista de los valores. |
| <code>has_coordinate_system</code> | <i>booleano</i> | En campos geoespaciales, esta propiedad define si este campo tiene un sistema de coordenadas |
| <code>coordinate_system</code> | <i>cadena</i> | En campos geoespaciales, esta propiedad con clave define el sistema de coordenadas para este campo. |

Tabla 95. Propiedades de *typenode* (continuación).

| Propiedad de <i>typenode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------|--|--|
| <i>custom_storage_type</i> | Unknown / MeasureType.UNKNOWN String / MeasureType.STRING Integer / MeasureType.INTEGER Real / MeasureType.REAL Time / MeasureType.TIME Date / MeasureType.DATE Timestamp / MeasureType.TIMESTAMP List / MeasureType.LIST | Esta propiedad con clave es similar a <i>custom_storage</i> en cuanto a que puede utilizarse para definir el almacenamiento de alteración temporal para el campo. La diferencia es que, en los scripts Python, la función de establecimiento puede pasar también uno de los valores <i>StorageType</i> , mientras que la función de obtención siempre devolverá los valores <i>StorageType</i> . |
| <i>custom_list_storage_type</i> | String / MeasureType.STRING Integer / MeasureType.INTEGER Real / MeasureType.REAL Time / MeasureType.TIME Date / MeasureType.DATE Timestamp / MeasureType.TIMESTAMP | Para campos de lista, esta propiedad con clave especifica el tipo de almacenamiento de los valores subyacentes. |
| <i>custom_list_depth</i> | <i>entero</i> | Para campos de lista, esta propiedad con clave especifica la profundidad del campo. |
| <i>max_list_length</i> | <i>entero</i> | Solo está disponible para datos con un nivel de medición de <i>Geoespacial</i> o <i>Colección</i> . Establezca la longitud máxima de la lista especificando el número de elementos que puede contener la lista. |
| <i>max_string_length</i> | <i>entero</i> | Solo está disponible para datos <i>sin tipos</i> y se utiliza al generar SQL para crear una tabla. Entre el valor de la serie más larga en los datos; esto genera una columna en la tabla que es lo suficientemente grande para contener la serie. |

Capítulo 12. Propiedades de nodos Gráfico

Propiedades comunes del nodo Gráfico

Esta sección describe las propiedades disponibles para los nodos de gráficos, incluidas las comunes y aquellas específicas de cada tipo de nodo.

Tabla 96. Propiedades comunes de nodos de gráficos

| Propiedades comunes de nodos de gráficos | Tipo de datos | Descripción de la propiedad |
|--|---|---|
| título | <i>cadena</i> | Especifica el título. Ejemplo: "Esto es un título". |
| caption | <i>cadena</i> | Especifica el pie. Por ejemplo: "Esto es un pie". |
| output_mode | Screen File | Determina si el resultado del nodo de gráficos se muestra o si se guarda en un archivo. |
| output_format | BMP JPEG PNG HTML output (.cou) | Especifica el tipo de resultado. El tipo exacto de resultado permitido para cada nodo varía. |
| full_filename | <i>cadena</i> | Especifica la ruta de destino y el nombre de archivo del resultado generado por el nodo de gráficos. |
| use_graph_size | <i>flag</i> | Controla si el tamaño del gráfico se ha establecido de manera explícita mediante las propiedades de ancho y altura a continuación. Afecta solamente a los gráficos que tienen salida a pantalla. No disponible para el nodo Distribución. |
| graph_width | <i>número</i> | Cuando use_graph_size es True, establece el ancho del gráfico en píxeles. |
| graph_height | <i>número</i> | Cuando use_graph_size es True, establece la altura del gráfico en píxeles. |

Desactivación de los campos opcionales

Los campos opcionales, como un campo de superposición para gráficos, se pueden desactivar estableciendo el valor de la propiedad en "" (cadena vacía), tal y como se muestra en el siguiente ejemplo:

```
plotnode.setPropertyValue("color_field", "")
```

Especificación de colores

Los colores de los títulos, pies, fondos y etiquetas se pueden especificar mediante las cadenas hexadecimales que comiencen con el símbolo almohadilla (#). Por ejemplo, para establecer el fondo del gráfico en cielo azul, debe utilizar la siguiente instrucción:

```
mygraphnode.setPropertyValue("graph_background", "#87CEEB")
```

Aquí, los dos primeros dígitos, 87, especifican el contenido rojo, los dos del medio, CE, especifican el contenido verde y los dos últimos, EB, el contenido azul. Cada dígito puede tomar un valor del rango 0-9 o A-F. Juntos, estos valores pueden especificar red-green-blue, o RGB o color.

Nota: Al especificar colores en RVA, puede utilizar el selector de campos en la interfaz de usuario para determinar el código de color correcto. Basta con colocarse sobre el color para ver un texto con la información deseada.

Propiedades de collectionnode



El nodo Colección muestra la distribución de valores de un campo numérico relativo a los valores de otro. (Crea gráficos parecidos a los histogramas.) Es útil para ilustrar una variable o un campo cuyos valores cambian con el tiempo. Con los gráficos 3D también puede incluir un eje simbólico que muestra las distribuciones por categoría.

Ejemplo

```
node = stream.create("collection", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("three_D", True)
node.setPropertyValue("collect_field", "Drug")
node.setPropertyValue("over_field", "Age")
node.setPropertyValue("by_field", "BP")
node.setPropertyValue("operation", "Sum")
# Sección "Superponer"
node.setPropertyValue("color_field", "Drug")
node.setPropertyValue("panel_field", "Sex")
node.setPropertyValue("animation_field", "")
# pestaña "Opciones"
node.setPropertyValue("range_mode", "Automatic")
node.setPropertyValue("range_min", 1)
node.setPropertyValue("range_max", 100)
node.setPropertyValue("bins", "ByNumber")
node.setPropertyValue("num_bins", 10)
node.setPropertyValue("bin_width", 5)
```

Tabla 97. propiedades de collectionnode

| Propiedad de collectionnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|------------------------------------|-----------------------------|
| over_field | campo | |
| over_label_auto | flag | |
| over_label | cadena | |
| collect_field | campo | |
| collect_label_auto | flag | |
| collect_label | cadena | |
| three_D | flag | |
| by_field | campo | |
| by_label_auto | flag | |
| by_label | cadena | |
| operation | Sum Media Mín Máx SDev | |
| color_field | cadena | |
| panel_field | cadena | |

Tabla 97. propiedades de collectionnode (continuación)

| Propiedad de collectionnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|--------------------------|---|
| animation_field | cadena | |
| range_mode | Automatic UserDefined | |
| range_min | número | |
| range_max | número | |
| bins | ByNumber ByWidth | |
| num_bins | número | |
| bin_width | número | |
| use_grid | flag | |
| graph_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| page_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |

Propiedades de distributionnode



El nodo Distribución muestra las instancias de valores simbólicos (categóricos), como el tipo de hipoteca o el género. Normalmente, podría usar el nodo Distribución para mostrar los desequilibrios de los datos, que pueden rectificarse mediante el nodo Equilibrar antes de crear un modelo.

Ejemplo

```
node = stream.create("distribution", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("plot", "Flags")
node.setPropertyValue("x_field", "Age")
node.setPropertyValue("color_field", "Drug")
node.setPropertyValue("normalize", True)
node.setPropertyValue("sort_mode", "ByOccurrence")
node.setPropertyValue("use_proportional_scale", True)
```

Tabla 98. propiedades de distributionnode

| Propiedades de distributionnode | Tipo de datos | Descripción de la propiedad |
|---------------------------------|----------------------------|-----------------------------|
| plot | SelectedFields Flags | |
| x_field | campo | |
| color_field | campo | Campo de superposición. |
| normalize | flag | |
| sort_mode | ByOccurrence Alphabetic | |
| use_proportional_scale | flag | |

Propiedades de evaluationnode



El nodo Evaluación ayuda a evaluar y comparar modelos predictivos. El diagrama de evaluación muestra la calidad con que los modelos predicen resultados particulares. Ordena registros en función del valor predicho y la confianza de la predicción. Divide el registro en grupos de igual tamaño (**cuantiles**) y, a continuación, representa el valor del criterio de negocio de cada cuantil de mayor a menor. El gráfico muestra múltiples modelos como líneas independientes.

Ejemplo

```
node = stream.create("evaluation", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("chart_type", "Gains")
node.setPropertyValue("cumulative", False)
node.setPropertyValue("field_detection_method", "Name")
node.setPropertyValue("inc_baseline", True)
node.setPropertyValue("n_tile", "Deciles")
node.setPropertyValue("style", "Point")
node.setPropertyValue("point_type", "Dot")
node.setPropertyValue("use_fixed_cost", True)
node.setPropertyValue("cost_value", 5.0)
node.setPropertyValue("cost_field", "Na")
node.setPropertyValue("use_fixed_revenue", True)
node.setPropertyValue("revenue_value", 30.0)
node.setPropertyValue("revenue_field", "Age")
node.setPropertyValue("use_fixed_weight", True)
node.setPropertyValue("weight_value", 2.0)
node.setPropertyValue("weight_field", "K")
```

Tabla 99. propiedades de evaluationnode.

| Propiedad de evaluationnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|---|-----------------------------|
| chart_type | Gains Response Lift Profit ROI ROC | |
| inc_baseline | tag | |
| field_detection_method | Metadata Name | |
| use_fixed_cost | tag | |
| cost_value | number | |
| cost_field | cadena | |
| use_fixed_revenue | tag | |
| revenue_value | number | |
| revenue_field | cadena | |
| use_fixed_weight | tag | |
| weight_value | number | |
| weight_field | campo | |

Tabla 99. propiedades de evaluationnode (continuación).

| Propiedad de evaluationnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|---|-----------------------------|
| n_tile | Quartiles Quintiles Deciles Vingtiles Percentiles 1000-tiles | |
| cumulative | tag | |
| style | Line Point | |
| point_type | Rectángulo Dot Triangle Hexagon Plus Pentagon Star BowTie HorizontalDash VerticalDash IronCross Factory House Cathedral OnionDome ConcaveTriangle OblateGlobe CatEye FourSidedPillow RoundRectangle Fan | |
| export_data | flag | |
| data_filename | cadena | |
| delimiter | cadena | |
| new_line | flag | |
| inc_field_names | flag | |
| inc_best_line | tag | |
| inc_business_rule | tag | |
| business_rule_condition | cadena | |
| plot_score_fields | tag | |
| score_fields | [campo1 ... campoN] | |
| target_field | campo | |
| use_hit_condition | tag | |
| hit_condition | cadena | |
| use_score_expression | tag | |
| score_expression | cadena | |
| caption_auto | tag | |

Propiedades de graphboardnode



El nodo Tablero ofrece muchos tipos diferentes de gráficos en un único nodo. Con este nodo puede seleccionar los campos de datos que desee explorar y seleccionar un gráfico de los disponibles para los datos seleccionados. El nodo filtra automáticamente cualquier tipo de gráfico que no sea compatible con las selecciones de campo.

Nota: Si establece una propiedad que no es válida para el tipo de gráfico (por ejemplo, si especifica `y_field` para un histograma), se ignora dicha propiedad.

Nota: En la IU, en la pestaña Detallado de muchos tipos de gráfico distintos, hay un campo **Resumen**; los scripts no soportan dicho campo en la actualidad.

Ejemplo

```
node = stream.create("graphboard", "My node")
node.setPropertyValue("graph_type", "Line")
node.setPropertyValue("x_field", "K")
node.setPropertyValue("y_field", "Na")
```

Tabla 100. Propiedades de graphboardnode

| Propiedades de graphboard | Tipo de datos | Descripción de la propiedad |
|---------------------------|--|---|
| graph_type | 2DDotplot 3DArea 3DBar 3DDensity 3DHistogram 3DPie 3DScatterplot Área ArrowMap Bar BarCounts BarCountsMap BarMap BinnedScatter Diagramas de caja Bubble ChoroplethMeans ChoroplethMedians ChoroplethSums ChoroplethValues ChoroplethCounts CoordinateMap CoordinateChoroplethMeans CoordinateChoroplethMedians CoordinateChoroplethSums CoordinateChoroplethValues CoordinateChoroplethCounts Dotplot Heatmap HexBinScatter Histograma Line LineChartMap LineOverlayMap Parallel Path Pie PieCountMap PieCounts PieMap PointOverlayMap PolygonOverlayMap Ribbon Scatterplot SPLOM Surface | Identifica el tipo de gráfico. |
| x_field | <i>campo</i> | Especifica una etiqueta personalizada para el eje x. Disponible solamente para etiquetas. |
| y_field | <i>campo</i> | Especifica una etiqueta personalizada para el eje y. Disponible solamente para etiquetas. |
| z_field | <i>campo</i> | Se utiliza en algunos gráficos 3D. |
| color_field | <i>campo</i> | Se utiliza en mapas de calor. |

Tabla 100. Propiedades de graphboardnode (continuación)

| Propiedades de graphboard | Tipo de datos | Descripción de la propiedad |
|---------------------------|---------------|--|
| size_field | campo | Se utiliza en gráficos de burbujas. |
| categories_field | campo | |
| values_field | campo | |
| rows_field | campo | |
| columns_field | campo | |
| campos | campo | |
| start_longitude_field | campo | Se utiliza con flechas en un mapa de referencia. |
| end_longitude_field | campo | |
| start_latitude_field | campo | |
| end_latitude_field | campo | |
| data_key_field | campo | Se utiliza en diversos mapas. |
| panelrow_field | cadena | |
| panelcol_field | cadena | |
| animation_field | cadena | |
| longitude_field | campo | Se utiliza en mapas de coordenadas. |
| latitude_field | campo | |
| map_color_field | campo | |
| | | |

Propiedades de histogramnode



El nodo Histograma muestra las instancias de valores de los campos numéricos. Se suele utilizar para explorar los datos antes de las manipulaciones y la generación de modelos. Al igual que con el nodo Distribución, con frecuencia el nodo Histograma detecta desequilibrios en los datos.

Ejemplo

```
node = stream.create("histogram", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("field", "Drug")
node.setPropertyValue("color_field", "Drug")
node.setPropertyValue("panel_field", "Sex")
node.setPropertyValue("animation_field", "")
# pestaña "Opciones"
node.setPropertyValue("range_mode", "Automatic")
node.setPropertyValue("range_min", 1.0)
node.setPropertyValue("range_max", 100.0)
node.setPropertyValue("num_bins", 10)
node.setPropertyValue("bin_width", 10)
node.setPropertyValue("normalize", True)
node.setPropertyValue("separate_bands", False)
```

Tabla 101. propiedades de histogramnode

| Propiedad de histogramnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---------------|-----------------------------|
| field | campo | |

Tabla 101. propiedades de histogramnode (continuación)

| Propiedad de histogramnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|--------------------------|--|
| color_field | campo | |
| panel_field | campo | |
| animation_field | campo | |
| range_mode | Automatic UserDefined | |
| range_min | número | |
| range_max | número | |
| bins | ByNumber ByWidth | |
| num_bins | número | |
| bin_width | número | |
| normalize | flag | |
| separate_bands | flag | |
| x_label_auto | flag | |
| x_label | cadena | |
| y_label_auto | flag | |
| y_label | cadena | |
| use_grid | flag | |
| graph_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| page_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| normal_curve | flag | Indica si se debe mostrar la curva de distribución normal en los resultados. |

Propiedades de mapvisualization



El nodo Visualización de mapas puede aceptar varias conexiones de entrada y visualizar datos geospaciales en un mapa como una serie de capas. Cada capa es un único campo geoespacial; por ejemplo, la capa base podría ser el mapa de un país, y por encima puede tener una capa de carreteras, una capa de ríos y una capa de ciudades.

Tabla 102. Propiedades de mapvisualization

| Propiedades de mapvisualization | Tipo de datos | Descripción de la propiedad |
|---------------------------------|---------------|--|
| tag | cadena | Establece el nombre de la etiqueta para la entrada. La etiqueta predeterminada es un número basado en el orden en el que las entradas estaban conectadas al nodo (la primera etiqueta de conexión es 1, la segunda etiqueta de conexión es 2, etc. |

Tabla 102. Propiedades de mapvisualization (continuación)

| Propiedades de mapvisualization | Tipo de datos | Descripción de la propiedad |
|---------------------------------|---------------|--|
| layer_field | campo | <p>Selecciona qué geocampo se muestra del conjunto de datos como una capa en el mapa. La selección predeterminada se basa en el orden de clasificación siguiente:</p> <ul style="list-style-type: none"> • Primero - Punto • Serie lineal • Polígono • Multipunto • Serie multilínea • Último - Multipolígono <p>Si existen dos campos con el mismo tipo de medida, de forma predeterminada, se seleccionará el primer campo alfabéticamente (por nombre).</p> |
| color_type | booleano | <p>Especifica si se va a aplicar un color estándar a todas las características del geocampo, o un campo de superposición que colorea las características basándose en valores de otro campo del conjunto de datos. Los valores posibles son standard u overlay. El valor predeterminado es standard.</p> |
| color | cadena | <p>Si standard está seleccionado para color_type, el desplegable contiene la misma paleta de colores que el orden de colores de la categoría de gráfico en la pestaña Visualización de opciones de usuario.</p> <p>El valor predeterminado es el color de categoría de gráfico 1.</p> |
| color_field | campo | <p>Si overlay está seleccionado para color_type, el desplegable contiene todos los campos del mismo conjunto de datos que el geocampo seleccionado como la capa.</p> |
| symbol_type | booleano | <p>Especifica si se va a aplicar un símbolo estándar a todos los registro del geocampo, o un símbolo de superposición que cambie el icono de símbolo para los puntos basándose en los valores de otro campo del conjunto de datos. Los valores posibles son standard u overlay. El valor predeterminado es standard.</p> |
| symbol | cadena | <p>Si standard está seleccionado para symbol_type, el desplegable contiene una selección de símbolos que se puede utilizar para mostrar puntos en el mapa.</p> |
| symbol_field | campo | <p>Si overlay está seleccionado para symbol_type, el desplegable contiene todos los campos nominales, ordinales o categóricos del mismo conjunto de datos como el geocampo seleccionado como la capa.</p> |

Tabla 102. Propiedades de mapvisualization (continuación)

| Propiedades de mapvisualization | Tipo de datos | Descripción de la propiedad |
|---------------------------------|---------------|--|
| size_type | booleano | Especifica si se va a aplicar un tamaño estándar a todos los registros del geocampo, o un tamaño de superposición que cambia el tamaño del icono de símbolo o el grosor de la línea basándose en valores de otro campo del conjunto de datos. Los valores posibles son standard u overlay. El valor predeterminado es standard. |
| tamaño | cadena | Si standard está seleccionado para size_type, para point o multipoint, el desplegable contiene una selección de tamaños para el símbolo seleccionado. Para linestring o multilinestring, el desplegable contiene una selección del grosor de línea. |
| size_field | campo | Si overlay está seleccionado para size_type, el desplegable contiene todos los campos del mismo conjunto de datos que el geocampo seleccionado como la capa. |
| transp_type | booleano | Especifica si se va a aplicar una transparencia estándar a todos los registros del geocampo, o una transferencia de superposición que cambia el nivel de transparencia para el símbolo, línea o polígono basándose en valores de otro campo del conjunto de datos. Los valores posibles son standard u overlay. El valor predeterminado es standard. |
| transp | entero | <p>Si standard está seleccionado para transp_type, el desplegable contiene una selección de niveles de transparencia que empieza en 0% (opaco) y va aumentando hasta 100% (transparente) en incrementos del 10 %. Establece la transparencia de puntos, líneas o polígonos en el mapa.</p> <p>Si overlay está seleccionado para size_type, el desplegable contiene todos los campos del mismo conjunto de datos que el geocampo seleccionado como la capa.</p> <p>Para points, multipoints, linestrings y multilinestrings, polygons y multipolygons (que están en la capa inferior), el valor predeterminado es 0%. Para polygons y multipolygons que no están en la capa inferior, el valor predeterminado es 50% (para evitar oscurecer las capas entre estos polígonos.)</p> |
| transp_field | campo | Si overlay está seleccionado para transp_type, el desplegable contiene todos los campos del mismo conjunto de datos que el geocampo seleccionado como la capa. |
| data_label_field | campo | Especifica el campo para utilizar como etiquetas de datos en el mapa. Por ejemplo, si la capa donde se aplica este valor es un capa de polígono, la etiqueta de datos podría ser el campo name – que contiene el nombre de cada polígono. Así si aquí se selecciona el campo name, se mostrarán estos nombres en el mapa. |

Tabla 102. Propiedades de mapvisualization (continuación)

| Propiedades de mapvisualization | Tipo de datos | Descripción de la propiedad |
|---|---------------|--|
| use_hex_binning | booleano | Habilita la agrupación hexadecimal y habilita todos los desplegados de agregación. Este valor está desactivado de forma predeterminada. |
| color_aggregation y transp_aggregation | cadena | <p>Si selecciona un campo de superposición para una capa de puntos que utiliza la agrupación hexadecimal, todos los valores para dicho campo se deben agregar para todos los puntos del hexágono. Por lo tanto, debe especificar una función de agregación para cualquier otro campo de superposición que desea aplicar al mapa.</p> <p>Las funciones de agregación disponibles son:</p> <p>Continuas (almacenamiento de Real o Entero)</p> <ul style="list-style-type: none"> • Suma • Media. • Mín • Máx • Mediana • Primer cuartil • Tercer cuartil <p>Continuas (almacenamiento de Hora, Fecha o Indicación de fecha y hora):</p> <ul style="list-style-type: none"> • Media. • Mín • Máx <p>Nominal/Categorico:</p> <ul style="list-style-type: none"> • Modo • Mín • Máx <p>Distintivo:</p> <ul style="list-style-type: none"> • True si es verdadero • False si es falso |
| custom_storage | cadena | Establece el tipo de almacenamiento general del campo. El valor predeterminado es List. Si se ha especificado List, los controles custom_value_storage y list_depth siguientes están inhabilitados. |
| custom_value_storage | cadena | Establece los tipos de almacenamiento de los elementos de la lista, en lugar de los campos como un todo. EL valor predeterminado es Real. |

Tabla 102. Propiedades de mapvisualization (continuación)

| Propiedades de mapvisualization | Tipo de datos | Descripción de la propiedad |
|---------------------------------|---------------|--|
| list_depth | entero | <p>Establece la profundidad del campo de lista. Su profundidad necesaria depende del tipo de geocampo, siguiendo estos criterios:</p> <ul style="list-style-type: none"> • Punto - 0 • Serie lineal - 1 • Polígono - 2 • Multipunto - 1 • Multi serie lineal - 2 • Multipolígono - 3 <p>Debe conocer el tipo de campo geoespacial que está volviendo a convertir a una lista y la profundidad necesaria para dicho tipo de campo. Si se ha establecido de forma incorrecta, el campo no se puede utilizar.</p> <p>El valor predeterminado es 0, el valor mínimo es 0, y el valor máximo es 10.</p> |

Propiedades de multiplotnode



El nodo G. múltiple crea un gráfico que muestra varios campos Y sobre un campo X único. Los campos Y están representados como líneas coloreadas; cada uno equivale a un nodo Gráfico con el estilo establecido en **Línea** y el Modo para X establecido en **Ordenar**. Los gráficos múltiples son útiles cuando quiere explorar la fluctuación de varias variables a través del tiempo.

Ejemplo

```
node = stream.create("multiplot", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("x_field", "Age")
node.setPropertyValue("y_fields", ["Drug", "BP"])
node.setPropertyValue("panel_field", "Sex")
# Sección "Superponer"
node.setPropertyValue("animation_field", "")
node.setPropertyValue("tooltip", "test")
node.setPropertyValue("normalize", True)
node.setPropertyValue("use_overlay_expr", False)
node.setPropertyValue("overlay_expression", "test")
node.setPropertyValue("records_limit", 500)
node.setPropertyValue("if_over_limit", "PlotSample")
```

Tabla 103. propiedades de multiplotnode

| Propiedad de multiplotnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---------------|-----------------------------|
| x_field | campo | |
| y_fields | lista | |
| panel_field | campo | |
| animation_field | campo | |
| normalize | flag | |
| use_overlay_expr | flag | |

Tabla 103. propiedades de *multiplotnode* (continuación)

| Propiedad de <i>multiplotnode</i> | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|-----------------------------------|---|
| <code>overlay_expression</code> | <i>cadena</i> | |
| <code>records_limit</code> | <i>número</i> | |
| <code>if_over_limit</code> | PlotBins PlotSample PlotAll | |
| <code>x_label_auto</code> | <i>flag</i> | |
| <code>x_label</code> | <i>cadena</i> | |
| <code>y_label_auto</code> | <i>flag</i> | |
| <code>y_label</code> | <i>cadena</i> | |
| <code>use_grid</code> | <i>flag</i> | |
| <code>graph_background</code> | <i>color</i> | Al principio de esta sección se describen los colores de gráficos estándar. |
| <code>page_background</code> | <i>color</i> | Al principio de esta sección se describen los colores de gráficos estándar. |

Propiedades de *plotnode*



El nodo Gráfico muestra la relación entre los campos numéricos. Puede crear un gráfico mediante puntos (un diagrama de dispersión) o líneas.

Ejemplo

```
node = stream.create("plot", "My node")
# Pestaña "Gráfico"
node.setPropertyValue("three_D", True)
node.setPropertyValue("x_field", "BP")
node.setPropertyValue("y_field", "Cholesterol")
node.setPropertyValue("z_field", "Drug")
# Sección "Superponer"
node.setPropertyValue("color_field", "Drug")
node.setPropertyValue("size_field", "Age")
node.setPropertyValue("shape_field", "")
node.setPropertyValue("panel_field", "Sex")
node.setPropertyValue("animation_field", "BP")
node.setPropertyValue("transp_field", "")
node.setPropertyValue("style", "Point")
# Pestaña "Resultados"
node.setPropertyValue("output_mode", "File")
node.setPropertyValue("output_format", "JPEG")
node.setPropertyValue("full_filename", "C:/temp/graph_output/plot_output.jpeg")
```

Tabla 104. propiedades de *plotnode*.

| Propiedad de <i>plotnode</i> | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------|---|
| <code>x_field</code> | <i>campo</i> | Especifica una etiqueta personalizada para el eje <i>x</i> . Disponible solamente para etiquetas. |
| <code>y_field</code> | <i>campo</i> | Especifica una etiqueta personalizada para el eje <i>y</i> . Disponible solamente para etiquetas. |

Tabla 104. propiedades de plotnode (continuación).

| Propiedad de plotnode | Tipo de datos | Descripción de la propiedad |
|-----------------------|---|--|
| three_D | tag | Especifica una etiqueta personalizada para el eje y. Disponible sólo para etiquetas en gráficos 3D. |
| z_field | campo | |
| color_field | campo | Campo de superposición. |
| size_field | campo | |
| shape_field | campo | |
| panel_field | campo | Especifica un campo de marcas o nominal para crear un gráfico independiente para cada categoría. Los gráficos aparecerán juntos en una ventana de resultados. |
| animation_field | campo | Especifica un campo de marcas o nominal para ilustrar las categorías de los valores de datos creando una serie de gráficos secuenciados mediante la animación. |
| transp_field | campo | Especifica un campo para ilustrar las categorías de los valores de datos utilizando un nivel de transparencia distinto para cada categoría. No disponible para gráficos de líneas. |
| overlay_type | Ninguno Smoother Función | Determina si se muestra una función superpuesta o suavizado LOESS. |
| overlay_expression | cadena | Especifica la expresión utilizada cuando overlay_type se establece en Function. |
| style | Point Line | |
| point_type | Rectángulo Dot Triangle Hexagon Plus Pentagon Star BowTie HorizontalDash VerticalDash IronCross Factory House Cathedral OnionDome ConcaveTriangle OblateGlobe CatEye FourSidedPillow RoundRectangle Fan | |
| x_mode | Ordenar Overlay AsRead | |
| x_range_mode | Automatic UserDefined | |
| x_range_min | number | |

Tabla 104. propiedades de plotnode (continuación).

| Propiedad de plotnode | Tipo de datos | Descripción de la propiedad |
|-----------------------|-----------------------------------|---|
| x_range_max | number | |
| y_range_mode | Automatic UserDefined | |
| y_range_min | number | |
| y_range_max | number | |
| z_range_mode | Automatic UserDefined | |
| z_range_min | number | |
| z_range_max | number | |
| jitter | tag | |
| records_limit | number | |
| if_over_limit | PlotBins PlotSample PlotAll | |
| x_label_auto | tag | |
| x_label | cadena | |
| y_label_auto | tag | |
| y_label | cadena | |
| z_label_auto | tag | |
| z_label | cadena | |
| use_grid | tag | |
| graph_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| page_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| use_overlay_expr | tag | Desaprobado en favor de overlay_type. |

Propiedades de timeplotnode



El nodo Gráfico de tiempo muestra uno o más conjuntos de datos de series temporales. Normalmente, primero se utilizaría un nodo Intervalos de tiempo para crear un campo *EtiquetaTiempo*, que se utilizaría para etiquetar el eje x.

Ejemplo

```
node = stream.create("timeplot", "My node")
node.setPropertyValue("y_fields", ["sales", "men", "women"])
node.setPropertyValue("panel", True)
node.setPropertyValue("normalize", True)
node.setPropertyValue("line", True)
node.setPropertyValue("smoother", True)
node.setPropertyValue("use_records_limit", True)
node.setPropertyValue("records_limit", 2000)
# Appearance settings
node.setPropertyValue("symbol_size", 2.0)
```

Tabla 105. propiedades de timeplotnode.

| Propiedad de timeplotnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|---|---|
| plot_series | Series Models | |
| use_custom_x_field | tag | |
| x_field | campo | |
| y_fields | lista | |
| panel | tag | |
| normalize | tag | |
| line | tag | |
| points | tag | |
| point_type | Rectángulo Dot Triangle Hexagon Plus Pentagon Star BowTie HorizontalDash VerticalDash IronCross Factory House Cathedral OnionDome ConcaveTriangle OblateGlobe CatEye FourSidedPillow RoundRectangle Fan | |
| suavizado | tag | Puede añadir suavizados al gráfico únicamente si establece panel en True. |
| use_records_limit | tag | |
| records_limit | entero | |
| symbol_size | number | Especifica el tamaño del símbolo. |
| panel_layout | Horizontal Vertical | |

Propiedades de webnode



El nodo Malla muestra la fuerza de las relaciones entre los valores de dos o más campos simbólicos (categóricos). El gráfico utiliza líneas de varios anchos para indicar la fuerza de la conexión. Podría utilizar un nodo Malla, por ejemplo, para explorar las relaciones existentes entre la compra de un conjunto de elementos en un sitio de comercio electrónico.

Ejemplo

```
node = stream.create("web", node "My ")
# Pestaña "Gráfico"
node.setPropertyValue("use_directed_web", True)
```

```

node.setPropertyValue("to_field", "Drug")
node.setPropertyValue("fields", ["BP", "Cholesterol", "Sex", "Drug"])
node.setPropertyValue("from_fields", ["BP", "Cholesterol", "Sex"])
node.setPropertyValue("true_flags_only", False)
node.setPropertyValue("line_values", "Absolute")
node.setPropertyValue("strong_links_heavier", True)
# pestaña "Opciones"
node.setPropertyValue("max_num_links", 300)
node.setPropertyValue("links_above", 10)
node.setPropertyValue("num_links", "ShowAll")
node.setPropertyValue("discard_links_min", True)
node.setPropertyValue("links_min_records", 5)
node.setPropertyValue("discard_links_max", True)
node.setPropertyValue("weak_below", 10)
node.setPropertyValue("strong_above", 19)
node.setPropertyValue("link_size_continuous", True)
node.setPropertyValue("web_display", "Circular")

```

Tabla 106. propiedades de webnode

| Propiedad de webnode | Tipo de datos | Descripción de la propiedad |
|----------------------|---|---|
| use_directed_web | flag | |
| campos | lista | |
| to_field | campo | |
| from_fields | lista | |
| true_flags_only | flag | |
| line_values | Absolute OverallPct PctLarger PctSmaller | |
| strong_links_heavier | flag | |
| num_links | ShowMaximum ShowLinksAbove ShowAll | |
| max_num_links | número | |
| links_above | número | |
| discard_links_min | flag | |
| links_min_records | número | |
| discard_links_max | flag | |
| links_max_records | número | |
| weak_below | número | |
| strong_above | número | |
| link_size_continuous | flag | |
| web_display | Circular Red Directed cuadrícula | |
| graph_background | color | Al principio de esta sección se describen los colores de gráficos estándar. |
| symbol_size | número | Especifica el tamaño del símbolo. |

Capítulo 13. Propiedades de nodos de modelado

Propiedades comunes de nodos de modelado

Las siguientes propiedades son comunes a algunos o todos los nodos de modelado. Las excepciones se indican en la documentación de los nodos de modelado individuales según sea adecuado.

Tabla 107. Propiedades comunes de nodos de modelado

| Propiedad | Valores | Descripción de la propiedad |
|--------------------------|---|--|
| custom_fields | <i>flag</i> | Si es verdadero, le permite especificar el objetivo, la entrada y otros campos del nodo actual. Si es falso, se utiliza la configuración actual de un nodo Tipo situado en un punto anterior de la ruta. |
| objetivo o targets | <i>campo</i> o [<i>field1 ... fieldN</i>] | Especifica un único campo objetivo o varios campos objetivo dependiendo del tipo de modelo. |
| inputs | [<i>field1 ... fieldN</i>] | Campos de entrada o predictor utilizados por el modelo. |
| partición | <i>campo</i> | |
| use_partitioned_data | <i>flag</i> | Si se ha definido un campo de partición, esta opción garantiza que sólo se utilizarán los datos de la partición de entrenamiento para la generación del modelo. |
| use_split_data | <i>flag</i> | |
| splits | [<i>campo1 ... campoN</i>] | Especifica el campo o campos para utilizar en el modelado de divisiones. Sólo funciona si use_split_data está establecido como True. |
| use_frequency | <i>flag</i> | Los modelos específicos utilizan campos de ponderación y frecuencia como se indica en cada tipo de modelo. |
| frequency_field | <i>campo</i> | |
| use_weight | <i>flag</i> | |
| weight_field | <i>campo</i> | |
| use_model_name | <i>flag</i> | |
| model_name | <i>cadena</i> | Nombre personalizado para nuevo modelo. |
| mode | Simple Valores avanzados | |

propiedades de anomalydetectionnode



El nodo Detección de anomalías identifica casos extraños, o valores atípicos, que no se ajustan a patrones de datos "normales". Con este nodo, es posible identificar valores atípicos aunque no se ajusten a ningún patrón previamente conocido o no se realice una búsqueda exacta.

Ejemplo

```
node = stream.create("anomalydetection", "My node")
node.setPropertyValue("anomaly_method", "PerRecords")
node.setPropertyValue("percent_records", 95)
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("peer_group_num_auto", True)
node.setPropertyValue("min_num_peer_groups", 3)
node.setPropertyValue("max_num_peer_groups", 10)
```

Tabla 108. propiedades de anomalydetectionnode

| Propiedades de anomalydetectionnode | Valores | Descripción de la propiedad |
|-------------------------------------|--|--|
| inputs | [campo1 ... campoN] | Los modelos de detección de anomalías criban registros basándose en los campos de entrada especificados. No utilizan un campo objetivo. Los campos de ponderación y frecuencia tampoco se usan. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| mode | Expert Simple | |
| anomaly_method | IndexLevel PerRecords NumRecords | Especifica el método utilizado para determinar el valor de corte para marcar los registros como anómalos. |
| index_level | número | Especifica el valor de corte mínimo con el que se van a marcar anomalías. |
| percent_records | número | Establece el umbral para marcar registros de acuerdo al porcentaje de registros en los datos de entrenamiento. |
| num_records | número | Establece el umbral para marcar registros de acuerdo al número de registros en los datos de entrenamiento. |
| num_fields | entero | El número de campos de los que se va a informar por cada registro anómalo. |
| impute_missing_values | flag | |
| adjustment_coeff | número | Valor que se usa para equilibrar la ponderación relativa asignada a los campos categóricos y continuos al calcular la distancia. |
| peer_group_num_auto | flag | Calcula automáticamente el número de grupos de homólogos. |
| min_num_peer_groups | entero | Especifica el número mínimo de grupos de homólogos empleado cuando peer_group_num_auto se establece en True. |
| max_num_per_groups | entero | Especifica el número máximo de grupos de homólogos. |
| num_peer_groups | entero | Especifica el número de grupos de homólogos empleado cuando peer_group_num_auto se establece en False. |

Tabla 108. propiedades de anomalydetectionnode (continuación)

| Propiedades de anomalydetectionnode | Valores | Descripción de la propiedad |
|-------------------------------------|---------|---|
| noise_level | número | Determina el modo en que se tratan los valores atípicos durante el clúster. Especifique un valor entre 0 y 0,5. |
| noise_ratio | número | Especifica la parte de memoria asignada al componente que debe usarse para el almacenamiento en búfer de ruido. Especifique un valor entre 0 y 0,5. |

propiedades de apriorinode



El nodo Apriori extrae un conjunto de reglas de los datos y destaca aquellas reglas con un mayor contenido de información. Apriori ofrece cinco métodos diferentes para la selección de reglas y utiliza un sofisticado esquema de indización para procesar eficientemente grandes conjuntos de datos. En los problemas de mucho volumen, Apriori se entrena más rápidamente, no tiene un límite arbitrario para el número de reglas que puede retener y puede gestionar reglas que tengan hasta 32 precondiciones. Apriori requiere que todos los campos de entrada y salida sean categóricos, pero ofrece un mejor rendimiento ya que está optimizado para este tipo de datos.

Ejemplo

```
node = stream.create("apriori", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("partition", "Test")
# Para no transaccionales
node.setPropertyValue("use_transactional_data", False)
node.setPropertyValue("consequents", ["Age"])
node.setPropertyValue("antecedents", ["BP", "Cholesterol", "Drug"])
# Para transaccionales
node.setPropertyValue("use_transactional_data", True)
node.setPropertyValue("id_field", "Age")
node.setPropertyValue("contiguous", True)
node.setPropertyValue("content_field", "Drug")
# "Model" tab
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "Apriori_bp_choles_drug")
node.setPropertyValue("min_supp", 7.0)
node.setPropertyValue("min_conf", 30.0)
node.setPropertyValue("max_antecedents", 7)
node.setPropertyValue("true_flags", False)
node.setPropertyValue("optimize", "Memory")
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("evaluation", "ConfidenceRatio")
node.setPropertyValue("lower_bound", 7)
```

Tabla 109. propiedades de apriorinode

| Propiedades de apriorinode | Valores | Descripción de la propiedad |
|----------------------------|--|---|
| consequents | <i>campo</i> | Los modelos Apriori utilizan Consecuentes y Antecedentes en lugar de los campos estándar objetivo y de entrada. Los campos de ponderación y frecuencia no se usan. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| antecedents | [<i>field1 ... fieldN</i>] | |
| min_supp | <i>número</i> | |
| min_conf | <i>número</i> | |
| max_antecedents | <i>número</i> | |
| true_flags | <i>flag</i> | |
| optimize | Speed Memory | |
| use_transactional_data | <i>flag</i> | |
| contiguous | <i>flag</i> | |
| id_field | <i>cadena</i> | |
| content_field | <i>cadena</i> | |
| mode | Simple Valores avanzados | |
| evaluación | RuleConfidence DifferenceToPrior ConfidenceRatio InformationDifference NormalizedChiSquare | |
| lower_bound | <i>número</i> | |
| optimize | Speed Memory | Se utiliza para especificar si la generación del modelo se debe optimizar para la velocidad o la memoria. |

propiedades associationrulesnode



El nodo Reglas de asociación es similar al nodo Apriori; sin embargo, a diferencia de Apriori, el nodo Reglas de asociación puede procesar datos de lista. Además, el nodo Reglas de asociación puede utilizarse con IBM SPSS Analytic Server para procesar datos de gran tamaño y aprovechar el proceso en paralelo de mayor velocidad.

Tabla 110. propiedades associationrulesnode

| Propiedades associationrulesnode | Tipo de datos | Descripción de la propiedad |
|----------------------------------|----------------------------|---|
| predicciones | <i>campo</i> | Los campos en esta lista sólo pueden aparecer como un predictor de una regla |
| condiciones | [<i>field1...fieldN</i>] | Los campos en esta lista sólo pueden aparecer como una condición de una regla |
| max_rule_conditions | <i>entero</i> | El número máximo de condiciones que pueden incluirse en una sola regla. Mínimo 1, máximo 9. |

Tabla 110. propiedades associationrulesnode (continuación)

| Propiedades associationrulesnode | Tipo de datos | Descripción de la propiedad |
|----------------------------------|---|--|
| max_rule_predictions | entero | El número máximo de predicciones que pueden incluirse en una sola regla. Mínimo 1, máximo 5. |
| max_num_rules | entero | El número máximo de reglas que pueden considerarse como parte de la generación de regla. Mínimo 1, máximo 10.000. |
| rule_criterion_top_n | Confianza Rulesupport Lift Conditionsupport Capacidad de despliegue | El criterio de reglas que determina el valor por el cual se eligen las reglas "N" superiores en el modelo. |
| true_flags | Booleana | Establecer en Y determina que sólo se considerarán los valores verdaderos para campos de distintivo durante la generación de la regla. |
| rule_criterion | Booleana | Establecer en Y determina que los valores del criterio de regla se utilizan para excluir reglas durante la generación de modelos. |
| min_confidence | número | 0,1 a 100 - el valor de porcentaje para el nivel de confianza mínimo necesario para una regla producida por el modelo. Si el modelo genera una regla con un nivel de confianza inferior al valor especificado aquí, la regla se descarta. |
| min_rule_support | número | 0,1 a 100 - el valor de porcentaje para el soporte de regla mínimo necesario para una regla producida por el modelo. Si el modelo genera una regla con un nivel de soporte de regla inferior al valor especificado, la regla se descarta. |
| min_condition_support | número | 0,1 a 100 - el valor de porcentaje para el soporte de condición mínima necesaria para una regla producida por el modelo. Si el modelo genera una regla con un nivel de soporte de condición inferior al valor especificado, la regla se descarta. |
| min_lift | entero | 1 a 10 – representa la elevación mínima necesaria para una regla producida por el modelo. Si el modelo genera una regla con un nivel de elevación inferior al valor especificado, la regla se descarta. |
| exclude_rules | Booleana | Se utiliza para seleccionar una lista de campos relacionados a partir de los cuales no desea que el modelo cree reglas. Ejemplo: set :gsarsnode.exclude_rules = [[field1,field2, field3],[field4, field5]] - donde cada lista de campos separados por [] es una fila en la tabla. |
| num_bins | entero | Establezca el número de intervalos automáticos en los que se agrupan los campos continuos. Mínimo 2, máximo 10. |

Tabla 110. propiedades associationrulesnode (continuación)

| Propiedades associationrulesnode | Tipo de datos | Descripción de la propiedad |
|----------------------------------|---|--|
| max_list_length | entero | Se aplica a cualquier campo de lista del que no se conoce la longitud máxima. Los elementos de la lista hasta el número especificado aquí se incluyen en la generación de modelos; los elementos adicionales se descartan. Mínimo 1, máximo 100. |
| output_confidence | Booleana | |
| output_rule_support | Booleana | |
| output_lift | Booleana | |
| output_condition_support | Booleana | |
| output_deployability | Booleana | |
| rules_to_display | upto all | El número máximo de reglas a visualizar en las tablas de salida. |
| display_upto | entero | Si upto se establece en rules_to_display, establezca el número de reglas a visualizar en las tablas de salida. El mínimo es 1. |
| field_transformations | Booleana | |
| records_summary | Booleana | |
| rule_statistics | Booleana | |
| most_frequent_values | Booleana | |
| most_frequent_fields | Booleana | |
| word_cloud | Booleana | |
| word_cloud_sort | Confianza Rulesupport Lift Conditionsupport Capacidad de despliegue | |
| word_cloud_display | entero | Mínimo 1, máximo 20. |
| max_predictions | entero | El número máximo de reglas que se pueden aplicar a cada entrada de la puntuación. |
| criterio | Confianza Rulesupport Lift Conditionsupport Capacidad de despliegue | Seleccione la medida utilizada para determinar la fuerza de las reglas. |
| allow_repeats | Booleana | Determine si las reglas con la misma predicción se incluyen en la puntuación. |
| check_input | NoPredictions Predictions NoCheck | |

propiedades de autotransformador



El nodo Clasificador automático crea y compara varios modelos diferentes para obtener resultados binarios (sí o no, abandono o no de clientes, etc.), lo que le permite seleccionar el mejor enfoque para un análisis determinado. Son compatibles varios algoritmos de modelado, por lo que es posible seleccionar los métodos que desee utilizar, las opciones específicas para cada uno y los criterios para comparar los resultados. El nodo genera un conjunto de modelos basado en las opciones especificadas y clasifica los mejores candidatos en función de los criterios que especifique.

Ejemplo

```
node = stream.create("autoclassifier", "My node")
node.setPropertyValue("ranking_measure", "Accuracy")
node.setPropertyValue("ranking_dataset", "Training")
node.setPropertyValue("enable_accuracy_limit", True)
node.setPropertyValue("accuracy_limit", 0.9)
node.setPropertyValue("calculate_variable_importance", True)
node.setPropertyValue("use_costs", True)
node.setPropertyValue("svm", False)
```

Tabla 111. propiedades de autotransformador.

| Propiedades de autotransformador | Valores | Descripción de la propiedad |
|----------------------------------|--|---|
| objetivo | <i>campo</i> | En objetivos de marca, el nodo Clasificador binario requiere un único campo objetivo y uno o más campos de entrada. También se puede especificar campos de ponderación y frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| ranking_measure | Exactitud Area_under_curve Profit Lift Num_variables | |
| ranking_dataset | Entrenamiento Test | |
| number_of_models | <i>entero</i> | Número de modelos que se incluirán en el nugget de modelo. Especifique un entero entre 1 y 100. |
| calculate_variable_importance | <i>tag</i> | |
| enable_accuracy_limit | <i>tag</i> | |
| accuracy_limit | <i>entero</i> | Entero entre 0 y 100. |
| enable_area_under_curve_limit | <i>tag</i> | |
| area_under_curve_limit | <i>number</i> | Número real entre 0,0 y 1,0. |
| enable_profit_limit | <i>tag</i> | |
| profit_limit | <i>number</i> | Entero mayor que 0. |
| enable_lift_limit | <i>tag</i> | |
| lift_limit | <i>number</i> | Número real mayor que 1,0. |
| enable_number_of_variables_limit | <i>tag</i> | |

Tabla 111. propiedades de autoclassifiernode (continuación).

| Propiedades de autoclassifiernode | Valores | Descripción de la propiedad |
|--|---------|---|
| number_of_variables_limit | number | Entero mayor que 0. |
| use_fixed_cost | tag | |
| fixed_cost | number | Número real mayor que 0.0. |
| variable_cost | campo | |
| use_fixed_revenue | tag | |
| fixed_revenue | number | Número real mayor que 0.0. |
| variable_revenue | campo | |
| use_fixed_weight | tag | |
| fixed_weight | number | Número real mayor que 0,0 |
| variable_weight | campo | |
| lift_percentile | number | Entero entre 0 y 100. |
| enable_model_build_time_limit | tag | |
| model_build_time_limit | number | Entero que indica el número máximo de minutos que se puede tardar en generar cada uno de los modelos. |
| enable_stop_after_time_limit | tag | |
| stop_after_time_limit | number | Número real que indica el número máximo de horas que puede tardar una ejecución del clasificador automático. |
| enable_stop_after_valid_model_produced | tag | |
| use_costs | tag | |
| <algorithm> | tag | Activa o desactiva el uso de un determinado algoritmo. |
| <algorithm>.<property> | cadena | Define un valor de propiedad para un algoritmo específico. Consulte el tema "Propiedades de ajustes de algoritmo" para obtener más información. |

Propiedades de ajustes de algoritmo

En el caso de los nodos Clasificador automático, Autonumérico y Agrupación en clústeres automática, las propiedades de determinados algoritmos utilizados por el nodo se pueden establecer utilizando el formato general:

```
autonode.setKeyedPropertyValue(<algoritmo>, <propiedad>, <valor>)
```

Por ejemplo:

```
node.setKeyedPropertyValue("neuralnetwork", "method", "MultilayerPerceptron")
```

Los nombres de algoritmos del nodo Clasificador automático son cart, chaid, quest, c50, logreg, decisionlist, bayesnet, discriminant, svm y knn.

Los nombres de algoritmos del nodo Autonumérico son cart, chaid, neuralnetwork, genlin, svm, regression, linear y knn.

Los nombres de algoritmos del nodo Autoclúster son twostep, k-means y kohonen.

Los nombres de las propiedades son los nombres estándar, según se han documentado para cada nodo de algoritmo.

Las propiedades de algoritmos que contienen puntos u otros signos de puntuación deben encerrarse entre comillas simples. Por ejemplo:

```
node.setKeyedPropertyValue("logreg", "tolerance", "1.0E-5")
```

También es posible asignar varios valores a una propiedad. Por ejemplo:

```
node.setKeyedPropertyValue("decisionlist", "search_direction", ["Up", "Down"])
```

Para activar o desactivar el uso de un determinado algoritmo:

```
node.setPropertyValue("chaid", True)
```

Nota: En los casos en los que determinadas opciones de algoritmos no están disponibles en el nodo Clasificador automático o cuando sólo se puede especificar un único valor, en lugar de un intervalo de valores, se aplican los mismos límites que tienen los scripts cuando se accede al nodo de la manera estándar.

propiedades de nodo de agrupación en clústeres automática



El nodo Agrupación en clústeres automática calcula y compara los modelos de agrupación en clústeres que identifican grupos de registros con características similares. El nodo funciona de la misma manera que otros nodos de modelado automático, permitiéndole experimentar con múltiples combinaciones de opciones en una única pasada de modelado. Los modelos se pueden comparar utilizando medidas básicas con las que se intenta filtrar y definir la utilidad de los modelos de clúster y proporcionar una medida según la importancia de campos concretos.

Ejemplo

```
node = stream.create("autocluster", "My node")
node.setPropertyValue("ranking_measure", "Silhouette")
node.setPropertyValue("ranking_dataset", "Training")
node.setPropertyValue("enable_silhouette_limit", True)
node.setPropertyValue("silhouette_limit", 5)
```

Tabla 112. propiedades de nodo de agrupación en clústeres automática

| Propiedades de autoclusternode | Valores | Descripción de la propiedad |
|--------------------------------|--|---|
| evaluación | campo | Nota: Solo nodo Agrupación en clústeres automática. Identifica el campo cuyo valor de importancia se calculará. Como alternativa, se puede utilizar para identificar la calidad con la que el clúster diferencia el valor de este campo y, por lo tanto; la calidad con la que el modelo predecirá este campo. |
| ranking_measure | Silhouette Num_clusters Size_smallest_cluster Size_largest_cluster Smallest_to_largest Importance | |
| ranking_dataset | Entrenamiento Prueba | |

Tabla 112. propiedades de nodo de agrupación en clústeres automática (continuación)

| Propiedades de autoclusternode | Valores | Descripción de la propiedad |
|-----------------------------------|---------------------------|--|
| summary_limit | entero | Número de modelos que se incluirán en el informe. Especifique un entero entre 1 y 100. |
| enable_silhouette_limit | flag | |
| silhouette_limit | entero | Entero entre 0 y 100. |
| enable_number_less_limit | flag | |
| number_less_limit | número | Número real entre 0,0 y 1,0. |
| enable_number_greater_limit | flag | |
| number_greater_limit | número | Entero mayor que 0. |
| enable_smallest_cluster_limit | flag | |
| smallest_cluster_units | Percentage Counts | |
| smallest_cluster_limit_percentage | número | |
| smallest_cluster_limit_count | entero | Entero mayor que 0. |
| enable_largest_cluster_limit | flag | |
| largest_cluster_units | Percentage Counts | |
| largest_cluster_limit_percentage | número | |
| largest_cluster_limit_count | entero | |
| enable_smallest_largest_limit | flag | |
| smallest_largest_limit | número | |
| enable_importance_limit | flag | |
| importance_limit_condition | Greater_than Less_than | |
| importance_limit_greater_than | número | Entero entre 0 y 100. |
| importance_limit_less_than | número | Entero entre 0 y 100. |
| <algorithm> | flag | Activa o desactiva el uso de un determinado algoritmo. |
| <algorithm>.<property> | cadena | Define un valor de propiedad para un algoritmo específico. Consulte el tema "Propiedades de ajustes de algoritmo" en la página 186 para obtener más información. |

propiedades de autonumericnode



El nodo Autonumérico calcula y compara modelos para resultados de rango numérico continuo utilizando cierto número de métodos diferentes. El nodo funciona de la misma manera que el nodo Clasificador automático, lo que le permite seleccionar los algoritmos que desee utilizar y experimentar con varias combinaciones de opciones en una única pasada de modelado. Los algoritmos admitidos incluyen redes neuronales, C&RT, CHAID, regresión lineal, regresión lineal generalizada y máquinas de vectores de soporte (SVM). Los modelos se pueden comparar basándose en la correlación, el error relativo o el número de variables utilizado.

Ejemplo

```

node = stream.create("autonumeric", "My node")
node.setPropertyValue("ranking_measure", "Correlation")
node.setPropertyValue("ranking_dataset", "Training")
node.setPropertyValue("enable_correlation_limit", True)
node.setPropertyValue("correlation_limit", 0.8)
node.setPropertyValue("calculate_variable_importance", True)
node.setPropertyValue("neuralnetwork", True)
node.setPropertyValue("chaid", False)

```

Tabla 113. propiedades de autonumericnode

| Propiedades de autonumericnode | Valores | Descripción de la propiedad |
|--------------------------------|-------------------------------|--|
| custom_fields | <i>flag</i> | Si es verdadero, se utilizará la configuración de campos personalizada en lugar de la configuración del nodo Tipo. |
| target | <i>campo</i> | El nodo Autonumérico requiere un único campo objetivo y uno o más campos de entrada. También se puede especificar campos de ponderación y frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| inputs | [<i>campo1 ... campo2</i>] | |
| partición | <i>campo</i> | |
| use_frequency | <i>flag</i> | |
| frequency_field | <i>campo</i> | |
| use_weight | <i>flag</i> | |
| weight_field | <i>campo</i> | |
| use_partitioned_data | <i>flag</i> | Si se ha definido un campo de partición, sólo se utilizarán los datos de entrenamiento para la generación del modelo. |
| ranking_measure | Correlation NumberOfFields | |
| ranking_dataset | Test Entrenamiento | |
| number_of_models | <i>entero</i> | Número de modelos que se incluirán en el nugget de modelo. Especifique un entero entre 1 y 100. |
| calculate_variable_importance | <i>flag</i> | |
| enable_correlation_limit | <i>flag</i> | |
| correlation_limit | <i>entero</i> | |
| enable_number_of_fields_limit | <i>flag</i> | |
| number_of_fields_limit | <i>entero</i> | |
| enable_relative_error_limit | <i>flag</i> | |
| relative_error_limit | <i>entero</i> | |
| enable_model_build_time_limit | <i>flag</i> | |
| model_build_time_limit | <i>entero</i> | |
| enable_stop_after_time_limit | <i>flag</i> | |
| stop_after_time_limit | <i>entero</i> | |

Tabla 113. propiedades de autonumericnode (continuación)

| Propiedades de autonumericnode | Valores | Descripción de la propiedad |
|--------------------------------|---------|--|
| stop_if_valid_model | flag | |
| <algorithm> | flag | Activa o desactiva el uso de un determinado algoritmo. |
| <algorithm>.<property> | cadena | Define un valor de propiedad para un algoritmo específico. Consulte el tema "Propiedades de ajustes de algoritmo" en la página 186 para obtener más información. |

Propiedades de bayesnetnode



El nodo Red bayesiana le permite crear un modelo de probabilidad combinando pruebas observadas y registradas con conocimiento del mundo real para establecer la probabilidad de instancias. El nodo se centra en las redes Naïve Bayes aumentado a árbol (TAN) y de manto de Markov que se utilizan principalmente para la clasificación.

Ejemplo

```
node = stream.create("bayesnet", "My node")
node.setPropertyValue("continue_training_existing_model", True)
node.setPropertyValue("structure_type", "MarkovBlanket")
node.setPropertyValue("use_feature_selection", True)
# pestaña Experto
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("all_probabilities", True)
node.setPropertyValue("independence", "Pearson")
```

Tabla 114. Propiedades de bayesnetnode

| Propiedades de bayesnetnode | Valores | Descripción de la propiedad |
|----------------------------------|----------------------|---|
| inputs | [campo1 ... campoN] | Los modelos de red bayesiana utilizan un único campo objetivo y uno o más campos de entrada. Los campos continuos se establecen en intervalos automáticamente. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue_training_existing_model | flag | |
| structure_type | TAN MarkovBlanket | Seleccione la estructura que desea utilizar cuando cree la red bayesiana. |
| use_feature_selection | flag | |
| parameter_learning_method | Likelihood Bayes | Especifica el método utilizado para calcular las tablas de probabilidad condicional entre nodos donde se conocen los valores de los elementos padre. |
| mode | Expert Simple | |
| missing_values | flag | |
| all_probabilities | flag | |

Tabla 114. Propiedades de bayesnetnode (continuación)

| Propiedades de bayesnetnode | Valores | Descripción de la propiedad |
|---------------------------------|-----------------------|---|
| independence | Likelihood Pearson | Especifica el método utilizado para determinar si las observaciones relacionadas de dos variables son independientes entre sí. |
| significance_level | número | Especifica el valor de corte para determinar la independencia. |
| maximal_conditioning_set | número | Establece el número máximo de variables de condición que se utilizarán para la comprobación de la independencia. |
| inputs_always_selected | [campo1 ... campoN] | Especifica qué campos del conjunto de datos se deben utilizar siempre al generar la red bayesiana. Nota: el campo objetivo siempre está seleccionado. |
| maximum_number_inputs | número | Especifica el número máximo de campos de entrada que se deben utilizar al generar la red bayesiana. |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

propiedades de buildr



El nodo Crear R le permite especificar script R personalizado para realizar la creación de modelos y la puntuación de modelos desplegados en IBM SPSS Modeler.

Ejemplo

```
node = stream.create("buildr", "My node")
node.setPropertyValue("score_syntax", "")
result<-predict(modelerModel,newdata=modelerData)
modelerData<-cbind(modelerData,result)
var1<-c(fieldName="NaPrediction",fieldLabel="",fieldStorage="real",fieldMeasure="",
fieldFormat="",fieldRole="")
modelerDataModel<-data.frame(modelerDataModel,var1)"")
```

Tabla 115. propiedades de buildr.

| Propiedades de buildr | Valores | Descripción de la propiedad |
|-----------------------|------------------------------------|--|
| build_syntax | cadena | Sintaxis de scripts R para la creación del modelo. |
| score_syntax | cadena | Sintaxis de scripts R para la puntuación del modelo. |
| convert_flags | StringsAndDoubles LogicalValues | Opción para convertir campos de distintos. |

Tabla 115. propiedades de buildr (continuación).

| Propiedades de buildr | Valores | Descripción de la propiedad |
|------------------------|--------------------|--|
| convert_datetime | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| convert_datetime_class | POSIXct POSIX1t | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |
| convert_missing | <i>tag</i> | Opción para convertir los valores que faltan al valor R NA. |
| output_html | <i>tag</i> | Opción para visualizar gráficos en una pestaña en el nugget de modelo R. |
| output_text | <i>flag</i> | Opción para escribir salida de texto de la consola R en una pestaña del modelo R. |

propiedades de c50node



El nodo C5.0 genera un árbol de decisión o un conjunto de reglas. El modelo divide la muestra basándose en el campo que ofrece la máxima ganancia de información en cada nivel. El campo objetivo debe ser categórico. Se permiten varias divisiones en más de dos subgrupos.

Ejemplo

```
node = stream.create("c50", "My node")
# "Model" tab
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "C5_Drug")
node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("output_type", "DecisionTree")
node.setPropertyValue("use_xval", True)
node.setPropertyValue("xval_num_folds", 3)
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("favor", "Generality")
node.setPropertyValue("min_child_records", 3)
# pestaña "Costes"
node.setPropertyValue("use_costs", True)
node.setPropertyValue("costs", [["drugA", "drugX", 2]])
```

Tabla 116. propiedades de c50node

| Propiedad de c50node | Valores | Descripción de la propiedad |
|----------------------|-------------------------|--|
| target | <i>campo</i> | Los modelos C50 utilizan un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de ponderación. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| output_type | DecisionTree RuleSet | |
| group_symbolics | <i>flag</i> | |
| use_boost | <i>flag</i> | |

Tabla 116. propiedades de c50node (continuación)

| Propiedad de c50node | Valores | Descripción de la propiedad |
|---------------------------------|-----------------------------|--|
| boost_num_trials | número | |
| use_xval | flag | |
| xval_num_folds | número | |
| mode | Simple Valores avanzados | |
| favor | Exactitud Generality | Generalización o precisión de favor. |
| expected_noise | número | |
| min_child_records | número | |
| pruning_severity | número | |
| use_costs | flag | |
| costes | structured | Ésta es una propiedad estructurada. |
| use_winnowing | flag | |
| use_global_pruning | flag | Activado (True) de forma predeterminada. |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

propiedades de carmanode



El modelo CARMA extrae un conjunto de reglas de los datos sin necesidad de especificar campos de entrada ni de objetivo. A diferencia de Apriori el nodo CARMA ofrece configuraciones de generación basadas en el soporte de las reglas (soporte tanto para el antecedente como el consecuente) en lugar de hacerlo sólo respecto al soporte del antecedente. Esto significa que las reglas generadas se pueden utilizar en una gama de aplicaciones más amplia, por ejemplo, para buscar una lista de productos o servicios (antecedentes) cuyo consecuente es el elemento que se desea promocionar durante esta temporada de vacaciones.

Ejemplo

```
node = stream.create("carma", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("use_transactional_data", True)
node.setPropertyValue("inputs", ["BP", "Cholesterol", "Drug"])
node.setPropertyValue("partition", "Test")
# "Model" tab
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "age_bp_drug")
node.setPropertyValue("use_partitioned_data", False)
node.setPropertyValue("min_supp", 10.0)
node.setPropertyValue("min_conf", 30.0)
node.setPropertyValue("max_size", 5)
# Opciones de experto
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("use_pruning", True)
```

```

node.setPropertyValue("pruning_value", 300)
node.setPropertyValue("vary_support", True)
node.setPropertyValue("estimated_transactions", 30)
node.setPropertyValue("rules_without_antecedents", True)

```

Tabla 117. propiedades de carmanode

| Propiedades de carmanode | Valores | Descripción de la propiedad |
|---------------------------|-----------------------------|--|
| inputs | [campo1 ... campon] | Los modelos CARMA utilizan una lista de campos de entrada, pero no de campos objetivo. Los campos de ponderación y frecuencia no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| id_field | campo | Campo utilizado como el campo de ID para la generación del modelo. |
| contiguous | flag | Se utiliza para especificar si los ID del campo de ID son contiguos. |
| use_transactional_data | flag | |
| content_field | campo | |
| min_supp | número (porcentaje) | Está relacionado con el soporte de regla en lugar de con el soporte de antecedentes. El valor por omisión es 20%. |
| min_conf | número (porcentaje) | El valor por omisión es 20%. |
| max_size | número | El valor por omisión es 10. |
| mode | Simple Valores avanzados | El valor predeterminado es Simple. |
| exclude_multiple | flag | Excluye las reglas con varios consecuentes. El valor predeterminado es False. |
| use_pruning | flag | El valor predeterminado es False. |
| pruning_value | número | El valor predeterminado es 500. |
| vary_support | flag | |
| estimated_transactions | entero | |
| rules_without_antecedents | flag | |

propiedades de cartnode



El nodo de árbol de clasificación y regresión (C&R) genera un árbol de decisión que permite predecir o clasificar observaciones futuras. El método utiliza la partición reiterada para dividir los registros de entrenamiento en segmentos minimizando las impurezas en cada paso, donde un nodo se considera "puro" si el 100% de los casos del nodo corresponden a una categoría específica del campo objetivo. Los campos de entrada y objetivo pueden ser continuos (rango numérico) o categóricos (nominal, ordinal o marca). Todas las divisiones son binarias (sólo se crean dos subgrupos).

Ejemplo

```

node = stream.createAt("cart", "My node", 200, 100)
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Drug")
node.setPropertyValue("inputs", ["Age", "BP", "Cholesterol"])

```

```

# "Build Options" tab, "Objective" panel
node.setPropertyValue("model_output_type", "InteractiveBuilder")
node.setPropertyValue("use_tree_directives", True)
node.setPropertyValue("tree_directives", """Grow Node Index 0 Children 1 2
Grow Node Index 2 Children 3 4""")
# "Build Options" tab, "Basics" panel
node.setPropertyValue("prune_tree", False)
node.setPropertyValue("use_std_err_rule", True)
node.setPropertyValue("std_err_multiplier", 3.0)
node.setPropertyValue("max_surrogates", 7)
# "Build Options" tab, "Stopping Rules" panel
node.setPropertyValue("use_percentage", True)
node.setPropertyValue("min_parent_records_pc", 5)
node.setPropertyValue("min_child_records_pc", 3)
# "Build Options" tab, "Advanced" panel
node.setPropertyValue("min_impurity", 0.0003)
node.setPropertyValue("impurity_measure", "Twoing")
# Pestaña "Opciones de modelo"
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Cart_Drug")

```

Tabla 118. propiedades de cartnode

| Propiedades de cartnode | Valores | Descripción de la propiedad |
|----------------------------------|--|--|
| target | <i>campo</i> | Los modelos de árbol C&R requieren un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue_training_existing_model | <i>flag</i> | |
| objective | Standard Aumento Agregación autodocimante psm | psm se utiliza para conjuntos de datos de grandes dimensiones y requiere una conexión al Servidor. |
| model_output_type | Single InteractiveBuilder | |
| use_tree_directives | <i>flag</i> | |
| tree_directives | <i>cadena</i> | Especifique directivas para desarrollar el árbol. Las directivas se pueden escribir entre comillas triples para evitar comillas o saltos de línea no deseados. Recuerde que las directivas pueden ser muy sensibles a las pequeñas modificaciones de las opciones de modelado o los datos y es posible que no se puedan generalizar para otros conjuntos de datos. |
| use_max_depth | Predeterminado Personalizado | |
| max_depth | <i>entero</i> | Máxima profundidad del árbol, desde 0 a 1000. Sólo se utiliza si use_max_depth = Custom. |
| prune_tree | <i>flag</i> | Poda del árbol para evitar sobreajustes. |

Tabla 118. propiedades de cartnode (continuación)

| Propiedades de cartnode | Valores | Descripción de la propiedad |
|---------------------------------|--|---|
| use_std_err | <i>flag</i> | Use la diferencia máxima en riesgos (en errores estándar). |
| std_err_multiplier | <i>número</i> | Diferencia máxima. |
| max_surrogates | <i>número</i> | Número máximo de sustitutos. |
| use_percentage | <i>flag</i> | |
| min_parent_records_pc | <i>número</i> | |
| min_child_records_pc | <i>número</i> | |
| min_parent_records_abs | <i>número</i> | |
| min_child_records_abs | <i>número</i> | |
| use_costs | <i>flag</i> | |
| costes | <i>structured</i> | Propiedad estructurada. |
| priors | Datos Equal Personalizado | |
| custom_priors | <i>structured</i> | Propiedad estructurada. |
| adjust_priors | <i>flag</i> | |
| trails | <i>número</i> | Número de modelos de componente para un aumento o agregación autodocimante. |
| set_ensemble_method | Voting HighestProbability HighestMeanProbability | Regla de combinación predeterminada para objetivos categóricos. |
| range_ensemble_method | Media Mediana | Regla de combinación predeterminada para objetivos continuos. |
| large_boost | <i>flag</i> | Aplicar aumento a conjunto de datos muy grandes. |
| min_impurity | <i>número</i> | |
| impurity_measure | Gini Twoing Ordered | |
| train_pct | <i>número</i> | Conjunto de prevención sobreajustado. |
| set_random_seed | <i>flag</i> | Opción replicar resultados. |
| seed | <i>número</i> | |
| calculate_variable_importance | <i>flag</i> | |
| calculate_raw_propensities | <i>flag</i> | |
| calculate_adjusted_propensities | <i>flag</i> | |
| adjusted_propensity_partition | Test Validation | |

propiedades de chaidnode



El nodo CHAID genera árboles de decisión utilizando estadísticos de chi-cuadrado para identificar las divisiones óptimas. A diferencia de los nodos C&RT y Árbol y QUEST, CHAID puede generar árboles no binarios, lo que significa que algunas divisiones generarán más de dos ramas. Los campos de entrada y objetivo pueden ser continuos (rango numérico) o categóricos. CHAID exhaustivo es una modificación de CHAID que examina con mayor precisión todas las divisiones posibles, aunque necesita más tiempo para realizar los cálculos.

Ejemplo

```

filenode = stream.createAt("variablefile", "My node", 100, 100)
filenode.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
node = stream.createAt("chaid", "My node", 200, 100)
stream.link(filenode, node)

node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Drug")
node.setPropertyValue("inputs", ["Age", "Na", "K", "Cholesterol", "BP"])
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "CHAID")
node.setPropertyValue("method", "Chaid")
node.setPropertyValue("model_output_type", "InteractiveBuilder")
node.setPropertyValue("use_tree_directives", True)
node.setPropertyValue("tree_directives", "Test")
node.setPropertyValue("split_alpha", 0.03)
node.setPropertyValue("merge_alpha", 0.04)
node.setPropertyValue("chi_square", "Pearson")
node.setPropertyValue("use_percentage", False)
node.setPropertyValue("min_parent_records_abs", 40)
node.setPropertyValue("min_child_records_abs", 30)
node.setPropertyValue("epsilon", 0.003)
node.setPropertyValue("max_iterations", 75)
node.setPropertyValue("split_merged_categories", True)
node.setPropertyValue("bonferroni_adjustment", True)

```

Tabla 119. propiedades de chaidnode

| Propiedad de chaidnode | Valores | Descripción de la propiedad |
|----------------------------------|--|--|
| target | <i>campo</i> | Los modelos CHAID requieren un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue_training_existing_model | <i>flag</i> | |
| objective | Standard Aumento Agregación autodocimante psm | psm se utiliza para conjuntos de datos de grandes dimensiones y requiere una conexión al Servidor. |
| model_output_type | Single InteractiveBuilder | |
| use_tree_directives | <i>flag</i> | |
| tree_directives | <i>cadena</i> | |
| method | Chaid ExhaustiveChaid | |

Tabla 119. propiedades de chaidnode (continuación)

| Propiedad de chaidnode | Valores | Descripción de la propiedad |
|---------------------------------|--|---|
| use_max_depth | Predeterminado Personalizado | |
| max_depth | entero | Máxima profundidad del árbol, desde 0 a 1000. Sólo se utiliza si use_max_depth = Custom. |
| use_percentage | flag | |
| min_parent_records_pc | número | |
| min_child_records_pc | número | |
| min_parent_records_abs | número | |
| min_child_records_abs | número | |
| use_costs | flag | |
| costes | structured | Propiedad estructurada. |
| trails | número | Número de modelos de componente para un aumento o agregación autodocimante. |
| set_ensemble_method | Voting HighestProbability HighestMeanProbability | Regla de combinación predeterminada para objetivos categóricos. |
| range_ensemble_method | Media Mediana | Regla de combinación predeterminada para objetivos continuos. |
| large_boost | flag | Aplicar aumento a conjunto de datos muy grandes. |
| split_alpha | número | Nivel de significancia para división. |
| merge_alpha | número | Nivel de significancia para fusión. |
| bonferroni_adjustment | flag | Los valores de significancia de ajuste utilizando el método de Bonferroni. |
| split_merged_categories | flag | Permitir segunda división de categorías fusionadas. |
| chi_square | Pearson LR | Método usado para calcular la estadística de chi cuadrado: Pearson o Razón de verosimilitud |
| epsilon | número | Cambio mínimo en frecuencias de casillas esperadas. |
| max_iterations | número | Número máximo de iteraciones para la convergencia. |
| set_random_seed | entero | |
| seed | número | |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |
| maximum_number_of_models | entero | |

propiedades de coxregnode



El nodo Regresión de Cox le permite crear un modelo de supervivencia para datos de tiempo hasta el evento en presencia de registros censurados. El modelo produce una función de supervivencia que predice la probabilidad de que el evento de interés se haya producido en el momento dado (t) para valores determinados de las variables de entrada.

Ejemplo

```
node = stream.create("coxreg", "My node")
node.setPropertyValue("survival_time", "tenure")
node.setPropertyValue("method", "BackwardsStepwise")
# pestaña Experto
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("removal_criterion", "Conditional")
node.setPropertyValue("survival", True)
```

Tabla 120. propiedades de coxregnode

| Propiedades de coxregnode | Valores | Descripción de la propiedad |
|---------------------------|---|---|
| survival_time | <i>campo</i> | Los modelos de regresión de Cox requieren un único campo con los tiempos de supervivencia. |
| target | <i>campo</i> | Los modelos de regresión de Cox requieren un único campo objetivo y uno o más campos de entrada. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| method | Intro Stepwise BackwardsStepwise | |
| grupos | <i>campo</i> | |
| model_type | MainEffects Personalizado | |
| custom_terms | ["BP*Sexo" "BP*Edad"] | |
| mode | Expert Simple | |
| max_iterations | <i>número</i> | |
| p_converge | 1.0E-4 1.0E-5 1.0E-6 1.0E-7 1.0E-8 0 | |
| p_converge | 1.0E-4 1.0E-5 1.0E-6 1.0E-7 1.0E-8 0 | |

Tabla 120. propiedades de coxregnode (continuación)

| Propiedades de coxregnode | Valores | Descripción de la propiedad |
|---------------------------|---|--|
| l_converge | 1.0E-1 1.0E-2 1.0E-3 1.0E-4 1.0E-5 0 | |
| removal_criterion | LR Wald Conditional | |
| probability_entry | número | |
| probability_removal | número | |
| output_display | EachStep LastStep | |
| ci_enable | flag | |
| ci_value | 90 95 99 | |
| correlación | flag | |
| display_baseline | flag | |
| survival | flag | |
| hazard | flag | |
| log_minus_log | flag | |
| one_minus_survival | flag | |
| separate_line | campo | |
| value | número o cadena | Si no se especifica ningún valor para un campo, se utilizará la opción predeterminada "Mean" para dicho campo. |

Propiedades de decisionlistnode



El nodo Lista de decisiones identifica subgrupos, o segmentos, que muestran una mayor o menor posibilidad de proporcionar un resultado binario relacionado con la población global. Por ejemplo, puede buscar clientes que tengan menos posibilidades de abandonar o más posibilidades de responder favorablemente a una campaña. Puede incorporar su conocimiento empresarial al modelo añadiendo sus propios segmentos personalizados y previsualizando modelos alternativos uno junto a otro para comparar los resultados. Los modelos de listas de decisiones constan de una lista de reglas en las que cada regla tiene una condición y un resultado. Las reglas se aplican en orden, y la primera regla que coincide determina el resultado.

Ejemplo

```
node = stream.create("decisionlist", "My node")
node.setPropertyValue("search_direction", "Down")
node.setPropertyValue("target_value", 1)
node.setPropertyValue("max_rules", 4)
node.setPropertyValue("min_group_size_pct", 15)
```

Tabla 121. Propiedades de decisionlistnode

| Propiedades de decisionlistnode | Valores | Descripción de la propiedad |
|---------------------------------|------------------------------|---|
| target | <i>campo</i> | Los modelos de listas de decisiones utilizan un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| model_output_type | Modelo InteractiveBuilder | |
| search_direction | Up Bajar | Hace referencia a la localización de segmentos, donde Up es el equivalente a Alta probabilidad y Down es el equivalente a Baja probabilidad. |
| target_value | <i>cadena</i> | Si no se especifica, se supondrá el valor true para las marcas. |
| max_rules | <i>entero</i> | Número máximo de segmentos sin incluir el resto. |
| min_group_size | <i>entero</i> | Tamaño mínimo del segmento. |
| min_group_size_pct | <i>número</i> | Tamaño mínimo del segmento como porcentaje. |
| confidence_level | <i>número</i> | Umbral mínimo que un campo de entrada tiene que mejorar la probabilidad de la respuesta (aumentar la elevación) para que merezca la pena añadirlo a la definición de un segmento. |
| max_segments_per_rule | <i>entero</i> | |
| mode | Simple Valores avanzados | |
| bin_method | EqualWidth EqualCount | |
| bin_count | <i>número</i> | |
| max_models_per_cycle | <i>entero</i> | Amplitud de búsqueda de las listas. |
| max_rules_per_cycle | <i>entero</i> | Amplitud de búsqueda de las reglas de segmentación. |
| segment_growth | <i>número</i> | |
| include_missing | <i>flag</i> | |
| final_results_only | <i>flag</i> | |
| reuse_fields | <i>flag</i> | Permite la reutilización de los atributos (los campos de entrada que aparecen en las reglas). |
| max_alternatives | <i>entero</i> | |
| calculate_raw_propensities | <i>flag</i> | |
| calculate_adjusted_propensities | <i>flag</i> | |
| adjusted_propensity_partition | Test Validation | |

propiedades de discriminantnode



El análisis discriminante realiza más supuestos rigurosos que regresiones logísticas pero puede ser una alternativa o un suplemento valioso al análisis de regresión logística si se cumplen dichos supuestos.

Ejemplo

```
node = stream.create("discriminant", "My node")
node.setPropertyValue("target", "custcat")
node.setPropertyValue("use_partitioned_data", False)
node.setPropertyValue("method", "Stepwise")
```

Tabla 122. propiedades de discriminantnode

| Propiedades de discriminantnode | Valores | Descripción de la propiedad |
|---------------------------------|--------------------------------|---|
| target | campo | Los modelos discriminantes requieren un único campo objetivo y uno o más campos de entrada. Los campos de ponderación y frecuencia no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| method | Intro Por pasos | |
| mode | Simple Valores avanzados | |
| prior_probabilities | AllEqual ComputeFromSizes | |
| covariance_matrix | WithinGroups SeparateGroups | |
| medias | flag | Opciones de estadísticos del cuadro de diálogo Salida avanzada. |
| univariate_anovas | flag | |
| box_m | flag | |
| within_group_covariance | flag | |
| within_groups_correlation | flag | |
| separate_groups_covariance | flag | |
| total_covariance | flag | |
| fishers | flag | |
| unstandardized | flag | |
| casewise_results | flag | Opciones de clasificación del cuadro de diálogo Salida avanzada. |
| limit_to_first | número | El valor predeterminado es 10. |
| summary_table | flag | |
| leave_one_classification | flag | |
| combined_groups | flag | |
| separate_groups_covariance | flag | Opción de matrices Covarianza de grupos separados . |

Tabla 122. propiedades de discriminantnode (continuación)

| Propiedades de discriminantnode | Valores | Descripción de la propiedad |
|---------------------------------|---|--|
| territorial_map | flag | |
| combined_groups | flag | Opción de gráfico Grupos combinados . |
| separate_groups | flag | Opción de gráfico Grupos separados . |
| summary_of_steps | flag | |
| F_pairwise | flag | |
| stepwise_method | WilksLambda UnexplainedVariance MahalanobisDistance SmallestF RaosV | |
| V_to_enter | número | |
| criterios | UseValue UseProbability | |
| F_value_entry | número | El valor predeterminado es 3.84. |
| F_value_removal | número | El valor predeterminado es 2.71. |
| probability_entry | número | El valor predeterminado es 0.05. |
| probability_removal | número | El valor predeterminado es 0,10. |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

Propiedades de extensionmodelnode



Con el nodo de Modelo de extensión, puede ejecutar scripts R o Python para Spark para crear y puntuar resultados.

Ejemplo de Python para Spark

```
#### Ejemplo de script de Python para Spark
import modeler.api
stream = modeler.script.stream()
node = stream.create("extension_build", "extension_build")
node.setPropertyValue("syntax_type", "Python")

build_script = """
import json
import spss.pyspark.runtime
from pyspark.mllib.regression import LabeledPoint
from pyspark.mllib.linalg import DenseVector
from pyspark.mllib.tree import DecisionTree

cxt = spss.pyspark.runtime.getContext()
df = cxt.getSparkInputData()
```

```

schema = df.dtypes[:]

target = "Drug"
predictors = ["Age","BP","Sex","Cholesterol","Na","K"]

def metaMap(row,schema):
    col = 0
    meta = []
    for (cname, ctype) in schema:
        if ctype == 'string':
            meta.append(set([row[col]]))
        else:
            meta.append((row[col],row[col]))
        col += 1
    return meta

def metaReduce(meta1,meta2,schema):
    col = 0
    meta = []
    for (cname, ctype) in schema:
        if ctype == 'string':
            meta.append(meta1[col].union(meta2[col]))
        else:
            meta.append((min(meta1[col][0],meta2[col][0]),max(meta1[col][1],meta2[col][1])))
        col += 1
    return meta

metadata = df.rdd.map(lambda row: metaMap(row,schema)).reduce(lambda x,y:metaReduce(x,y,schema))

def setToList(v):
    if isinstance(v,set):
        return list(v)
    return v

metadata = map(lambda x: setToList(x), metadata)
print metadata

lookup = {}
for i in range(0,len(schema)):
    lookup[schema[i][0]] = i

def row2LabeledPoint(dm,lookup,target,predictors,row):
    target_index = lookup[target]
    tval = dm[target_index].index(row[target_index])
    pvals = []
    for predictor in predictors:
        predictor_index = lookup[predictor]
        if isinstance(dm[predictor_index],list):
            pval = dm[predictor_index].index(row[predictor_index])
        else:
            pval = row[predictor_index]
        pvals.append(pval)
    return LabeledPoint(tval,DenseVector(pvals))

# número de recuento de clases de destino
predictorClassCount = len(metadata[lookup[target]])

# definir función para extraer información de predictor categórico de modelo de datos
def getCategoricalFeatureInfo(dm,lookup,predictors):
    info = {}
    for i in range(0,len(predictors)):
        predictor = predictors[i]
        predictor_index = lookup[predictor]
        if isinstance(dm[predictor_index],list):
            info[i] = len(dm[predictor_index])
    return info

```

```
# convertir marco de datos a un RDD que contiene LabeledPoint
lps = df.rdd.map(lambda row: row2LabeledPoint(metadata,lookup,target,predictors,row))

treeModel = DecisionTree.trainClassifier(
    lps,
    numClasses=predictorClassCount,
    categoricalFeaturesInfo=getCategoricalFeatureInfo(metadata, lookup, predictors),
    impurity='gini',
    maxDepth=5,
    maxBins=100)

_outputPath = cxt.createTemporaryFolder()
treeModel.save(cxt.getSparkContext(), _outputPath)
cxt.setModelContentFromPath("TreeModel", _outputPath)
cxt.setModelContentFromString("model.dm", json.dumps(metadata), mimeType="application/json")\
    .setModelContentFromString("model.structure", treeModel.toDebugString())

"""

node.setPropertyValue("python_build_syntax", build_script)
```

Ejemplo de R

```
#### Ejemplo de script de R
node.setPropertyValue("syntax_type", "R")
node.setPropertyValue("r_build_syntax", """modelerModel <- lm(modelerData$Na~modelerData$K,modelerData)
modelerDataModel
modelerModel
""")
```

Tabla 123. Propiedades de extensión modelnode

| Propiedades de extensión modelnode | Valores | Descripción de la propiedad |
|------------------------------------|------------------------------------|--|
| syntax_type | <i>R</i> <i>Python</i> | Especifica qué script se ejecuta: R o Python (R es el valor predeterminado). |
| r_build_syntax | <i>cadena</i> | Sintaxis de scripts R para la creación de modelos. |
| r_score_syntax | <i>cadena</i> | Sintaxis de scripts R para la puntuación de modelos. |
| python_build_syntax | <i>cadena</i> | Sintaxis de scripts Python para la creación de modelos. |
| python_score_syntax | <i>cadena</i> | Sintaxis de scripts Python para la puntuación de modelos. |
| convert_flags | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| convert_missing | <i>flag</i> | Opción para convertir los valores que faltan al valor R NA. |
| convert_datetime | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| convert_datetime_class | POSIXct POSIX1t | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |
| output_html | <i>flag</i> | Opción para visualizar gráficos en una pestaña en el nugget de modelo R. |
| output_text | <i>flag</i> | Opción para escribir salida de texto de la consola R en una pestaña del modelo R. |

propiedades de factornode



El nodo PCA/Factorial proporciona técnicas eficaces de reducción de datos para reducir la complejidad de los datos. Análisis de componentes principales (PCA) busca combinaciones lineales de los campos de entrada que realizan el mejor trabajo a la hora de capturar la varianza en todo el conjunto de campos, en el que los componentes son ortogonales (perpendiculares) entre ellos. Análisis factorial intenta identificar factores subyacentes que expliquen el patrón de correlaciones dentro de un conjunto de campos observados. Para los dos métodos, el objetivo es encontrar un número pequeño de campos derivados que resuma de forma eficaz la información del conjunto original de campos.

Ejemplo

```
node = stream.create("factor", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("inputs", ["BP", "Na", "K"])
node.setPropertyValue("partition", "Test")
# "Model" tab
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Factor_Age")
node.setPropertyValue("use_partitioned_data", False)
node.setPropertyValue("method", "GLS")
# Opciones de experto
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("complete_records", True)
node.setPropertyValue("matrix", "Covariance")
node.setPropertyValue("max_iterations", 30)
node.setPropertyValue("extract_factors", "ByFactors")
node.setPropertyValue("min_eigenvalue", 3.0)
node.setPropertyValue("max_factor", 7)
node.setPropertyValue("sort_values", True)
node.setPropertyValue("hide_values", True)
node.setPropertyValue("hide_below", 0.7)
# Sección "Rotación"
node.setPropertyValue("rotation", "DirectOblimin")
node.setPropertyValue("delta", 0.3)
node.setPropertyValue("kappa", 7.0)
```

Tabla 124. propiedades de factornode

| Propiedad de factornode | Valores | Descripción de la propiedad |
|-------------------------|---|--|
| inputs | [field1 ... fieldN] | Los modelos PCA/Factorial utilizan una lista de campos de entrada, pero no de campos objetivo. Los campos de ponderación y frecuencia no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| method | PC ULS GLS ML PAF Alpha Image | |
| mode | Simple Valores avanzados | |

Tabla 124. propiedades de factornode (continuación)

| Propiedad de factornode | Valores | Descripción de la propiedad |
|-------------------------|---|--|
| max_iterations | número | |
| complete_records | flag | |
| matrix | Correlation Covariance | |
| extract_factors | ByEigenvalues ByFactors | |
| min_eigenvalue | número | |
| max_factor | número | |
| rotación | Ninguno Varimax DirectOblimin Equamax Quartimax Promax | |
| delta | número | Si selecciona DirectOblimin como el tipo de datos de rotación, podrá especificar un valor para delta. Si no especifica ningún valor, se utilizará el valor predeterminado para delta. |
| kappa | número | Si selecciona Promax como el tipo de datos de rotación, podrá especificar un valor para kappa. Si no especifica ningún valor, se utilizará el valor predeterminado para kappa. |
| sort_values | flag | |
| hide_values | flag | |
| hide_below | número | |

propiedades de featureselectionnode



El nodo Selección de características filtra los campos de entrada para su eliminación en función de un conjunto de criterios (como el porcentaje de valores perdidos); a continuación, clasifica el grado de importancia del resto de entradas de acuerdo con un objetivo específico. Por ejemplo, a partir de un conjunto de datos dado con cientos de entradas potenciales, ¿cuáles tienen mayor probabilidad de ser útiles para el modelado de resultados de pacientes?

Ejemplo

```
node = stream.create("featureselection", "My node")
node.setPropertyValue("screen_single_category", True)
node.setPropertyValue("max_single_category", 95)
node.setPropertyValue("screen_missing_values", True)
node.setPropertyValue("max_missing_values", 80)
node.setPropertyValue("criteria", "Likelihood")
node.setPropertyValue("unimportant_below", 0.8)
```

```

node.setPropertyValue("important_above", 0.9)
node.setPropertyValue("important_label", "Check Me Out!")
node.setPropertyValue("selection_mode", "TopN")
node.setPropertyValue("top_n", 15)

```

Para obtener un ejemplo más detallado que cree y aplique un modelo de Selección de características, consulte en.

Tabla 125. propiedades de featureselectionnode

| Propiedad de featureselectionnode | Valores | Descripción de la propiedad |
|-----------------------------------|---|---|
| target | campo | Los modelos de selección de características ordenan predictores por rangos con respecto a su objetivo específico. Los campos de ponderación y frecuencia no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| screen_single_category | flag | En caso de establecer True, filtra campos que tienen demasiados registros dentro de la misma categoría respecto al número total de registros. |
| max_single_category | número | Especifica el umbral que se utiliza cuando screen_single_category es True. |
| screen_missing_values | flag | Si se establece en True, filtra campos con demasiados valores perdidos, expresados como un porcentaje del número total de registros. |
| max_missing_values | número | |
| screen_num_categories | flag | Si se establece en True, filtra campos con demasiadas categorías respecto al número total de registros. |
| max_num_categories | número | |
| screen_std_dev | flag | Si se establece en True, filtra campos con una desviación estándar menor o igual que el mínimo especificado. |
| min_std_dev | número | |
| screen_coeff_of_var | flag | Si se establece en True, filtra campos con un coeficiente de varianza menor o igual que el mínimo especificado. |
| min_coeff_of_var | número | |
| criterios | Pearson Likelihood CramersV Lambda | Al clasificar los predictores categóricos en función de un objetivo categórico, especifica la medida en la que se basa el valor de importancia. |
| unimportant_below | número | Especifica los valores p de umbral utilizados para clasificar las variables como importantes, marginales o sin importancia. Acepta valores de 0,0 a 1,0. |
| important_above | número | Acepta valores de 0,0 a 1,0. |
| unimportant_label | cadena | Especifica la etiqueta para la clasificación como 'Sin importancia'. |
| marginal_label | cadena | |

Tabla 125. propiedades de `featureselectionnode` (continuación)

| Propiedad de <code>featureselectionnode</code> | Valores | Descripción de la propiedad |
|--|--|--|
| <code>important_label</code> | <i>cadena</i> | |
| <code>selection_mode</code> | ImportanceLevel ImportanceValue TopN | |
| <code>select_important</code> | <i>flag</i> | Si <code>selection_mode</code> se establece en ImportanceLevel, determina si se seleccionan los campos importantes. |
| <code>select_marginal</code> | <i>flag</i> | Si <code>selection_mode</code> se establece en ImportanceLevel, determina si se seleccionan los campos marginales. |
| <code>select_unimportant</code> | <i>flag</i> | Si <code>selection_mode</code> se establece en ImportanceLevel, determina si se seleccionan los campos sin importancia. |
| <code>importance_value</code> | <i>número</i> | Si <code>selection_mode</code> se establece en ImportanceValue, determina el valor de corte que se va a usar. Acepta valores de 0 a 100. |
| <code>top_n</code> | <i>entero</i> | Si <code>selection_mode</code> se establece en TopN, determina el valor de corte que se va a usar. Acepta valores de 0 a 1000. |

propiedades de `genlinnode`



El modelo lineal generalizado amplía el modelo lineal general, de manera que la variable dependiente está relacionada linealmente con los factores y las covariables mediante una determinada función de enlace. Además, el modelo permite que la variable dependiente tenga una distribución no normal. Cubre la funcionalidad de un amplio número de modelos estadísticos, incluyendo regresión lineal, regresión logística, modelos log lineales para recuento de datos y modelos de supervivencia censurados por intervalos.

Ejemplo

```
node = stream.create("genlin", "My node")
node.setPropertyValue("model_type", "MainAndAllTwoWayEffects")
node.setPropertyValue("offset_type", "Variable")
node.setPropertyValue("offset_field", "Claimant")
```

Tabla 126. propiedades de `genlinnode`

| Propiedades de <code>genlinnode</code> | Valores | Descripción de la propiedad |
|--|--------------|--|
| <code>target</code> | <i>campo</i> | Los modelos lineales generalizados requieren un único campo objetivo, que debe ser un campo nominal o marca, y uno o más campos de entrada. También se puede especificar un campo de ponderación. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| <code>use_weight</code> | <i>flag</i> | |
| <code>weight_field</code> | <i>campo</i> | El tipo de campo es únicamente continuo. |
| <code>target_represents_trials</code> | <i>flag</i> | |

Tabla 126. propiedades de genlinnode (continuación)

| Propiedades de genlinnode | Valores | Descripción de la propiedad |
|---------------------------|---|--|
| trials_type | Variable FixedValue | |
| trials_field | campo | El tipo de campo es continuo, marca u ordinal. |
| trials_number | número | El valor predeterminado es 10. |
| model_type | MainEffects MainAndAllTwoWayEffects | |
| offset_type | Variable FixedValue | |
| offset_field | campo | El tipo de campo es únicamente continuo. |
| offset_value | número | Debe ser un número real. |
| base_category | Last Primero | |
| include_intercept | flag | |
| mode | Simple Valores avanzados | |
| distribution | BINOMIAL GAMMA IGAUSS NEGBIN NORMAL POISSON TWEEDIE MULTINOMIAL | IGAUSS: De Gauss inversa. NEGBIN: Negative binomial. |
| negbin_para_type | Especifica Estimate | |
| negbin_parameter | número | El valor predeterminado es 1. Debe contener un número real no negativo. |
| tweedie_parameter | número | |
| link_function | IDENTITY CLOGLOG LOG LOGC LOGIT NEGBIN NLOGLOG ODDSPower PROBIT POWER CUMCAUCHIT CUMCLOGLOG CUMLOGIT CUMNLOGLOG CUMPROBIT | CLOGLOG: log-log complementario. LOGC: complemento log. NEGBIN: Negative binomial. NLOGLOG: Log-log negativo. CUMCAUCHIT: Cauchit acumulada. CUMCLOGLOG: Log-log complementario acumulado. CUMLOGIT: Logit acumulado. CUMNLOGLOG: Log-log negativo acumulado. CUMPROBIT: Probit acumulado. |
| potencia | número | El valor debe ser real y distinto de cero. |
| method | Hybrid Fisher NewtonRaphson | |

Tabla 126. propiedades de *genlinode* (continuación)

| Propiedades de <i>genlinode</i> | Valores | Descripción de la propiedad |
|--|---|---|
| <code>max_fisher_iterations</code> | <i>número</i> | El valor predeterminado es 1; sólo se admiten enteros positivos. |
| <code>scale_method</code> | MaxLikelihoodEstimate Deviance PearsonChiSquare FixedValue | |
| <code>scale_value</code> | <i>número</i> | El valor predeterminado es 1; debe ser mayor que 0. |
| <code>covariance_matrix</code> | ModelEstimator RobustEstimator | |
| <code>max_iterations</code> | <i>número</i> | El valor predeterminado es 100; sólo enteros no negativos. |
| <code>max_step_halving</code> | <i>número</i> | El valor predeterminado es 5; sólo enteros positivos. |
| <code>check_separation</code> | <i>flag</i> | |
| <code>start_iteration</code> | <i>número</i> | El valor predeterminado es 20; sólo se admiten enteros positivos. |
| <code>estimates_change</code> | <i>flag</i> | |
| <code>estimates_change_min</code> | <i>número</i> | El valor predeterminado es 1E-006; sólo se admiten números positivos. |
| <code>estimates_change_type</code> | Absolute Relative | |
| <code>loglikelihood_change</code> | <i>flag</i> | |
| <code>loglikelihood_change_min</code> | <i>número</i> | Sólo se admiten números positivos. |
| <code>loglikelihood_change_type</code> | Absolute Relative | |
| <code>hessian_convergence</code> | <i>flag</i> | |
| <code>hessian_convergence_min</code> | <i>número</i> | Sólo se admiten números positivos. |
| <code>hessian_convergence_type</code> | Absolute Relative | |
| <code>case_summary</code> | <i>flag</i> | |
| <code>contrast_matrices</code> | <i>flag</i> | |
| <code>descriptive_statistics</code> | <i>flag</i> | |
| <code>estimable_functions</code> | <i>flag</i> | |
| <code>model_info</code> | <i>flag</i> | |
| <code>iteration_history</code> | <i>flag</i> | |
| <code>goodness_of_fit</code> | <i>flag</i> | |
| <code>print_interval</code> | <i>número</i> | El valor predeterminado es 1; debe ser un entero positivo. |
| <code>model_summary</code> | <i>flag</i> | |
| <code>lagrange_multiplier</code> | <i>flag</i> | |
| <code>parameter_estimates</code> | <i>flag</i> | |
| <code>include_exponential</code> | <i>flag</i> | |
| <code>covariance_estimates</code> | <i>flag</i> | |

Tabla 126. propiedades de genlinnode (continuación)

| Propiedades de genlinnode | Valores | Descripción de la propiedad |
|---------------------------------|--|------------------------------------|
| correlation_estimates | flag | |
| analysis_type | TypeI TypeIII TypeIAndTypeIII | |
| statistics | Wald LR | |
| citype | Wald Profile | |
| tolerancelevel | número | El valor predeterminado es 0.0001. |
| confidence_interval | número | El valor predeterminado es 95. |
| loglikelihood_function | Completa Kernel | |
| singularity_tolerance | 1E-007 1E-008 1E-009 1E-010 1E-011 1E-012 | |
| value_order | Ascending Descending DataOrder | |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

Propiedades de glmmnode



Un modelo lineal mixto generalizado (GLMM) amplía el modelo lineal de modo que el objetivo pueda tener una distribución no normal, esté linealmente relacionado con los factores y covariables mediante una función de enlace especificada y las observaciones se puedan correlacionar. Los modelos lineales mixtos generalizados cubren una amplia variedad de modelos, desde modelos de regresión lineal simple hasta modelos multinivel complejos para datos longitudinales no normales.

Tabla 127. Propiedades de glmmnode.

| Propiedades de glmmnode | Valores | Descripción de la propiedad |
|-------------------------|---------------------|---|
| residual_subject_spec | estructurado | La combinación de valores de los campos categóricos especificados que definen de forma exclusiva los sujetos del conjunto de datos. |
| repeated_measures | estructurado | Campos utilizados para identificar observaciones repetidas. |
| residual_group_spec | [field1 ... fieldN] | Campos que definen conjuntos independientes de parámetros de covarianza de efectos repetidos. |

Tabla 127. Propiedades de *glmmnode* (continuación).

| Propiedades de <i>glmmnode</i> | Valores | Descripción de la propiedad |
|--------------------------------|--|---|
| residual_covariance_type | Diagonal AR1 ARMA11 COMPOUND_SYMMETRY IDENTITY TOEPLITZ UNSTRUCTURED VARIANCE_COMPONENTS | Especifica la estructura de covarianza de residuos. |
| custom_target | <i>tag</i> | Indica si se puede utilizar un objetivo definido en el nodo anterior (<i>false</i>) o un objetivo personalizado especificado por <i>target_field</i> (<i>true</i>). |
| target_field | <i>campo</i> | Campo a utilizar como objetivo si <i>custom_target</i> es <i>true</i> . |
| use_trials | <i>tag</i> | Indica si hay que utilizar el campo adicional o el valor que especifica el número de ensayos cuando la respuesta objetivo es un número de eventos que tienen lugar en un conjunto de ensayos. El valor predeterminado es <i>false</i> . |
| use_field_or_value | Campo Value | Indica si se utiliza el campo (valor predeterminado) o valor para especificar el número de ensayos. |
| trials_field | <i>campo</i> | Campo a utilizar para especificar el número de ensayos. |
| trials_value | <i>entero</i> | Valor a utilizar para especificar el número de ensayos. Si se especifica, el valor mínimo es 1. |
| use_custom_target_reference | <i>tag</i> | Indica si hay que utilizar la categoría de referencia personalizada para un objetivo categórico. El valor predeterminado es <i>false</i> . |
| target_reference_value | <i>cadena</i> | Categoría de referencia a utilizar si <i>use_custom_target_reference</i> es <i>true</i> . |
| dist_link_combination | Nominal Logit GammaLog BinomialLogit PoissonLog BinomialProbit NegbinLog BinomialLogC Custom | Modelos comunes para la distribución de valores de objetivo. Seleccione Custom para especificar una distribución de la lista proporcionada por <i>target_distribution</i> . |
| target_distribution | Normal Binomial Multinomial Gamma Inverso NegativeBinomial Poisson | La distribución de valores de objetivo cuando <i>dist_link_combination</i> es Custom. |

Tabla 127. Propiedades de *glmnode* (continuación).

| Propiedades de <i>glmnode</i> | Valores | Descripción de la propiedad |
|-------------------------------|--|---|
| link_function_type | Identity LogC Log CLOGLOG Logit NLOGLOG PROBIT POWER CAUCHIT | Función de enlace para relacionar valores de objetivo a los predictores. Si target_distribution es Binomial podrá utilizarse cualquiera de las funciones de enlace listadas. Si target_distribution es Multinomial podrán utilizarse CLOGLOG, CAUCHIT, LOGIT, NLOGLOG o PROBIT. Si target_distribution es cualquier cosa distinta de Binomial o Multinomial podrán utilizarse IDENTITY, LOG o POWER. |
| link_function_param | number | Valor del parámetro de función de enlace que hay que utilizar. Sólo es aplicable si normal_link_function o link_function_type es POWER. |
| use_predefined_inputs | tag | Indica si los campos de efectos fijos deben ser aquellos definidos anteriormente como campos de entrada (true) o han de ser los campos fixed_effects_list (false). El valor predeterminado es false. |
| fixed_effects_list | estructurado | Si use_predefined_inputs es falso, especifica los campos de entrada que se han de utilizar como campos de efectos fijos. |
| use_intercept | tag | Si es true, el valor predeterminado. incluye la interceptación en el modelo. |
| random_effects_list | estructurado | Lista de campos para especificar como efectos aleatorios. |
| regression_weight_field | campo | Campo a utilizar como campo de ponderación de análisis. |
| use_offset | Ninguno offset_value offset_field | Indica cómo se especifica la compensación. El valor None significa que no se ha utilizado compensación. |
| offset_value | number | El valor que se ha de utilizar para desplazamiento si use_offset se establece en offset_value. |
| offset_field | campo | El valor que se ha de utilizar para desplazamiento si use_offset se establece en offset_field. |
| target_category_order | Ascending Descending Datos | Orden de clasificación para objetivos categóricos. El valor Data especifica que se utiliza el orden de clasificación de los datos. El valor predeterminado es Ascending. |
| inputs_category_order | Ascending Descending Datos | Orden de clasificación para predictores categóricos. El valor Data especifica que se utiliza el orden de clasificación de los datos. El valor predeterminado es Ascending. |

Tabla 127. Propiedades de *glmmnode* (continuación).

| Propiedades de <i>glmmnode</i> | Valores | Descripción de la propiedad |
|--|---|---|
| <code>max_iterations</code> | <i>entero</i> | Número máximo de iteraciones que ejecutará el algoritmo. Un número entero no negativo; el valor predeterminado es 100. |
| <code>confidence_level</code> | <i>entero</i> | Nivel de confianza utilizado para calcular estimaciones de intervalo de los coeficientes del modelo. Un número entero no negativo; el valor máximo es 100, el valor predeterminado es 95. |
| <code>degrees_of_freedom_method</code> | Fixed Varied | Especifica cómo se calculan los grados de libertad para la prueba de significación. |
| <code>test_fixed_effects_coefficients</code> | Modelo Robust | Método para calcular la matriz de covarianza de las estimaciones de los parámetros. |
| <code>use_p_converge</code> | <i>flag</i> | Opción para la convergencia de parámetros. |
| <code>p_converge</code> | <i>número</i> | Blanco, o cualquier valor positivo. |
| <code>p_converge_type</code> | Absolute Relative | |
| <code>use_l_converge</code> | <i>flag</i> | Opción para la convergencia log-likelihood. |
| <code>l_converge</code> | <i>número</i> | Blanco, o cualquier valor positivo. |
| <code>l_converge_type</code> | Absolute Relative | |
| <code>use_h_converge</code> | <i>flag</i> | Opción para la convergencia hessiana. |
| <code>h_converge</code> | <i>número</i> | Blanco, o cualquier valor positivo. |
| <code>h_converge_type</code> | Absolute Relative | |
| <code>max_fisher_steps</code> | <i>entero</i> | |
| <code>singularity_tolerance</code> | <i>número</i> | |
| <code>use_model_name</code> | <i>tag</i> | Indica si hay que especificar un nombre personalizado para el modelo (<i>true</i>) o si se ha de utilizar el nombre generado por el sistema (<i>false</i>). El valor predeterminado es <i>false</i> . |
| <code>model_name</code> | <i>cadena</i> | Si <code>use_model_name</code> es <i>true</i> , especifica el nombre de modelo que se va a utilizar. |
| <code>confidence</code> | <i>onProbability</i> <i>onIncrease</i> | Base para calcular el valor de confianza de la puntuación: probabilidad más alta predicha, o la diferencia entre la probabilidad más alta predicha y la segunda probabilidad más alta. |
| <code>score_category_probabilities</code> | <i>tag</i> | Si es <i>true</i> , genera las probabilidades predichas para objetivos categóricos. El valor predeterminado es <i>false</i> . |
| <code>max_categories</code> | <i>entero</i> | Si <code>score_category_probabilities</code> es <i>true</i> , especifica el número máximo de categorías que se han de guardar. |

Tabla 127. Propiedades de glmmnode (continuación).

| Propiedades de glmmnode | Valores | Descripción de la propiedad |
|------------------------------|----------------------------------|---|
| score_propensity | tag | Si es true, produce puntuaciones de propensión para campos de objetivo de marca que indican la probabilidad del resultado "true" para el campo. |
| emeans | estructura | Para cada campo categórico de la lista de efectos fijos, especifica si hay que producir medias marginales estimadas. |
| covariance_list | estructura | Para cada campo continuo de la lista de efectos fijos, especifica si hay que usar la media o un valor personalizado al calcular medias marginales estimadas. |
| mean_scale | Original Transformed | Especifica si las medias marginales estimadas se calculan basándose en la escala original del objetivo (valor predeterminado) o en la transformación de la función de enlace. |
| comparison_adjustment_method | DMS SEQBONFERRONI SEQSIDAK | Método de ajuste que hay que utilizar al realizar pruebas de hipótesis con varios contrastes. |

Propiedades de gle



Un modelo lineal mixto generalizado (GLMM) amplía el modelo lineal de modo que el objetivo pueda tener una distribución no normal, esté linealmente relacionado con los factores y covariables mediante una función de enlace especificada y las observaciones se puedan correlacionar. Los modelos lineales mixtos generalizados cubren una amplia variedad de modelos, desde modelos de regresión lineal simple hasta modelos multinivel complejos para datos longitudinales no normales.

Tabla 128. Propiedades de gle

| Propiedades de gle | Valores | Descripción de la propiedad |
|---------------------------|----------------|---|
| custom_target | flag | Indica si se puede utilizar un objetivo definido en el nodo anterior (false) o un objetivo personalizado especificado por target_field (true). |
| target_field | campo | Campo a utilizar como objetivo si custom_target es true. |
| use_trials | flag | Indica si hay que utilizar el campo adicional o el valor que especifica el número de ensayos cuando la respuesta objetivo es un número de eventos que tienen lugar en un conjunto de ensayos. El valor predeterminado es false. |
| use_trials_field_or_value | Campo Valor | Indica si se utiliza el campo (valor predeterminado) o valor para especificar el número de ensayos. |
| trials_field | campo | Campo a utilizar para especificar el número de ensayos. |
| trials_value | entero | Valor a utilizar para especificar el número de ensayos. Si se especifica, el valor mínimo es 1. |

Tabla 128. Propiedades de gle (continuación)

| Propiedades de gle | Valores | Descripción de la propiedad |
|-----------------------------|--|--|
| use_custom_target_reference | <i>flag</i> | Indica si hay que utilizar la categoría de referencia personalizada para un objetivo categórico. El valor predeterminado es false. |
| target_reference_value | <i>cadena</i> | Categoría de referencia a utilizar si use_custom_target_reference es true. |
| dist_link_combination | NormalIdentity GammaLog PoissonLog NegbinLog TweedieIdentity NominalLogit BinomialLogit BinomialProbit BinomialLogC CUSTOM | Modelos comunes para la distribución de valores de objetivo. Elija CUSTOM para especificar una distribución en la lista proporcionada por target_distribution. |
| target_distribution | Normal Binomial Multinomial Gamma INVERSE_GAUSS NEG_BINOMIAL Poisson TWEEDIE UNKNOWN | La distribución de valores de objetivo cuando dist_link_combination es Custom. |
| link_function_type | UNKNOWN IDENTITY LOG LOGIT PROBIT COMPL_LOG_LOG POWER LOG_COMPL NEG_LOG_LOG ODDS_POWER NEG_BINOMIAL GEN_LOGIT CUMUL_LOGIT CUMUL_PROBIT CUMUL_COMPL_LOG_LOG CUMUL_NEG_LOG_LOG CUMUL_CAUCHIT | Función de enlace para relacionar valores de objetivo a los predictores. Si target_distribution es Binomial puede utilizar: UNKNOWN IDENTITY LOG LOGIT PROBIT COMPL_LOG_LOG POWER LOG_COMPL NEG_LOG_LOG ODDS_POWER Si target_distribution es NEG_BINOMIAL puede utilizar: NEG_BINOMIAL. Si target_distribution es UNKNOWN, puede utilizar: GEN_LOGIT CUMUL_LOGIT CUMUL_PROBIT CUMUL_COMPL_LOG_LOG CUMUL_NEG_LOG_LOG CUMUL_CAUCHIT |
| link_function_param | <i>número</i> | Valor del parámetro Tweedie que hay que utilizar. Sólo es aplicable si normal_link_function o link_function_type es POWER. |

Tabla 128. Propiedades de gle (continuación)

| Propiedades de gle | Valores | Descripción de la propiedad |
|---------------------------------|--|---|
| tweedie_param | número | Valor del parámetro de función de enlace que hay que utilizar. Solo es aplicable si dist_link_combination está establecido en TweedieIdentity, o link_function_type es TWEEDIE. |
| use_predefined_inputs | flag | Indica si los campos de efectos del modelo deben ser aquellos definidos anteriormente como campos de entrada (true) o han de ser los campos fixed_effects_list (false). |
| model_effects_list | structured | Si use_predefined_inputs es falso, especifica los campos de entrada que se han de utilizar como campos de efectos del modelo. |
| use_intercept | flag | Si es true, el valor predeterminado. incluye la interceptación en el modelo. |
| regression_weight_field | campo | Campo a utilizar como campo de ponderación de análisis. |
| use_offset | Ninguno Value Variable | Indica cómo se especifica la compensación. El valor None significa que no se ha utilizado compensación. |
| offset_value | número | El valor que se ha de utilizar para desplazamiento si use_offset se establece en offset_value. |
| offset_field | campo | El valor que se ha de utilizar para desplazamiento si use_offset se establece en offset_field. |
| target_category_order | Ascending Descending | Orden de clasificación para objetivos categóricos. El valor predeterminado es Ascending. |
| inputs_category_order | Ascending Descending | Orden de clasificación para predictores categóricos. El valor predeterminado es Ascending. |
| max_iterations | entero | Número máximo de iteraciones que ejecutará el algoritmo. Un número entero no negativo; el valor predeterminado es 100. |
| confidence_level | número | Nivel de confianza utilizado para calcular estimaciones de intervalo de los coeficientes del modelo. Un número entero no negativo; el valor máximo es 100, el valor predeterminado es 95. |
| test_fixed_effects_coefficients | Modelo robusta | Método para calcular la matriz de covarianza de las estimaciones de los parámetros. |
| detect_outliers | flag | Si es verdadero, el algoritmo encuentra valores atípicos de influencia para todas las distribuciones excepto para la distribución multinomial. |
| conduct_trend_analysis | flag | Si es verdadero, el algoritmo realiza un análisis de tendencias para el gráfico de dispersión. |
| estimation_method | FISHER_SCORING NEWTON_RAPHSON HYBRID | Especifica el algoritmo de estimación de máxima verosimilitud. |
| max_fisher_iterations | entero | Si se utiliza FISHER_SCORING estimation_method, el número máximo de iteraciones. Mínimo 0, máximo 20. |

Tabla 128. Propiedades de gle (continuación)

| Propiedades de gle | Valores | Descripción de la propiedad |
|-----------------------------|---|---|
| scale_parameter_method | MLE FIXED DEVIANCE PEARSON_CHISQUARE | Especifique el método que se va a utilizar para la estimación del parámetro de escala. |
| scale_value | número | Solo está disponible si scale_parameter_method está establecido en Fixed. |
| negative_binomial_method | MLE FIXED | Especifique el método que se va a utilizar para la estimación del parámetro auxiliar binomial negativo. |
| negative_binomial_value | número | Solo está disponible si negative_binomial_method está establecido en Fixed. |
| use_p_converge | flag | Opción para la convergencia de parámetros. |
| p_converge | número | Blanco, o cualquier valor positivo. |
| p_converge_type | flag | Verdadero = Absoluto, Falso = Relativo |
| use_l_converge | flag | Opción para la convergencia log-likelihood. |
| l_converge | número | Blanco, o cualquier valor positivo. |
| l_converge_type | flag | Verdadero = Absoluto, Falso = Relativo |
| use_h_converge | flag | Opción para la convergencia hessiana. |
| h_converge | número | Blanco, o cualquier valor positivo. |
| h_converge_type | flag | Verdadero = Absoluto, Falso = Relativo |
| max_iterations | entero | Número máximo de iteraciones que ejecutará el algoritmo. Un número entero no negativo; el valor predeterminado es 100. |
| sing_tolerance | entero | |
| use_model_selection | flag | Habilita los controles de umbral de parámetro y de método de selección de modelos. |
| method | LASSO ELASTIC_NET FORWARD_STEPWISE RIDGE | Determina el método de selección de modelos, o si está utilizando Ridge, el método de regularización utilizado. |
| detect_two_way_interactions | flag | Si es True, el modelo detectará automáticamente interacciones bidireccionales entre campos de entrada. Este control solo debería estar habilitado si el modelo es solo efectos principales (es decir, donde el usuario no ha creado ningún efecto de orden superior) y si el method seleccionado es Paso adelante, Lazo, o Red elástica. |

Tabla 128. Propiedades de gle (continuación)

| Propiedades de gle | Valores | Descripción de la propiedad |
|----------------------------|---------------|--|
| automatic_penalty_params | <i>flag</i> | Solo está disponible si la selección del modelo method es Lazo o red elástica. Utilice esta función para especificar los parámetros de penalización asociados a los métodos de selección de variables Lasso o Red elástica. Si es True, se utilizan los valores predeterminados. Si es False, se pueden especificar los parámetros de penalización y los valores personalizados habilitados. |
| lasso_penalty_param | <i>número</i> | Solo está disponible si la selección del modelo method es Lazo o Red elástica y automatic_penalty_params es False. Especifique el valor del parámetro de penalización para Lasso. |
| elastic_net_penalty_param1 | <i>número</i> | Solo está disponible si la selección del modelo method es Lazo o Red elástica y automatic_penalty_params es False. Especifique el valor del parámetro de penalización para el parámetro 1 de Red elástica. |
| elastic_net_penalty_param2 | <i>número</i> | Solo está disponible si la selección del modelo method es Lazo o Red elástica y automatic_penalty_params es False. Especifique el valor del parámetro de penalización para el parámetro 2 de Red elástica. |
| probability_entry | <i>número</i> | Solo está disponible si el method seleccionado es Paso adelante. Especifique el nivel de significación del criterio estadístico F para la inclusión de efectos. |
| probability_removal | <i>número</i> | Solo está disponible si el method seleccionado es Paso adelante. Especifique el nivel de significación del criterio estadístico F para la eliminación de efectos. |
| use_max_effects | <i>flag</i> | Solo está disponible si el method seleccionado es Paso adelante. Habilita el control max_effects. Si es False, el número de efectos predeterminado incluido debería ser igual al número total de efectos proporcionados al modelo, menos la interceptación. |
| max_effects | <i>entero</i> | Especifique el número máximo de efectos cuando se utiliza el método de creación de paso adelante. |
| use_max_steps | <i>flag</i> | Habilita el control max_steps. Si es False, el número predeterminado de pasos debe ser igual a tres veces el número de efectos proporcionado al modelo, excluyendo la interceptación. |

Tabla 128. Propiedades de gle (continuación)

| Propiedades de gle | Valores | Descripción de la propiedad |
|--------------------|---------|--|
| max_steps | entero | Especifique el número máximo de pasos que se van a realizar cuando se utiliza el method de creación de paso adelante. |
| use_model_name | flag | Indica si hay que especificar un nombre personalizado para el modelo (true) o si se ha de utilizar el nombre generado por el sistema false). El valor predeterminado es false. |
| model_name | cadena | Si use_model_name es true, especifica el nombre de modelo que se va a utilizar. |
| usePI | flag | Si true, se calcula la importancia de predictor. |

propiedades de kmeansnode



El nodo K-medias agrupa conjuntos de datos en grupos distintos (o clústeres). El método define un número fijo de clústeres, de forma iterativa asigna registros a los clústeres y ajusta los centros de los clústeres hasta que no se pueda mejorar el modelo. En lugar de intentar predecir un resultado, los modelos de *k*-medias utilizan un proceso conocido como aprendizaje no supervisado para revelar los patrones del conjunto de campos de entrada.

Ejemplo

```
node = stream.create("kmeans", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("inputs", ["Cholesterol", "BP", "Drug", "Na", "K", "Age"])
# "Model" tab
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Kmeans_allinputs")
node.setPropertyValue("num_clusters", 9)
node.setPropertyValue("gen_distance", True)
node.setPropertyValue("cluster_label", "Number")
node.setPropertyValue("label_prefix", "Kmeans_")
node.setPropertyValue("optimize", "Speed")
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("stop_on", "Custom")
node.setPropertyValue("max_iterations", 10)
node.setPropertyValue("tolerance", 3.0)
node.setPropertyValue("encoding_value", 0.3)
```

Tabla 129. propiedades de kmeansnode

| Propiedad de kmeansnode | Valores | Descripción de la propiedad |
|-------------------------|---------------------|---|
| inputs | [field1 ... fieldN] | Los modelos de K-Medias realizan un análisis de clústeres en un conjunto de campos de entrada pero no utilizan ningún campo objetivo. Los campos de ponderación y frecuencia no se usan. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| num_clusters | número | |
| gen_distance | flag | |

Tabla 129. propiedades de kmeansnode (continuación)

| Propiedad de kmeansnode | Valores | Descripción de la propiedad |
|-------------------------|---------------------------------|---|
| cluster_label | Cadena Number | |
| label_prefix | <i>cadena</i> | |
| mode | Simple Valores avanzados | |
| stop_on | Predeterminado Personalizado | |
| max_iterations | <i>número</i> | |
| tolerance | <i>número</i> | |
| encoding_value | <i>número</i> | |
| optimize | Speed Memory | Se utiliza para especificar si la generación del modelo se debe optimizar para la velocidad o la memoria. |

propiedades de knnnode



El nodo k de modelado de vecino (KNN) asocia el nuevo caso con la categoría o valor de los objetos k junto a él en el espacio de predictores, donde k es un entero. Los casos parecidos están próximos y los que no lo son están alejados entre sí.

Ejemplo

```
node = stream.create("knn", "My node")
# Objectives tab
node.setPropertyValue("objective", "Custom")
# Settings tab - Neighbors panel
node.setPropertyValue("automatic_k_selection", False)
node.setPropertyValue("fixed_k", 2)
node.setPropertyValue("weight_by_importance", True)
# Settings tab - Analyze panel
node.setPropertyValue("save_distances", True)
```

Tabla 130. propiedades de knnnode

| Propiedades de knnnode | Valores | Descripción de la propiedad |
|------------------------|--|---|
| de errores | PredictTarget IdentifyNeighbors | |
| objective | Balance Speed Exactitud Personalizado | |
| normalize_ranges | <i>flag</i> | |
| use_case_labels | <i>flag</i> | Seleccione esta casilla de verificación para activar la siguiente opción. |
| case_labels_field | <i>campo</i> | |
| identify_focal_cases | <i>flag</i> | Seleccione esta casilla de verificación para activar la siguiente opción. |
| focal_cases_field | <i>campo</i> | |

Tabla 130. propiedades de knnnode (continuación)

| Propiedades de knnnode | Valores | Descripción de la propiedad |
|---------------------------------|------------------------|--|
| automatic_k_selection | flag | |
| fixed_k | entero | Se activa únicamente si el valor de automatic_k_selection es False. |
| minimum_k | entero | Se activa únicamente si el valor de automatic_k_selection es True. |
| maximum_k | entero | |
| distance_computation | Euclidean CityBlock | |
| weight_by_importance | flag | |
| range_predictions | Media Mediana | |
| perform_feature_selection | flag | |
| forced_entry_inputs | [field1 ... fieldN] | |
| stop_on_error_ratio | flag | |
| number_to_select | entero | |
| minimum_change | número | |
| validation_fold_assign_by_field | flag | |
| number_of_folds | entero | Sólo se activa si el valor de validation_fold_assign_by_field es False |
| set_random_seed | flag | |
| random_seed | número | |
| folds_field | campo | Sólo se activa si el valor de validation_fold_assign_by_field es True |
| all_probabilities | flag | |
| save_distances | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

propiedades de kohonennode



El nodo Kohonen genera un tipo de red neuronal que se puede usar para agrupar un conjunto de datos en grupos distintos. Cuando la red se termina de entrenar, los registros que son similares se deberían cerrar juntos en el mapa de resultados, mientras que los registros que son diferentes aparecerían aparte. Puede observar el número de observaciones capturadas por cada unidad en el nugget de modelo para identificar unidades fuertes. Esto le proporcionará una idea del número apropiado de clústeres.

Ejemplo

```
node = stream.create("kohonen", "My node")
# "Model" tab
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "Symbolic Cluster")
node.setPropertyValue("stop_on", "Time")
```

```

node.setPropertyValue("time", 1)
node.setPropertyValue("set_random_seed", True)
node.setPropertyValue("random_seed", 12345)
node.setPropertyValue("optimize", "Speed")
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("width", 3)
node.setPropertyValue("length", 3)
node.setPropertyValue("decay_style", "Exponential")
node.setPropertyValue("phase1_neighborhood", 3)
node.setPropertyValue("phase1_eta", 0.5)
node.setPropertyValue("phase1_cycles", 10)
node.setPropertyValue("phase2_neighborhood", 1)
node.setPropertyValue("phase2_eta", 0.2)
node.setPropertyValue("phase2_cycles", 75)

```

Tabla 131. propiedades de kohonenode

| Propiedad de kohonenode | Valores | Descripción de la propiedad |
|-------------------------|-----------------------------|--|
| inputs | [field1 ... fieldN] | Los modelos Kohonen utilizan una lista de campos de entrada, pero no de campos objetivo. Los campos de frecuencia y ponderación no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue | flag | |
| show_feedback | flag | |
| stop_on | Predeterminado Hora | |
| time | número | |
| optimize | Speed Memory | Se utiliza para especificar si la generación del modelo se debe optimizar para la velocidad o la memoria. |
| cluster_label | flag | |
| mode | Simple Valores avanzados | |
| width | número | |
| longitud | número | |
| decay_style | Lineal Exponential | |
| phase1_neighborhood | número | |
| phase1_eta | número | |
| phase1_cycles | número | |
| phase2_neighborhood | número | |
| phase2_eta | número | |
| phase2_cycles | número | |

Propiedades de linearnode



Los modelos de regresión lineal predicen un objetivo continuo tomando como base las relaciones lineales entre el destino y uno o más predictores.

Ejemplo

```
node = stream.create("linear", "My node")
# Build Options tab - Objectives panel
node.setPropertyValue("objective", "Standard")
# Build Options tab - Model Selection panel
node.setPropertyValue("model_selection", "BestSubsets")
node.setPropertyValue("criteria_best_subsets", "ASE")
# Build Options tab - Ensembles panel
node.setPropertyValue("combining_rule_categorical", "HighestMeanProbability")
```

Tabla 132. Propiedades de linearnode.

| Propiedades de linearnode | Valores | Descripción de la propiedad |
|----------------------------------|--|--|
| objetivo | <i>campo</i> | Especifica un campo de objetivo único. |
| inputs | <i>[field1 ... fieldN]</i> | Campos de predictor utilizados por el modelo. |
| continue_training_existing_model | <i>tag</i> | |
| objective | Standard Agregación autodocimante Aumento psm | psm se utiliza para conjuntos de datos de grandes dimensiones y requiere una conexión al Servidor. |
| use_auto_data_preparation | <i>tag</i> | |
| confidence_level | <i>number</i> | |
| model_selection | ForwardStepwise BestSubsets Ninguno | |
| criteria_forward_stepwise | AICC Estadísticas F R cuadrado corregido ASE | |
| probability_entry | <i>number</i> | |
| probability_removal | <i>number</i> | |
| use_max_effects | <i>tag</i> | |
| max_effects | <i>number</i> | |
| use_max_steps | <i>tag</i> | |
| max_steps | <i>number</i> | |
| criteria_best_subsets | AICC R cuadrado corregido ASE | |
| combining_rule_continuous | Media Mediana | |

Tabla 132. Propiedades de linearnode (continuación).

| Propiedades de linearnode | Valores | Descripción de la propiedad |
|---------------------------|---------|-----------------------------|
| component_models_n | number | |
| use_random_seed | tag | |
| random_seed | number | |
| use_custom_model_name | tag | |
| custom_model_name | cadena | |
| use_custom_name | tag | |
| custom_name | cadena | |
| tooltip | cadena | |
| palabras clave | cadena | |
| annotation | cadena | |

Propiedades de linearasnode



Los modelos de regresión lineal predicen un objetivo continuo tomando como base las relaciones lineales entre el destino y uno o más predictores.

Tabla 133. Propiedades de linearasnode

| Propiedades de linearasnode | Valores | Descripción de la propiedad |
|-----------------------------|--|--|
| objetivo | campo | Especifica un campo de objetivo único. |
| inputs | [field1 ... fieldN] | Campos de predictor utilizados por el modelo. |
| weight_field | campo | Campo de análisis usado por el modelo. |
| custom_fields | flag | El valor predeterminado es TRUE. |
| intercept | flag | El valor predeterminado es TRUE. |
| detect_2way_interaction | flag | Indica si debe tenerse en cuenta la interacción bidireccional. El valor predeterminado es TRUE. |
| cin | número | Intervalo de confianza usado para calcular las estimaciones de los coeficientes del modelo. Especifique un valor mayor que 0 y menor que 100. El valor predeterminado es 95. |
| factor_order | ascending descending | Orden de clasificación para predictores categóricos. El valor predeterminado es ascending. |
| var_select_method | ForwardStepwise BestSubsets none | El método de selección de modelo que se va a usar. El valor predeterminado es ForwardStepwise. |

Tabla 133. Propiedades de linearasnode (continuación)

| Propiedades de linearasnode | Valores | Descripción de la propiedad |
|-------------------------------|---|--|
| criteria_for_forward_stepwise | AICC Estadísticas F R cuadrado corregido ASE | La estadística utilizada para determinar si un efecto debe añadirse o eliminarse del modelo. El valor predeterminado es AdjustedRSquare. |
| pin | número | El efecto con el valor p más pequeño con respecto a este umbral de pin especificado se añade al modelo. El valor predeterminado es 0.05. |
| pout | número | Cualquier efecto del modelo con un valor p superior a este umbral de pout especificado se eliminará. El valor predeterminado es 0,10. |
| use_custom_max_effects | flag | Indica si debe utilizarse el número máximo de efectos en el modelo final. El valor predeterminado es FALSE. |
| max_effects | número | Número máximo de efectos por usar en el modelo final. El valor predeterminado es 1. |
| use_custom_max_steps | flag | Indica si debe utilizarse el número máximo de pasos. El valor predeterminado es FALSE. |
| max_steps | número | El número máximo de pasos antes de que se detenga el algoritmo escalonado. El valor predeterminado es 1. |
| criteria_for_best_subsets | AICC R cuadrado corregido ASE | La modalidad de criterios a utilizar. El valor predeterminado es AdjustedRSquare. |

Propiedades de logregnode



La regresión logística es una técnica de estadístico para clasificar los registros en función los valores de los campos de entrada. Es análoga a la regresión lineal pero toma un campo objetivo categórico en lugar de uno numérico.

Ejemplo multinomial

```
node = stream.create("logreg", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Drug")
node.setPropertyValue("inputs", ["BP", "Cholesterol", "Age"])
node.setPropertyValue("partition", "Test")
# "Model" tab
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Log_reg Drug")
node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("method", "Stepwise")
node.setPropertyValue("logistic_procedure", "Multinomial")
node.setPropertyValue("multinomial_base_category", "BP")
node.setPropertyValue("model_type", "FullFactorial")
node.setPropertyValue("custom_terms", [["BP", "Sex"], ["Age"], ["Na", "K"]])
node.setPropertyValue("include_constant", False)
```

```

# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("scale", "Pearson")
node.setPropertyValue("scale_value", 3.0)
node.setPropertyValue("all_probabilities", True)
node.setPropertyValue("tolerance", "1.0E-7")
# "Convergence..." section
node.setPropertyValue("max_iterations", 50)
node.setPropertyValue("max_steps", 3)
node.setPropertyValue("l_converge", "1.0E-3")
node.setPropertyValue("p_converge", "1.0E-7")
node.setPropertyValue("delta", 0.03)
# "Output..." section
node.setPropertyValue("summary", True)
node.setPropertyValue("likelihood_ratio", True)
node.setPropertyValue("asymptotic_correlation", True)
node.setPropertyValue("goodness_fit", True)
node.setPropertyValue("iteration_history", True)
node.setPropertyValue("history_steps", 3)
node.setPropertyValue("parameters", True)
node.setPropertyValue("confidence_interval", 90)
node.setPropertyValue("asymptotic_covariance", True)
node.setPropertyValue("classification_table", True)
# "Stepping" options
node.setPropertyValue("min_terms", 7)
node.setPropertyValue("use_max_terms", True)
node.setPropertyValue("max_terms", 10)
node.setPropertyValue("probability_entry", 3)
node.setPropertyValue("probability_removal", 5)
node.setPropertyValue("requirements", "Containment")

```

Ejemplo binomial

```

node = stream.create("logreg", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Cholesterol")
node.setPropertyValue("inputs", ["BP", "Drug", "Age"])
node.setPropertyValue("partition", "Test")
# "Model" tab
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "Log_reg Cholesterol")
node.setPropertyValue("multinomial_base_category", "BP")
node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("binomial_method", "Forwards")
node.setPropertyValue("logistic_procedure", "Binomial")
node.setPropertyValue("binomial_categorical_input", "Sex")
node.setKeyedPropertyValue("binomial_input_contrast", "Sex", "Simple")
node.setKeyedPropertyValue("binomial_input_category", "Sex", "Last")
node.setPropertyValue("include_constant", False)
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("scale", "Pearson")
node.setPropertyValue("scale_value", 3.0)
node.setPropertyValue("all_probabilities", True)
node.setPropertyValue("tolerance", "1.0E-7")
# "Convergence..." section
node.setPropertyValue("max_iterations", 50)
node.setPropertyValue("l_converge", "1.0E-3")
node.setPropertyValue("p_converge", "1.0E-7")
# "Output..." section

```

```

node.setPropertyValue("binomial_output_display", "at_each_step")
node.setPropertyValue("binomial_goodness_of_fit", True)
node.setPropertyValue("binomial_iteration_history", True)
node.setPropertyValue("binomial_parameters", True)
node.setPropertyValue("binomial_ci_enable", True)
node.setPropertyValue("binomial_ci", 85)
# "Stepping" options
node.setPropertyValue("binomial_removal_criterion", "LR")
node.setPropertyValue("binomial_probability_removal", 0.2)

```

Tabla 134. Propiedades de logregnode.

| Propiedades de logregnode | Valores | Descripción de la propiedad |
|----------------------------|--|--|
| objetivo | <i>campo</i> | Los modelos de regresión logística requieren un único campo objetivo y uno o más campos de entrada. Los campos de frecuencia y ponderación no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| logistic_procedure | Binomial Multinomial | |
| include_constant | <i>tag</i> | |
| mode | Simple Expert | |
| método | Intro Stepwise Forwards Backwards BackwardsStepwise | |
| binomial_method | Intro Forwards Backwards | |
| model_type | MainEffects FullFactorial Custom | Si FullFactorial se especifica como el tipo de modelo, no se ejecutarán los métodos por pasos, aunque así se indique. En su lugar, el método utilizado será Enter. Si el tipo de modelo se establece en Custom pero no se ha especificado ningún campo personalizado, se generará un modelo de efectos principales. |
| custom_terms | <i>[[BP Sexo][BP][Edad]]</i> | |
| multinomial_base_category | <i>cadena</i> | Especifica cómo se determina la categoría de referencia. |
| binomial_categorical_input | <i>cadena</i> | |
| binomial_input_contrast | Indicator Simple Diferencia Helmert Repeated Polinómico Desviación | Propiedad con clave para la entrada categórica que especifica cómo se determina el contraste. |

Tabla 134. Propiedades de logregnode (continuación).

| Propiedades de logregnode | Valores | Descripción de la propiedad |
|------------------------------|---|---|
| binomial_input_category | Primero Last | Propiedad con clave para la entrada categórica que especifica cómo se determina la categoría de referencia. |
| scale | Ninguno UserDefined Pearson Deviance | |
| scale_value | <i>number</i> | |
| all_probabilities | <i>tag</i> | |
| tolerance | 1.0E-5 1.0E-6 1.0E-7 1.0E-8 1.0E-9 1.0E-10 | |
| min_terms | <i>number</i> | |
| use_max_terms | <i>tag</i> | |
| max_terms | <i>number</i> | |
| entry_criterion | Puntuación LR | |
| removal_criterion | LR Wald | |
| probability_entry | <i>number</i> | |
| probability_removal | <i>number</i> | |
| binomial_probability_entry | <i>number</i> | |
| binomial_probability_removal | <i>number</i> | |
| requirements | HierarchyDiscrete HierarchyAll Containment Ninguno | |
| max_iterations | <i>number</i> | |
| max_steps | <i>number</i> | |
| p_converge | 1.0E-4 1.0E-5 1.0E-6 1.0E-7 1.0E-8 0 | |
| l_converge | 1.0E-1 1.0E-2 1.0E-3 1.0E-4 1.0E-5 0 | |
| delta | <i>number</i> | |
| iteration_history | <i>tag</i> | |
| history_steps | <i>number</i> | |

Tabla 134. Propiedades de logregnode (continuación).

| Propiedades de logregnode | Valores | Descripción de la propiedad |
|--------------------------------|------------------------------|-----------------------------|
| summary | tag | |
| likelihood_ratio | tag | |
| asymptotic_correlation | tag | |
| goodness_fit | tag | |
| parameters | tag | |
| confidence_interval | number | |
| asymptotic_covariance | tag | |
| classification_table | tag | |
| stepwise_summary | tag | |
| info_criteria | tag | |
| monotonicity_measures | tag | |
| binomial_output_display | at_each_step at_last_step | |
| binomial_goodness_of_fit | tag | |
| binomial_parameters | tag | |
| binomial_iteration_history | tag | |
| binomial_classification_plots | tag | |
| binomial_ci_enable | tag | |
| binomial_ci | number | |
| binomial_residual | valores atípicos all | |
| binomial_residual_enable | tag | |
| binomial_outlier_threshold | number | |
| binomial_classification_cutoff | number | |
| binomial_removal_criterion | LR Wald Conditional | |
| calculate_variable_importance | tag | |
| calculate_raw_propensities | tag | |

propiedades de lsvmnode



El nodo Máquina de vectores de soporte lineal (LSVM) le permite clasificar datos en uno o dos grupos sin que haya un ajuste por exceso. LSVM es lineal y funciona bien con conjuntos de datos grandes, como aquellos con un gran número de registros.

Tabla 135. propiedades de lsvmnode

| Propiedades lsvmnode | Valores | Descripción de la propiedad |
|----------------------|---------|---|
| intercept | flag | Incluye la interceptación del modelo. El valor predeterminado es True. |

Tabla 135. propiedades de lsvmnode (continuación)

| Propiedades lsvmnode | Valores | Descripción de la propiedad |
|-------------------------------|-------------------------|---|
| target_order | Ascending Descending | Especifica el orden de clasificación del objetivo categórico. Si ignora para objetivos continuos. El valor predeterminado es Ascending. |
| precision | número | Sólo se utiliza si el nivel de medición del campo objetivo es Continuo. Especifica el parámetro relacionado con la sensibilidad de la pérdida de regresión. El mínimo es 0 y no hay máximo. El valor predeterminado es 0.1. |
| exclude_missing_values | flag | Si es True, se excluye un registro si falta algún valor único. El valor predeterminado es False. |
| penalty_function | L1 L2 | Especifica el tipo de función de penalización utilizado. El valor predeterminado es L2. |
| lambda | número | Parámetro de penalización (regularización). |
| calculate_variable_importance | flag | En modelos que producen una medida adecuada de importancia, esta opción muestra un gráfico que indica la importancia relativa de cada predictor al estimar el modelo. En modelos que producen una medida adecuada de importancia, esta opción muestra un gráfico que indica la importancia relativa de cada predictor al estimar el modelo. Tenga en cuenta que la importancia variable puede ser tardar más en calcularse en algunos modelos, especialmente al trabajar con grandes conjuntos de datos y está desactivada de forma predeterminada para algunos modelos como resultado. La importancia variable no está disponible en los modelos de lista de decisiones. |

propiedades de neuralnetnode

Importante: En este release está disponible una versión más reciente del nodo de modelado Red neuronal, con características mejoradas, y se describe en la siguiente sección (*red neuronal*). Aunque aún puede generar y puntuar un modelo con la versión anterior, recomendamos que actualice sus scripts para que se use la nueva versión. Los detalles de la versión anterior se conservan aquí como referencia.

Ejemplo

```
node = stream.create("neuralnet", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("targets", ["Drug"])
node.setPropertyValue("inputs", ["Age", "Na", "K", "Cholesterol", "BP"])
# "Model" tab
```

```

node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("method", "Dynamic")
node.setPropertyValue("train_pct", 30)
node.setPropertyValue("set_random_seed", True)
node.setPropertyValue("random_seed", 12345)
node.setPropertyValue("stop_on", "Time")
node.setPropertyValue("accuracy", 95)
node.setPropertyValue("cycles", 200)
node.setPropertyValue("time", 3)
node.setPropertyValue("optimize", "Speed")
# sección "Opciones de experto para método múltiple"
node.setPropertyValue("m_topologies", "5 30 5; 2 20 3, 1 10 1")
node.setPropertyValue("m_non_pyramids", False)
node.setPropertyValue("m_persistence", 100)

```

Tabla 136. propiedades de neuralnetnode

| Propiedad de neuralnetnode | Valores | Descripción de la propiedad |
|----------------------------|--|--|
| targets | [field1 ... fieldN] | El nodo Red neuronal espera uno o varios campos objetivo y uno o más campos de entrada. Los campos de frecuencia y ponderación se omiten. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| method | Quick Dynamic Multiple Prune ExhaustivePrune RBFN | |
| prevent_overtrain | flag | |
| train_pct | número | |
| set_random_seed | flag | |
| random_seed | número | |
| mode | Simple Valores avanzados | |
| stop_on | Predeterminado Exactitud Cycles Hora | Modo de parada. |
| exactitud | número | Precisión de parada. |
| cycles | número | Ciclos para entrenar. |
| time | número | Tiempo para entrenar (minutos). |
| continue | flag | |
| show_feedback | flag | |
| binary_encode | flag | |
| use_last_model | flag | |
| gen_logfile | flag | |
| logfile_name | cadena | |
| alpha | número | |
| initial_eta | número | |

Tabla 136. propiedades de neuralnetnode (continuación)

| Propiedad de neuralnetnode | Valores | Descripción de la propiedad |
|---------------------------------|---------------------|--|
| high_eta | número | |
| low_eta | número | |
| eta_decay_cycles | número | |
| hid_layers | One Two Three | |
| hl_units_one | número | |
| hl_units_two | número | |
| hl_units_three | número | |
| persistence | número | |
| m_topologies | cadena | |
| m_non_pyramids | flag | |
| m_persistence | número | |
| p_hid_layers | One Two Three | |
| p_hl_units_one | número | |
| p_hl_units_two | número | |
| p_hl_units_three | número | |
| p_persistence | número | |
| p_hid_rate | número | |
| p_hid_pers | número | |
| p_inp_rate | número | |
| p_inp_pers | número | |
| p_overall_pers | número | |
| r_persistence | número | |
| r_num_clusters | número | |
| r_eta_auto | flag | |
| r_alpha | número | |
| r_eta | número | |
| optimize | Speed Memory | Se utiliza para especificar si la generación del modelo se debe optimizar para la velocidad o la memoria. |
| calculate_variable_importance | flag | Nota: La propiedad sensitivity_analysis utilizada en versiones anteriores se ha desaprobadado en favor de esta propiedad. La propiedad anterior se sigue admitiendo, pero se recomienda calculate_variable_importance. |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

Propiedades de neuralnetwork



El nodo Red neuronal utiliza un modelo simplificado que emula el modo en que el cerebro humano procesa la información: Funciona simultaneando un número elevado de unidades simples de procesamiento interconectadas que parecen versiones abstractas de neuronas. Las redes neuronales son dispositivos eficaces de cálculo de funciones generales y requieren un conocimiento matemático o estadístico mínimo para entrenarlas o aplicarlas.

Ejemplo

```
node = stream.create("neuralnetwork", "My node")
# Build Options tab - Objectives panel
node.setPropertyValue("objective", "Standard")
# Build Options tab - Ensembles panel
node.setPropertyValue("combining_rule_categorical", "HighestMeanProbability")
```

Tabla 137. Propiedades de neuralnetwork

| Propiedades de neuralnetworknode | Valores | Descripción de la propiedad |
|----------------------------------|--|---|
| targets | [field1 ... fieldN] | Especifica campos objetivo. |
| inputs | [field1 ... fieldN] | Campos de predictor utilizados por el modelo. |
| splits | [campo1 ... fieldN] | Especifica el campo o campos para utilizar en el modelado de divisiones. |
| use_partition | flag | Si se ha definido un campo de partición, esta opción garantiza que sólo se utilizarán los datos de la partición de entrenamiento para la generación del modelo. |
| continue | flag | Continuar entrenando modelo existente. |
| objective | Standard Agregación autodocimante Aumento psm | psm se utiliza para conjuntos de datos de grandes dimensiones y requiere una conexión al Servidor. |
| method | MultilayerPerceptron RadialBasisFunction | |
| use_custom_layers | flag | |
| first_layer_units | número | |
| second_layer_units | número | |
| use_max_time | flag | |
| max_time | número | |
| use_max_cycles | flag | |
| max_cycles | número | |
| use_min_accuracy | flag | |
| min_accuracy | número | |
| combining_rule_categorical | Voting HighestProbability HighestMeanProbability | |
| combining_rule_continuous | Media Mediana | |
| component_models_n | número | |

Tabla 137. Propiedades de neuralnetwork (continuación)

| Propiedades de neuralnetworknode | Valores | Descripción de la propiedad |
|----------------------------------|--|-----------------------------|
| overfit_prevention_pct | número | |
| use_random_seed | flag | |
| random_seed | número | |
| missing_values | listwiseDeletion missingValueImputation | |
| use_model_name | booleano | |
| model_name | cadena | |
| confidence | onProbability onIncrease | |
| score_category_probabilities | flag | |
| max_categories | número | |
| score_propensity | flag | |
| use_custom_name | flag | |
| custom_name | cadena | |
| tooltip | cadena | |
| palabras clave | cadena | |
| annotation | cadena | |

propiedades de questnode



El nodo QUEST proporciona un método de clasificación binario para generar árboles de decisión; está diseñado para reducir el tiempo de procesamiento necesario para realizar los análisis de C&RT y reducir la tendencia de los métodos de clasificación de árboles para favorecer a las entradas que permitan realizar más divisiones. Los campos de entrada pueden ser continuos (rango numérico), sin embargo el campo objetivo debe ser categórico. Todas las divisiones son binarias.

Ejemplo

```
node = stream.create("quest", "My node")
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Drug")
node.setPropertyValue("inputs", ["Age", "Na", "K", "Cholesterol", "BP"])
node.setPropertyValue("model_output_type", "InteractiveBuilder")
node.setPropertyValue("use_tree_directives", True)
node.setPropertyValue("max_surrogates", 5)
node.setPropertyValue("split_alpha", 0.03)
node.setPropertyValue("use_percentage", False)
node.setPropertyValue("min_parent_records_abs", 40)
node.setPropertyValue("min_child_records_abs", 30)
node.setPropertyValue("prune_tree", True)
node.setPropertyValue("use_std_err", True)
node.setPropertyValue("std_err_multiplier", 3)
```

Tabla 138. propiedades de questnode

| Propiedad de questnode | Valores | Descripción de la propiedad |
|----------------------------------|--|--|
| target | <i>campo</i> | Los modelos QUEST requieren un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue_training_existing_model | <i>flag</i> | |
| objective | Standard Aumento Agregación autodocimante psm | psm se utiliza para conjuntos de datos de grandes dimensiones y requiere una conexión al Servidor. |
| model_output_type | Single InteractiveBuilder | |
| use_tree_directives | <i>flag</i> | |
| tree_directives | <i>cadena</i> | |
| use_max_depth | Predeterminado Personalizado | |
| max_depth | <i>entero</i> | Máxima profundidad del árbol, desde 0 a 1000. Sólo se utiliza si use_max_depth = Custom. |
| prune_tree | <i>flag</i> | Poda del árbol para evitar sobreajustes. |
| use_std_err | <i>flag</i> | Use la diferencia máxima en riesgos (en errores estándar). |
| std_err_multiplier | <i>número</i> | Diferencia máxima. |
| max_surrogates | <i>número</i> | Número máximo de sustitutos. |
| use_percentage | <i>flag</i> | |
| min_parent_records_pc | <i>número</i> | |
| min_child_records_pc | <i>número</i> | |
| min_parent_records_abs | <i>número</i> | |
| min_child_records_abs | <i>número</i> | |
| use_costs | <i>flag</i> | |
| costes | <i>structured</i> | Propiedad estructurada. |
| priors | Datos Equal Personalizado | |
| custom_priors | <i>structured</i> | Propiedad estructurada. |
| adjust_priors | <i>flag</i> | |
| trails | <i>número</i> | Número de modelos de componente para un aumento o agregación autodocimante. |
| set_ensemble_method | Voting HighestProbability HighestMeanProbability | Regla de combinación predeterminada para objetivos categóricos. |
| range_ensemble_method | Media Mediana | Regla de combinación predeterminada para objetivos continuos. |

Tabla 138. propiedades de questnode (continuación)

| Propiedad de questnode | Valores | Descripción de la propiedad |
|---------------------------------|--------------------|--|
| large_boost | flag | Aplicar aumento a conjunto de datos muy grandes. |
| split_alpha | número | Nivel de significancia para división. |
| train_pct | número | Conjunto de prevención sobreajustado. |
| set_random_seed | flag | Opción replicar resultados. |
| seed | número | |
| calculate_variable_importance | flag | |
| calculate_raw_propensities | flag | |
| calculate_adjusted_propensities | flag | |
| adjusted_propensity_partition | Test Validation | |

propiedades randomtrees



El nodo Árboles aleatorios es similar al nodo C&RT existente; sin embargo, el nodo Árboles aleatorios se ha diseñado para procesar grandes cantidades de datos (Big Data) para crear un solo árbol y mostrar el modelo resultante en el visor de la salida que se ha añadido en SPSS Modeler versión 17. El nodo Árboles aleatorios genera un árbol de decisión que se utiliza para predecir o clasificar observaciones futuras. El método utiliza la partición reiterada para dividir los registros de entrenamiento en segmentos minimizando las impurezas en cada paso, donde un nodo se considera *puro* si el 100% de los casos del nodo corresponden a una categoría específica del campo objetivo. Los campos de entrada y objetivo pueden ser continuos (rango numérico) o categóricos (nominal, ordinal o marca). Todas las divisiones son binarias (sólo se crean dos subgrupos).

Tabla 139. propiedades randomtrees

| Propiedades randomtrees | Valores | Descripción de la propiedad |
|----------------------------|---------|---|
| target | campo | En el nodo Árboles aleatorios, los modelos requieren un único objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| number_of_models | entero | Determina el número de modelos a construir como parte del modelado de conjunto. |
| use_number_of_predictors | flag | Determina si se utiliza number_of_predictors. |
| number_of_predictors | entero | Especifica el número de predictores que se van a utilizar al crear modelos de división. |
| use_stop_rule_for_accuracy | flag | Determina si la creación del modelo se detiene cuando no se puede mejorar la precisión. |
| sample_size | número | Reduzca este valor para mejorar el rendimiento del procesar conjuntos de datos muy grandes. |

Tabla 139. propiedades randomtrees (continuación)

| Propiedades randomtrees | Valores | Descripción de la propiedad |
|-------------------------|--|--|
| handle_imbalanced_data | flag | Si el objetivo del modelo es un resultado de marca determinado, y la proporción del resultado deseado con respecto a un resultado no deseado es muy pequeña, los datos se desequilibran y el muestreo de simulación realizado por el modelo puede afectar a la exactitud del mismo. Habilite el manejo de datos desequilibrados para que el modelo capture una proporción mayor del resultado deseado y genere un modelo más sólido. |
| use_weighted_sampling | flag | Si se establece en <i>False</i> , las variables para cada nodo se seleccionan aleatoriamente con la misma probabilidad. Si es <i>True</i> , las variables se ponderan y se seleccionan de acuerdo con ello. |
| max_node_number | entero | Número máximo de nodos permitidos en árboles individuales. Si el número va a superarse en la división siguiente, el crecimiento del árbol se detiene. |
| max_depth | entero | Máxima profundidad del árbol antes de que se detenga el crecimiento. |
| min_child_node_size | entero | Determina el número mínimo de registros permitidos en un nodo hijo después de que se divida el nodo padre. Si un nodo hijo va a contener menos registros de los especificados aquí, el nodo padre no se dividirá. |
| use_costs | flag | |
| costes | structured | Propiedad estructurada. El formato es una lista de 3 valores: el valor real, el valor predicho y el coste si dicha predicción es errónea. Por ejemplo: tree.setPropertyValue("costs", [{"drugA", "drugB", 3.0}, {"drugX", "drugY", 4.0}]) |
| default_cost_increase | ninguno lineal square personalizada | Nota: sólo se habilita para objetivos ordinales. Establezca valores predeterminados en la matriz de costes. |
| max_pct_missing | entero | Si el porcentaje de valores faltantes en cualquier entrada es mayor que el valor especificado aquí, la entrada se excluye. Mínimo 0, máximo 100. |
| exclude_single_cat_pct | entero | Si un valor de categoría representa un porcentaje mayor de los registros que el especificado aquí, todo el campo se excluye de la creación de modelos. Mínimo 1, máximo 99. |
| max_category_number | entero | Si el número de categorías de un campo supera este valor, el campo se excluye de la creación de modelos. El mínimo es 2. |

Tabla 139. propiedades randomtrees (continuación)

| Propiedades randomtrees | Valores | Descripción de la propiedad |
|-------------------------|---------|--|
| min_field_variation | número | Si el coeficiente de variación de un campo continuo es inferior a este valor, el campo se excluye de la creación de modelos. |
| num_bins | entero | Sólo se utiliza si los datos están formados por entradas continuas. Establezca el número de intervalos de frecuencia que deben utilizarse para las entradas; las opciones son 2, 4, 5, 10, 20, 25, 50 o 100. |
| topN | entero | Especifica el número de reglas para informar. El valor predeterminado es 50, con un mínimo de 1 y un máximo de 1000. |

Propiedades de regressionnode



La regresión lineal es una técnica de estadístico común utilizada para resumir datos y realizar predicciones ajustando una superficie o línea recta que minimice las discrepancias existentes entre los valores de salida reales y los predichos.

Nota: El nodo Lineal reemplazará al nodo Regresión en una versión futura. Recomendamos que a partir de ahora utilice modelos lineales para la regresión lineal.

Ejemplo

```
node = stream.create("regression", "My node")
# "Fields" tab
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("target", "Age")
node.setPropertyValue("inputs", ["Na", "K"])
node.setPropertyValue("partition", "Test")
node.setPropertyValue("use_weight", True)
node.setPropertyValue("weight_field", "Drug")
# "Model" tab
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Regression Age")
node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("method", "Stepwise")
node.setPropertyValue("include_constant", False)
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("complete_records", False)
node.setPropertyValue("tolerance", "1.0E-3")
# "Stepping..." section
node.setPropertyValue("stepping_method", "Probability")
node.setPropertyValue("probability_entry", 0.77)
node.setPropertyValue("probability_removal", 0.88)
node.setPropertyValue("F_value_entry", 7.0)
node.setPropertyValue("F_value_removal", 8.0)
# "Output..." section
node.setPropertyValue("model_fit", True)
node.setPropertyValue("r_squared_change", True)
node.setPropertyValue("selection_criteria", True)
node.setPropertyValue("descriptives", True)
```

```

node.setPropertyValue("p_correlations", True)
node.setPropertyValue("collinearity_diagnostics", True)
node.setPropertyValue("confidence_interval", True)
node.setPropertyValue("covariance_matrix", True)
node.setPropertyValue("durbin_watson", True)

```

Tabla 140. Propiedades de regressionnode

| Propiedad de regressionnode | Valores | Descripción de la propiedad |
|-----------------------------|---|--|
| target | campo | Los modelos de regresión requieren un único campo objetivo y uno o más campos de entrada. También se puede especificar un campo de ponderación. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| method | Intro Stepwise Backwards Adelante | |
| include_constant | flag | |
| use_weight | flag | |
| weight_field | campo | |
| mode | Simple Valores avanzados | |
| complete_records | flag | |
| tolerance | 1.0E-1 1.0E-2 1.0E-3 1.0E-4 1.0E-5 1.0E-6 1.0E-7 1.0E-8 1.0E-9 1.0E-10 1.0E-11 1.0E-12 | Utilice comillas dobles para los argumentos. |
| stepping_method | useP useF | useP : utilizar la probabilidad de F useF: utilizar el valor F |
| probability_entry | número | |
| probability_removal | número | |
| F_value_entry | número | |
| F_value_removal | número | |
| selection_criteria | flag | |
| confidence_interval | flag | |
| covariance_matrix | flag | |
| collinearity_diagnostics | flag | |
| regression_coefficients | flag | |
| exclude_fields | flag | |
| durbin_watson | flag | |
| model_fit | flag | |

Tabla 140. Propiedades de regressionnode (continuación)

| Propiedad de regressionnode | Valores | Descripción de la propiedad |
|-------------------------------|---------|-----------------------------|
| r_squared_change | flag | |
| p_correlations | flag | |
| descriptives | flag | |
| calculate_variable_importance | flag | |

propiedades de sequencenode



El nodo Secuencia encuentra reglas de asociación en datos secuenciales o en datos ordenados en el tiempo. Una secuencia es una lista de conjuntos de elementos que tiende a producirse en un orden previsible. Por ejemplo, si un cliente compra una cuchilla y una loción para después del afeitado, probablemente comprará crema para afeitarse la próxima vez que vaya a comprar. El nodo Secuencia se basa en el algoritmo de reglas de asociación de CARMA, que utiliza un método de dos pasos para encontrar las secuencias.

Ejemplo

```
node = stream.create("sequence", "My node")
# "Fields" tab
node.setPropertyValue("id_field", "Age")
node.setPropertyValue("contiguous", True)
node.setPropertyValue("use_time_field", True)
node.setPropertyValue("time_field", "Date1")
node.setPropertyValue("content_fields", ["Drug", "BP"])
node.setPropertyValue("partition", "Test")
# "Model" tab
node.setPropertyValue("use_model_name", True)
node.setPropertyValue("model_name", "Sequence_test")
node.setPropertyValue("use_partitioned_data", False)
node.setPropertyValue("min_supp", 15.0)
node.setPropertyValue("min_conf", 14.0)
node.setPropertyValue("max_size", 7)
node.setPropertyValue("max_predictions", 5)
# "Expert" tab
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("use_max_duration", True)
node.setPropertyValue("max_duration", 3.0)
node.setPropertyValue("use_pruning", True)
node.setPropertyValue("pruning_value", 4.0)
node.setPropertyValue("set_mem_sequences", True)
node.setPropertyValue("mem_sequences", 5.0)
node.setPropertyValue("use_gaps", True)
node.setPropertyValue("min_item_gap", 20.0)
node.setPropertyValue("max_item_gap", 30.0)
```

Tabla 141. propiedades de sequencenode

| Propiedad de sequencenode | Valores | Descripción de la propiedad |
|---------------------------|-----------------------------|--|
| id_field | campo | Para crear un modelo de secuencias, es necesario especificar un campo de ID, un campo de tiempo opcional y uno o varios campos de contenido. Los campos de ponderación y frecuencia no se usan. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| time_field | campo | |
| use_time_field | flag | |
| content_fields | [field1 ... fieldn] | |
| contiguous | flag | |
| min_supp | número | |
| min_conf | número | |
| max_size | número | |
| max_predictions | número | |
| mode | Simple Valores avanzados | |
| use_max_duration | flag | |
| max_duration | número | |
| use_gaps | flag | |
| min_item_gap | número | |
| max_item_gap | número | |
| use_pruning | flag | |
| pruning_value | número | |
| set_mem_sequences | flag | |
| mem_sequences | entero | |

propiedades de slrmnode



El nodo Modelo de respuesta de autoaprendizaje (SLRM) permite crear un modelo en el que un solo caso nuevo o un pequeño número de casos nuevos se pueden utilizar para volver a calcular el modelo sin tener que entrenar de nuevo el modelo utilizando todos los datos.

Ejemplo

```
node = stream.create("slrm", "My node")
node.setPropertyValue("target", "Offer")
node.setPropertyValue("target_response", "Response")
node.setPropertyValue("inputs", ["Cust_ID", "Age", "Ave_Bal"])
```

Tabla 142. propiedades de slrmnode

| Propiedades de slrmnode | Valores | Descripción de la propiedad |
|----------------------------------|-------------------------|--|
| target | campo | El campo objetivo debe ser un campo nominal o marca. También se puede especificar un campo de frecuencia. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| target_response | campo | El tipo debe ser marca. |
| continue_training_existing_model | flag | |
| target_field_values | flag | Utilizar todos: Usar todos los valores del origen. Especifique: Son necesarios determinados valores. |
| target_field_values_specify | [campo1 ... campoN] | |
| include_model_assessment | flag | |
| model_assessment_random_seed | número | Debe ser un número real. |
| model_assessment_sample_size | número | Debe ser un número real. |
| model_assessment_iterations | número | Número de iteraciones. |
| display_model_evaluation | flag | |
| max_predictions | número | |
| randomization | número | |
| scoring_random_seed | número | |
| sort | Ascending Descending | Especifica si se mostrarán primero las ofertas con las puntuaciones más altas o más bajas. |
| model_reliability | flag | |
| calculate_variable_importance | flag | |

propiedades de statisticsmodelnode



El nodo Modelo Statistics permite analizar y trabajar con sus datos ejecutando los procedimientos de IBM SPSS Statistics que producen PMML. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Las propiedades de este nodo están descritas en "propiedades de statisticsmodelnode" en la página 342.

propiedades de stpnode



El nodo Predicción espacio-temporal (STP) utiliza datos que contienen datos de ubicación, campos de entrada para la predicción (predictores), un campo de hora y un campo de objetivo. Cada ubicación tiene muchas filas en los datos que representan los valores de cada predictor en cada tiempo de medición. Después de analizar los datos, se puede utilizar para predecir los valores de objetivo en cualquier ubicación dentro de los datos de forma que se utilizan en el análisis.

Tabla 143. propiedades de stpnode

| Propiedades de stpnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|---|--|
| Pestaña Campos | | |
| target | campo | Este es el campo de destino. |
| location | campo | Campo de ubicación del modelo. Sólo se permiten campos geoespaciales. |
| location_label | campo | Campo categórico a utilizar en la salida para etiquetar las ubicaciones elegidas en location |
| time_field | campo | Campo de hora del modelo. Sólo se permiten campos con medición continua, y el tipo de almacenamiento debe ser hora, fecha, indicación de fecha y hora o entero. |
| inputs | [campo1 ... campoN] | Lista de campos de entrada. |
| Pestaña Intervalos de tiempo | | |
| interval_type_timestamp | Años Trimestres Meses Weeks Days Hours Minutes Seconds | |
| interval_type_date | Años Trimestres Meses Weeks Days | |
| interval_type_time | Hours Minutes Seconds | Limita el número de días por semana que se tienen en cuenta al crear el índice de hora que utiliza STP para el cálculo |
| interval_type_integer | Períodos (Sólo campos de índice de hora, almacenamiento Entero) | Intervalo en el que se convertirá el conjunto de datos. La selección disponible depende del tipo de almacenamiento del campo elegido como time_field para el modelo. |
| period_start | entero | |
| start_month | Enero Febrero Marzo Abril Mayo Junio Julio Agosto Septiembre Octubre Noviembre Diciembre | El mes desde el que el modelo empezará a indexar (por ejemplo, si se establece en March pero el primer registro del conjunto de datos es January, el modelo omitirá los primeros dos registros y comenzará a indexar en marzo. |

Tabla 143. propiedades de stpnode (continuación)

| Propiedades de stpnode | Tipo de datos | Descripción de la propiedad |
|------------------------|--|---|
| week_begins_on | Sunday Monday Tuesday Wednesday Jueves Friday Saturday | Punto de partida para el índice temporal creado por STP a partir de los datos |
| days_per_week | entero | Mínimo 1, máximo 7, con incrementos de 1 |
| hours_per_day | entero | El número de horas que el modelo cuenta en un día. Si se establece en 10, el modelo empezará a indexar en la hora day_begins_at y continuará indexando durante 10 horas, y luego saltará al siguiente valor que coincida con el valor day_begins_at, etc. |
| day_begins_at | 00:00 01:00 02:00 03:00 ... 23:00 | Establece el valor de hora desde el que el modelo inicia la indexación. |
| interval_increment | 1 2 3 4 5 6 10 12 15 20 30 | Este valor de incremento es de minutos o segundos. Determina dónde el modelo crea índices a partir de los datos. Así que con un incremento de 30 y el intervalo de tipo seconds, el modelo crea un índice a partir de los datos cada 30 segundos. |
| data_matches_interval | Booleana | <p>Si se establece en N, la conversión de los datos al interval_type normal se produce antes de que se construya el modelo.</p> <p>Si los datos ya tiene el formato correcto, e interval_type y sus valores asociados coinciden con sus datos, establézcalo en Y para evitar la conversión o la agregación de los datos.</p> <p>Si lo establece en Y, se inhabilitan todos los controles de agregación.</p> |
| agg_range_default | Sum Media Min Max Mediana 1stQuartile 3rdQuartile | Determina el método de agregación predeterminado que se utiliza para los campos continuos. Los campos continuos que no estén incluidos específicamente en la agregación predeterminada se agregarán usando el método aquí indicado. |

Tabla 143. propiedades de stpnode (continuación)

| Propiedades de stpnode | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---|--|
| custom_agg | [[campo, método de agregación],[..] Demo: [['x5' 'FirstQuartile'] ['x4' 'Sum']] | Propiedad estructurada: Parámetro de script: custom_agg Por ejemplo: set :stpnode.custom_agg = [[campo1 función] [campo2 función]] Donde función es la función de agregación a utilizar con dicho campo. |
| Pestaña Procedimientos básicos | | |
| include_intercept | flag | |
| max_autoregressive_lag | entero | Mínimo 1, máximo 5, en incrementos de 1. Es el número de registros previos necesarios para una predicción. Por lo tanto, si se establece en 5, por ejemplo, entonces los 5 registros se utilizan para crear un nuevo pronóstico. El número de registros especificado aquí a partir de los datos de construcción se incorporan en el modelo y, por lo tanto, el usuario no necesita proporcionar los datos de nuevo al puntuar el modelo. |
| estimation_method | Parametric Nonparametric | Método para modelar la matriz de covarianzas espacial |
| parametric_model | Gaussian Exponential PoweredExponential | Parámetro de orden para el modelo de covarianza espacial de tipo Parametric |
| exponential_power | número | Nivel alimentación para el modelo PoweredExponential. Mínimo 1, máximo 2. |
| Pestaña Avanzada | | |
| max_missing_values | entero | Porcentaje máximo de registros con valores faltantes que se permite en el modelo. |
| significación | número | Nivel de significación para pruebas de hipótesis en la construcción del modelo. Especifica el valor de significación para todas las pruebas en la estimación del modelo STP, incluidas dos pruebas de Bondad de ajuste, pruebas F de efectos y pruebas T de coeficiente. |
| Pestaña Salida | | |
| model_specifications | flag | |
| temporal_summary | flag | |

Tabla 143. propiedades de stpnode (continuación)

| Propiedades de stpnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|---------------|--|
| location_summary | flag | Determina si la tabla Resumen de ubicación se incluye en la salida del modelo. |
| model_quality | flag | |
| test_mean_structure | flag | |
| mean_structure_coefficients | flag | |
| autoregressive_coefficients | flag | |
| test_decay_space | flag | |
| parametric_spatial_covariance | flag | |
| correlations_heat_map | flag | |
| correlations_map | flag | |
| location_clusters | flag | |
| similarity_threshold | número | Umbral en el cual los clústeres de salida se consideran lo suficientemente parecidos para que se fusionen en un único clúster. |
| max_number_clusters | entero | Límite superior para el número de clústeres que se pueden incluir en la salida del modelo. |
| Pestaña Opciones de modelo | | |
| use_model_name | flag | |
| model_name | cadena | |
| uncertainty_factor | número | Mínimo 0, máximo 100. Determina el aumento de la incertidumbre (error) aplicado a las predicciones en el futuro. Es el límite superior e inferior para las predicciones. |

propiedades de svmnode



El nodo Máquina de vectores de soporte (SVM) le permite clasificar datos en uno o dos grupos sin que haya un ajuste por exceso. SVM funciona bien con conjuntos de datos grandes, como aquellos con un gran número de campos de entrada.

Ejemplo

```
node = stream.create("svm", "My node")
# pestaña Experto
node.setPropertyValue("mode", "Expert")
node.setPropertyValue("all_probabilities", True)
node.setPropertyValue("kernel", "Polynomial")
node.setPropertyValue("gamma", 1.5)
```

Tabla 144. propiedades de svmnode.

| Propiedades de svmnode | Valores | Descripción de la propiedad |
|------------------------|---------|-----------------------------|
| all_probabilities | tag | |

Tabla 144. propiedades de svmnode (continuación).

| Propiedades de svmnode | Valores | Descripción de la propiedad |
|---------------------------------|---|---|
| stopping_criteria | 1.0E-1 1.0E-2 1.0E-3 (valor predeterminado) 1.0E-4 1.0E-5 1.0E-6 | Determina cuándo detener el algoritmo de optimización. |
| regularization | number | También se conoce como el parámetro C. |
| precision | number | Sólo se utiliza si el nivel de medición del campo objetivo es Continuo. |
| kernel | RBF (valor predeterminado) Polinómico Sigmoide Lineal | Tipo de función kernel utilizada para la transformación. |
| rbf_gamma | number | Sólo se utiliza si kernel es RBF. |
| gamma | number | Sólo se utiliza si kernel es Polinómico o Sigmoide. |
| bias | number | |
| grado | number | Sólo se utiliza si kernel es Polinómico. |
| calculate_variable_importance | tag | |
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |
| adjusted_propensity_partition | Test Validation | |

Propiedades de tcmnode



El modelado causal temporal intenta descubrir relaciones causales clave en datos de series temporales. En el modelado causal temporal, especifique un conjunto de series de objetivos y un conjunto de entradas candidato para estos objetivos. El procedimiento crea un modelo de serie temporal autorregresivo para cada objetivo e incluye solo estas entradas que tienen la relación causal más significativa con el objetivo.

Tabla 145. propiedades de tcmnode

| Propiedades de tcmnode | Valores | Descripción de la propiedad |
|------------------------|-----------------------------|-----------------------------|
| custom_fields | Booleana | |
| dimensionlist | [dimension1 ... dimensionN] | |
| data_struct | Múltiple Única | |
| metric_fields | campos | |
| both_target_and_input | [f1 ... fN] | |
| targets | [f1 ... fN] | |
| candidate_inputs | [f1 ... fN] | |
| forced_inputs | [f1 ... fN] | |

Tabla 145. propiedades de tcmnode (continuación)

| Propiedades de tcmnode | Valores | Descripción de la propiedad |
|--------------------------|---|-----------------------------|
| use_timestamp | Marca de tiempo Period | |
| input_interval | Ninguno Desconocido Año Trimestre Mes Semana Día Hora Hour_nonperiod Minuto Minute_nonperiod Segundo Second_nonperiod | |
| period_field | <i>cadena</i> | |
| period_start_value | <i>entero</i> | |
| num_days_per_week | <i>entero</i> | |
| start_day_of_week | Sunday Monday Martes Miércoles Jueves Viernes Saturday | |
| num_hours_per_day | <i>entero</i> | |
| start_hour_of_day | <i>entero</i> | |
| timestamp_increments | <i>entero</i> | |
| cyclic_increments | <i>entero</i> | |
| cyclic_periods | <i>lista</i> | |
| output_interval | Ninguno Año Trimestre Mes Semana Día Hora Minuto Segundo | |
| is_same_interval | El mismo Notsame | |
| cross_hour | <i>Booleana</i> | |
| aggregate_and_distribute | <i>lista</i> | |
| aggregate_default | Media Sum Mode Min Max | |
| distribute_default | Media Sum | |

Tabla 145. propiedades de tcmnode (continuación)

| Propiedades de tcmnode | Valores | Descripción de la propiedad |
|------------------------------|---|-----------------------------|
| group_default | Media Sum Mode Min Max | |
| missing_imput | Linear_interp Series_mean K_mean K_meridian Linear_trend Ninguno | |
| k_mean_param | entero | |
| k_median_param | entero | |
| missing_value_threshold | entero | |
| conf_level | entero | |
| max_num_predictor | entero | |
| max_lag | entero | |
| epsilon | número | |
| threshold | entero | |
| is_re_est | Booleana | |
| num_targets | entero | |
| percent_targets | entero | |
| fields_display | lista | |
| series_display | lista | |
| network_graph_for_target | Booleana | |
| sign_level_for_target | número | |
| fit_and_outlier_for_target | Booleana | |
| sum_and_para_for_target | Booleana | |
| impact_diag_for_target | Booleana | |
| impact_diag_type_for_target | Efecto Cause Both | |
| impact_diag_level_for_target | entero | |
| series_plot_for_target | Booleana | |
| res_plot_for_target | Booleana | |
| top_input_for_target | Booleana | |
| forecast_table_for_target | Booleana | |
| same_as_for_target | Booleana | |
| network_graph_for_series | Booleana | |
| sign_level_for_series | número | |
| fit_and_outlier_for_series | Booleana | |
| sum_and_para_for_series | Booleana | |
| impact_diagram_for_series | Booleana | |

Tabla 145. propiedades de tcnode (continuación)

| Propiedades de tcnode | Valores | Descripción de la propiedad |
|---------------------------------|---------------------------------------|-----------------------------|
| impact_diagram_type_for_series | Efecto Cause Both | |
| impact_diagram_level_for_series | entero | |
| series_plot_for_series | Booleana | |
| residual_plot_for_series | Booleana | |
| forecast_table_for_series | Booleana | |
| outlier_root_cause_analysis | Booleana | |
| causal_levels | entero | |
| outlier_table | Interactive Lista dinámica Both | |
| rmsp_error | Booleana | |
| bic | Booleana | |
| r_square | Booleana | |
| outliers_over_time | Booleana | |
| series_transormation | Booleana | |
| use_estimation_period | Booleana | |
| estimation_period | Times Observación | |
| observations | lista | |
| observations_type | Latest Más antiguo | |
| observations_num | entero | |
| observations_exclude | entero | |
| extend_records_into_future | Booleana | |
| forecastperiods | entero | |
| max_num_distinct_values | entero | |
| display_targets | FIXEDNUMBER PERCENTAGE | |
| goodness_fit_measure | ROOTMEAN BIC RSQUARE | |
| top_input_for_series | Booleana | |
| aic | Booleana | |
| rmse | Booleana | |

Propiedades ts



El nodo Serie temporal estima modelos de suavizado exponencial, modelos autorregresivos integrados de media móvil (ARIMA) univariados y modelos ARIMA (o de función de transferencia) multivariados para series temporales y genera previsiones. Este nodo Serie temporal es similar al nodo Serie temporal anterior que estaba en desuso en SPSS Modeler versión 18. Sin embargo, este nodo Serie temporal más reciente se ha diseñado para emplear la potencia de IBM SPSS Analytic Server para procesar grandes cantidades de datos y mostrar el modelo resultante en el visor de resultados que se ha añadido en SPSS Modeler versión 17.

Tabla 146. Propiedades ts

| Propiedades ts | Valores | Descripción de la propiedad |
|--------------------|---|--|
| targets | <i>campo</i> | El nodo Serie temporal prevé uno o más objetivos, utilizando opcionalmente uno o más campos de entrada como predictores. Los campos de frecuencia y ponderación no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| candidate_inputs | [<i>field1 ... fieldN</i>] | Campos de entrada o predictor utilizados por el modelo. |
| use_period | <i>flag</i> | |
| date_time_field | <i>campo</i> | |
| input_interval | Ninguno Desconocido Año Trimestre Mes Semana Día Hora Hour_nonperiod Minuto Minute_nonperiod Segundo Second_nonperiod | |
| period_field | <i>campo</i> | |
| period_start_value | <i>entero</i> | |
| num_days_per_week | <i>entero</i> | |
| start_day_of_week | Sunday Monday Martes Miércoles Jueves Viernes Saturday | |
| num_hours_per_day | <i>entero</i> | |
| start_hour_of_day | <i>entero</i> | |

Tabla 146. Propiedades ts (continuación)

| Propiedades ts | Valores | Descripción de la propiedad |
|--------------------------|--|--|
| timestamp_increments | entero | |
| cyclic_increments | entero | |
| cyclic_periods | lista | |
| output_interval | Ninguno Año Trimestre Mes Semana Día Hora Minuto Segundo | |
| is_same_interval | flag | |
| cross_hour | flag | |
| aggregate_and_distribute | lista | |
| aggregate_default | Media Sum Mode Min Max | |
| distribute_default | Media Sum | |
| group_default | Media Sum Mode Min Max | |
| missing_imput | Linear_interp Series_mean K_mean K_median Linear_trend | |
| k_span_points | entero | |
| use_estimation_period | flag | |
| estimation_period | Observaciones Times | |
| date_estimation | lista | Solo está disponible si se utiliza date_time_field |
| period_estimation | lista | Solo está disponible si se utiliza use_period |
| observations_type | Latest Más antiguo | |
| observations_num | entero | |
| observations_exclude | entero | |
| method | ExpertModeler Exsmooth Arima | |

Tabla 146. Propiedades ts (continuación)

| Propiedades ts | Valores | Descripción de la propiedad |
|----------------------------------|---|---|
| expert_modeler_method | ExpertModeler Exsmooth Arima | |
| consider_seasonal | flag | |
| detect_outliers | flag | |
| expert_outlier_additive | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_innovational | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_transient | flag | |
| expert_outlier_seasonal_additive | flag | |
| expert_outlier_local_trend | flag | |
| expert_outlier_additive_patch | flag | |
| consider_newesmodels | flag | |
| exsmooth_model_type | Simple HoltsLinearTrend BrownsLinearTrend DampedTrend SimpleSeasonal WintersAdditive WintersMultiplicative DampedTrendAdditive DampedTrendMultiplicative MultiplicativeTrendAdditive MultiplicativeSeasonal MultiplicativeTrendMultiplicative MultiplicativeTrend | Especifica el método de Suavizado exponencial. El valor predeterminado es Simple. |
| futureValue_type_method | Compute specify | |
| exsmooth_transformation_type | Ninguno SquareRoot NaturalLog | |
| arima.p | entero | |
| arima.d | entero | |
| arima.q | entero | |
| arima.sp | entero | |
| arima.sd | entero | |
| arima.sq | entero | |
| arima_transformation_type | Ninguno SquareRoot NaturalLog | |
| arima_include_constant | flag | |
| tf_arima.p. nombredcampo | entero | Para funciones de transferencia. |
| tf_arima.d. nombredcampo | entero | Para funciones de transferencia. |

Tabla 146. Propiedades ts (continuación)

| Propiedades ts | Valores | Descripción de la propiedad |
|---|-------------------------------------|--|
| tf_arima.q. <i>nombredecampo</i> | entero | Para funciones de transferencia. |
| tf_arima.sp. <i>nombredecampo</i> | entero | Para funciones de transferencia. |
| tf_arima.sd. <i>nombredecampo</i> | entero | Para funciones de transferencia. |
| tf_arima.sq. <i>nombredecampo</i> | entero | Para funciones de transferencia. |
| tf_arima.delay. <i>nombredecampo</i> | entero | Para funciones de transferencia. |
| tf_arima.transformation_type. <i>nombredecampo</i> | Ninguno SquareRoot NaturalLog | Para funciones de transferencia. |
| arma_detect_outliers | flag | |
| arma_outlier_additive | flag | |
| arma_outlier_level_shift | flag | |
| arma_outlier_innovational | flag | |
| arma_outlier_transient | flag | |
| arma_outlier_seasonal_additive | flag | |
| arma_outlier_local_trend | flag | |
| arma_outlier_additive_patch | flag | |
| max_lags | entero | |
| cal_PI | flag | |
| conf_limit_pct | real | |
| eventos | campos | |
| continue | flag | |
| scoring_model_only | flag | Se utiliza para modelos con cifras muy grandes (cientos de miles) o series temporales. |
| forecastperiods | entero | |
| extend_records_into_future | flag | |
| extend_metric_values | campos | Le permite proporcionar valores futuros para los predictores. |
| conf_limits | flag | |
| noise_res | flag | |

Tabla 146. Propiedades ts (continuación)

| Propiedades ts | Valores | Descripción de la propiedad |
|-------------------|---------|--|
| max_models_output | entero | Controla cuántos modelos se muestran en el resultado. El valor predeterminado es 10. Los modelos no se muestran en el resultado si el número total de modelos creados supera este valor. Los modelos aún están disponibles para la puntuación. |

Propiedades timeseriesnode (en desuso)



Nota: Esta nodo Serie temporal original estaba en desuso en la versión 18 de SPSS Modeler y se sustituye por el nuevo nodo Serie temporal que se ha designado para emplear la potencia de IBM SPSS Analytic Server y procesar grandes cantidades de datos (Big Data). El nodo Serie temporal estima modelos de suavizado exponencial, modelos autorregresivos integrados de media móvil (ARIMA) univariados y modelos ARIMA (o de función de transferencia) multivariados para series temporales y genera previsiones. Un nodo Serie temporal debe ir siempre precedido por un nodo Intervalos de tiempo.

Ejemplo

```
node = stream.create("timeseries", "My node")
node.setPropertyValue("method", "Exsmooth")
node.setPropertyValue("exsmooth_model_type", "HoltsLinearTrend")
node.setPropertyValue("exsmooth_transformation_type", "None")
```

Tabla 147. propiedades de timeseriesnode

| Propiedades de timeseriesnode | Valores | Descripción de la propiedad |
|-------------------------------|---|--|
| targets | campo | El nodo Serie temporal prevé uno o más objetivos, utilizando opcionalmente uno o más campos de entrada como predictores. Los campos de frecuencia y ponderación no se usan. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| continue | flag | |
| method | ExpertModeler Exsmooth Arima Reuse | |
| expert_modeler_method | flag | |
| consider_seasonal | flag | |
| detect_outliers | flag | |
| expert_outlier_additive | flag | |

Tabla 147. propiedades de timeseriesnode (continuación)

| Propiedades de timeseriesnode | Valores | Descripción de la propiedad |
|--|--|----------------------------------|
| expert_outlier_level_shift | flag | |
| expert_outlier_innovational | flag | |
| expert_outlier_level_shift | flag | |
| expert_outlier_transient | flag | |
| expert_outlier_seasonal_additive | flag | |
| expert_outlier_local_trend | flag | |
| expert_outlier_additive_patch | flag | |
| exsmooth_model_type | Simple HoltLinearTrend BrownLinearTrend DampedTrend SimpleSeasonal WintersAdditive WintersMultiplicative | |
| exsmooth_transformation_type | Ninguno SquareRoot NaturalLog | |
| arma_p | entero | |
| arma_d | entero | |
| arma_q | entero | |
| arma_sp | entero | |
| arma_sd | entero | |
| arma_sq | entero | |
| arma_transformation_type | Ninguno SquareRoot NaturalLog | |
| arma_include_constant | flag | |
| tf_arma_p. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_d. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_q. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_sp. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_sd. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_sq. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_delay. nombredecampo | entero | Para funciones de transferencia. |
| tf_arma_transformation_type. nombredecampo | Ninguno SquareRoot NaturalLog | Para funciones de transferencia. |
| arma_detect_outlier_mode | Ninguno Automatic | |

Tabla 147. propiedades de timeseriesnode (continuación)

| Propiedades de timeseriesnode | Valores | Descripción de la propiedad |
|---------------------------------|---------|--|
| arima_outlier_additive | flag | |
| arima_outlier_level_shift | flag | |
| arima_outlier_innovational | flag | |
| arima_outlier_transient | flag | |
| arima_outlier_seasonal_additive | flag | |
| arima_outlier_local_trend | flag | |
| arima_outlier_additive_patch | flag | |
| conf_limit_pct | real | |
| max_lags | entero | |
| eventos | campos | |
| scoring_model_only | flag | Se utiliza para modelos con cifras muy grandes (cientos de miles) o series temporales. |

propiedades de treeas



El nodo Tree-AS es similar al nodo CHAID existente; sin embargo, el nodo Tree-AS se ha designado para procesar grandes cantidades de datos (Big Data) para crear un solo árbol y mostrar el modelo resultante en el visor de salida que se ha añadido en SPSS Modeler versión 17. El nodo genera un árbol de decisiones utilizando estadísticas de chi-cuadrado (CHAID) para identificar divisiones opcionales. Este uso de CHAID puede generar árboles no binarios, lo que significa que algunas divisiones generarán más de dos ramas. Los campos de entrada y objetivo pueden ser continuos (rango numérico) o categóricos. CHAID exhaustivo es una modificación de CHAID que examina con mayor precisión todas las divisiones posibles, aunque necesita más tiempo para realizar los cálculos.

Tabla 148. propiedades de treeas

| Propiedades de treeas | Valores | Descripción de la propiedad |
|-----------------------|---------------------------|--|
| target | campo | En el nodo Tree-AS, los modelos CHAID requieren un único objetivo y uno o más campos de entrada. También se puede especificar un campo de frecuencia. Consulte el tema “Propiedades comunes de nodos de modelado” en la página 179 para obtener más información. |
| method | chaid exhaustive_chaid | |
| max_depth | entero | Profundidad máxima del árbol, de 0 a 20. El valor predeterminado es 5. |
| num_bins | entero | Sólo se utiliza si los datos están formados por entradas continuas. Establezca el número de intervalos de frecuencia que deben utilizarse para las entradas; las opciones son 2, 4, 5, 10, 20, 25, 50 o 100. |

Tabla 148. propiedades de treeas (continuación)

| Propiedades de treeas | Valores | Descripción de la propiedad |
|----------------------------|--------------------------------|--|
| record_threshold | entero | El número de registros en los que el modelo pasa de usar valores p a tamaños del efecto mientras se construye el árbol. El valor predeterminado es de 1.000.000; se incrementa o decrementa de 10.000 en 10.000. |
| split_alpha | número | Nivel de significancia para división. El valor debe estar comprendido entre 0.01 y 0.99. |
| merge_alpha | número | Nivel de significancia para fusión. El valor debe estar comprendido entre 0.01 y 0.99. |
| bonferroni_adjustment | flag | Los valores de significancia de ajuste utilizando el método de Bonferroni. |
| effect_size_threshold_cont | número | Establece el umbral del tamaño del efecto cuando se dividen los nodos y fusionan las categorías al usar un objetivo continuo. El valor debe estar comprendido entre 0.01 y 0.99. |
| effect_size_threshold_cat | número | Establezca el umbral de tamaño de efecto al dividir nodos y fusionar categorías al utilizar un objetivo categórico. El valor debe estar comprendido entre 0.01 y 0.99. |
| split_merged_categories | flag | Permitir segunda división de categorías fusionadas. |
| grouping_sig_level | número | Se usa para determinar cómo se forman los grupos de nodos o cómo se identifican los nodos inusuales. |
| chi_square | pearson likelihood_ratio | Método usado para calcular la estadística de chi cuadrado: Pearson o Razón de verosimilitud |
| minimum_record_use | use_percentage use_absolute | |
| min_parent_records_pc | número | El valor predeterminado es de 2. El mínimo es 1 y el máximo 100, en incrementos de 1. El valor de la rama padre debe ser superior que el de la rama hija. |
| min_child_records_pc | número | El valor predeterminado es 1. Mínimo 1, máximo 100, en incrementos de 1. |
| min_parent_records_abs | número | El valor predeterminado es 100. Mínimo 1, máximo 100, en incrementos de 1. El valor de la rama padre debe ser superior a la rama hija. |
| min_child_records_abs | número | El valor predeterminado es 50. Mínimo 1, máximo 100, e incrementos de 1. |
| epsilon | número | Cambio mínimo en frecuencias de casillas esperadas. |
| max_iterations | número | Número máximo de iteraciones para la convergencia. |
| use_costs | flag | |

Tabla 148. propiedades de treeas (continuación)

| Propiedades de treeas | Valores | Descripción de la propiedad |
|-----------------------|--|---|
| costes | <i>structured</i> | Propiedad estructurada. El formato es una lista de 3 valores: el valor real, el valor predicho y el coste si dicha predicción es errónea. Por ejemplo: tree.setPropertyValue("costes", [{"drugA", "drugB", 3.0}, {"drugX", "drugY", 4.0}]) |
| default_cost_increase | ninguno lineal square personalizada | Nota: sólo se habilita para objetivos ordinales. Establezca valores predeterminados en la matriz de costes. |
| calculate_conf | <i>flag</i> | |
| display_rule_id | <i>flag</i> | Añade un campo en el resultado de puntuación que indica el ID para el nodo terminal al que se asigna cada registro. |

Propiedades de twostepnode



El nodo Bietápico es un método de agrupación en clústeres de dos pasos. El primer paso es hacer una única pasada por los datos para comprimir los datos de entrada de la fila en un conjunto de subclústeres administrable. El segundo paso utiliza un método de agrupación en clústeres jerárquica para fundir progresivamente los subclústeres en clústeres cada vez más grandes. El bietápico tiene la ventaja de estimar automáticamente el número óptimo de clústeres para los datos de entrenamiento. Puede gestionar tipos de campos mixtos y grandes conjuntos de datos eficazmente.

Ejemplo

```
node = stream.create("twostep", "My node")
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("inputs", ["Age", "K", "Na", "BP"])
node.setPropertyValue("partition", "Test")
node.setPropertyValue("use_model_name", False)
node.setPropertyValue("model_name", "TwoStep_Drug")
node.setPropertyValue("use_partitioned_data", True)
node.setPropertyValue("exclude_outliers", True)
node.setPropertyValue("cluster_label", "String")
node.setPropertyValue("label_prefix", "TwoStep_")
node.setPropertyValue("cluster_num_auto", False)
node.setPropertyValue("max_num_clusters", 9)
node.setPropertyValue("min_num_clusters", 3)
node.setPropertyValue("num_clusters", 7)
```

Tabla 149. propiedades de twostepnode

| Propiedad de twostepnode | Valores | Descripción de la propiedad |
|--------------------------|----------------------------|--|
| inputs | <i>[field1 ... fieldN]</i> | Los modelos bietápicos utilizan una lista de campos de entrada, pero no de campos objetivo. Los campos de ponderación y frecuencia no se reconocen. Consulte el tema "Propiedades comunes de nodos de modelado" en la página 179 para obtener más información. |
| standardize | <i>flag</i> | |

Tabla 149. propiedades de twostepnode (continuación)

| Propiedad de twostepnode | Valores | Descripción de la propiedad |
|--------------------------|----------------------------|-----------------------------|
| exclude_outliers | flag | |
| percentage | número | |
| cluster_num_auto | flag | |
| min_num_clusters | número | |
| max_num_clusters | número | |
| num_clusters | número | |
| cluster_label | Cadena Number | |
| label_prefix | cadena | |
| distance_measure | Euclidean Loglikelihood | |
| clustering_criterion | AIC BIC | |

Propiedades de twostepAS



El clúster bietápico es una herramienta de exploración diseñada para descubrir las agrupaciones naturales (o clústeres) de un conjunto de datos que, de otra manera, no sería posible detectar. El algoritmo que emplea este procedimiento incluye varias atractivas características que lo hacen diferente de las técnicas de agrupación en clústeres tradicionales, como el tratamiento de variables categóricas y continuas, la selección automática de número de clústeres y la escalabilidad.

Tabla 150. propiedades de twostepAS

| Propiedades de twostepAS | Valores | Descripción de la propiedad |
|------------------------------|---|---|
| inputs | [f1 ... fN] | Los modelos bietápicos utilizan una lista de campos de entrada, pero no de objetivos. Los campos de ponderación y frecuencia no se reconocen. |
| use_predefined_roles | Booleano | Default=True |
| use_custom_field_assignments | Booleano | Default=False |
| cluster_num_auto | Booleano | Default=True |
| min_num_clusters | número entero | Default=2 |
| max_num_clusters | número entero | Default=15 |
| num_clusters | número entero | Default=5 |
| clustering_criterion | AIC BIC | |
| automatic_clustering_method | use_clustering_criterion_setting Distance_jump Mínimo Máximo | |

Tabla 150. propiedades de twostepAS (continuación)

| Propiedades de twostepAS | Valores | Descripción de la propiedad |
|---|---|-----------------------------|
| feature_importance_method | use_clustering_criterion_setting effect_size | |
| use_random_seed | Booleano | |
| random_seed | número entero | |
| distance_measure | Euclidean Loglikelihood | |
| include_outlier_clusters | Booleano | Default=True |
| num_cases_in_feature_tree_leaf_is_less_than | número entero | Default=10 |
| top_perc_outliers | número entero | Default=5 |
| initial_dist_change_threshold | número entero | Default=0 |
| leaf_node_maximum_branches | número entero | Default=8 |
| non_leaf_node_maximum_branches | número entero | Default=8 |
| max_tree_depth | número entero | Default=3 |
| adjustment_weight_on_measurement_level | número entero | Default=6 |
| memory_allocation_mb | número | Default=512 |
| delayed_split | Booleano | Default=True |
| fields_to_standardize | [f1 ... fN] | |
| adaptive_feature_selection | Booleano | Default=True |
| featureMisPercent | número entero | Default=70 |
| coefRange | número | Default=0,05 |
| percCasesSingleCategory | número entero | Default=95 |
| numCases | número entero | Default=24 |
| include_model_specifications | Booleano | Default=True |
| include_record_summary | Booleano | Default=True |
| include_field_transformations | Booleano | Default=True |
| excluded_inputs | Booleano | Default=True |
| evaluate_model_quality | Booleano | Default=True |
| show_feature_importance_bar_chart | Booleano | Default=True |
| show_feature_importance_word_cloud | Booleano | Default=True |
| show_outlier_clusters_interactive_table_and_chart | Booleano | Default=True |
| show_outlier_clusters_pivot_table | Booleano | Default=True |
| across_cluster_feature_importance | Booleano | Default=True |
| across_cluster_profiles_pivot_table | Booleano | Default=True |
| withinprofiles | Booleano | Default=True |
| cluster_distances | Booleano | Default=True |
| cluster_label | Cadena Número | |
| label_prefix | Cadena | |

Capítulo 14. Propiedades del nodo de nugget de modelo

Los nodos de nugget de modelo comparten las mismas propiedades comunes que los otros nodos. Consulte el tema “Propiedades de nodos comunes” en la página 71 para obtener más información.

Propiedades de `applyanomalydetectionnode`

Los nodos de modelado Detección de anomalías pueden utilizarse para generar un nugget de modelo Detección de anomalías. El nombre de script de este nugget de modelo es `applyanomalydetectionnode`. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de `anomalydetectionnode`” en la página 179

Tabla 151. propiedades de `applyanomalydetectionnode`.

| Propiedades de <code>applyanomalydetectionnode</code> | Valores | Descripción de la propiedad |
|---|---------------------------------------|---|
| <code>anomaly_score_method</code> | FlagAndScore FlagOnly ScoreOnly | Determina que resultados se crean para puntuación. |
| <code>num_fields</code> | <i>entero</i> | Campos para informar. |
| <code>discard_records</code> | <i>tag</i> | Indica si los registros se descartan del resultado o no. |
| <code>discard_anomalous_records</code> | <i>tag</i> | Indicador de cuando descartar los registros anómalos o <i>no</i> anómalos. El valor predeterminado es <i>off</i> , que significa que se descartan los registros <i>no</i> anómalos. En caso contrario, si es <i>on</i> , se descartan los registros anómalos. Esta propiedad se activa sólo si la propiedad <code>discard_records</code> se activa. |

Propiedades de `applyapriorinode`

Los nodos de modelado Apriori pueden utilizarse para generar un nugget de modelo Apriori. El nombre de script de este nugget de modelo es `applyapriorinode`. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de `apriorinode`” en la página 181

Tabla 152. propiedades de `applyapriorinode`.

| Propiedades de <code>applyapriorinode</code> | Valores | Descripción de la propiedad |
|--|--|-----------------------------|
| <code>max_predictions</code> | <i>número (entero)</i> | |
| <code>ignore_unmatched</code> | <i>tag</i> | |
| <code>allow_repeats</code> | <i>tag</i> | |
| <code>check_basket</code> | NoPredictions Predictions NoCheck | |
| <code>criterio</code> | Confianza Soporte RuleSupport Lift Capacidad de despliegue | |

Propiedades de applyassociationrulesnode

El nodo de modelado de reglas de asociación se puede utilizar para generar un nugget de modelo de reglas de asociación. El nombre de script de este nugget de modelo es *applyassociationrulesnode*. Para obtener más información sobre los scripts para propio nodo de modelado, consulte “propiedades associationrulesnode” en la página 182.

Tabla 153. propiedades de applyassociationrulesnode

| Propiedades de applyassociationrulesnode | Tipo de datos | Descripción de la propiedad |
|--|---|---|
| max_predictions | entero | El número máximo de reglas que se pueden aplicar a cada entrada de la puntuación. |
| criterio | Confianza Rulesupport Lift Conditionsupport Capacidad de despliegue | Seleccione la medida utilizada para determinar la fuerza de las reglas. |
| allow_repeats | Booleana | Determine si las reglas con la misma predicción se incluyen en la puntuación. |
| check_input | NoPredictions Predictions NoCheck | |

Propiedades de applyautoclassifiernode

Los nodos de modelado de clasificador automático se pueden utilizar para crear un nugget de modelo Clasificador automático. El nombre de script de este nugget de modelo es *applyautoclassifiernode*. Para obtener más información sobre los scripts para el propio nodo de modelado, “propiedades de autoclassifiernode” en la página 185

Tabla 154. propiedades de applyautoclassifiernode.

| Propiedades de applyautoclassifiernode | Valores | Descripción de la propiedad |
|--|--|---|
| flag_ensemble_method | Voting ConfidenceWeightedVoting RawPropensityWeightedVoting HighestConfidence AverageRawPropensity | Especifica el método utilizado para determinar la puntuación del conjunto. Este conjunto sólo se aplica si el objetivo seleccionado es un campo de marca. |
| flag_voting_tie_selection | Random HighestConfidence RawPropensity | Si se selecciona un método de votación, especifica cómo se resolverán los empates. Este conjunto sólo se aplica si el objetivo seleccionado es un campo de marca. |
| set_ensemble_method | Voting ConfidenceWeightedVoting HighestConfidence | Especifica el método utilizado para determinar la puntuación del conjunto. Este conjunto sólo se aplica si el objetivo seleccionado es un campo de conjunto. |
| set_voting_tie_selection | Random HighestConfidence | Si se selecciona un método de votación, especifica cómo se resolverán los empates. Este conjunto sólo se aplica si el objetivo seleccionado es un campo nominal. |

Propiedades de applyautoclusternode

Los nodos de modelado de Clúster automático se pueden utilizar para crear un nugget de modelo Clúster automático. El nombre de script de este nugget de modelo es *applyautoclusternode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de nodo de agrupación en clústeres automática” en la página 187

Propiedades de applyautonumericnode

Los nodos de modelado autonumérico se pueden utilizar para crear un nugget de modelo Autonumérico. El nombre de script de este nugget de modelo es *applyautonumericnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, “propiedades de autonumericnode” en la página 188

Tabla 155. propiedades de applyautonumericnode.

| Propiedades de applyautonumericnode | Valores | Descripción de la propiedad |
|-------------------------------------|---------|-----------------------------|
| calculate_standard_error | tag | |

Propiedades de applybayesnetnode

Los nodos de modelado de red bayesiana pueden utilizarse para generar un nugget de modelo de red bayesiana. El nombre de script de este nugget de modelo es *applybayesnetnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de bayesnetnode” en la página 190.

Tabla 156. propiedades de applybayesnetnode.

| Propiedades de applybayesnetnode | Valores | Descripción de la propiedad |
|----------------------------------|---------|-----------------------------|
| all_probabilities | tag | |
| raw_propensity | tag | |
| adjusted_propensity | tag | |
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |

Propiedades de applyc50node

Los nodos de modelado C5.0 pueden utilizarse para generar un nugget de modelo C5.0. El nombre de script de este nugget de modelo es *applyc50node*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de c50node” en la página 192.

Tabla 157. propiedades de applyc50node.

| Propiedades de applyc50node | Valores | Descripción de la propiedad |
|---------------------------------|--------------------------|--|
| sql_generate | Nunca NoMissingValues | Se utiliza para establecer las opciones de generación de SQL durante la ejecución del conjunto de reglas. |
| calculate_conf | tag | Disponible cuando la generación de SQL está activada. Esta propiedad incluye los cálculos de confianza en el árbol generado. |
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |

Propiedades de applycarmanode

Los nodos de modelado CARMA pueden utilizarse para generar un nugget de modelo CARMA. El nombre de script de este nugget de modelo es *applycarmanode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de carmanode” en la página 193.

Propiedades de applycartnode

Se pueden utilizar los nodos de modelado C&RT para generar un nugget de modelo C&RT. El nombre de script de este nugget de modelo es *applycartnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de cartnode” en la página 194.

Tabla 158. propiedades de applycartnode.

| Propiedades de applycartnode | Valores | Descripción de la propiedad |
|---------------------------------|---|--|
| enable_sql_generation | Nunca MissingValues NoMissingValues | Se utiliza para establecer las opciones de generación de SQL durante la ejecución del conjunto de reglas. |
| calculate_conf | tag | Disponible cuando la generación de SQL está activada. Esta propiedad incluye los cálculos de confianza en el árbol generado. |
| display_rule_id | tag | Añade un campo en el resultado de puntuación que indica el ID para el nodo terminal al que se asigna cada registro. |
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |

Propiedades de applychaidnode

Los nodos de modelado CHAID pueden utilizarse para generar un nugget de modelo CHAID. El nombre de script de este nugget de modelo es *applychaidnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de chaidnode” en la página 197.

Tabla 159. propiedades de applychaidnode.

| Propiedades de applychaidnode | Valores | Descripción de la propiedad |
|---------------------------------|------------------------|---|
| enable_sql_generation | Nunca MissingValues | Se utiliza para establecer las opciones de generación de SQL durante la ejecución del conjunto de reglas. |
| calculate_conf | tag | |
| display_rule_id | tag | Añade un campo en el resultado de puntuación que indica el ID para el nodo terminal al que se asigna cada registro. |
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |

Propiedades de applycoxregnode

Los nodos de modelado Cox pueden utilizarse para generar un nugget de modelo Cox. El nombre de script de este nugget de modelo es *applycoxregnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de coxregnode” en la página 199.

Tabla 160. propiedades de *applycoxregnode*.

| Propiedades de <i>applycoxregnode</i> | Valores | Descripción de la propiedad |
|---------------------------------------|----------------------|-----------------------------|
| future_time_as | Intervalos Campos | |
| time_interval | <i>number</i> | |
| num_future_times | <i>entero</i> | |
| time_field | <i>campo</i> | |
| past_survival_time | <i>campo</i> | |
| all_probabilities | <i>tag</i> | |
| cumulative_hazard | <i>tag</i> | |

Propiedades de applydecisionlistnode

Los nodos de modelado Lista de decisiones pueden utilizarse para generar un nugget de modelo Lista de decisiones. El nombre de script de este nugget de modelo es *applydecisionlistnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de decisionlistnode” en la página 200.

Tabla 161. propiedades de *applydecisionlistnode*.

| Propiedades de <i>applydecisionlistnode</i> | Valores | Descripción de la propiedad |
|---|------------|---|
| enable_sql_generation | <i>tag</i> | Cuando se establece en true, IBM SPSS Modeler intenta enviar el modelo Lista de decisiones a SQL. |
| calculate_raw_propensities | <i>tag</i> | |
| calculate_adjusted_propensities | <i>tag</i> | |

Propiedades de applydiscriminantnode

Los nodos de modelado Discriminante pueden utilizarse para generar un nugget de modelo Discriminante. El nombre de script de este nugget de modelo es *applydiscriminantnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de discriminantnode” en la página 202.

Tabla 162. propiedades de *applydiscriminantnode*.

| Propiedades de <i>applydiscriminantnode</i> | Valores | Descripción de la propiedad |
|---|------------|-----------------------------|
| calculate_raw_propensities | <i>tag</i> | |
| calculate_adjusted_propensities | <i>tag</i> | |

Propiedades de applyextension



Los nodos de Modelo de extensión se pueden utilizar para generar un nugget de modelo de Extensión. El nombre de script de este nugget de modelo es *applyextension*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de extensionmodelnode” en la página 203.

Ejemplo de Python para Spark

```
#### Ejemplo de script de Python para Spark
applyModel = stream.findByType("extension_apply", None)

score_script = """
import json
import spss.pyspark.runtime
from pyspark.mllib.regression import LabeledPoint
from pyspark.mllib.linalg import DenseVector
from pyspark.mllib.tree import DecisionTreeModel
from pyspark.sql.types import StringType, StructField

cxt = spss.pyspark.runtime.getContext()

if cxt.isComputeDataModelOnly():
    _schema = cxt.getSparkInputSchema()
    _schema.fields.append(StructField("Prediction", StringType(), nullable=True))
    cxt.setSparkOutputSchema(_schema)
else:
    df = cxt.getSparkInputData()

    _modelPath = cxt.getModelContentToPath("TreeModel")
    metadata = json.loads(cxt.getModelContentToString("model.dm"))

    schema = df.dtypes[:]
    target = "Drug"
    predictors = ["Age", "BP", "Sex", "Cholesterol", "Na", "K"]

    lookup = {}
    for i in range(0, len(schema)):
        lookup[schema[i][0]] = i

    def row2LabeledPoint(dm, lookup, target, predictors, row):
        target_index = lookup[target]
        tval = dm[target_index].index(row[target_index])
        pvals = []
        for predictor in predictors:
            predictor_index = lookup[predictor]
            if isinstance(dm[predictor_index], list):
                pval = row[predictor_index] in dm[predictor_index] and
                    dm[predictor_index].index(row[predictor_index]) or -1
            else:
                pval = row[predictor_index]
            pvals.append(pval)
        return LabeledPoint(tval, DenseVector(pvals))

    # convertir marco de datos a un RDD que contiene LabeledPoint
    lps = df.rdd.map(lambda row: row2LabeledPoint(metadata, lookup, target, predictors, row))
    treeModel = DecisionTreeModel.load(cxt.getSparkContext(), _modelPath);
    # puntuar el modelo, produce un RDD que contiene sólo valores dobles
    predictions = treeModel.predict(lps.map(lambda lp: lp.features))

    def addPrediction(x, dm, lookup, target):
```

```

result = []
for _idx in range(0, len(x[0])):
    result.append(x[0][_idx])
result.append(dm[lookup[target]][int(x[1])])
return result

_schema = cxt.getSparkInputSchema()
_schema.fields.append(StructField("Prediction", StringType(), nullable=True))
rdd2 = df.rdd.zip(predictions).map(lambda x:addPrediction(x, metadata, lookup, target))
outDF = cxt.getSparkSQLContext().createDataFrame(rdd2, _schema)

cxt.setSparkOutputData(outDF)
"""
applyModel.setPropertyValue("python_syntax", score_script)

```

Ejemplo de R

```

#### Ejemplo de script de R
applyModel.setPropertyValue("r_syntax", """
result<-predict(modelerModel,newdata=modelerData)
modelerData<-cbind(modelerData,result)
var1<-c(fieldName="NaPrediction",fieldLabel="",fieldStorage="real",fieldMeasure="",
fieldFormat="",fieldRole="")
modelerDataModel<-data.frame(modelerDataModel,var1)""")

```

Tabla 163. Propiedades de *applyextension*

| Propiedades de <i>applyextension</i> | Valores | Descripción de la propiedad |
|--------------------------------------|------------------------------------|--|
| <i>r_syntax</i> | <i>cadena</i> | Sintaxis de scripts R para la puntuación del modelo. |
| <i>python_syntax</i> | <i>cadena</i> | Sintaxis de scripts Python para la puntuación de modelo. |
| <i>use_batch_size</i> | <i>flag</i> | Habilitar uso de proceso por lotes. |
| <i>batch_size</i> | <i>entero</i> | Especifique el número de registros de datos que se deben incluir en cada lote. |
| <i>convert_flags</i> | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| <i>convert_missing</i> | <i>flag</i> | Opción para convertir valores perdidos al valor NA de R. |
| <i>convert_datetime</i> | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| <i>convert_datetime_class</i> | POSIXct POSIXlt | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |

Propiedades de *applyfactornode*

Los nodos de modelado PCA/Factorial pueden utilizarse para generar un nugget de modelo PCA/Factorial. El nombre de script de este nugget de modelo es *applyfactornode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de factornode” en la página 206.

Propiedades de applyfeatureselectionnode

Los nodos de modelado Selección de características pueden utilizarse para generar un nugget de modelo Selección de características. El nombre de script de este nugget de modelo es *applyfeatureselectionnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de featureselectionnode” en la página 207.

Tabla 164. propiedades de applyfeatureselectionnode.

| Propiedades de applyfeatureselectionnode | Valores | Descripción de la propiedad |
|--|---------|---|
| selected_ranked_fields | | especifica qué campos clasificados se comprueban en el explorador de modelos. |
| selected_screened_fields | | Especifica qué campos filtrados se comprueban en el explorador de modelos. |

Propiedades de applygeneralizedlinearnode

Los nodos de modelado lineal generalizado (genlin) pueden utilizarse para generar un nugget de modelo lineal generalizado. El nombre de script de este nugget de modelo es *applygeneralizedlinearnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de genlinnode” en la página 209.

Tabla 165. propiedades de applygeneralizedlinearnode.

| Propiedades de applygeneralizedlinearnode | Valores | Descripción de la propiedad |
|---|---------|-----------------------------|
| calculate_raw_propensities | tag | |
| calculate_adjusted_propensities | tag | |

Propiedades de applyglmnode

Los nodos de modelado GLMM pueden utilizarse para generar un nugget de modelo GLMM. El nombre de script de este nugget de modelo es *applyglmnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de glmnode” en la página 212.

Tabla 166. Propiedades de applyglmnode.

| Propiedades de applyglmnode | Valores | Descripción de la propiedad |
|------------------------------|-----------------------------|--|
| confidence | onProbability onIncrease | Base para calcular el valor de confianza de la puntuación: probabilidad más alta predicha, o la diferencia entre la probabilidad más alta predicha y la segunda probabilidad más alta. |
| score_category_probabilities | tag | Si es True, genera las probabilidades predichas para objetivos categóricos. Se crea un campo para cada categoría. El valor predeterminado es False. |
| max_categories | entero | Número máximo de categorías para el que se van a predecir las probabilidades. Sólo se utiliza si score_category_probabilities es True. |

Tabla 166. Propiedades de *applyglmnode* (continuación).

| Propiedades de <i>applyglmnode</i> | Valores | Descripción de la propiedad |
|------------------------------------|---------------|--|
| score_propensity | tag | Si se establece en True, genera puntuaciones de propensión en bruto (probabilidad de resultado "True") para modelos con objetivos de marca. Si las particiones están en vigor, también genera puntuaciones de propensión ajustadas en función de la partición de prueba. El valor predeterminado es False. |
| enable_sql_generation | udf native | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Las opciones son para retrotraer a la base de datos y puntuar utilizando un adaptador de puntuación de SPSS® Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado) o para puntuar en SPSS Modeler. El valor predeterminado es udf. |

Propiedades de *applygle*

Los nodos de modelado GLE pueden utilizarse para generar un nugget de modelo GLE. El nombre de script de este nugget de modelo es *applygle*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte "Propiedades de *gle*" en la página 216.

Tabla 167. Propiedades de *applygle*

| Propiedades de <i>applygle</i> | Valores | Descripción de la propiedad |
|--------------------------------|---------------|---|
| enable_sql_generation | udf native | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Elija si desea realizar una retrotracción a la base de datos y puntuar utilizando un adaptador de puntuación de SPSS Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado) o puntuar dentro de SPSS Modeler. |

Propiedades de *applykmeansnode*

Los nodos de modelado K-medias pueden utilizarse para generar un nugget de modelo K-medias. El nombre de script de este nugget de modelo es *applykmeansnode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte "propiedades de *kmeansnode*" en la página 221.

Propiedades de applyknnnode

Los nodos de modelado KNN pueden utilizarse para generar un nugget de modelo KNN. El nombre de script de este nugget de modelo es *applyknnnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de knnnode” en la página 222.

Tabla 168. propiedades de *applyknnnode*.

| Propiedades de <i>applyknnnode</i> | Valores | Descripción de la propiedad |
|------------------------------------|------------|-----------------------------|
| all_probabilities | <i>tag</i> | |
| save_distances | <i>tag</i> | |

Propiedades de applykohonennode

Los nodos de modelado Kohonen pueden utilizarse para generar un nugget de modelo Kohonen. El nombre de script de este nugget de modelo es *applykohonennode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de c50node” en la página 192.

Propiedades de applylinearnode

Los nodos de modelado lineal pueden utilizarse para generar un nugget de modelo lineal. El nombre de script de este nugget de modelo es *applylinearnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de linearnode” en la página 225.

Tabla 169. Propiedades de *applylinearnode*.

| Propiedades de <i>linear</i> | Valores | Descripción de la propiedad |
|------------------------------|--------------------------|---|
| use_custom_name | <i>tag</i> | |
| custom_name | <i>cadena</i> | |
| enable_sql_generation | udf native puresql | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Las opciones se van a retrotraer a la base de datos y la puntuación mediante el adaptador de puntuación SPSS® Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado), para puntuar en SPSS Modeler, o para retrotraer a la base de datos y puntuar mediante SQL. El valor predeterminado es <i>udf</i> . |

Propiedades de applylinearasnode

Los nodos de modelado Linear-AS pueden utilizarse para generar un nugget de modelo Linear-AS. El nombre de script de este nugget de modelo es *applylinearasnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de linearasnode” en la página 226.

Tabla 170. Propiedades de *applylinearasnode*

| Propiedad <i>applylinearasnode</i> | Valores | Descripción de la propiedad |
|------------------------------------|---------------|---|
| enable_sql_generation | udf native | El valor predeterminado es <i>udf</i> . |

Propiedades de applylogregnode

Los nodos de modelado Regresión logística pueden utilizarse para generar un nugget de modelo Regresión logística. El nombre de script de este nugget de modelo es *applylogregnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de logregnode” en la página 227.

Tabla 171. propiedades de applylogregnode.

| Propiedades de applylogregnode | Valores | Descripción de la propiedad |
|--------------------------------|-------------|-----------------------------|
| calculate_raw_propensities | <i>tag</i> | |
| calculate_conf | <i>flag</i> | |
| enable_sql_generation | <i>flag</i> | |

Propiedades de applysvmnode

Los nodos de modelado LSVM pueden utilizarse para generar un nugget de modelo LSVM. El nombre de script de este nugget de modelo es *applysvmnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de svmnode” en la página 231.

Tabla 172. propiedades de applysvmnode

| Propiedades de applysvmnode | Valores | Descripción de la propiedad |
|-----------------------------|---------------|---|
| calculate_raw_propensities | <i>flag</i> | Especifica si deben calcularse puntuaciones de propensión en bruto. |
| enable_sql_generation | udf native | Especifica si debe puntuarse utilizando el adaptador de puntuación (si está instalado) o en proceso, o puntuar fuera de la base de datos. |

Propiedades de applyneuralnetnode

Los nodos de modelado Red neuronal pueden utilizarse para generar un nugget de modelo Red neuronal. El nombre de script de este nugget de modelo es *applyneuralnetnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de neuralnetnode” en la página 232.

Precaución: Una versión más reciente del nodo de modelado Red neural, con características mejoradas, está disponible en esta versión y se describe en la sección siguiente (*applyneuralnetwork*). Aunque la versión anterior sigue estando disponible, le recomendamos actualizar sus scripts para que se usen la nueva versión. En este documento se incluyen detalles de la versión anterior como referencia, pero en versiones futuras dejará de ser compatible.

Tabla 173. propiedades de applyneuralnetnode.

| Propiedades de applyneuralnetnode | Valores | Descripción de la propiedad |
|-----------------------------------|-----------------------|--|
| calculate_conf | <i>tag</i> | Disponible cuando la generación de SQL está activada. Esta propiedad incluye los cálculos de confianza en el árbol generado. |
| enable_sql_generation | <i>tag</i> | |
| nn_score_method | Diferencia SoftMax | |
| calculate_raw_propensities | <i>tag</i> | |
| calculate_adjusted_propensities | <i>tag</i> | |

propiedades de applyneuralnetworknode

Los nodos de modelado Red neuronal pueden utilizarse para generar un nugget de modelo Red neuronal. El nombre de script de este nugget de modelo es *applyneuralnetworknode*. Si desea más información sobre cómo crear scripts para el propio nodo de modelado, consulte Propiedades neuralnetworknode

Tabla 174. propiedades de applyneuralnetworknode

| Propiedades de applyneuralnetworknode | Valores | Descripción de la propiedad |
|---------------------------------------|-----------------------------|---|
| use_custom_name | flag | |
| custom_name | cadena | |
| confidence | onProbability onIncrease | |
| score_category_probabilities | flag | |
| max_categories | número | |
| score_propensity | flag | |
| enable_sql_generation | udf native puresql | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Las opciones se van a retrotraer a la base de datos y la puntuación mediante el adaptador de puntuación SPSS® Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado), para puntuar en SPSS Modeler, o para retrotraer a la base de datos y puntuar mediante SQL. El valor predeterminado es udf. |

Propiedades de applyocsvmnode

Los nodos SVM de una clase se pueden utilizar para generar un nugget de modelo SVM de una clase. El nombre de script de este nugget de modelo es *applyocsvmnode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de ocsvmnode” en la página 348.

Propiedades de applyquestnode

Los nodos de modelado QUEST pueden utilizarse para generar un nugget de modelo QUEST. El nombre de script de este nugget de modelo es *applyquestnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de questnode” en la página 236.

Tabla 175. propiedades de applyquestnode.

| Propiedades de applyquestnode | Valores | Descripción de la propiedad |
|-------------------------------|---|---|
| enable_sql_generation | Nunca MissingValues NoMissingValues | Se utiliza para establecer las opciones de generación de SQL durante la ejecución del conjunto de reglas. |
| calculate_conf | tag | |
| display_rule_id | tag | Añade un campo en el resultado de puntuación que indica el ID para el nodo terminal al que se asigna cada registro. |

Tabla 175. propiedades de *applyquestnode* (continuación).

| Propiedades de <i>applyquestnode</i> | Valores | Descripción de la propiedad |
|--|------------|-----------------------------|
| <code>calculate_raw_propensities</code> | <i>tag</i> | |
| <code>calculate_adjusted_propensities</code> | <i>tag</i> | |

Propiedades de *applyr*

Los nodos de modelado R pueden utilizarse para generar un nugget de modelo R. El nombre de script de este nugget de modelo es *applyr*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de *buildr*” en la página 191.

Tabla 176. propiedades de *applyr*

| Propiedades de <i>applyr</i> | Valores | Descripción de la propiedad |
|-------------------------------------|------------------------------------|--|
| <code>score_syntax</code> | <i>cadena</i> | Sintaxis de scripts R para la puntuación del modelo. |
| <code>convert_flags</code> | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| <code>convert_datetime</code> | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| <code>convert_datetime_class</code> | POSIXct POSIXlt | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |
| <code>convert_missing</code> | <i>flag</i> | Opción para convertir los valores que faltan al valor R NA. |
| <code>use_batch_size</code> | <i>flag</i> | Habilite el uso del proceso por lotes |
| <code>batch_size</code> | <i>entero</i> | Especifique el número de registros de datos que se incluirán en cada lote |

Propiedades *applyrandomtrees*

El nodo de modelador Árboles aleatorios se puede utilizar para generar un nugget del modelo Árboles aleatorios. El nombre de script de este nugget de modelo es *applyrandomtrees*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades *randomtrees*” en la página 238.

Tabla 177. propiedades *applyrandomtrees*

| Propiedades <i>applyrandomtrees</i> | Valores | Descripción de la propiedad |
|-------------------------------------|---------------|---|
| <code>calculate_conf</code> | <i>flag</i> | Esta propiedad incluye cálculos de confianza en el árbol generado. |
| <code>enable_sql_generation</code> | udf native | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Elija si desea realizar una retrotracción a la base de datos y puntuar utilizando un adaptador de puntuación de SPSS Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado) o puntuar dentro de SPSS Modeler. |

Propiedades de applyregressionnode

Los nodos de modelado Regresión lineal pueden utilizarse para generar un nugget de modelo Regresión lineal. El nombre de script de este nugget de modelo es *applyregressionnode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de regressionnode” en la página 240.

propiedades de applyselflearningnode

Los nodos de modelado de modelo de respuesta de autoaprendizaje (SLRM) pueden utilizarse para generar un nugget de modelo SLRM. El nombre de script de este nugget de modelo es *applyselflearningnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de slrmnode” en la página 243.

Tabla 178. propiedades de applyselflearningnode.

| Propiedades de applyselflearningnode | Valores | Descripción de la propiedad |
|--------------------------------------|-------------------------|--|
| max_predictions | <i>number</i> | |
| randomization | <i>number</i> | |
| scoring_random_seed | <i>number</i> | |
| sort | ascending descending | Especifica si se mostrarán primero las ofertas con las puntuaciones más altas o más bajas. |
| model_reliability | <i>tag</i> | Tiene en cuenta la opción de fiabilidad del modelo de la pestaña Configuración. |

Propiedades de applysequencenode

Los nodos de modelado Secuencia pueden utilizarse para generar un nugget de modelo Secuencia. El nombre de script de este nugget de modelo es *applysequencenode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de sequencenode” en la página 242.

Propiedades de applysvmnode

Los nodos de modelado SVM pueden utilizarse para generar un nugget de modelo SVM. El nombre de script de este nugget de modelo es *applysvmnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de svmnode” en la página 248.

Tabla 179. propiedades de applysvmnode.

| Propiedades de applysvmnode | Valores | Descripción de la propiedad |
|---------------------------------|------------|-----------------------------|
| all_probabilities | <i>tag</i> | |
| calculate_raw_propensities | <i>tag</i> | |
| calculate_adjusted_propensities | <i>tag</i> | |

Propiedades de applystpnode

El nodo de modelado STP puede utilizarse para generar un nugget de modelo asociado, que muestra la salida del modelo en el Visor de salida. El nombre de script de este nugget de modelo es *applystpnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de stpnode” en la página 244.

Tabla 180. propiedades de *applystpnode*

| Propiedades de <i>applystpnode</i> | Tipo de datos | Descripción de la propiedad |
|------------------------------------|-----------------|-----------------------------|
| <i>uncertainty_factor</i> | <i>Booleana</i> | Mínimo 0, máximo 100. |

Propiedades de *applytcmnode*

Los nodos de modelado temporal causal (TCM) pueden usarse para generar un nugget de modelo TCM. El nombre de script de este nugget de modelo es *applytcmnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de *tcmnode*” en la página 249.

Tabla 181. Propiedades de *applytcmnode*

| Propiedades de <i>applytcmnode</i> | Valores | Descripción de la propiedad |
|------------------------------------|-----------------|-----------------------------|
| <i>ext_future</i> | <i>booleano</i> | |
| <i>ext_future_num</i> | <i>entero</i> | |
| <i>noise_res</i> | <i>booleano</i> | |
| <i>conf_limits</i> | <i>booleano</i> | |
| <i>target_fields</i> | <i>lista</i> | |
| <i>target_series</i> | <i>lista</i> | |

Propiedades *applyts*

El nodo de modelado Serie temporal se puede utilizar para generar un nugget de modelo Serie temporal. El nombre de scripts de este nugget de modelo es *applyts*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades *ts*” en la página 253.

Tabla 182. propiedades *applyts*

| Propiedades <i>applyts</i> | Valores | Descripción de la propiedad |
|------------------------------------|-----------------|-----------------------------|
| <i>extend_records_into_future</i> | <i>Booleana</i> | |
| <i>ext_future_num</i> | <i>entero</i> | |
| <i>compute_future_values_input</i> | <i>Booleana</i> | |
| <i>forecastperiods</i> | <i>entero</i> | |
| <i>noise_res</i> | <i>booleano</i> | |
| <i>conf_limits</i> | <i>booleano</i> | |
| <i>target_fields</i> | <i>lista</i> | |
| <i>target_series</i> | <i>lista</i> | |
| <i>includeTargets</i> | <i>campo</i> | |

Propiedades *applytimeseriesnode* (en desuso)

El nodo de modelado Serie temporal se puede utilizar para generar un nugget de modelo Serie temporal. El nombre de script de este nugget de modelo es *applytimeseriesnode*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades *timeseriesnode* (en desuso)” en la página 257.

Tabla 183. propiedades de *applytimeseriesnode*.

| Propiedades de <i>applytimeseriesnode</i> | Valores | Descripción de la propiedad |
|---|------------|-----------------------------|
| <i>calculate_conf</i> | <i>tag</i> | |

Tabla 183. propiedades de *applytimeseriesnode* (continuación).

| Propiedades de <i>applytimeseriesnode</i> | Valores | Descripción de la propiedad |
|---|---------|-----------------------------|
| calculate_residuals | tag | |

Propiedades de *applytreeas*

Los nodos de modelado Tree-AS pueden utilizarse para generar un nugget de modelo Tree-AS. El nombre de script de este nugget de modelo es *applytreeas*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “propiedades de treeas” en la página 259.

Tabla 184. propiedades de *applytreeas*

| Propiedades de <i>applytreeas</i> | Valores | Descripción de la propiedad |
|-----------------------------------|---------------|---|
| calculate_conf | flag | Esta propiedad incluye cálculos de confianza en el árbol generado. |
| display_rule_id | flag | Añade un campo en el resultado de puntuación que indica el ID para el nodo terminal al que se asigna cada registro. |
| enable_sql_generation | udf native | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Elija si desea realizar una retrotracción a la base de datos y puntuar utilizando un adaptador de puntuación de SPSS Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado) o puntuar dentro de SPSS Modeler. |

Propiedades de *applytwostepnode*

Los nodos de modelado Bietápico pueden utilizarse para generar un nugget de modelo Bietápico. El nombre de script de este nugget de modelo es *applytwostepnode*. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de twostepnode” en la página 261.

Propiedades de *applytwostepAS*

Los nodos de modelado Bietápico AS pueden utilizarse para generar un nugget de modelo Bietápico AS. El nombre de script de este nugget de modelo es *applytwostepAS*. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de twostepAS” en la página 262.

Tabla 185. Propiedades de *applytwostepAS*

| Propiedades de <i>applytwostepAS</i> | Valores | Descripción de la propiedad |
|--------------------------------------|---------------|--|
| enable_sql_generation | udf native | Se utiliza para establecer las opciones de generación de SQL durante la ejecución de ruta. Las opciones son para retrotraer a la base de datos y puntuar utilizando un adaptador de puntuación de SPSS® Modeler Server (si está conectado a una base de datos con un adaptador de puntuación instalado) o para puntuar en SPSS Modeler. El valor predeterminado es udf. |

Propiedades de `applyxgboosttreenode`

El nodo XGBoost Tree se puede utilizar para generar un nugget de modelo de XGBoost Tree. El nombre de script de este nugget de modelo es `applyxgboosttreenode`. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de `xgboosttreenode`” en la página 345.

Propiedades de `applyxgboostlinearnode`

Los nodos XGBoost Linear se pueden utilizar para generar un nugget de modelo de XGBoost Linear. El nombre de script de este nugget de modelo es `applyxgboostlinearnode`. No existe ninguna otra propiedad para este nugget de modelo. Para obtener más información sobre los scripts para el propio nodo de modelado, consulte “Propiedades de `xboostlinearnode`” en la página 347.

Capítulo 15. Propiedades de nodos de modelado de bases de datos

IBM SPSS Modeler admite la integración con herramientas de modelado y minería de datos disponibles en proveedores de bases de datos, incluidos Microsoft SQL Server Analysis Services, Oracle Data Mining e IBM Netezza Analytics. Podrá crear y almacenar modelos mediante algoritmos nativos de bases de datos, todo desde la aplicación IBM SPSS Modeler. Los modelos de base de datos también se pueden crear y manipular a través de scripts utilizando las propiedades descritas en esta sección.

Por ejemplo, el siguiente fragmento de script muestra la creación de un modelo de árboles de decisión de Microsoft mediante la interfaz de scripts de IBM SPSS Modeler:

```
stream = modeler.script.stream()
msbuilder = stream.createAt("mstreenode", "MSBuilder", 200, 200)

msbuilder.setPropertyValue("analysis_server_name", 'localhost')
msbuilder.setPropertyValue("analysis_database_name", 'TESTDB')
msbuilder.setPropertyValue("mode", 'Expert')
msbuilder.setPropertyValue("datasource", 'LocalServer')
msbuilder.setPropertyValue("target", 'Drug')
msbuilder.setPropertyValue("inputs", ['Age', 'Sex'])
msbuilder.setPropertyValue("unique_field", 'IDX')
msbuilder.setPropertyValue("custom_fields", True)
msbuilder.setPropertyValue("model_name", 'MSDRUG')

typenode = stream.findByType("type", None)
stream.link(typenode, msbuilder)
results = []
msbuilder.run(results)
msapplier = stream.createModelApplierAt(results[0], "Drug", 200, 300)
tablenode = stream.createAt("table", "Results", 300, 300)
stream.linkBetween(msapplier, typenode, tablenode)
msapplier.setPropertyValue("sql_generate", True)
tablenode.run([])
```

Propiedades de nodos de modelado de Microsoft

Propiedades de nodos de modelado de Microsoft

Propiedades comunes

Las siguientes propiedades son comunes a los nodos de modelado de bases de datos de Microsoft.

Tabla 186. Propiedades comunes de nodos de Microsoft

| Propiedades comunes de nodo de Microsoft | Valores | Descripción de la propiedad |
|--|---------------|--|
| analysis_database_name | <i>cadena</i> | Nombre de la base de datos de Analysis Services. |
| analysis_server_name | <i>cadena</i> | Nombre del host de Analysis Services. |
| use_transactional_data | <i>flag</i> | Especifica si los datos de entrada están en formato tabular o transaccional. |
| inputs | <i>lista</i> | Campos de entrada de datos tabulares. |
| target | <i>campo</i> | Campo predicho (no aplicable a nodo Clúster de MS o nodos de Agrupación en clústeres de secuencias). |

Tabla 186. Propiedades comunes de nodos de Microsoft (continuación)

| Propiedades comunes de nodo de Microsoft | Valores | Descripción de la propiedad |
|--|-------------------|--|
| unique_field | <i>campo</i> | Campos clave. |
| msas_parameters | <i>structured</i> | Parámetros del algoritmo. Consulte el tema “Parámetros del algoritmo” en la página 285 para obtener más información. |
| with_drillthrough | <i>flag</i> | Opción Con exploración. |

Árbol de decisión de MS

No hay propiedades específicas para los nodos del tipo `mstreenode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Clúster de MS

No hay propiedades específicas para los nodos del tipo `msclusternode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Reglas de asociación de MS

Las siguientes propiedades específicas están disponibles para los nodos del tipo `msassocnode`:

Tabla 187. propiedades de `msassocnode`

| Propiedades de <code>msassocnode</code> | Valores | Descripción de la propiedad |
|---|--------------|--|
| id_field | <i>campo</i> | Identifica todas las transacciones en los datos. |
| trans_inputs | <i>lista</i> | Los campos de entrada de datos transaccionales. |
| transactional_target | <i>campo</i> | Campo predicho (datos transaccionales). |

Bayesiano ingenuo de MS

No hay propiedades específicas para los nodos del tipo `msbayesnode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Regresión lineal de MS

No hay propiedades específicas para los nodos del tipo `msregressionnode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Red neuronal de MS

No hay propiedades específicas para los nodos del tipo `msneuralnetworknode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Regresión logística de MS

No hay propiedades específicas para los nodos del tipo `mslogisticnode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Series temporales de MS

No hay propiedades específicas para los nodos del tipo `mstimeSeriesNode`. Consulte las propiedades comunes de Microsoft que se indican al comienzo de esta sección.

Clúster de secuencias de MS

Las siguientes propiedades específicas están disponibles para los nodos del tipo `mssequenceclusternode`:

Tabla 188. propiedades de `mssequenceclusternode`

| Propiedades de <code>mssequenceclusternode</code> | Valores | Descripción de la propiedad |
|---|--------------|--|
| <code>id_field</code> | <i>campo</i> | Identifica todas las transacciones en los datos. |
| <code>input_fields</code> | <i>lista</i> | Los campos de entrada de datos transaccionales. |
| <code>sequence_field</code> | <i>campo</i> | Identificador de secuencia. |
| <code>target_field</code> | <i>campo</i> | Campo predicho (datos tabulares). |

Parámetros del algoritmo

Cada tipo de modelo de base de datos de Microsoft tiene parámetros concretos que se pueden establecer mediante la propiedad `msas_parameters`. Por ejemplo:

```
stream = modeler.script.stream()
msregressionnode = stream.findByType("msregression", None)
msregressionnode.setPropertyValue("msas_parameters", [
["MAXIMUM_INPUT_ATTRIBUTES", 255],
["MAXIMUM_OUTPUT_ATTRIBUTES", 255]])
```

Estos parámetros se derivan de SQL Server. Para ver los parámetros relevantes para cada nodo:

1. Coloque un nodo de origen de base de datos en el lienzo.
2. Abra el nodo de origen de base de datos.
3. Seleccione un origen válido en la lista desplegable **Origen de datos**.
4. Seleccione una tabla válida en la lista **Nombre de tabla**.
5. Pulse en **Aceptar** para cerrar el nodo de origen de base de datos.
6. Conecte un nodo de modelado de bases de datos de Microsoft cuyas propiedades desee conocer.
7. Abra el nodo de modelado de bases de datos.
8. Seleccione la pestaña **Experto**.

Aparecerán las propiedades `msas_parameters` disponibles de este nodo.

Propiedades de nugget de modelo de Microsoft

Las siguientes propiedades son para los nugget de modelo creados mediante los nodos de modelado de bases de datos de Microsoft.

Árbol de decisión de MS

Tabla 189. Propiedades de Árbol de decisión de MS

| Propiedades de <code>aplymstreenode</code> | Valores | Descripción |
|--|---------------|--|
| <code>analysis_database_name</code> | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| <code>analysis_server_name</code> | <i>cadena</i> | Nombre del host del servidor de análisis. |

Tabla 189. Propiedades de Árbol de decisión de MS (continuación)

| Propiedades de aplymstreenode | Valores | Descripción |
|--------------------------------------|---------------|--|
| datasource | <i>cadena</i> | Nombre del origen de datos (DSN) ODBC de SQL Server. |
| sql_generate | <i>flag</i> | Activa la generación de SQL. |

Regresión lineal de MS

Tabla 190. Propiedades de Regresión lineal de MS

| Propiedades de aplymsregressionnode | Valores | Descripción |
|--|---------------|--|
| analysis_database_name | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| analysis_server_name | <i>cadena</i> | Nombre del host del servidor de análisis. |

Red neuronal de MS

Tabla 191. Propiedades de Red neuronal de MS

| Propiedades de aplymsneuralnetworknode | Valores | Descripción |
|---|---------------|--|
| analysis_database_name | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| analysis_server_name | <i>cadena</i> | Nombre del host del servidor de análisis. |

Regresión logística de MS

Tabla 192. Propiedades de Regresión logística de MS

| Propiedades de aplymslogisticnode | Valores | Descripción |
|--|---------------|--|
| analysis_database_name | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| analysis_server_name | <i>cadena</i> | Nombre del host del servidor de análisis. |

Serie temporales de MS

Tabla 193. Propiedades de MS Time Series

| Propiedades de aplymstimeseriesnode | Valores | Descripción |
|--|---------------|--|
| analysis_database_name | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| analysis_server_name | <i>cadena</i> | Nombre del host del servidor de análisis. |

Tabla 193. Propiedades de MS Time Series (continuación)

| Propiedades de <code>aplymstimeseriesnode</code> | Valores | Descripción |
|--|---|---|
| <code>start_from</code> | <code>new_prediction</code> <code>historical_prediction</code> | Especifica si se realizarán predicciones futuras o históricas. |
| <code>new_step</code> | <i>número</i> | Define el período de tiempo inicial de predicciones futuras. |
| <code>historical_step</code> | <i>número</i> | Define el período de tiempo inicial de predicciones históricas. |
| <code>end_step</code> | <i>número</i> | Define el período de tiempo final de las predicciones. |

Clúster de secuencias de MS

Tabla 194. Propiedades de Agrupación en clústeres de secuencias de MS

| propiedades de <code>aplymssequenceclusternode</code> | Valores | Descripción |
|---|---------------|--|
| <code>analysis_database_name</code> | <i>cadena</i> | Este nodo se puede puntuar directamente en una ruta. Esta propiedad se utiliza para identificar el nombre de la base de datos de Analysis Services. |
| <code>analysis_server_name</code> | <i>cadena</i> | Nombre del host del servidor de análisis. |

Propiedades de nodos de modelado de Oracle

Propiedades de nodos de modelado de Oracle

Las siguientes propiedades son comunes a los nodos de modelado de bases de datos de Oracle.

Tabla 195. Propiedades comunes de nodos de Oracle.

| Propiedades comunes de nodos de Oracle | Valores | Descripción de la propiedad |
|--|------------------------|---|
| <code>objetivo</code> | <i>campo</i> | |
| <code>inputs</code> | <i>Lista de campos</i> | |
| <code>partición</code> | <i>campo</i> | Campo usado para dividir los datos en muestras independientes para las fases de entrenamiento, comprobación y validación en la generación del modelo. |
| <code>datasource</code> | | |
| <code>nombre de usuario</code> | | |
| <code>contraseña</code> | | |
| <code>epassword</code> | | |
| <code>use_model_name</code> | <i>tag</i> | |
| <code>model_name</code> | <i>cadena</i> | Nombre personalizado para nuevo modelo. |
| <code>use_partitioned_data</code> | <i>tag</i> | Si se ha definido un campo de partición, esta opción garantiza que sólo se utilizarán los datos de la partición de entrenamiento para la generación del modelo. |
| <code>unique_field</code> | <i>campo</i> | |

Tabla 195. Propiedades comunes de nodos de Oracle (continuación).

| Propiedades comunes de nodos de Oracle | Valores | Descripción de la propiedad |
|--|------------------|--|
| auto_data_prep | tag | Activa o desactiva la característica de preparación de datos automática de Oracle (solamente para bases de datos 11g). |
| costs | estructurado | Propiedad estructurada de la siguiente forma: [[drugA drugB 1.5] [drugA drugC 2.1]], donde los argumentos en [] son costes pronosticados reales. |
| mode | Simple Expert | Hace que se ignoren ciertas propiedades si se establece como Simple, como se indica en las propiedades de nodos individuales. |
| use_prediction_probability | tag | |
| prediction_probability | cadena | |
| use_prediction_set | tag | |

Bayesiano ingenuo de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo oranbnode.

Tabla 196. propiedades de oranbnode

| Propiedades de oranbnode | Valores | Descripción de la propiedad |
|--------------------------|---------------------------------|---|
| singleton_threshold | número | 0.0–1.0.* |
| pairwise_threshold | número | 0.0–1.0.* |
| priors | Datos Equal Personalizado | |
| custom_priors | structured | Propiedad estructurada de la siguiente forma: set :oranbnode.custom_priors = [[drugA 1] [drugB 2] [drugC 3] [drugX 4] [drugY 5]] |

* Propiedad ignorada si Modo se establece como Simple.

Bayesiano adaptativo de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo oraabnnode.

Tabla 197. propiedades de oraabnnode

| Propiedades de oraabnnode | Valores | Descripción de la propiedad |
|----------------------------|---|---------------------------------|
| model_type | SingleFeature MultiFeature NaiveBayes | |
| use_execution_time_limit | flag | * |
| execution_time_limit | entero | El valor debe ser mayor que 0.* |
| max_naive_bayes_predictors | entero | El valor debe ser mayor que 0.* |
| max_predictors | entero | El valor debe ser mayor que 0.* |
| priors | Datos Equal Personalizado | |

Tabla 197. propiedades de oraabnnode (continuación)

| Propiedades de oraabnnode | Valores | Descripción de la propiedad |
|---------------------------|-------------------|--|
| custom_priors | <i>structured</i> | Propiedad estructurada de la siguiente forma: set :oraabnnode.custom_priors = [[drugA 1][drugB 2][drugC 3][drugX 4][drugY 5]] |

* Propiedad ignorada si Modo se establece como Simple.

Máquinas de vectores de soporte de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo orasvmnode.

Tabla 198. propiedades de orasvmnode

| Propiedades de orasvmnode | Valores | Descripción de la propiedad |
|---------------------------|---------------------------------|---|
| active_learning | Enable Disable | |
| kernel_function | Lineal Gaussian Sistema | |
| normalization_method | zscore minmax none | |
| kernel_cache_size | <i>entero</i> | Solamente kernel gaussiano. El valor debe ser mayor que 0.* |
| convergence_tolerance | <i>número</i> | El valor debe ser mayor que 0.* |
| use_standard_deviation | <i>flag</i> | Solamente kernel gaussiano.* |
| standard_deviation | <i>número</i> | El valor debe ser mayor que 0.* |
| use_epsilon | <i>flag</i> | Solamente modelos de regresión.* |
| epsilon | <i>número</i> | El valor debe ser mayor que 0.* |
| use_complexity_factor | <i>flag</i> | * |
| complexity_factor | <i>número</i> | * |
| use_outlier_rate | <i>flag</i> | Solamente variantes de una clase.* |
| outlier_rate | <i>número</i> | Solamente variantes de una clase. 0.0–1.0.* |
| weights | Datos Equal Personalizado | |
| custom_weights | <i>structured</i> | Propiedad estructurada de la siguiente forma: set :orasvmnode.custom_weights = [[drugA 1][drugB 2][drugC 3][drugX 4][drugY 5]] |

* Propiedad ignorada si Modo se establece como Simple.

Modelos lineales generalizados de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo `oraglmnode`.

Tabla 199. Propiedades de `oraglmnode`

| Propiedades de <code>oraglmnode</code> | Valores | Descripción de la propiedad |
|--|---------------------------------------|-----------------------------|
| <code>normalization_method</code> | zscore minmax none | |
| <code>missing_value_handling</code> | ReplaceWithMean UseCompleteRecords | |
| <code>use_row_weights</code> | <i>flag</i> | * |
| <code>row_weights_field</code> | <i>campo</i> | * |
| <code>save_row_diagnostics</code> | <i>flag</i> | * |
| <code>row_diagnostics_table</code> | <i>cadena</i> | * |
| <code>coefficient_confidence</code> | <i>número</i> | * |
| <code>use_reference_category</code> | <i>flag</i> | * |
| <code>reference_category</code> | <i>cadena</i> | * |
| <code>ridge_regression</code> | Auto Off On | * |
| <code>parameter_value</code> | <i>número</i> | * |
| <code>vif_for_ridge</code> | <i>flag</i> | * |

* Propiedad ignorada si Modo se establece como Simple.

Árbol de decisión de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo `oradecisiontreenode`.

Tabla 200. Propiedades de `oradecisiontreenode`

| Propiedades de <code>oradecisiontreenode</code> | Valores | Descripción de la propiedad |
|---|-----------------|---------------------------------|
| <code>use_costs</code> | <i>flag</i> | |
| <code>impurity_metric</code> | Entropy Gini | |
| <code>term_max_depth</code> | <i>entero</i> | 2–20.* |
| <code>term_minpct_node</code> | <i>número</i> | 0.0–10.0.* |
| <code>term_minpct_split</code> | <i>número</i> | 0.0–20.0.* |
| <code>term_minrec_node</code> | <i>entero</i> | El valor debe ser mayor que 0.* |
| <code>term_minrec_split</code> | <i>entero</i> | El valor debe ser mayor que 0.* |
| <code>display_rule_ids</code> | <i>flag</i> | * |

* Propiedad ignorada si Modo se establece como Simple.

O-clúster de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo oraoclusternode.

Tabla 201. Propiedades de oraoclusternode

| Propiedades de oraoclusternode | Valores | Descripción de la propiedad |
|--------------------------------|---------|---------------------------------|
| max_num_clusters | entero | El valor debe ser mayor que 0. |
| max_buffer | entero | El valor debe ser mayor que 0.* |
| sensitivity | número | 0.0–1.0.* |

* Propiedad ignorada si Modo se establece como Simple.

K-medias de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo orakmeansnode.

Tabla 202. Propiedades de orakmeansnode

| Propiedades de orakmeansnode | Valores | Descripción de la propiedad |
|------------------------------|--------------------------|---------------------------------------|
| num_clusters | entero | El valor debe ser mayor que 0. |
| normalization_method | zscore minmax none | |
| distance_function | Euclidean Cosine | |
| iteraciones | entero | 0–20.* |
| conv_tolerance | número | 0.0–0.5.* |
| split_criterion | Variance Size | El valor predeterminado es Variance.* |
| num_bins | entero | El valor debe ser mayor que 0.* |
| block_growth | entero | 1–5.* |
| min_pct_attr_support | número | 0.0–1.0.* |

* Propiedad ignorada si Modo se establece como Simple.

NMF de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo oranmfnode.

Tabla 203. Propiedades de oranmfnode

| Propiedades de oranmfnode | Valores | Descripción de la propiedad |
|---------------------------|----------------|--|
| normalization_method | minmax none | |
| use_num_features | flag | * |
| num_features | entero | 0–1. El algoritmo estima el valor predeterminado a partir de los datos.* |
| random_seed | número | * |
| num_iterations | entero | 0–500.* |

Tabla 203. Propiedades de oranmfnode (continuación)

| Propiedades de oranmfnode | Valores | Descripción de la propiedad |
|---------------------------|---------|-----------------------------|
| conv_tolerance | número | 0.0–0.5.* |
| display_all_features | flag | * |

* Propiedad ignorada si Modo se establece como Simple.

Apriori de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo oraapriorinode.

Tabla 204. Propiedades de oraapriorinode

| Propiedades de oraapriorinode | Valores | Descripción de la propiedad |
|-------------------------------|---------|-----------------------------|
| content_field | campo | |
| id_field | campo | |
| max_rule_length | entero | 2–20. |
| min_confidence | número | 0.0–1.0. |
| min_support | número | 0.0–1.0. |
| use_transactional_data | flag | |

Longitud mínima de la descripción de Oracle (LMD)

No hay propiedades específicas para los nodos del tipo oramdlnode. Consulte las propiedades comunes de Oracle que se indican al comienzo de esta sección.

Importancia del atributo de Oracle (AI)

Las siguientes propiedades están disponibles para los nodos del tipo oraainode.

Tabla 205. Propiedades de oraainode

| Propiedades de oraainode | Valores | Descripción de la propiedad |
|--------------------------|--|--|
| custom_fields | flag | Si es verdadero, le permite especificar el objetivo, la entrada y otros campos del nodo actual. Si es falso, se utiliza la configuración actual de un nodo Tipo situado en un punto anterior de la ruta. |
| selection_mode | ImportanceLevel ImportanceValue TopN | |
| select_important | flag | Si selection_mode se establece en ImportanceLevel, determina si se seleccionan los campos importantes. |
| important_label | cadena | Especifica la etiqueta para la clasificación como "important". |
| select_marginal | flag | Si selection_mode se establece en ImportanceLevel, determina si se seleccionan los campos marginales. |
| marginal_label | cadena | Especifica la etiqueta para la clasificación como "marginal". |
| important_above | número | 0.0–1.0. |

Tabla 205. Propiedades de oraainode (continuación)

| Propiedades de oraainode | Valores | Descripción de la propiedad |
|--------------------------|---------|---|
| select_unimportant | flag | Si selection_mode se establece en ImportanceLevel, determina si se seleccionan los campos sin importancia. |
| unimportant_label | cadena | Especifica la etiqueta para la clasificación como "unimportant". |
| unimportant_below | número | 0.0–1.0. |
| importance_value | número | Si selection_mode se establece en ImportanceValue, determina el valor de corte que se va a usar. Acepta valores de 0 a 100. |
| top_n | número | Si selection_mode se establece en TopN, determina el valor de corte que se va a usar. Acepta valores de 0 a 1000. |

Propiedades de nugget de modelo de Oracle

Las siguientes propiedades son para los nugget de modelo creados mediante los modelos de Oracle.

Bayesiano ingenuo de Oracle

No hay propiedades específicas para los nodos del tipo applyoranbnode.

Bayesiano adaptativo de Oracle

No hay propiedades específicas para los nodos del tipo applyoranbnode.

Máquinas de vectores de soporte de Oracle

No hay propiedades específicas para los nodos del tipo applyorasvmnode.

Árbol de decisión de Oracle

Las siguientes propiedades están disponibles para los nodos del tipo applyoradecisiontreenode.

Tabla 206. propiedades de applyoradecisiontreenode

| Propiedades de applyoradecisiontreenode | Valores | Descripción de la propiedad |
|---|---------|-----------------------------|
| use_costs | flag | |
| display_rule_ids | flag | |

O-clúster de Oracle

No hay propiedades específicas para los nodos del tipo applyoraoclusternode.

K-medias de Oracle

No hay propiedades específicas para los nodos del tipo applyorakmeansnode.

NMF de Oracle

La siguiente propiedad está disponible para los nodos del tipo `applyoranmfnode`:

Tabla 207. propiedades de `applyoranmfnode`

| Propiedades de <code>applyoranmfnode</code> | Valores | Descripción de la propiedad |
|---|-------------|-----------------------------|
| <code>display_all_features</code> | <i>flag</i> | |

Apriori de Oracle

Este nugget de modelo no se puede aplicar en los scripts.

LMD de Oracle

Este nugget de modelo no se puede aplicar en los scripts.

Propiedades de nodos de modelado de IBM Netezza Analytics

Propiedades de nodos de modelado de Netezza

Las siguientes propiedades son comunes a los nodos de modelado de bases de datos de IBM Netezza.

Tabla 208. Propiedades comunes de nodos de Netezza.

| Propiedades comunes de nodos de Netezza | Valores | Descripción de la propiedad |
|---|----------------------------|---|
| <code>custom_fields</code> | <i>tag</i> | Si es verdadero, le permite especificar el objetivo, la entrada y otros campos del nodo actual. Si es falso, se utiliza la configuración actual de un nodo Tipo situado en un punto anterior de la ruta. |
| <code>inputs</code> | <i>[campo1 ... campoN]</i> | Campos de entrada o predictor utilizados por el modelo. |
| <code>objetivo</code> | <i>campo</i> | Campo de destino (continuo o categórico). |
| <code>record_id</code> | <i>campo</i> | El campo que se debe utilizar como identificador de registros exclusivo. |
| <code>use_upstream_connection</code> | <i>tag</i> | Si es verdadero (valor predeterminado), los detalles de conexión especificados en un nodo anterior. No se utiliza si se especifica <code>move_data_to_connection</code> . |
| <code>move_data_connection</code> | <i>tag</i> | Si es verdadero, transfiere los datos a la base de datos especificada mediante <code>connection</code> . No se utiliza si se especifica <code>use_data_upstream_connection</code> . |
| <code>connection</code> | <i>estructurado</i> | La cadena de conexión para la base de datos de Netezza donde se almacena el modelo. Propiedad estructurada de la siguiente forma: ['odbc' '<dsn>' '<nombreusuario>' '<psw>' '<nombrecat>' '<atribos_conex>' [true false]] donde: <dsn> es el nombre del origen de datos <username> y <psw> son el nombre de usuario y la contraseña para la base de datos <catname> es el nombre de catálogo <conn_attribs> son los atributos de conexión true false indica si la contraseña es necesaria. |

Tabla 208. Propiedades comunes de nodos de Netezza (continuación).

| Propiedades comunes de nodos de Netezza | Valores | Descripción de la propiedad |
|---|---------|---|
| table_name | cadena | Nombre de la base de datos donde se debe almacenar el modelo. |
| use_model_name | tag | Si es true, utiliza el nombre que especifica mediante model_name como el nombre del modelo, de lo contrario el sistema crea el nombre del modelo. |
| model_name | cadena | Nombre personalizado para nuevo modelo. |
| include_input_fields | tag | Si es verdadero, transmite todos los campos de entrada siguientes, de lo contrario solo transmite record_id y los campos generados por el modelo. |

Árbol de decisión de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo netezzadectreenode.

Tabla 209. Propiedades de netezzadectreenode

| Propiedades de netezzadectreenode | Valores | Descripción de la propiedad |
|-----------------------------------|-----------------|--|
| impurity_measure | Entropy Gini | La medición de impureza, utilizada para valorar la mejor ubicación para dividir el árbol. |
| max_tree_depth | entero | Número máximo de niveles que puede alcanzar el crecimiento de árbol. El valor predeterminado es 62 (el máximo posible). |
| min_improvement_splits | número | Mejoras mínimas que se pueden realizar en la impureza de la división. El valor predeterminado es 0.01. |
| min_instances_split | entero | Número mínimo de registros por dividir antes de realizar la división. El valor predeterminado es 2 (el mínimo posible). |
| weights | structured | Ponderaciones relativas para clases. Propiedad estructurada de la siguiente forma: set :netezza_dectree.weights = [[drugA 0.3][drugB 0.6]] El valor predeterminado es la ponderación de 1 para todas las clases. |
| pruning_measure | Acc wAcc | El valor predeterminado es Acc (precisión). wAcc alternativo (precisión ponderada) tiene en cuenta las ponderaciones de clase mientras se aplica la poda. |

Tabla 209. Propiedades de netezzadectreenode (continuación)

| Propiedades de netezzadectreenode | Valores | Descripción de la propiedad |
|-----------------------------------|---|---|
| prune_tree_options | allTrainingData partitionTrainingData useOtherTable | El valor predeterminado es utilizar allTrainingData para calcular la precisión del modelo. Utilice partitionTrainingData para especificar un porcentaje de datos de prueba por utilizar, o useOtherTable para utilizar un conjunto de datos de prueba desde una tabla específica de la base de datos. |
| perc_training_data | número | Si prune_tree_options se establece en partitionTrainingData, especifica el porcentaje de datos que se utilizará para entrenamiento. |
| prune_seed | entero | Semilla aleatoria que se debe utilizar para replicar los resultados del análisis si prune_tree_options se establece en partitionTrainingData; el valor predeterminado es 1. |
| pruning_table | cadena | Nombre de tabla de un conjunto de datos de poda separado para estimar la precisión del modelo. |
| compute_probabilities | flag | Si es verdadero, produce un campo de nivel de confianza (probabilidad) y el campo de predicción. |

K-medias de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo netezzakmeansnode.

Tabla 210. Propiedades de netezzakmeansnode

| Propiedades de netezzakmeansnode | Valores | Descripción de la propiedad |
|----------------------------------|--|---|
| distance_measure | Euclidean Manhattan Canberra máximo | Método que se debe utilizar para medir la distancia entre puntos de datos. |
| num_clusters | entero | Número de clústeres que se deben crear; el valor predeterminado es 3. |
| max_iterations | entero | Número de iteraciones de algoritmos después de los cuáles se debe detener la prueba del modelo; el valor predeterminado es 5. |
| rand_seed | entero | Semilla aleatoria que se debe utilizar para replicar los resultados del análisis; el valor predeterminado es 12345. |

Red bayesiana de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezzabayesnode`.

Tabla 211. Propiedades de `netezzabayesnode`

| Propiedades de <code>netezzabayesnode</code> | Valores | Descripción de la propiedad |
|--|--|---|
| <code>base_index</code> | <i>entero</i> | Identificador numérico asignado al primer campo de entrada de gestión interna; el valor predeterminado es 777. |
| <code>sample_size</code> | <i>entero</i> | Tamaño de la muestra que se tomará si el número de atributos es muy grande; el valor predeterminado es 10.000. |
| <code>display_additional_information</code> | <i>flag</i> | Si es verdadero, muestra información adicional sobre el progreso en un cuadro de diálogo de mensaje. |
| <code>type_of_prediction</code> | <i>best vecinos nn-neighbors</i> | Tipo de algoritmo de predicción que se utilizará: el mejor (vecino más correlacionado), vecinos (predicción ponderada de vecinos), o vecinos NN (vecinos no nulos). |

Bayesiano ingenuo de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezzanaivebayesnode`.

Tabla 212. propiedades de `netezzanaivebayesnode`

| Propiedades de <code>netezzanaivebayesnode</code> | Valores | Descripción de la propiedad |
|---|-------------|---|
| <code>compute_probabilities</code> | <i>flag</i> | Si es verdadero, produce un campo de nivel de confianza (probabilidad) y el campo de predicción. |
| <code>use_m_estimation</code> | <i>flag</i> | Si es verdadero, utiliza la técnica m-estimation para evitar probabilidades de cero durante el cálculo. |

KNN de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezzaknnnode`.

Tabla 213. Propiedades de `netezzaknnnode`

| Propiedades de <code>netezzaknnnode</code> | Valores | Descripción de la propiedad |
|--|--|---|
| <code>weights</code> | <i>structured</i> | Propiedad estructurada que se utiliza para asignar ponderaciones a clases individuales. Ejemplo: <code>set :netezzaknnnode.weights = [[drugA 0.3][drugB 0.6]]</code> |
| <code>distance_measure</code> | Euclidean Manhattan Canberra Máximo | Método que se utiliza para medir la distancia entre puntos de datos. |
| <code>num_nearest_neighbors</code> | <i>entero</i> | Número de vecinos más próximos de un caso concreto; el valor predeterminado es 3. |
| <code>standardize_measurements</code> | <i>flag</i> | Si es verdadero, estandariza las mediciones de campos de entrada continuos antes de calcular los valores de distancia. |

Tabla 213. Propiedades de netezzaknnnode (continuación)

| Propiedades de netezzaknnnode | Valores | Descripción de la propiedad |
|-------------------------------|---------|---|
| use_coresets | flag | Si es verdadero, utiliza el muestreo del conjunto principal para acelerar el cálculo de conjuntos de datos grandes. |

Clúster divisivo de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo netezzadivclusternode.

Tabla 214. Propiedades de netezzadivclusternode

| Propiedades de netezzadivclusternode | Valores | Descripción de la propiedad |
|--------------------------------------|--|---|
| distance_measure | Euclidean Manhattan Canberra Máximo | Método que se utiliza para medir la distancia entre puntos de datos. |
| max_iterations | entero | Número máximo de iteraciones de algoritmo que se ejecutarán antes de detener el entrenamiento del modelo; el valor predeterminado es 5. |
| max_tree_depth | entero | El número máximo de niveles en los que se puede subdividir el conjunto de datos; el valor predeterminado es 3. |
| rand_seed | entero | Semilla aleatoria, se utiliza para replicar los análisis; el valor predeterminado es 12345. |
| min_instances_split | entero | El número mínimo de registros que se pueden dividir, el valor predeterminado es 5. |
| nivel | entero | El nivel de jerarquía en el que se guardan los registros; el valor predeterminado es -1. |

PCA de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo netezzapcanode.

Tabla 215. Propiedades de netezzapcanode

| Propiedades de netezzapcanode | Valores | Descripción de la propiedad |
|-------------------------------|---------|--|
| center_data | flag | Si es verdadero (opción predeterminada), ejecuta el centrado de datos (también conocido como "sustracción de media") antes del análisis. |
| perform_data_scaling | flag | Si es verdadero, ejecuta la adaptación de los datos antes del análisis. De esta forma el análisis será menos arbitrario si las diferentes variables se miden en unidades diferentes. |
| force_eigensolve | flag | Si es verdadero, utiliza un método menos preciso, pero más rápido para encontrar componentes principales. |
| pc_number | entero | El número principal de componentes al que se reducirá el conjunto de datos; el valor predeterminado es 1. |

Árbol de regresión de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezzaregtreenode`.

Tabla 216. Propiedades de `netezzaregtreenode`

| Propiedades de <code>netezzaregtreenode</code> | Valores | Descripción de la propiedad |
|--|---|--|
| <code>max_tree_depth</code> | <i>entero</i> | Número máximo de niveles que puede crecer el árbol por debajo del nodo raíz; el valor predeterminado es 10. |
| <code>split_evaluation_measure</code> | Variance | Medida de clase de impureza, se utiliza para evaluar la mejor ubicación para dividir el árbol; el valor predeterminado (y la única opción actualmente) es Variance. |
| <code>min_improvement_splits</code> | <i>número</i> | Cantidad mínima para reducir la impureza antes de que se cree la nueva división en el árbol. |
| <code>min_instances_split</code> | <i>entero</i> | El número mínimo de registros que se pueden dividir. |
| <code>pruning_measure</code> | mse r2 pearson spearman | Método que se utilizará para la poda. |
| <code>prune_tree_options</code> | allTrainingData partitionTrainingData useOtherTable | El valor predeterminado es utilizar <code>allTrainingData</code> para calcular la precisión del modelo. Utilice <code>partitionTrainingData</code> para especificar un porcentaje de datos de prueba por utilizar, o <code>useOtherTable</code> para utilizar un conjunto de datos de prueba desde una tabla específica de la base de datos. |
| <code>perc_training_data</code> | <i>número</i> | Si <code>prune_tree_options</code> se establece en <code>PercTrainingData</code> , especifica el porcentaje de datos que se utilizará para entrenamiento. |
| <code>prune_seed</code> | <i>entero</i> | Semilla aleatoria que se debe utilizar para replicar los resultados del análisis si <code>prune_tree_options</code> se establece en <code>PercTrainingData</code> ; el valor predeterminado es 1. |
| <code>pruning_table</code> | <i>cadena</i> | Nombre de tabla de un conjunto de datos de poda separado para estimar la precisión del modelo. |
| <code>compute_probabilities</code> | <i>flag</i> | Si es verdadero, especifica que las varianzas de las clases asignadas se deben incluir en el resultado. |

Regresión lineal de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezza.lineregressionnode`.

Tabla 217. Propiedades de `netezza.lineregressionnode`

| Propiedades de <code>netezza.lineregressionnode</code> | Valores | Descripción de la propiedad |
|--|-------------|---|
| <code>use_svd</code> | <i>flag</i> | Si es verdadero, utiliza la matriz de descomposición de valores singulares en lugar de la matriz original, para mayor velocidad y precisión numérica. |
| <code>include_intercept</code> | <i>flag</i> | Si es verdadero (valor predeterminado), aumenta la precisión global de la solución. |
| <code>calculate_model_diagnostics</code> | <i>flag</i> | Si es verdadero, calcula el diagnóstico del modelo. |

Series temporales de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo `netezza.timeseriesnode`.

Tabla 218. Propiedades de `netezza.timeseriesnode`

| Propiedades de <code>netezza.timeseriesnode</code> | Valores | Descripción de la propiedad |
|--|---|---|
| <code>time_points</code> | <i>campo</i> | El campo de entrada que contiene los valores de fecha u hora de la serie temporal. |
| <code>time_series_ids</code> | <i>campo</i> | El campo de entrada que contiene diversos ID de series temporales; utilice esta opción si la entrada contiene más de una serie temporal. |
| <code>model_table</code> | <i>campo</i> | Nombre de la tabla de base de datos en la que se guardará el modelo de series temporales de Netezza. |
| <code>description_table</code> | <i>campo</i> | Nombre de la tabla de entrada que contiene los nombres y las descripciones de las series temporales. |
| <code>seasonal_adjustment_table</code> | <i>campo</i> | Nombre de la tabla de salida en la que se guardarán los valores ajustados calculados por los algoritmos de suavizado exponencial o de descomposición de tendencia estacional. |
| <code>algorithm_name</code> | SpectralAnalysis o spectral ExponentialSmoothing o esmoothing ARIMA SeasonalTrendDecomposition o std | Algoritmo que hay que utilizar para el modelado de series temporales. |
| <code>trend_name</code> | N A DA M DM | Tipo de tendencia del suavizado exponencial: N - none A - aditivo DA -aditivo amortiguado M - multiplicativo DM - multiplicativo amortiguado |

Tabla 218. Propiedades de *netezatimeseriesnode* (continuación)

| Propiedades de <i>netezatimeseriesnode</i> | Valores | Descripción de la propiedad |
|--|---|--|
| seasonality_type | N A M | Tipo de estacionalidad del suavizado exponencial: N - none A - aditivo M - multiplicativo |
| interpolation_method | lineal cubicspline exponential spline | Método de interpolación que hay que utilizar. |
| timerange_setting | SD SP | Valor de rango de tiempo que se debe utilizar: SD - determinado por el sistema (utiliza el rango completo de datos de series de tiempo) SP - especificado por el usuario mediante <i>earliest_time</i> y <i>latest_time</i> |
| earliest_time | <i>entero</i> | Valores de inicio y finalización si <i>timerange_setting</i> es SP. El formato debe seguir el valor <i>time_points</i> . Por ejemplo, si el campo <i>time_points</i> contiene una fecha, éste también debería ser una fecha. Ejemplo: set NZ_DT1.timerange_setting = 'SP' set NZ_DT1.earliest_time = '1921-01-01' set NZ_DT1.latest_time = '2121-01-01' |
| latest_time | <i>fecha</i> <i>hora</i> <i>marca de tiempo</i> | |
| arima_setting | SD SP | Valor para el algoritmo ARIMA (sólo se utiliza si <i>algorithm_name</i> se establece en ARIMA): SD - determinado por el sistema SP - especificado por el usuario Si se utiliza <i>arima_setting</i> = SP, utilice los parámetros siguientes para establecer los valores estacionales y no estacionales. Ejemplo (solo no estacionales): set NZ_DT1.algorithm_name = 'arima' set NZ_DT1.arima_setting = 'SP' set NZ_DT1.p_symbol = 'lesseq' set NZ_DT1.p = '4' set NZ_DT1.d_symbol = 'lesseq' set NZ_DT1.d = '2' set NZ_DT1.q_symbol = 'lesseq' set NZ_DT1.q = '4' |

Tabla 218. Propiedades de `netezatimeseriesnode` (continuación)

| Propiedades de <code>netezatimeseriesnode</code> | Valores | Descripción de la propiedad |
|--|--|---|
| <code>p_symbol</code> | <code>sin</code> | ARIMA - operador para los parámetros <code>p</code> , <code>d</code> , <code>q</code> , <code>sp</code> , <code>sd</code> y <code>sq</code> : <code>less</code> - menor que <code>eq</code> - igual a <code>lesseq</code> - menor o igual que |
| <code>d_symbol</code> | <code>eq</code> | |
| <code>q_symbol</code> | <code>lesseq</code> | |
| <code>sp_symbol</code> | | |
| <code>sd_symbol</code> | | |
| <code>sq_symbol</code> | | |
| <code>p</code> | <i>entero</i> | ARIMA: grados no estacionales de autocorrelación. |
| <code>q</code> | <i>entero</i> | ARIMA: valor de derivación no estacional. |
| <code>d</code> | <i>entero</i> | ARIMA: número no estacional de órdenes de media móvil presentes en el modelo. |
| <code>sp</code> | <i>entero</i> | ARIMA: grados estacionales de autocorrelación. |
| <code>sq</code> | <i>entero</i> | ARIMA: valor de derivación estacional. |
| <code>sd</code> | <i>entero</i> | ARIMA: número estacional de órdenes de media móvil presentes en el modelo. |
| <code>advanced_setting</code> | SD SP | Determina cómo se manejan los valores avanzados: SD - determinado por el sistema SP - especificado por el usuario mediante <code>period</code> , <code>units_period</code> y <code>forecast_setting</code> . Ejemplo: <pre>set NZ_DT1.advanced_setting = 'SP' set NZ_DT1.period = 5 set NZ_DT1.units_period = 'd'</pre> |
| <code>punto</code> | <i>entero</i> | Longitud de ciclo estacional, especificado junto con <code>units_period</code> . No aplicable para análisis espectrales. |
| <code>units_period</code> | <code>ms</code> <code>s</code> <code>min</code> <code>h</code> <code>d</code> <code>semana</code> <code>q</code> <code>y</code> | Unidades en que se expresa <code>period</code> : <code>ms</code> - milisegundos <code>s</code> - segundos <code>min</code> - minutos <code>h</code> - horas <code>d</code> - días <code>wk</code> - semanas <code>q</code> - trimestres <code>y</code> - años Por ejemplo, para una serie temporal semanal utilice 1 para <code>period</code> y <code>wk</code> para <code>units_period</code> . |
| <code>forecast_setting</code> | <code>forecasthorizon</code> <code>forecasttimes</code> | Especifica cómo se han de realizar las previsiones. |

Tabla 218. Propiedades de *netezzatimeseriesnode* (continuación)

| Propiedades de <i>netezzatimeseriesnode</i> | Valores | Descripción de la propiedad |
|---|--|---|
| forecast_horizon | <i>entero</i> <i>fecha</i> <i>hora</i> <i>marca de tiempo</i> | Si <i>forecast_setting</i> = <i>forecasthorizon</i> , especifica el valor de punto final para la previsión. El formato debe seguir el valor <i>time_points</i> . Por ejemplo, si el campo <i>time_points</i> contiene una fecha, éste también debería ser una fecha. |
| forecast_times | <i>entero</i> <i>fecha</i> <i>hora</i> <i>marca de tiempo</i> | Si <i>forecast_setting</i> = <i>forecasttimes</i> , especifica los valores que se utilizarán para hacer previsiones. El formato debe seguir el valor <i>time_points</i> . Por ejemplo, si el campo <i>time_points</i> contiene una fecha, éste también debería ser una fecha. |
| include_history | <i>flag</i> | Indica si se deben incluir los valores históricos en los resultados. |
| include_interpolated_values | <i>flag</i> | Indica si se deben incluir los valores interpolados en los resultados. No es aplicable si <i>include_history</i> es <i>false</i> . |

Lineal generalizado de Netezza

Las siguientes propiedades están disponibles para los nodos del tipo *netezzaglmnode*.

Tabla 219. Propiedades de *netezzaglmnode*

| Propiedades de <i>netezzaglmnode</i> | Valores | Descripción de la propiedad |
|--------------------------------------|---|---|
| dist_family | <i>bernoulli</i> <i>de gauss</i> <i>poisson</i> <i>negativebinomial</i> <i>wald</i> <i>gamma</i> | El tipo de distribución. El valor predeterminado es <i>bernoulli</i> . |
| dist_params | <i>número</i> | Valor del parámetro de distribución que hay que utilizar. Sólo se aplica si <i>distribution</i> es <i>Negativebinomial</i> . |
| trials | <i>entero</i> | Sólo se aplica si <i>distribution</i> es <i>Binomial</i> . Cuando la respuesta objetivo es un número de eventos que tienen lugar en un conjunto de ensayos, el campo <i>target</i> contiene el número de eventos, y el campo <i>trials</i> contiene el número de ensayos. |

Tabla 219. Propiedades de netezzaglmnode (continuación)

| Propiedades de netezzaglmnode | Valores | Descripción de la propiedad |
|-------------------------------|---|--|
| model_table | campo | Nombre de la tabla de base de datos en la que se guardará el modelo lineal generalizado de Netezza. |
| maxit | entero | Número máximo de iteraciones que debe ejecutar el algoritmo; el valor predeterminado es 20. |
| eps | número | Valor del error máximo (en notación científica) en el que el algoritmo debería dejar de buscar el modelo de mejor ajuste. El valor predeterminado es -3, lo que significa 1E-3 o 0,001. |
| tol | número | El valor (en notación científica) por debajo del que los errores se tratan como si su valor fuera cero. El valor predeterminado es -7, lo que significa que los valores de error por debajo de 1E-7 (o 0,0000001) se cuentan como insignificantes. |
| link_func | identidad inverse invnegative invsquare sqrt power oddspower anotaciones clog loglog cloglog logit probit gaussit cauchit canbinom cangeom cannegbinom | Función de enlace que se ha de utilizar; el valor predeterminado es logit. |
| link_params | número | Valor del parámetro de función de enlace que hay que utilizar. Sólo se aplica si link_function es power u oddspower. |
| interaction | [[[nombrescol1],[niveles1]], [[nombrescol2],[niveles2]], ...[[nombrescolN],[nivelesN]],] | Especifica las interacciones entre los campos. colnames es una lista de campos de entrada, y level es siempre 0 para cada campo. Ejemplo: [[["K", "BP", "Sex", "K"], [0,0,0,0]], [["Age", "Na"], [0,0]]] |
| intercept | flag | Si es true, incluye la interceptación en el modelo. |

Propiedades de nugget de modelo de Netezza

Las siguientes propiedades son comunes a los nuggets del modelo de la base de datos de Netezza.

Tabla 220. Propiedades comunes de nugget de nodos de Netezza

| Propiedades comunes de nugget de modelo de Netezza | Valores | Descripción de la propiedad |
|--|---------|---|
| connection | cadena | La cadena de conexión para la base de datos de Netezza donde se almacena el modelo. |
| table_name | cadena | Nombre de la base de datos donde se almacenará el modelo. |

Otras las propiedades del nugget de modelo son las mismas que las del nodo de modelado correspondiente.

Los nombres de script de los nuggets de modelo son los siguientes.

Tabla 221. Nombres de script de nuggets de modelos Netezza

| Nugget de modelo | Nombre de script |
|---------------------|-------------------------------|
| Árbol de decisiones | applynetezadectreenode |
| K-medias | applynetezzakmeansnode |
| Red bayesiana | applynetezabayesnode |
| bayesiano ingenuo | applynetezanaivebayesnode |
| KNN | applynetezaknnnode |
| Clúster divisivo | applynetezadivclusternode |
| PCA | applynetezapcanode |
| Árbol de regresión | applynetezaregtreenode |
| Regresión lineal | applynetezalineregressionnode |
| Serie temporal | applynetezatimeseriesnode |
| Lineal generalizado | applynetezaglmnode |

Capítulo 16. Propiedades del nodo de resultados

Las propiedades de nodos de resultados se diferencian un poco de las de otros tipos de nodos. En lugar de hacer referencia a una opción determinada de nodo, las propiedades de nodos de resultados almacenan una referencia en el objeto de resultado. Esto resulta útil al tomar un valor de una tabla y establecerlo como un parámetro de ruta.

Esta sección describe las propiedades de scripts disponibles para los nodos de resultados.

propiedades de analysisnode



El nodo Análisis evalúa la capacidad de los modelos predictivos para generar predicciones precisas. Los nodos Análisis realizan varias comparaciones entre los valores predichos y los valores reales para uno o más nugget de modelo. También pueden comparar modelos predictivos entre sí.

Ejemplo

```
node = stream.create("analysis", "My node")
# Pestaña Análisis
node.setPropertyValue("coincidence", True)
node.setPropertyValue("performance", True)
node.setPropertyValue("confidence", True)
node.setPropertyValue("threshold", 75)
node.setPropertyValue("improve_accuracy", 3)
node.setPropertyValue("inc_user_measure", True)
# "Definir medida del usuario..."
node.setPropertyValue("user_if", "@TARGET = @PREDICTED")
node.setPropertyValue("user_then", "101")
node.setPropertyValue("user_else", "1")
node.setPropertyValue("user_compute", ["Mean", "Sum"])
node.setPropertyValue("by_fields", ["Drug"])
# Pestaña "Resultados"
node.setPropertyValue("output_format", "HTML")
node.setPropertyValue("full_filename", "C:/output/analysis_out.html")
```

Tabla 222. propiedades de analysisnode.

| Propiedad de analysisnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|--|---|
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| use_output_name | tag | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | cadena | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| output_format | Text (.txt) HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. |
| by_fields | lista | |

Tabla 222. propiedades de analysisnode (continuación).

| Propiedad de analysisnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|-------------------------|--|
| full_filename | cadena | Nombre del archivo de resultados, si se trata de resultados HTML, de datos o de disco. |
| coincidence | tag | |
| performance | tag | |
| evaluation_binary | flag | |
| confidence | tag | |
| umbral | number | |
| improve_accuracy | number | |
| inc_user_measure | tag | |
| user_if | expr | |
| user_then | expr | |
| user_else | expr | |
| user_compute | [Mean Sum Min Max SDev] | |

propiedades de dataauditnode



El nodo Auditoría de datos permite echar un primer vistazo exhaustivo a los datos, incluyendo estadísticos de resumen, histogramas y distribución para cada campo, así como información sobre valores atípicos, valores perdidos y extremos. Los resultados se muestran en una matriz fácil de leer que se puede ordenar y utilizar para generar nodos de preparación de datos y gráficos de tamaño completo.

Ejemplo

```

filenode = stream.createAt("variablefile", "File", 100, 100)
filenode.setPropertyValue("full_filename", "$CLEO_DEMOS/DRUG1n")
node = stream.createAt("dataaudit", "My node", 196, 100)
stream.link(filenode, node)
node.setPropertyValue("custom_fields", True)
node.setPropertyValue("fields", ["Age", "Na", "K"])
node.setPropertyValue("display_graphs", True)
node.setPropertyValue("basic_stats", True)
node.setPropertyValue("advanced_stats", True)
node.setPropertyValue("median_stats", False)
node.setPropertyValue("calculate", ["Count", "Breakdown"])
node.setPropertyValue("outlier_detection_method", "std")
node.setPropertyValue("outlier_detection_std_outlier", 1.0)
node.setPropertyValue("outlier_detection_std_extreme", 3.0)
node.setPropertyValue("output_mode", "Screen")

```

Tabla 223. propiedades de dataauditnode.

| Propiedad de dataauditnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---------------------|-----------------------------|
| custom_fields | tag | |
| campos | [campo1 ... campoN] | |
| overlay | campo | |

Tabla 223. propiedades de dataauditnode (continuación).

| Propiedad de dataauditnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|--|
| display_graphs | <i>flag</i> | Se utiliza para activar o desactivar la representación de gráficos en la matriz de resultados. |
| basic_stats | <i>tag</i> | |
| advanced_stats | <i>tag</i> | |
| median_stats | <i>tag</i> | |
| calculate | Count Breakdown | Se utiliza para calcular valores perdidos. Seleccione uno de los métodos de cálculo, ambos o ninguno. |
| outlier_detection_method | std iqr | Se utiliza para especificar el método de detección de valores atípicos y extremos. |
| outlier_detection_std_outlier | <i>number</i> | Si outlier_detection_method es std, especifica el número que se utilizará para definir los valores atípicos. |
| outlier_detection_std_extreme | <i>number</i> | Si outlier_detection_method es std, especifica el número que se utilizará para definir los valores extremos. |
| outlier_detection_iqr_outlier | <i>number</i> | Si outlier_detection_method es iqr, especifica el número que se utilizará para definir los valores atípicos. |
| outlier_detection_iqr_extreme | <i>number</i> | Si outlier_detection_method es iqr, especifica el número que se utilizará para definir los valores extremos. |
| use_output_name | <i>tag</i> | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | <i>cadena</i> | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | Formatted (.tab) Delimited (.csv) HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. |
| paginate_output | <i>flag</i> | Si output_format es HTML, los resultados se separarán por páginas. |
| lines_per_page | <i>número</i> | Si se usa con paginate_output, especifica las líneas por página del resultado. |
| full_filename | <i>cadena</i> | |

Propiedades de extensionoutputnode



El nodo de Resultados de extensión le permite analizar los datos y los resultados de la puntuación de modelo utilizando su propio script R o Python para Spark personalizado. El resultado del análisis puede ser en texto o en gráficos. El resultado se añade a la pestaña **Resultado** del panel de gestor. De forma alternativa, el resultado se puede redirigir a un archivo.

Ejemplo de Python para Spark

```
#### Ejemplo de script de Python para Spark
import modeler.api
stream = modeler.script.stream()
node = stream.create("extension_output", "extension_output")
node.setPropertyValue("syntax_type", "Python")

python_script = """
import json
import spss.pyspark.runtime

cxt = spss.pyspark.runtime.getContext()
df = cxt.getSparkInputData()
schema = df.dtypes[:]
print df
"""

node.setPropertyValue("python_syntax", python_script)
```

Ejemplo de R

```
#### Ejemplo de script de R
node.setPropertyValue("syntax_type", "R")
node.setPropertyValue("r_syntax", "print(modelerData$Age)")
```

Tabla 224. Propiedades de extensionoutputnode

| Propiedades de extensionoutputnode | Tipo de datos | Descripción de la propiedad |
|------------------------------------|------------------------------------|--|
| syntax_type | R Python | Especifica qué script se ejecuta: R o Python (R es el valor predeterminado). |
| r_syntax | string | Sintaxis de scripts R para la puntuación del modelo. |
| python_syntax | string | Sintaxis de scripts Python para la puntuación de modelo. |
| convert_flags | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| convert_missing | flag | Opción para convertir valores perdidos al valor NA de R. |
| convert_datetime | flag | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| convert_datetime_class | POSIXct POSIXlt | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |

Tabla 224. Propiedades de *extensionoutputnode* (continuación)

| Propiedades de <i>extensionoutputnode</i> | Tipo de datos | Descripción de la propiedad |
|---|---------------------|--|
| output_to | Screen File | Especifique el tipo de salida (Screen o File). |
| output_type | Graph Text | Especifique si se debe producir salida gráfica o de texto. |
| full_filename | <i>string</i> | Nombre de archivo a utilizar para la salida generada. |
| graph_file_type | HTML COU | Tipo de archivo para el archivo de salida (.html o .cou). |
| text_file_type | HTML TEXT COU | Especifique el tipo de archivo para la salida de texto (.html, .txt o .cou). |

propiedades de *matrixnode*



El nodo Matriz crea una tabla que muestra las relaciones entre campos. Se suele utilizar normalmente para mostrar las relaciones entre dos campos simbólicos, pero también puede mostrar relaciones entre campos de marcas o entre campos numéricos.

Ejemplo

```
node = stream.create("matrix", "My node")
# Pestaña "Configuración"
node.setPropertyValue("fields", "Numerics")
node.setPropertyValue("row", "K")
node.setPropertyValue("column", "Na")
node.setPropertyValue("cell_contents", "Function")
node.setPropertyValue("function_field", "Age")
node.setPropertyValue("function", "Sum")
# Pestaña "Aspecto"
node.setPropertyValue("sort_mode", "Ascending")
node.setPropertyValue("highlight_top", 1)
node.setPropertyValue("highlight_bottom", 5)
node.setPropertyValue("display", ["Counts", "Expected", "Residuals"])
node.setPropertyValue("include_totals", True)
# Pestaña "Resultados"
node.setPropertyValue("full_filename", "C:/output/matrix_output.html")
node.setPropertyValue("output_format", "HTML")
node.setPropertyValue("paginate_output", True)
node.setPropertyValue("lines_per_page", 50)
```

Tabla 225. propiedades de *matrixnode*.

| Propiedades de <i>matrixnode</i> | Tipo de datos | Descripción de la propiedad |
|----------------------------------|---|-----------------------------|
| campos | Seleccionado Flags Caracteres numéricos | |
| row | <i>campo</i> | |
| column | <i>campo</i> | |

Tabla 225. propiedades de matrixnode (continuación).

| Propiedades de matrixnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|---|--|
| include_missing_values | tag | Determina si los valores perdidos por el sistema (nulos) o no especificados por el usuario (vacíos) se incluyen en los resultados de fila y columna. |
| cell_contents | CrossTabs Función | |
| function_field | cadena | |
| function | Sum Media Mín Máx SDev | |
| sort_mode | Unsorted Ascending Descending | |
| highlight_top | number | Si no es cero, es verdadero. |
| highlight_bottom | number | Si no es cero, es verdadero. |
| display | [Counts Expected Residuos RowPct ColumnPct TotalPct] | |
| include_totals | tag | |
| use_output_name | tag | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | cadena | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | Formatted (.tab) Delimited (.csv) HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. Los formatos Formatted y Delimited aceptan el modificador transposed, mediante el cual se transponen filas y columnas en la tabla. |
| paginate_output | flag | Si output_format es HTML, los resultados se separarán por páginas. |
| lines_per_page | número | Si se usa con paginate_output, especifica las líneas por página del resultado. |
| full_filename | cadena | |

propiedades de meansnode



El nodo Medias compara las medias de grupos independientes o de pares de campos relacionados para probar si existen diferencias significativas. Por ejemplo, puede comparar los ingresos medios antes y después de poner en marcha una promoción o comparar los ingresos de los clientes que no recibieron esa promoción con los que sí lo hicieron.

Ejemplo

```
node = stream.create("means", "My node")
node.setPropertyValue("means_mode", "BetweenFields")
node.setPropertyValue("paired_fields", [["OPEN_BAL", "CURR_BAL"]])
node.setPropertyValue("label_correlations", True)
node.setPropertyValue("output_view", "Advanced")
node.setPropertyValue("output_mode", "File")
node.setPropertyValue("output_format", "HTML")
node.setPropertyValue("full_filename", "C:/output/means_output.html")
```

Tabla 226. propiedades de meansnode.

| Propiedad de meansnode | Tipo de datos | Descripción de la propiedad |
|--------------------------|---|--|
| means_mode | BetweenGroups BetweenFields | Especifica el tipo de estadístico de las medias que se va a ejecutar en los datos. |
| test_fields | [field1 ... fieldn] | Especifica el campo de prueba si means_mode se establece en BetweenGroups. |
| grouping_field | campo | Especifica el campo de agrupación. |
| paired_fields | [[field1 field2] [campo3 campo4] ...] | Especifica los pares de campos que se usan si means_mode se establece en BetweenFields. |
| label_correlations | tag | Determina si las etiquetas de correlación se muestran en el resultado. Esta configuración se aplica únicamente si means_mode se establece en BetweenFields. |
| correlation_mode | Probability Absolute | Determina si las correlaciones deben etiquetarse según la probabilidad o según el valor absoluto. |
| weak_label | cadena | |
| medium_label | cadena | |
| strong_label | cadena | |
| weak_below_probability | number | Si correlation_mode se establece en Probability, determina el valor de corte para las correlaciones débiles. Debe tratarse de un valor comprendido entre 0 y 1; por ejemplo, 0,90. |
| strong_above_probability | number | Valor de corte para correlaciones fuertes. |

Tabla 226. propiedades de meansnode (continuación).

| Propiedad de meansnode | Tipo de datos | Descripción de la propiedad |
|------------------------|---|--|
| weak_below_absolute | number | Si correlation_mode se establece en Absolute, especifica el valor de corte para las correlaciones débiles. Debe tratarse de un valor comprendido entre 0 y 1; por ejemplo, 0,90. |
| strong_above_absolute | number | Valor de corte para correlaciones fuertes. |
| unimportant_label | cadena | |
| marginal_label | cadena | |
| important_label | cadena | |
| unimportant_below | number | Valor de corte para una importancia del campo baja. Debe tratarse de un valor comprendido entre 0 y 1; por ejemplo, 0,90. |
| important_above | number | |
| use_output_name | tag | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | cadena | Nombre que se va a usar. |
| output_mode | Screen File | Especifica la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | Formatted (.tab) Delimited (.csv) HTML (.html) Output (.cou) | Especifica el tipo de resultado. |
| full_filename | cadena | |
| output_view | Simple Advanced | Determina si el resultado muestra la vista simple o la avanzada. |

propiedades de reportnode



El nodo Informe crea informes con formato que contienen texto fijo, así como datos y otras expresiones derivadas de los datos. Puede especificar el formato del informe utilizando plantillas de texto para definir el texto fijo y las construcciones de resultados de datos. Puede proporcionar formato de texto personalizado utilizando etiquetas HTML de la plantilla y configurando opciones en la pestaña Resultado. Puede incluir valores de datos y otros resultados condicionales mediante el uso de expresiones CLEM en la plantilla.

Ejemplo

```
node = stream.create("report", "My node")
node.setPropertyValue("output_format", "HTML")
node.setPropertyValue("full_filename", "C:/report_output.html")
node.setPropertyValue("lines_per_page", 50)
node.setPropertyValue("title", "Report node created by a script")
node.setPropertyValue("highlights", False)
```

Tabla 227. propiedades de reportnode.

| Propiedad de reportnode | Tipo de datos | Descripción de la propiedad |
|-------------------------|--|---|
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | HTML (.html) Text (.txt) Output (.cou) | Se utiliza para especificar el tipo de salida de archivo. |
| format | Automático Personalizado | Se utiliza para elegir si la salida se formatea automáticamente o si se formatea utilizando el HTML incluido en la plantilla. Para utilizar el formato HTML en la plantilla, especifique Personalizado. |
| use_output_name | tag | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | cadena | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| text | cadena | |
| full_filename | cadena | |
| highlights | tag | |
| título | cadena | |
| lines_per_page | number | |

propiedades de routputnode



El nodo Routput permite analizar datos y resultados de la puntuación del modelo utilizando su propio script R personalizado. El resultado del análisis puede ser en texto o en gráficos. El resultado se añade a la pestaña **Resultado** del panel de gestor. De forma alternativa, el resultado se puede redirigir a un archivo.

Tabla 228. propiedades de routputnode

| Propiedades de routputnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|------------------------------------|-----------------------------|
| syntax | cadena | |
| convert_flags | StringsAndDoubles LogicalValues | |
| convert_datetime | flag | |
| convert_datetime_class | POSIXct POSIXlt | |
| convert_missing | flag | |
| output_name | Auto Custom | |
| custom_name | cadena | |

Tabla 228. propiedades de routputnode (continuación)

| Propiedades de routputnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|---------------------|-----------------------------|
| output_to | Screen File | |
| output_type | Graph Text | |
| full_filename | <i>cadena</i> | |
| graph_file_type | HTML COU | |
| text_file_type | HTML TEXT COU | |

propiedades de setglobalsnode



El nodo Val. globales explora los datos y calcula los valores de resumen que se pueden utilizar en expresiones CLEM. Por ejemplo, puede utilizar este nodo para calcular estadísticos para un campo denominado *edad* y, a continuación, utilizar la media global de *edad* en expresiones CLEM insertando la función @GLOBAL_MEAN(*edad*).

Ejemplo

```
node = stream.create("setglobals", "My node")
node.setKeyedPropertyValue("globals", "Na", ["Max", "Sum", "Mean"])
node.setKeyedPropertyValue("globals", "K", ["Max", "Sum", "Mean"])
node.setKeyedPropertyValue("globals", "Age", ["Max", "Sum", "Mean", "SDev"])
node.setPropertyValue("clear_first", False)
node.setPropertyValue("show_preview", True)
```

Tabla 229. propiedades de setglobalsnode.

| Propiedad de setglobalsnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|----------------------------|---|
| globals | [Sum Mean Min Max SDev] | Propiedad estructurada en la que los campos que se van a establecer deben hacer referencia a la sintaxis siguiente: node.setKeyedPropertyValue("globals", "Age", ["Max", "Sum", "Mean", "SDev"]) |
| clear_first | <i>tag</i> | |
| show_preview | <i>tag</i> | |

propiedades de simevalnode



El nodo de evaluación de simulación evalúa un campo de destino predicho y presenta información sobre la distribución y correlación del campo de destino.

Tabla 230. propiedades de simevalnode.

| propiedades de simevalnode | Tipo de datos | Descripción de la propiedad |
|----------------------------|-----------------------------------|-----------------------------|
| target | campo | |
| iteration | campo | |
| presorted_by_iteration | booleano | |
| max_iterations | número | |
| tornado_fields | [field1...fieldN] | |
| plot_pdf | booleano | |
| plot_cdf | booleano | |
| show_ref_mean | booleano | |
| show_ref_median | booleano | |
| show_ref_sigma | booleano | |
| num_ref_sigma | número | |
| show_ref_pct | booleano | |
| ref_pct_bottom | número | |
| ref_pct_top | número | |
| show_ref_custom | booleano | |
| ref_custom_values | [number1...numberN] | |
| category_values | Category Probabilities Both | |
| category_groups | Categories Iterations | |
| create_pct_table | booleano | |
| pct_table | Quartiles Intervals Custom | |
| pct_intervals_num | número | |
| pct_custom_values | [number1...numberN] | |

propiedades de simfitnode



El nodo de simulación de ajuste examina la distribución de las estadísticas de los datos de cada campo y genera (o actualiza) un nodo de generación de simulación, con la mejor distribución de ajuste asignada a cada campo. El nodo de generación de simulación se puede utilizar, a continuación, para generar datos simulados.

Tabla 231. propiedades de simfitnode.

| propiedades de simfitnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|---------------------------|--|
| build | Nodo XMLExport Both | |
| use_source_node_name | booleano | |
| source_node_name | cadena | El nombre personalizado del nodo de origen que se está generando o actualizando. |

Tabla 231. propiedades de simfitnode (continuación).

| propiedades de simfitnode | Tipo de datos | Descripción de la propiedad |
|---------------------------|--------------------------------------|-----------------------------|
| use_cases | All LimitFirstN | |
| use_case_limit | entero | |
| fit_criterion | AndersonDarling KolmogorovSmirnov | |
| num_bins | entero | |
| parameter_xml_filename | cadena | |
| generate_parameter_import | booleano | |

propiedades de statisticsnode



El nodo Estadísticos ofrece información básica de resumen acerca de los campos numéricos. Calcula estadísticos de resumen para campos individuales y correlaciones entre campos.

Ejemplo

```
node = stream.create("statistics", "My node")
# Pestaña "Configuración"
node.setPropertyValue("examine", ["Age", "BP", "Drug"])
node.setPropertyValue("statistics", ["mean", "sum", "sdev"])
node.setPropertyValue("correlate", ["BP", "Drug"])
# "Etiquetas de correlación..." section
node.setPropertyValue("label_correlations", True)
node.setPropertyValue("weak_below_absolute", 0.25)
node.setPropertyValue("weak_label", "lower quartile")
node.setPropertyValue("strong_above_absolute", 0.75)
node.setPropertyValue("medium_label", "middle quartiles")
node.setPropertyValue("strong_label", "upper quartile")
# Pestaña "Resultados"
node.setPropertyValue("full_filename", "c:/output/statistics_output.html")
node.setPropertyValue("output_format", "HTML")
```

Tabla 232. propiedades de statisticsnode.

| Propiedad de statisticsnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------|--|---|
| use_output_name | tag | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | cadena | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | Text (.txt) HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. |
| full_filename | cadena | |

Tabla 232. propiedades de *statisticsnode* (continuación).

| Propiedad de <i>statisticsnode</i> | Tipo de datos | Descripción de la propiedad |
|------------------------------------|---|---|
| examine | <i>lista</i> | |
| correlate | <i>lista</i> | |
| statistics | [count mean sum min max range variance sdev semean median mode] | |
| correlation_mode | Probability Absolute | Determina si las correlaciones deben etiquetarse según la probabilidad o según el valor absoluto. |
| label_correlations | <i>tag</i> | |
| weak_label | <i>cadena</i> | |
| medium_label | <i>cadena</i> | |
| strong_label | <i>cadena</i> | |
| weak_below_probability | <i>number</i> | Si <i>correlation_mode</i> se establece en Probability, determina el valor de corte para las correlaciones débiles. Debe tratarse de un valor comprendido entre 0 y 1; por ejemplo, 0,90. |
| strong_above_probability | <i>number</i> | Valor de corte para correlaciones fuertes. |
| weak_below_absolute | <i>number</i> | Si <i>correlation_mode</i> se establece en Absolute, especifica el valor de corte para las correlaciones débiles. Debe tratarse de un valor comprendido entre 0 y 1; por ejemplo, 0,90. |
| strong_above_absolute | <i>number</i> | Valor de corte para correlaciones fuertes. |

Propiedades de *statisticsoutputnode*



El nodo Resultados de Statistics le permite llamar a un procedimiento de IBM SPSS Statistics para analizar los datos de IBM SPSS Modeler. Se puede acceder a una gran variedad de procedimientos analíticos de IBM SPSS Statistics. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Las propiedades de este nodo están descritas en “Propiedades de *statisticsoutputnode*” en la página 342.

propiedades de *tablenode*



El nodo Tabla muestra los datos en formato de tabla, que también se puede escribir en un archivo. Esto es útil en cualquier momento en que necesite inspeccionar sus valores de datos o exportarlos en un formato fácilmente legible.

Ejemplo

```
node = stream.create("table", "My node")
node.setPropertyValue("highlight_expr", "Age > 30")
node.setPropertyValue("output_format", "HTML")
node.setPropertyValue("transpose_data", True)
node.setPropertyValue("full_filename", "C:/output/table_output.htm")
node.setPropertyValue("paginate_output", True)
node.setPropertyValue("lines_per_page", 50)
```

Tabla 233. propiedades de *tablenode*.

| Propiedad de <i>tablenode</i> | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|---|
| full_filename | <i>cadena</i> | Nombre del archivo de resultados, si se trata de resultados HTML, de datos o de disco. |
| use_output_name | <i>tag</i> | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | <i>cadena</i> | Si <i>use_output_name</i> es verdadero, especifica el nombre que se va a utilizar. |
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | Formatted (.tab) Delimited (.csv) HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. |
| transpose_data | <i>tag</i> | Transpone los datos antes de exportarlos de manera que las filas representan campos y las columnas, registros. |
| paginate_output | <i>flag</i> | Si <i>output_format</i> es HTML, los resultados se separarán por páginas. |
| lines_per_page | <i>número</i> | Si se usa con <i>paginate_output</i> , especifica las líneas por página del resultado. |
| highlight_expr | <i>cadena</i> | |
| output | <i>cadena</i> | Propiedad de sólo lectura que mantiene una referencia en la última tabla creada por el nodo. |
| value_labels | [[Valor CadenaEtiqueta] [Valor CadenaEtiqueta] ...] | Se utiliza para especificar etiquetas para los pares de valores. |
| display_places | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se muestra (sólo se aplica a campos con almacenamiento REAL). Un valor de -1 utilizará el valor predeterminado de la ruta. |
| export_places | <i>entero</i> | Establece el número de cifras decimales para el campo cuando se exporta (sólo se aplica a campos con almacenamiento REAL). Un valor de -1 utilizará el valor predeterminado de la ruta. |
| decimal_separator | DEFAULT PERIOD COMMA | Establece el separador decimal para el campo (sólo se aplica a campos con almacenamiento REAL). |

Tabla 233. propiedades de tablenode (continuación).

| Propiedad de tablenode | Tipo de datos | Descripción de la propiedad |
|------------------------|---|---|
| date_format | "DDMAA" "MDDYY" "AAMDD" "YYMDD" "YYYYDDD" DAY MONTH "DD-MM-YY" "DD-MM-YYYY" "MM-DD-YY" "MM-DD-AAAA" "DD-MES-YY" "DD-MES-YYYY" "YYYY-MM-DD" "DD.MM.YY" "DD.MM.AAAA" "MM.DD.YYYY" "DD.MES.YY" "DD.MES.YYYY" "DD/MM/YY" "DD/MM/AAAA" "MM/DD/YY" "MM/DD/YYYY" "DD/MES/YY" "DD/MES/YYYY" MON YYYY q Q YYYY ww WK YYYY | Establece el formato de fecha para el campo (sólo se aplica a campos con almacenamiento FECHA o MARCADETIEMPO). |
| time_format | "HHMMSS" "HHMM" "MMSS" "HH:MM:SS" "HH:MM" "MM:SS" "(H)H:(M)M:(S)S" "(H)H:(M)M" "(M)M:(S)S" "HH.MM.SS" "HH.MM" "MM.SS" "(H)H.(M)M.(S)S" "(H)H.(M)M" "(M)M.(S)S" | Establece el formato de hora para el campo (sólo se aplica a campos con almacenamiento TIME o TIMESTAMP). |
| column_width | entero | Establece el ancho de columna para el campo. Un valor de -1 establecerá el ancho de columna en Auto. |
| justify | AUTO CENTER LEFT RIGHT | Establece la justificación de columna para el campo. |

propiedades de transformnode



El nodo Transformación permite seleccionar y previsualizar los resultados de las transformaciones antes de aplicarlas a los campos seleccionados.

Ejemplo

```
node = stream.create("transform", "My node")
node.setPropertyValue("fields", ["AGE", "INCOME"])
node.setPropertyValue("formula", "Select")
node.setPropertyValue("formula_log_n", True)
node.setPropertyValue("formula_log_n_offset", 1)
```

Tabla 234. propiedades de transformnode.

| propiedades de transformnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|-------------------------------------|--|
| campos | [<i>field1</i> ... <i>fieldn</i>] | Campos que se utilizarán en la transformación. |
| formula | Todos Select | Indica si se deben calcular todas las transformaciones o sólo las seleccionadas. |
| formula_inverse | <i>flag</i> | Indica si se debe utilizar la transformación inversa. |
| formula_inverse_offset | <i>número</i> | Indica el desplazamiento de los datos que se utilizará en la fórmula. De forma predeterminada es 0, a menos que el usuario especifique un valor. |
| formula_log_n | <i>flag</i> | Indica si debe utilizarse la transformación \log_n . |
| formula_log_n_offset | <i>número</i> | |
| formula_log_10 | <i>flag</i> | Indica si debe utilizarse la transformación \log_{10} . |
| formula_log_10_offset | <i>número</i> | |
| formula_exponential | <i>flag</i> | Indica si se debe utilizar la transformación exponencial (e^x). |
| formula_square_root | <i>flag</i> | Indica si se debe utilizar la transformación de raíz cuadrada. |
| use_output_name | <i>flag</i> | Especifica si se utiliza un nombre de resultado personalizado. |
| output_name | <i>string</i> | Si use_output_name es verdadero, especifica el nombre que se va a utilizar. |
| output_mode | Screen File | Se utiliza para especificar la ubicación objetivo para el resultado generado desde el nodo de resultados. |
| output_format | HTML (.html) Output (.cou) | Se utiliza para especificar el tipo de resultado. |

Tabla 234. propiedades de transformnode (continuación).

| propiedades de transformnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------|--|
| paginate_output | <i>flag</i> | Si output_format es HTML, los resultados se separarán por páginas. |
| lines_per_page | <i>número</i> | Si se usa con paginate_output, especifica las líneas por página del resultado. |
| full_filename | <i>cadena</i> | Indica el nombre de archivo que se utilizará para el resultado de archivo. |

Capítulo 17. Propiedades de nodos Exportar

Propiedades de nodos Exportar comunes

Las siguientes propiedades son comunes a todos los nodos de exportación:

Tabla 235. Propiedades comunes de nodos de exportación

| Propiedad | Valores | Descripción de la propiedad |
|------------------------|-------------------------|---|
| publish_path | <i>cadena</i> | Introduzca el nombre raíz que se utilizará para la imagen publicada y los archivos de parámetros. |
| publish_metadata | <i>flag</i> | Especifica si un archivo de metadatos se crea y que describe las entradas y los resultados de la imagen y sus modelos de datos. |
| publish_use_parameters | <i>flag</i> | Especifica si se incluyen parámetros de ruta en el archivo *.par. |
| publish_parameters | <i>lista de cadenas</i> | Especifica los parámetros que se van a incluir. |
| execute_mode | export_data publish | Especifica si el nodo se ejecuta sin publicar la ruta, o si la ruta se publica automáticamente cuando se ejecuta el nodo. |

Propiedades de asexport

La exportación de Analytic Server permite ejecutar una ruta en el sistema de archivos distribuido de Hadoop (HDFS).

Ejemplo

```
node.setPropertyValue("use_default_as", False)
node.setPropertyValue("connection",
["false", "9.119.141.141", "9080", "analyticserver", "ibm", "admin", "admin", "false", "", "", "", ""])
```

Tabla 236. Propiedades de asexport.

| Propiedades de asexport | Tipo de datos | Descripción de la propiedad |
|-------------------------|-----------------|---|
| data_source | <i>cadena</i> | Nombre del origen de datos. |
| export_mode | <i>cadena</i> | Especifica si se añaden (append) los datos exportados al origen de datos existente o si se sobrescriben (overwrite) al origen de datos existente. |
| use_default_as | <i>booleano</i> | Si se establece en True, utiliza la conexión de Analytic Server predeterminada configurada en el archivo options.cfg de servidor. Si se establece en False, utiliza la conexión de este nodo. |

Tabla 236. Propiedades de asexport (continuación).

| Propiedades de asexport | Tipo de datos | Descripción de la propiedad |
|-------------------------|---|---|
| connection | ["string","string","string", "string","string","string","string", "string","string","string", "string", ,"string"] | Una propiedad de la lista que contiene los detalles de conexión de Analytic Server. El formato es: ["is_secure_connect", "server_url", "server_port", "context_root", "consumer", "user_name", "password", "use-kerberos-auth", "kerberos-krb5-config-file-path", "kerberos-jaas-config-file-path", "kerberos-krb5-service-principal-name", "enable-kerberos-debug"] Donde: is_secure_connect: indica si se utiliza la conexión segura y es true o false. use-kerberos-auth: indica si se utiliza autenticación Kerberos y es true o false. enable-kerberos-debug: indica si se utiliza el modo de depuración de la autenticación Kerberos y es true o false. |

Propiedades del nodo de exportación Cognos



El nodo Exportar de IBM Cognos exporta datos en un formato que pueden leer las bases de datos de IBM Cognos.

Para este nodo, debe definir una conexión de Cognos y una conexión ODBC.

Conexión de Cognos

Las propiedades de la conexión de Cognos son las siguientes.

Tabla 237. Propiedades del nodo de exportación Cognos

| Propiedades del nodo de exportación Cognos | Tipo de datos | Descripción de la propiedad |
|--|---|---|
| cognos_connection | [<i>"cadena"</i> , <i>"distintivo"</i> , <i>"cadena"</i> , <i>"cadena"</i> , <i>"cadena"</i>] | <p>Una propiedad de la lista que contiene los detalles de conexión para el servidor de Cognos. El formato es: ["Cognos_server_URL", login_mode, "namespace", "username", "password"]</p> <p>donde: Cognos_server_URL es la URL del servidor de Cognos que contiene el origen. login_mode indica si se utiliza el inicio de sesión anónimo, y es true o false; si se establece en true, los campos siguientes deben establecerse en "". namespace especifica el proveedor de autenticación de seguridad utilizado para registrarse en el servidor. username y password son los utilizados para registrarse en el servidor de Cognos.</p> <p>En lugar de login_mode, también hay disponibles las modalidades siguientes:</p> <ul style="list-style-type: none"> anonymousMode. Por ejemplo: ['Cognos_server_url', 'anonymousMode', "namespace", "username", "password"] credentialMode. Por ejemplo: ['Cognos_server_url', 'credentialMode', "namespace", "username", "password"] storedCredentialMode. Por ejemplo: ['Cognos_server_url', 'storedCredentialMode', "stored_credential_name"] <p>Donde stored_credential_name es el nombre de una credencial de Cognos del repositorio.</p> |
| cognos_package_name | <i>cadena</i> | La ruta y el nombre del paquete de Cognos al que está exportando datos, por ejemplo: /Public Folders/MyPackage |
| cognos_datasource | <i>cadena</i> | |
| cognos_export_mode | Publicar ExportFile | |
| cognos_filename | <i>cadena</i> | |

Conexión ODBC

Las propiedades de la conexión ODBC son idénticas a las indicadas para `databaseexportnode` en la sección siguiente, a excepción de la propiedad `datasource`, que no es válida.

propiedades de `databaseexportnode`



El nodo Exportar base de datos escribe datos en orígenes de datos relacionales compatibles con ODBC. Para escribir en un origen de datos ODBC, el origen de datos debe existir y debe tener permiso para escribir en él.

Ejemplo

```
...
Se asume que se ha configurado un origen de datos denominado "Miorigendedatos"
...
stream = modeler.script.stream()
db_exportnode = stream.createAt("databaseexport", "DB Export", 200, 200)
applynn = stream.findByType("applyneuralnetwork", None)
stream.link(applynn, db_exportnode)

# pestaña Exportar
db_exportnode.setPropertyValue("username", "user")
db_exportnode.setPropertyValue("datasource", "MyDatasource")
db_exportnode.setPropertyValue("password", "password")
db_exportnode.setPropertyValue("table_name", "predictions")
db_exportnode.setPropertyValue("write_mode", "Create")
db_exportnode.setPropertyValue("generate_import", True)
db_exportnode.setPropertyValue("drop_existing_table", True)
db_exportnode.setPropertyValue("delete_existing_rows", True)
db_exportnode.setPropertyValue("default_string_size", 32)

# Cuadro de diálogo Esquema
db_exportnode.setKeyedPropertyValue("type", "region", "VARCHAR(10)")
db_exportnode.setKeyedPropertyValue("export_db_primarykey", "id", True)
db_exportnode.setPropertyValue("use_custom_create_table_command", True)
db_exportnode.setPropertyValue("custom_create_table_command", "My SQL Code")

# Cuadro de diálogo Índices
db_exportnode.setPropertyValue("use_custom_create_index_command", True)
db_exportnode.setPropertyValue("custom_create_index_command", "CREATE BITMAP INDEX <index-name>
ON <table-name> <(index-columns)>")
db_exportnode.setKeyedPropertyValue("indexes", "MYINDEX", [{"fields", ["id", "region"]}]
```

Tabla 238. propiedades de `databaseexportnode`.

| Propiedades de <code>databaseexportnode</code> | Tipo de datos | Descripción de la propiedad |
|--|---------------|-----------------------------|
| <code>datasource</code> | <i>cadena</i> | |
| nombre de usuario | <i>cadena</i> | |
| contraseña | <i>cadena</i> | |

Tabla 238. propiedades de databaseexportnode (continuación).

| Propiedades de databaseexportnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|----------------------------|--|
| epassword | <i>cadena</i> | Este intervalo es de sólo lectura durante la ejecución. Para generar una contraseña codificada, utilice la herramienta Contraseña disponible del menú Herramientas. Consulte el tema “Generación de una contraseña codificada” en la página 52 para obtener más información. |
| table_name | <i>cadena</i> | |
| write_mode | Create Append Fundir | |
| map | <i>cadena</i> | Correlaciona un nombre de campo de ruta a un nombre de columna de la base de datos (sólo es válido si write_mode es Merge). Para una fusión, todos los campos se deben estar correlacionados para que se exporten. Los nombres de campos que no existen en la base de datos se añaden como nuevas columnas. |
| key_fields | <i>lista</i> | Especifica el campo de ruta que se utiliza para la clave; la propiedad map muestra los elementos que se corresponden con la base de datos. |
| join | Base de datos Añadir | |
| drop_existing_table | <i>tag</i> | |
| delete_existing_rows | <i>tag</i> | |
| default_string_size | <i>entero</i> | |
| type | | Propiedad estructurada que se utiliza para establecer el tipo de esquema. |
| generate_import | <i>tag</i> | |
| use_custom_create_table_command | <i>tag</i> | Utilice el intervalo <i>custom_create_table</i> para modificar el comando de SQL estándar CREATE TABLE. |
| custom_create_table_command | <i>cadena</i> | Especifica el comando de cadena a utilizar en lugar del comando de SQL estándar CREATE TABLE. |
| use_batch | <i>tag</i> | Las siguientes propiedades son opciones avanzadas para la carga masiva de la base de datos. Un valor verdadero para Use_batch desactiva fila a fila las confirmaciones en la base de datos. |
| batch_size | <i>number</i> | Especifica el número de registros para enviar a la base de datos antes de confirmar en la memoria. |

Tabla 238. propiedades de databaseexportnode (continuación).

| Propiedades de databaseexportnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|-----------------------------|---|
| bulk_loading | Off ODBC External | Especifica el tipo de carga masiva. A continuación se muestran las opciones adicionales para ODBC y External. |
| not_logged | <i>flag</i> | |
| odbc_binding | Row Columna | Especifique el enlace a lo largo de las filas o de las columnas para la carga masiva a través de ODBC. |
| loader_delimit_mode | Tabulador Space Other | Especifique el tipo de delimitador para la carga masiva a través de un programa externo. Seleccione Other junto con la propiedad loader_other_delimiter para especificar los delimitadores, como la coma (,). |
| loader_other_delimiter | <i>cadena</i> | |
| specify_data_file | <i>flag</i> | Una marca verdadera activa la siguiente propiedad data_file, en la que puede especificar el nombre de archivo y la ruta de acceso en la que se va a escribir al realizar la carga masiva en la base de datos. |
| data_file | <i>cadena</i> | |
| specify_loader_program | <i>flag</i> | Una marca verdadera activa la siguiente propiedad loader_program, en la que puede especificar el nombre y la ubicación de un programa o script del cargador externo. |
| loader_program | <i>cadena</i> | |
| gen_logfile | <i>flag</i> | Una marca verdadera activa la siguiente propiedad logfile_name, en la que puede especificar el nombre de un archivo en el servidor para generar un registro de errores. |
| logfile_name | <i>cadena</i> | |
| check_table_size | <i>flag</i> | Una marca verdadera permite la comprobación de la tabla para garantizar que el aumento del tamaño de la tabla de la base de datos se corresponde con el número de filas exportadas desde IBM SPSS Modeler. |
| loader_options | <i>cadena</i> | Especifique los argumentos adicionales, como -comment y -specialdir, en el programa cargador. |
| export_db_primarykey | <i>tag</i> | Determina si el campo especificado es una clave primaria. |
| use_custom_create_index_command | <i>tag</i> | Si se establece true, se activa SQL personalizado para todos los índices. |

Tabla 238. propiedades de databaseexportnode (continuación).

| Propiedades de databaseexportnode | Tipo de datos | Descripción de la propiedad |
|---|---------------|--|
| custom_create_index_command | cadena | Especifica el comando de SQL empleado para crear índices cuando SQL personalizado está activado. (Este valor puede anularse para determinados índices, tal y como se indica a continuación.) |
| indexes.INDEXNAME.fields | | Crea el índice especificado si procede y enumera los nombres de campos que se van a incluir en él. |
| INDEXNAME "use_custom_create_index_command" | tag | Se usa para activar o desactivar el SQL personalizado para un índice específico. Consulte los ejemplos situados a continuación de la siguiente tabla. |
| INDEXNAME "custom_create_index_command" | cadena | Especifica el SQL personalizado que se usa para el índice específico. Consulte los ejemplos situados a continuación de la siguiente tabla. |
| indexes.INDEXNAME.remove | tag | Si se establece True, se elimina el índice específico del grupo de índices. |
| table_space | cadena | Especifica el espacio de tabla que se creará. |
| use_partition | tag | Especifica que se utilizará el campo Distribuir Hash. |
| partition_field | cadena | Especifica el contenido del campo Distribuir Hash. |

Nota: Para algunas bases de datos, puede especificar que se crearán tablas de bases de datos para la exportación con compresión (por ejemplo, el equivalente a CREATE TABLE MYTABLE (...) COMPRESS YES; en SQL). Las propiedades use_compression y compression_mode se proporcionan para dar soporte a esta característica, como se indica a continuación.

Tabla 239. Propiedades de databaseexportnode utilizando funciones de compresión.

| Propiedades de databaseexportnode | Tipo de datos | Descripción de la propiedad |
|-----------------------------------|--|--|
| use_compression | Booleana | Si se establece en True, crea tablas para la exportación con compresión. |
| compression_mode | Row Page | Establece el nivel de compresión de las bases de datos de SQL Server. |
| | Predeterminado Direct_Load_Operations All_Operations Básico OLTP Query_High Query_Low Archive_High Archive_Low | Establece el nivel de compresión de las bases de datos de Oracle. Tenga en cuenta que los valores OLTP, Query_High, Query_Low, Archive_High y Archive_Low requieren un mínimo de Oracle 11gR2. |

Ejemplo que muestra cómo cambiar el comando CREATE INDEX para un índice específico:

```
db_exportnode.setKeyedPropertyValue("indexes", "MYINDEX", ["use_custom_create_index_command",
True])db_exportnode.setKeyedPropertyValue("indexes", "MYINDEX", ["custom_create_index_command",
"CREATE BITMAP INDEX <index-name> ON <table-name> <(index-columns)>"])
```

De forma alternativa, esto puede hacerse a través de una tabla hash:

```
db_exportnode.setKeyedPropertyValue("indexes", "MYINDEX", [{"fields":["id", "region"],
"use_custom_create_index_command":True, "custom_create_index_command":"CREATE INDEX
<index-name> ON
<nombre-tabla> <(columnasindex-columns)>"}])
```

Propiedades de datacollectionexportnode



El nodo de exportación Data Collection abre los datos en el formato utilizado por el software de investigación de mercados Data Collection. Debe estar instalada una biblioteca de datos de Data Collection para utilizar este nodo.

Ejemplo

```
stream = modeler.script.stream()
datacollectionexportnode = stream.createAt("datacollectionexport", "Data Collection", 200, 200)
datacollectionexportnode.setPropertyValue("metadata_file", "c:\\museums.mdd")
datacollectionexportnode.setPropertyValue("merge_metadata", "Overwrite")
datacollectionexportnode.setPropertyValue("casedata_file", "c:\\museumdata.sav")
datacollectionexportnode.setPropertyValue("generate_import", True)
datacollectionexportnode.setPropertyValue("enable_system_variables", True)
```

Tabla 240. propiedades de datacollectionexportnode

| propiedades de datacollectionexportnode | Tipo de datos | Descripción de la propiedad |
|---|---------------------------|--|
| metadata_file | cadena | Nombre del archivo de metadatos que se va a exportar. |
| merge_metadata | Overwrite MergeCurrent | |
| enable_system_variables | flag | Especifica si el archivo <i>.mdd</i> exportado debe incluir las variables de sistema de Data Collection. |
| casedata_file | cadena | Nombre del archivo <i>.sav</i> donde se exportan los datos de casos. |
| generate_import | flag | |

Propiedades de excelexportnode



El nodo de exportación Excel genera datos en el formato de archivo *.xlsx* de Microsoft Excel. Si lo desea, puede elegir iniciar automáticamente Excel y abrir el archivo exportado cuando se ejecute el nodo.

Ejemplo

```
stream = modeler.script.stream()
excelexportnode = stream.createAt("excelexport", "Excel", 200, 200)
excelexportnode.setPropertyValue("full_filename", "C:/output/myexport.xlsx")
```

```

excelexportnode.setPropertyValue("excel_file_type", "Excel2007")
excelexportnode.setPropertyValue("inc_field_names", True)
excelexportnode.setPropertyValue("inc_labels_as_cell_notes", False)
excelexportnode.setPropertyValue("launch_application", True)
excelexportnode.setPropertyValue("generate_import", True)

```

Tabla 241. Propiedades de excelexportnode

| Propiedad de excelexportnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|------------------|---|
| full_filename | cadena | |
| excel_file_type | Excel2007 | |
| export_mode | Create Append | |
| inc_field_names | flag | Especifica si los nombres de campos deben incluirse en la primera fila de la hoja de trabajo. |
| start_cell | cadena | Especifica la casilla de inicio de la exportación. |
| worksheet_name | cadena | Nombre de la hoja de trabajo que se va a escribir. |
| launch_application | flag | Determina si Excel debe invocarse para el archivo resultante. Tenga en cuenta que deberá especificar la ruta para iniciar Excel en el cuadro de diálogo Aplicaciones de ayuda (menú Herramientas, Aplicaciones de ayuda). |
| generate_import | flag | Determina si debe crearse un nodo Importar a Excel que leerá el archivo de datos exportado. |

Propiedades de extensionexportnode



Con el nodo de Exportación de extensión, puede ejecutar scripts R o Python para Spark para exportar datos.

Ejemplo de Python para Spark

```

#### Ejemplo de script de Python para Spark
import modeler.api
stream = modeler.script.stream()
node = stream.create("extension_export", "extension_export")
node.setPropertyValue("syntax_type", "Python")

python_script = """import spss.pyspark.runtime
from pyspark.sql import SQLContext
from pyspark.sql.types import *

cxt = spss.pyspark.runtime.getContext()
df = cxt.getSparkInputData()
print df.dtypes[:]
_newDF = df.select("Age","Drug")

```

```
print _newDF.dtypes[:]

df.select("Age", "Drug").write.save("c:/data/ageAndDrug.json", format="json")
"""

node.setPropertyValue("python_syntax", python_script)
```

Ejemplo de R

```
#### Ejemplo de script de R
node.setPropertyValue("syntax_type", "R")
node.setPropertyValue("r_syntax", ""write.csv(modelerData, "C:/export.csv)""")
```

Tabla 242. Propiedades de *extensionexportnode*

| Propiedades de <i>extensionexportnode</i> | Tipo de datos | Descripción de la propiedad |
|---|------------------------------------|--|
| syntax_type | R <i>Python</i> | Especifica qué script se ejecuta: R o Python (R es el valor predeterminado). |
| r_syntax | <i>string</i> | Sintaxis de scripts R a ejecutar. |
| python_syntax | <i>string</i> | Sintaxis de scripts Python a ejecutar. |
| convert_flags | StringsAndDoubles LogicalValues | Opción para convertir campos de distintivos. |
| convert_missing | <i>flag</i> | Opción para convertir valores perdidos al valor NA de R. |
| convert_datetime | <i>flag</i> | Opción para convertir las variables con los formatos de fecha o de fecha y hora para formatos de fecha/hora R. |
| convert_datetime_class | POSIXct POSIXlt | Opciones para especificar a qué formato se convierten las variables con los formatos de fecha o de fecha y hora. |

Propiedades de *outputfilenode*



El nodo Archivo sin formato produce datos en un archivo de texto delimitado. Esto es útil para exportar datos que se pueden leer con otro software de hoja de cálculo o de análisis.

Ejemplo

```
stream = modeler.script.stream()
outputfile = stream.createAt("outputfile", "File Output", 200, 200)
outputfile.setPropertyValue("full_filename", "c:/output/flatfile_output.txt")
outputfile.setPropertyValue("write_mode", "Append")
outputfile.setPropertyValue("inc_field_names", False)
outputfile.setPropertyValue("use_newline_after_records", False)
outputfile.setPropertyValue("delimiter_mode", "Tab")
outputfile.setPropertyValue("other_delimiter", ",")
outputfile.setPropertyValue("quote_mode", "Double")
outputfile.setPropertyValue("other_quote", "*")
outputfile.setPropertyValue("decimal_symbol", "Period")
outputfile.setPropertyValue("generate_import", True)
```

Tabla 243. propiedades de outputfilenode

| Propiedades de outputfilenode | Tipo de datos | Descripción de la propiedad |
|-------------------------------|---|-----------------------------------|
| full_filename | cadena | Nombre del archivo de resultados. |
| write_mode | Overwrite Append | |
| inc_field_names | flag | |
| use_newline_after_records | flag | |
| delimit_mode | Comma Tab Space Other | |
| other_delimiter | carácter | |
| quote_mode | Ninguno Single Double Other | |
| other_quote | flag | |
| generate_import | flag | |
| codificación | StreamDefault SystemDefault "UTF-8" | |

Propiedades de sasexportnode



El nodo Exportar SAS produce datos en formato SAS, para leerlos en SAS o en un paquete de software compatible con SAS. Hay tres formatos de archivo SAS disponibles: SAS para Windows/OS2, SAS para UNIX o SAS versión 7/8.

Ejemplo

```
stream = modeler.script.stream()
sasexportnode = stream.createAt("sasexport", "SAS Export", 200, 200)
sasexportnode.setPropertyValue("full_filename", "c:/output/SAS_output.sas7bdat")
sasexportnode.setPropertyValue("format", "SAS8")
sasexportnode.setPropertyValue("export_names", "NamesAndLabels")
sasexportnode.setPropertyValue("generate_import", True)
```

Tabla 244. propiedades de sasexportnode

| Propiedades de sasexportnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------------------------|--|
| format | Windows UNIX SAS7 SAS8 | Campos de etiquetas de propiedad de variantes. |
| full_filename | cadena | |
| export_names | NamesAndLabels NamesAsLabels | Se utiliza para correlacionar los nombres de campos de IBM SPSS Modeler que se vayan a exportar a nombres de variables de IBM SPSS Statistics o SAS. |

Tabla 244. propiedades de sasexportnode (continuación)

| Propiedades de sasexportnode | Tipo de datos | Descripción de la propiedad |
|------------------------------|---------------|-----------------------------|
| generate_import | flag | |

Propiedades de statisticsexportnode



El nodo Exportar Statistics ofrece los resultados en formato IBM SPSS Statistics *.sav* o *.zsav*. Los archivos *.sav* o *.zsav* se pueden leer con IBM SPSS Statistics Base y otros productos. Este es también el formato utilizado para los archivos caché de IBM SPSS Modeler.

Las propiedades de este nodo están descritas en “Propiedades de statisticsexportnode” en la página 343.

Propiedades del nodo tm1odataexport



El nodo Exportar de IBM Cognos TM1 exporta datos en un formato que pueden leer las bases de datos de IBM Cognos TM1.

Tabla 245. Propiedades del nodo tm1odataexport

| Propiedades del nodo tm1odataexport | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|------------------------------------|---|
| admin_host | cadena | El URL del nombre de host de la API REST. |
| server_name | cadena | El nombre del servidor TM1 seleccionado de admin_host. |
| credential_type | inputCredential o storedCredential | Se utiliza para indicar el tipo de credencial. |
| input_credential | lista | Cuando credential_type es inputCredential; especifique el dominio, el nombre de usuario y la contraseña. |
| stored_credential_name | cadena | Cuando credential_type es storedCredential; especifique el nombre de la credencial en el servidor C&DS. |
| selected_cube | campo | El nombre del cubo al que está exportando datos. Por ejemplo: TM1_export.setPropertyValue("selected_cube", "plan_BudgetPlan") |

Tabla 245. Propiedades del nodo tm1odataexport (continuación)

| Propiedades del nodo tm1odataexport | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|---------------|--|
| spss_field_to_tm1_element_mapping | lista | <p>El elemento de tm1 al que hay que correlacionar debe formar parte de la dimensión de columna para la vista de cubo seleccionada. El formato es: <code>[[[Campo_1, Dimensión_1, False], [Elemento_1, Dimensión_2, True], ...], [[Campo_2, ElementoMedidaExistente, False], [Campo_3, NuevoElementoMedida, True], ...]]</code></p> <p>Existen 2 listas para describir la información de correlación. La correlación de un elemento de hoja a una dimensión se corresponde al ejemplo 2 siguiente:</p> <p>Ejemplo 1: La primera lista: (<code>[[Campo_1, Dimensión_1, False], [Elemento_1, Dimensión_2, True], ...]</code>) se utiliza para la información de mapas de la dimensión TM1.</p> <p>Cada lista de 3 valores indica información de correlación de dimensiones. El tercer valor booleano se utiliza para indicar si selecciona un elemento de una dimensión. Por ejemplo: <code>"[Campo_1, Dimensión_1, False]"</code> significa que Campo_1 está correlacionado con Dimensión_1; <code>"[Elemento_1, Dimensión_2, True]"</code> significa que Elemento_1 está seleccionado para Dimensión_2.</p> <p>Ejemplo 2: La segunda lista: (<code>[[Campo_2, ElementoMedidaExistente, False], [Campo_3, NuevoElementoMedida, True], ...]</code>) se utiliza para la información de correlación de elemento de dimensión de medida TM1.</p> <p>Cada lista de 3 valores indica información de correlación de elementos de medida. El tercer valor booleano se utiliza para indicar la necesidad de crear un elemento nuevo. <code>"[Campo_2, ElementoMedidaExistente, False]"</code> significa que Campo_2 está correlacionado con ElementoMedidaExistente; <code>"[Campo_3, NuevoElementoMedida, True]"</code> significa que NuevoElementoMedida debe ser una dimensión de medida seleccionada en <code>selected_measure</code> y el Campo_3 está correlacionado con este.</p> |
| selected_measure | cadena | <p>Especifique la dimensión de medida.</p> <p>Ejemplo: <code>setProperty("selected_measure", "Measures")</code></p> |

Propiedades del nodo tm1export (en desuso)



El nodo Exportar de IBM Cognos TM1 exporta datos en un formato que pueden leer las bases de datos de IBM Cognos TM1.

Nota: Este nodo estaba en desuso en Modeler 18.0. El nombre del script del nodo de sustitución es *tm1odataexport*.

Tabla 246. Propiedades del nodo *tm1export*.

| Propiedades del nodo <i>tm1export</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---|--|
| pm_host | <i>cadena</i> | <p>Nota: sólo para la versión 16.0 y 17.0</p> <p>Nombre del host. Por ejemplo: <code>TM1_export.setPropertyValue("pm_host", 'http://9.191.86.82:9510/pmhub/pm')</code></p> |
| tm1_connection | <i>["campo", "campo", ..., "campo"]</i> | <p>Nota: sólo para la versión 16.0 y 17.0</p> <p>Una propiedad de la lista que contiene los detalles de conexión para el servidor de TM1. El formato es: ["TM1_Server_Name", "tm1_username", "tm1_password"]</p> <p>Por ejemplo: <code>TM1_export.setPropertyValue("tm1_connection", ['Planning Sample', "admin" "apple"])</code></p> |
| selected_cube | <i>campo</i> | <p>El nombre del cubo al que está exportando datos. Por ejemplo: <code>TM1_export.setPropertyValue("selected_cube", "plan_BudgetPlan")</code></p> |

Tabla 246. Propiedades del nodo *tm1export* (continuación).

| Propiedades del nodo <i>tm1export</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---------------|---|
| <i>spssfield_tm1element_mapping</i> | <i>list</i> | <p>El elemento de <i>tm1</i> al que hay que correlacionar debe formar parte de la dimensión de columna para la vista de cubo seleccionada. El formato es: <code>[[[Campo_1, Dimensión_1, False], [Elemento_1, Dimensión_2, True], ...], [[Campo_2, ElementoMedidaExistente, False], [Campo_3, NuevoElementoMedida, True], ...]]</code></p> <p>Existen 2 listas para describir la información de correlación. La correlación de un elemento de hoja a una dimensión se corresponde al ejemplo 2 siguiente:</p> <p>Ejemplo 1: La primera lista: <code>([[Campo_1, Dimensión_1, False], [Elemento_1, Dimensión_2, True], ...])</code> se utiliza para la información de mapas de la dimensión TM1.</p> <p>Cada lista de 3 valores indica información de correlación de dimensiones. El tercer valor booleano se utiliza para indicar si selecciona un elemento de una dimensión. Por ejemplo: <code>"[Campo_1, Dimensión_1, False]"</code> significa que <code>Campo_1</code> está correlacionado con <code>Dimensión_1</code>; <code>"[Elemento_1, Dimensión_2, True]"</code> significa que <code>Elemento_1</code> está seleccionado para <code>Dimensión_2</code>.</p> <p>Ejemplo 2: La segunda lista: <code>([[Campo_2, ElementoMedidaExistente, False], [Campo_3, NuevoElementoMedida, True], ...])</code> se utiliza para la información de correlación de elemento de dimensión de medida TM1.</p> <p>Cada lista de 3 valores indica información de correlación de elementos de medida. El tercer valor booleano se utiliza para indicar la necesidad de crear un elemento nuevo. <code>"[Campo_2, ElementoMedidaExistente, False]"</code> significa que <code>Campo_2</code> está correlacionado con <code>ElementoMedidaExistente</code>; <code>"[Campo_3, NuevoElementoMedida, True]"</code> significa que <code>NuevoElementoMedida</code> debe ser una dimensión de medida seleccionada en <code>selected_measure</code> y el <code>Campo_3</code> está correlacionado con este.</p> |
| <i>selected_measure</i> | <i>cadena</i> | <p>Especifique la dimensión de medida.</p> <p>Ejemplo: <code>setProperty("selected_measure", "Measures")</code></p> |

Propiedades de *xmlexportnode*



El nodo de exportación XML exporta datos a un archivo en formato XML. También puede crear un nodo de origen XML para leer los datos exportados a la ruta.

Ejemplo

```
stream = modeler.script.stream()
xmlexportnode = stream.createAt("xmlexport", "XML Export", 200, 200)
xmlexportnode.setPropertyValue("full_filename", "c:/export/data.xml")
xmlexportnode.setPropertyValue("map", [{"/catalog/book/genre", "genre"},
["/catalog/book/title", "title"]])
```

Tabla 247. propiedades de `xmlexportnode`

| propiedades de <code>xmlexportnode</code> | Tipo de datos | Descripción de la propiedad |
|---|---------------|---|
| <code>full_filename</code> | <i>cadena</i> | (obligatorio) Ruta completa y nombre de archivo del archivo XML para exportar. |
| <code>use_xml_schema</code> | <i>flag</i> | Especifica si utilizar un esquema XML (archivo XSD o DTD) para controlar la estructura de los datos exportados. |
| <code>full_schema_filename</code> | <i>cadena</i> | Ruta completa y nombre de archivo del archivo XSD o DTD que se quiere utilizar. Es obligatorio si <code>use_xml_schema</code> está establecido como true. |
| <code>generate_import</code> | <i>flag</i> | Genera un nodo de origen XML que leerá el archivo de datos exportados de nuevo en la ruta. |
| <code>registros</code> | <i>cadena</i> | Expresión de XPath que denota el límite de registro. |
| <code>map</code> | <i>cadena</i> | Correlaciona el nombre de campo a la estructura XML. |

Capítulo 18. Propiedades de nodos de IBM SPSS Statistics

Propiedades de statisticsimportnode



El nodo Archivo Statistics lee los datos desde un formato de archivo *.sav* o *.zsav* que utiliza IBM SPSS Statistics y archivos caché guardados en IBM SPSS Modeler, que también puede utilizar el mismo formato.

Ejemplo

```
stream = modeler.script.stream()
statisticsimportnode = stream.createAt("statisticsimport", "SAV Import", 200, 200)
statisticsimportnode.setPropertyValue("full_filename", "C:/data/drug1n.sav")
statisticsimportnode.setPropertyValue("import_names", True)
statisticsimportnode.setPropertyValue("import_data", True)
```

Tabla 248. propiedades de statisticsimportnode.

| propiedades de statisticsimportnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|---------------------------------|---|
| full_filename | <i>cadena</i> | El nombre completo del archivo, incluyendo la ruta. |
| contraseña | <i>cadena</i> | La contraseña. El parámetro password se debe establecer antes que el parámetro file_encrypted. |
| file_encrypted | <i>tag</i> | Indica si el archivo está protegido con contraseña. |
| import_names | NamesAndLabels LabelsAsNames | Método para gestionar nombres y etiquetas de variables. |
| import_data | DataAndLabels LabelsAsData | Método para gestionar valores y etiquetas. |
| use_field_format_for_storage | <i>Booleana</i> | Especifica si se utiliza la información de formato de campo de IBM SPSS Statistics al importar. |

propiedades de statistictransformnode



El nodo Transformación Statistics ejecuta una selección de comandos de sintaxis de IBM SPSS Statistics en los orígenes de datos de IBM SPSS Modeler. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Ejemplo

```
stream = modeler.script.stream()
statistictransformnode = stream.createAt("statistictransform", "Transform", 200, 200)
statistictransformnode.setPropertyValue("syntax", "COMPUTE NewVar = Na + K.")
statistictransformnode.setKeyedPropertyValue("new_name", "NewVar", "Mixed Drugs")
statistictransformnode.setPropertyValue("check_before_saving", True)
```

Tabla 249. propiedades de statisticstransformnode

| propiedades de statisticstransformnode | Tipo de datos | Descripción de la propiedad |
|--|---------------|--|
| syntax | <i>cadena</i> | |
| check_before_saving | <i>flag</i> | Valida la sintaxis introducida antes de guardar las entradas. Muestra un mensaje de error si la sintaxis no es válida. |
| default_include | <i>flag</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |
| include | <i>flag</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |
| new_name | <i>cadena</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |

propiedades de statisticsmodelnode



El nodo Modelo Statistics permite analizar y trabajar con sus datos ejecutando los procedimientos de IBM SPSS Statistics que producen PMML. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Ejemplo

```
stream = modeler.script.stream()
statisticsmodelnode = stream.createAt("statisticsmodel", "Model", 200, 200)
statisticsmodelnode.setPropertyValue("syntax", "COMPUTE NewVar = Na + K.")
statisticsmodelnode.setKeyedPropertyValue("new_name", "NewVar", "Mixed Drugs")
```

| propiedades de statisticsmodelnode | Tipo de datos | Descripción de la propiedad |
|------------------------------------|---------------|---|
| syntax | <i>cadena</i> | |
| default_include | <i>flag</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |
| include | <i>flag</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |
| new_name | <i>cadena</i> | Consulte el tema “propiedades de filternode” en la página 142 para obtener más información. |

Propiedades de statisticsoutputnode



El nodo Resultados de Statistics le permite llamar a un procedimiento de IBM SPSS Statistics para analizar los datos de IBM SPSS Modeler. Se puede acceder a una gran variedad de procedimientos analíticos de IBM SPSS Statistics. Este nodo requiere una copia de IBM SPSS Statistics con licencia.

Ejemplo

```
stream = modeler.script.stream()
statisticsoutputnode = stream.createAt("statisticsoutput", "Output", 200, 200)
statisticsoutputnode.setPropertyValue("syntax", "SORT CASES BY Age(A) Sex(A) BP(A)
Cholesterol(A)")
statisticsoutputnode.setPropertyValue("use_output_name", False)
statisticsoutputnode.setPropertyValue("output_mode", "File")
statisticsoutputnode.setPropertyValue("full_filename", "Cases by Age, Sex and Medical History")
statisticsoutputnode.setPropertyValue("file_type", "HTML")
```

Tabla 250. propiedades de *statisticsoutputnode*

| propiedades de <i>statisticsoutputnode</i> | Tipo de datos | Descripción de la propiedad |
|--|--------------------|--|
| mode | Dialog Syntax | Selecciona la opción "cuadro de diálogo de IBM SPSS Statistics" o el editor Sintaxis |
| syntax | <i>cadena</i> | |
| use_output_name | <i>flag</i> | |
| output_name | <i>cadena</i> | |
| output_mode | Screen File | |
| full_filename | <i>cadena</i> | |
| file_type | HTML SPV SPW | |

Propiedades de *statisticsexportnode*



El nodo Exportar Statistics ofrece los resultados en formato IBM SPSS Statistics *.sav* o *.zsav*. Los archivos *.sav* o *.zsav* se pueden leer con IBM SPSS Statistics Base y otros productos. Este es también el formato utilizado para los archivos caché de IBM SPSS Modeler.

Ejemplo

```
stream = modeler.script.stream()
statisticsexportnode = stream.createAt("statisticsexport", "Export", 200, 200)
statisticsexportnode.setPropertyValue("full_filename", "c:/output/SPSS_Statistics_out.sav")
statisticsexportnode.setPropertyValue("field_names", "Names")
statisticsexportnode.setPropertyValue("launch_application", True)
statisticsexportnode.setPropertyValue("generate_import", True)
```

Tabla 251. propiedades de *statisticsexportnode*.

| propiedades de <i>statisticsexportnode</i> | Tipo de datos | Descripción de la propiedad |
|--|---------------|---|
| full_filename | <i>cadena</i> | |
| file_type | sav zsav | Guardar el archivo en formato <i>sav</i> o <i>zsav</i> . Por ejemplo: <code>statisticsexportnode.setPropertyValue("file_type", "sav")</code> |
| encrypt_file | <i>flag</i> | Indica si el archivo está protegido con contraseña. |
| password | <i>cadena</i> | La contraseña. |

Tabla 251. propiedades de statisticsexportnode (continuación).

| propiedades de statisticsexportnode | Tipo de datos | Descripción de la propiedad |
|-------------------------------------|---------------------------------|--|
| launch_application | <i>flag</i> | |
| export_names | NamesAndLabels NamesAsLabels | Se utiliza para correlacionar los nombres de campos de IBM SPSS Modeler que se vayan a exportar a nombres de variables de IBM SPSS Statistics o SAS. |
| generate_import | <i>flag</i> | |

Capítulo 19. Propiedades de nodo Python

Propiedades de smotenode



El nodo SMOTE (Synthetic Minority Over-sampling Technique) proporciona un algoritmo de sobremuestreo para tratar con conjuntos de datos desequilibrados. Proporciona un método avanzado para equilibrar los datos. El nodo de proceso SMOTE en SPSS Modeler se implementa en Python y necesita la biblioteca de Python `imbalanced-learn`®.

Tabla 252. Propiedades de smotenode

| Propiedades de smotenode | Tipo de datos | Descripción de la propiedad |
|--------------------------|-----------------|--|
| target_field | <i>field</i> | Campo objetivo. |
| sample_ratio | <i>string</i> | Habilita un valor de cociente personalizado. Las dos opciones son Automático (<code>sample_ratio_auto</code>) o Establecer cociente (<code>sample_ratio_manual</code>). |
| sample_ratio_value | <i>float</i> | El cociente es el número de muestras en la clase de minoría sobre el número de muestras en la clase de mayoría. Debe ser mayor que 0 y menor que o igual a 1. El valor predeterminado es auto. |
| random_seed | <i>integer</i> | Semilla utilizada por el generador de números aleatorios. |
| k_neighbours | <i>integer</i> | Número de vecinos más cercanos a utilizar para construir muestras sintéticas. El valor predeterminado es 5. |
| m_neighbours | <i>integer</i> | Número de vecinos más cercanos a utilizar para determinar si una muestra de minoría está en peligro. Esta opción sólo se habilita con los tipos de algoritmo de SMOTE <code>borderline1</code> y <code>borderline2</code> . El valor predeterminado es 10. |
| algorithm_kind | <i>string</i> | Tipo de algoritmo de SMOTE: regular, <code>borderline1</code> o <code>borderline2</code> . |
| use_partition | <i>Booleana</i> | Si se establece en true, sólo se utilizarán datos de entrenamiento para la generación de modelos. El valor predeterminado es true. |

Propiedades de xgboosttreenode



XGBoost Tree® es una implementación avanzada de un algoritmo de aumento de gradiente con un modelo de árbol como modelo base. Los algoritmos de aumento conocen de forma iterativa los clasificadores débiles y, a continuación, los añaden a un clasificador fuerte final. XGBoost Tree es muy flexible y proporciona muchos parámetros que pueden ser abrumadores para la mayoría de usuarios, de modo que el nodo XGBoost Tree en SPSS Modeler expone las características principales y los parámetros utilizados comúnmente. El nodo se implementa en Python.

Tabla 253. Propiedades de *xgboosttreenode*

| Propiedades de <i>xgboosttreenode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|----------------|--|
| TargetField | <i>field</i> | Campos objetivo. |
| InputFields | <i>field</i> | Campos de entrada. |
| treeMethod | <i>string</i> | Método de árbol para la generación de modelos. Los valores posibles son <i>auto</i> , <i>exact</i> o <i>approx</i> . El valor predeterminado es <i>auto</i> . |
| numBoostRound | <i>integer</i> | Valor de redondeo de aumento de número para la generación de modelos. Especifique un valor entre 1 y 1000. El valor predeterminado es 10. |
| maxDepth | <i>integer</i> | Profundidad máxima para el crecimiento del árbol. Especifique un valor de 1 o superior. El valor predeterminado es 6. |
| minChildWeight | <i>Double</i> | Ponderación hijo mínima para el crecimiento del árbol. Especifique un valor de 0 o superior. El valor predeterminado es 1. |
| maxDeltaStep | <i>Double</i> | Paso delta máximo para el crecimiento del árbol. Especifique un valor de 0 o superior. El valor predeterminado es 0. |
| objectiveType | <i>string</i> | Tipo de objetivo para la tarea de aprendizaje. Los valores posibles son <i>reg:linear</i> , Δ <i>reg:logistic</i> , Δ <i>reg:gamma</i> , Δ <i>reg:tweedie</i> , <i>count:poisson</i> , Δ <i>rank:pairwise</i> , <i>binary:logistic</i> o <i>multi</i> . Tenga en cuenta que para objetivos de marca, sólo se puede utilizar <i>binary:logistic</i> o <i>multi</i> . Si se utiliza <i>multi</i> , el resultado de puntuación mostrará los tipos de objetivo de XGBoost <i>multi:softmax</i> y <i>multi:softprob</i> . |
| random_seed | <i>integer</i> | Semilla de aleatorización. Cualquier número entre 0 y 9999999. El valor predeterminado es 0. |
| sampleSize | <i>Double</i> | La submuestra para el sobreajuste de control. Especifique un valor entre 0,1 y 1,0. El valor predeterminado es 0,1. |
| eta | <i>Double</i> | Eta para el sobreajuste de control. Especifique un valor entre 0 y 1. El valor predeterminado es 0,3. |
| gamma | <i>Double</i> | Gamma para el sobreajuste de control. Especifique cualquier número 0 o superior. El valor predeterminado es 6. |
| colsSampleRatio | <i>Double</i> | Muestra de columna por árbol para el sobreajuste de control. Especifique un valor entre 0,01 y 1. El valor predeterminado es 1. |
| colsSampleLevel | <i>Double</i> | Muestra de columna por nivel para el sobreajuste de control. Especifique un valor entre 0,01 y 1. El valor predeterminado es 1. |
| lambda | <i>Double</i> | Lambda para el sobreajuste de control. Especifique cualquier número 0 o superior. El valor predeterminado es 1. |

Tabla 253. Propiedades de *xgboosttreenode* (continuación)

| Propiedades de <i>xgboosttreenode</i> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|---------------|--|
| alpha | Double | Alfa para el sobreajuste de control. Especifique cualquier número 0 o superior. El valor predeterminado es 0. |
| scalePosWeight | Double | Ponderación de posición de escala para manejar conjuntos de datos desequilibrados. El valor predeterminado es 1. |

Propiedades de *xboostlinearnode*



XGBoost Linear[®] es una implementación avanzada de un algoritmo de aumento de gradiente con un modelo lineal como modelo base. Los algoritmos de aumento conocen de forma iterativa los clasificadores débiles y, a continuación, los añaden a un clasificador fuerte final. El nodo XGBoost Linear en SPSS Modeler se implementa en Python.

Tabla 254. Propiedades de *xgboostlinearnode*

| Propiedades de <i>xgboostlinearnode</i> | Tipo de datos | Descripción de la propiedad |
|---|---------------|--|
| TargetField | field | |
| InputFields | field | |
| alpha | Double | Parámetro de aumento lineal alfa. Especifique cualquier número 0 o superior. El valor predeterminado es 0. |
| lambda | Double | Parámetro de aumento lineal lambda. Especifique cualquier número 0 o superior. El valor predeterminado es 1. |
| lambdaBias | Double | Parámetro de aumento lineal lambda bias. Especifique cualquier número. El valor predeterminado es 0. |
| numBoostRound | integer | Valor de redondeo de aumento de número para la generación de modelos. Especifique un valor entre 1 y 1000. El valor predeterminado es 10. |
| objectiveType | string | Tipo de objetivo para la tarea de aprendizaje. Los valores posibles son <code>reg:linear</code> , <code>reg:logistic</code> , <code>reg:gamma</code> , <code>reg:tweedie</code> , <code>count:poisson</code> , <code>rank:pairwise</code> , <code>binary:logistic</code> o <code>multi</code> . Tenga en cuenta que para objetivos de marca, sólo se puede utilizar <code>binary:logistic</code> o <code>multi</code> . Si se utiliza <code>multi</code> , el resultado de puntuación mostrará los tipos de objetivo de XGBoost <code>multi:softmax</code> y <code>multi:softprob</code> . |
| random_seed | integer | Semilla de aleatorización. Cualquier número entre 0 y 9999999. El valor predeterminado es 0. |

Propiedades de ocsvmnode



El nodo SVM de una clase utiliza un algoritmo de aprendizaje no supervisado. El nodo se puede utilizar para la detección de novedad. Detectará el límite flexible de un conjunto de muestras proporcionado, para clasificar a continuación los puntos nuevos como pertenecientes o no a dicho conjunto. Este nodo de modelado SVM de una clase en SPSS Modeler se implementa en Python y necesita la biblioteca `scikit-learn` de Python.

Tabla 255. Propiedades de ocsvmnode

| Propiedades de ocsvmnode | Tipo de datos | Descripción de la propiedad |
|--------------------------|-----------------|---|
| role_use | <i>string</i> | Especifique predefined para utilizar roles predefinidos o custom para utilizar asignaciones de campo personalizado. El valor predeterminado es predefined. |
| inputs | <i>field</i> | Lista de los nombres de campo para entrada. |
| splits | <i>field</i> | Lista de nombre de campo para división. |
| use_partition | <i>Booleana</i> | Especifique true o false. El valor predeterminado es true. Si se establece en true, sólo se utilizarán datos de entrenamiento al generar el modelo. |
| mode_type | <i>string</i> | Modo. Los valores posibles son simple o expert. Todos los parámetros de la pestaña Experto estarán inhabilitados si se especifica simple. |
| stopping_criteria | <i>string</i> | Serie de notación científica. Los valores posibles son 1.0E-1, 1.0E-2, 1.0E-3, 1.0E-4, 1.0E-5 o 1.0E-6. El valor predeterminado es 1.0E-3. |
| precision | <i>float</i> | Precisión de regresión (nu). Límite en la fracción de errores de entrenamiento y vectores de soporte. Especifique un número mayor que 0 y menor que o igual a 1,0. El valor predeterminado es 0,1. |
| kernel | <i>string</i> | Tipo de kernel a utilizar en el algoritmo. Los valores posibles son linear, poly, rbf, sigmoid o precomputed. El valor predeterminado es rbf. |
| enable_gamma | <i>Booleana</i> | Habilita el parámetro gamma. Especifique true o false. El valor predeterminado es true. |
| gamma | <i>float</i> | Este parámetro sólo está habilitado para los kernels rbf, poly y sigmoid. Si el parámetro enable_gamma se establece en false, este parámetro se establecerá en auto. Si se establece en true, el valor predeterminado es 0,1. |
| coef0 | <i>float</i> | Término independiente en la función de kernel. Este parámetro sólo está habilitado para el kernel poly y el kernel sigmoid. El valor predeterminado es 0,0. |
| degree | <i>integer</i> | Grado de función de kernel polinómica. Este parámetro sólo está habilitado para el kernel poly. Especifique cualquier entero. El valor predeterminado es 3. |

Tabla 255. Propiedades de `ocsvmnode` (continuación)

| Propiedades de <code>ocsvmnode</code> | Tipo de datos | Descripción de la propiedad |
|---------------------------------------|-----------------|--|
| <code>shrinking</code> | <i>Booleana</i> | Especifica si se debe utilizar la opción heurística de reducción. Especifique <code>true</code> o <code>false</code> . El valor predeterminado es <code>false</code> . |
| <code>enable_cache_size</code> | <i>Booleana</i> | Habilita el parámetro <code>cache_size</code> . Especifique <code>true</code> o <code>false</code> . El valor predeterminado es <code>false</code> . |
| <code>cache_size</code> | <i>float</i> | Tamaño de caché de kernel en MB. El valor predeterminado es 200. |
| <code>enable_random_seed</code> | <i>Booleana</i> | Habilita el parámetro <code>random_seed</code> . Especifique <code>true</code> o <code>false</code> . El valor predeterminado es <code>false</code> . |
| <code>random_seed</code> | <i>integer</i> | Semilla de aleatorización a utilizar al reorganizar datos para la estimación de probabilidad. Especifique cualquier entero. |
| <code>pc_type</code> | <i>string</i> | Tipo del gráfico de coordenadas paralelas. Los valores posibles son <code>independent</code> o <code>general</code> . |
| <code>lines_amount</code> | <i>integer</i> | Número máximo de líneas a incluir en el gráfico. Especifique un entero entre 1 y 1000. |
| <code>lines_fields_custom</code> | <i>Booleana</i> | Habilita el parámetro <code>lines_fields</code> , que le permite especificar campos personalizados para mostrarlos en el resultado gráfico. Si se establece en <code>false</code> , se mostrarán todos los campos. Si se establece en <code>true</code> , sólo se mostrarán los campos especificados con el parámetro <code>lines_fields</code> . Por razones de rendimiento, se visualizará un máximo de 20 campos. |
| <code>lines_fields</code> | <i>field</i> | Lista de nombres de campo a incluir en el gráfico como ejes verticales. |

Capítulo 20. Propiedades de supernodo

En las siguientes tablas se describen las propiedades específicas de los Supernodos. Tenga en cuenta que las propiedades de nodos comunes se aplican también a los Supernodos.

Tabla 256. Propiedades del supernodo de terminal

| Nombre de la propiedad | Tipo de propiedad / Lista de valores | Descripción de la propiedad |
|------------------------|--------------------------------------|-----------------------------|
| execute_method | Script Normal | |
| script | cadena | |

Parámetros de Supernodos

Puede utilizar scripts para crear o establecer parámetros de Supernodo utilizando el formato general:
`mySuperNode.setParameterValue("minvalue", 30)`

Puede recuperar el valor del parámetro con:

```
value mySuperNode.getParameterValue("minvalue")
```

Búsqueda de los supernodos existentes

Puede encontrar supernodos en rutas utilizando la función `findByType()` :

```
source_supernode = modeler.script.stream().findByType("source_super", None)
process_supernode = modeler.script.stream().findByType("process_super", None)
terminal_supernode = modeler.script.stream().findByType("terminal_super", None)
```

Configuración de las propiedades de nodos encapsulados

Puede establecer las propiedades de determinados nodos encapsulados dentro un supernodo accediendo al diagrama hijo dentro del Supernodo. Por ejemplo, imaginemos que tiene un Supernodo de origen que incluye un nodo de archivo de variables encapsulado para leer los datos. Puede pasar el nombre del archivo para leer (especificado mediante la propiedad `full_filename`) accediendo al diagrama hijo y buscando el nodo relevante como se indica a continuación:

```
childDiagram = source_supernode.getChildDiagram()
varfilenode = childDiagram.findByType("variablefile", None)
varfilenode.setPropertyValue("full_filename", "c:/mydata.txt")
```

Creación de supernodos

Si desea crear un supernodo y su contenido de cero, puede hacerlo de forma similar a la creación del supernodo, accediendo al diagrama hijo y creando los nodos que desee. También debe asegurarse de que los nodos del diagrama del supernodo estén también vinculados a los nodos de los conectores de entrada y/o de salida. Por ejemplo, si desea crear un proceso Supernodo:

```
process_supernode = modeler.script.stream().createAt("process_super", "My SuperNode", 200, 200)
childDiagram = process_supernode.getChildDiagram()
filternode = childDiagram.createAt("filter", "My Filter", 100, 100)
childDiagram.linkFromInputConnector(filternode)
childDiagram.linkToOutputConnector(filternode)
```

Apéndice A. Referencia de nombres de nodo

Esta sección ofrece una referencia de todos los nombres de script de los nodos de IBM SPSS Modeler.

Nombres de nugget de modelo

Se puede hacer referencia a los nugget de modelo (también denominados modelos generados) según el tipo, como con los objetos de nodo y de resultado. Las siguientes tablas muestran los nombres de referencia de los objetos del modelo.

Tenga en cuenta que estos nombres se utilizan específicamente para hacer referencia a los nugget de modelo en la paleta Modelos (en la esquina superior derecha de la ventana de IBM SPSS Modeler). Para hacer referencia a los nodos de modelo que se han añadido a una ruta para la puntuación, se utiliza un conjunto diferente de nombres con el prefijo `apply...`. Consulte el tema Propiedades de nodos de nugget de modelo para obtener más información.

Nota: En circunstancias normales, se recomienda hacer referencia a los modelos por nombre y tipo para evitar confusiones.

Tabla 257. Nombres de nugget de modelo (paleta de modelado).

| Nombre del modelo | Modelo |
|-------------------|------------------------------------|
| anomalydetection | Anomalía |
| a priori | A priori |
| autoclassifier | Clasificado automático |
| autocluster | Agrupación en clústeres automática |
| autonumeric | Autonumérico |
| bayesnet | Red bayesiana |
| c50 | C5.0 |
| carma | Carma |
| árbol cr | Árbol C&R |
| chaid | CHAID |
| coxreg | Regresión de Cox |
| decisionlist | Lista de decisiones |
| discriminant | Discriminante |
| factor | PCA/Factorial |
| featureselection | Sel. características |
| genlin | Regresión lineal generalizada |
| glmm | GLMM |
| kmeans | K-medias |
| knn | <i>k</i> : vecino más cercano |
| kohonen | Kohonen |
| lineal | Lineal |
| logreg | Regresión logística |
| neuralnetwork | Red neuronal |

Tabla 257. Nombres de nugget de modelo (paleta de modelado) (continuación).

| Nombre del modelo | Modelo |
|-------------------|--|
| quest | QUEST |
| regresión | Regresión lineal |
| secuencia | Secuencia |
| slrm | Modelo de respuesta de autoaprendizaje |
| statisticsmodel | Modelo de IBM SPSS Statistics |
| svm | Máquina de vectores de soporte |
| timeseries | Serie temporal |
| twostep | Dos fases |

Tabla 258. Nombres de nugget de modelo (paleta de modelado de bases de datos).

| Nombre del modelo | Modelo |
|-----------------------|--------------------------------|
| db2imcluster | Clúster de IBM ISW |
| db2imlog | Regresión logística de IBM ISW |
| db2imnb | Bayesiano ingenuo de IBM ISW |
| db2imreg | Regresión de IBM ISW |
| db2imtree | Árbol de decisión de IBM ISW |
| msassoc | Reglas de asociación de MS |
| msbayes | Bayesiano ingenuo de MS |
| mscluster | Clúster de MS |
| mslogistic | Regresión logística de MS |
| msneuralnetwork | Red neuronal de MS |
| msregression | Regresión lineal de MS |
| mssequencecluster | Clúster de secuencias de MS |
| mstimeseries | Serie temporales de MS |
| mstree | Árbol de decisión de MS |
| netezzabayes | Red bayesiana de Netezza |
| netezzadectree | Árbol de decisión de Netezza |
| netezzadivcluster | Clúster divisivo de Netezza |
| netezzaglm | Lineal generalizado de Netezza |
| netezzakmeans | K-medias de Netezza |
| netezzaknn | KNN de Netezza |
| netezzalineregression | Regresión lineal de Netezza |
| netezzanaiwebayes | Bayesiano ingenuo de Netezza |
| netezzapca | PCA de Netezza |
| netezzaregtree | Árbol de regresión de Netezza |
| netezzatimeseries | Serie temporales de Netezza |
| oraabn | Bayesiano adaptativo de Oracle |
| oraai | Oracle AI |
| oradecisiontree | Árbol de decisión de Oracle |
| oraglm | GLM de Oracle |

Tabla 258. Nombres de nugget de modelo (paleta de modelado de bases de datos) (continuación).

| Nombre del modelo | Modelo |
|-------------------|-----------------------------|
| orakmeans | K-medias de Oracle |
| oranb | Bayesiano ingenuo de Oracle |
| oranmf | NMF de Oracle |
| oracluster | O-clúster de Oracle |
| orasvm | SVM de Oracle |

Evitar nombres duplicados del modelo

Al utilizar los scripts para manipular los modelos generados, debe tener en cuenta que el hecho de permitir nombres de modelo duplicados puede originar referencias ambiguas. Para evitarlo, resulta útil utilizar nombres exclusivos para los modelos generados en los scripts.

Para configurar las opciones de los nombres de modelo duplicados:

1. Seleccione en los menús:
Herramientas > Opciones de usuario
2. Pulse en la pestaña **Notificaciones**.
3. Seleccione **Sustituir modelo anterior** para restringir los nombres duplicados de los modelos generados.

El comportamiento de la ejecución de scripts puede variar entre SPSS Modeler y IBM SPSS Collaboration and Deployment Services cuando haya referencias de modelo ambiguas. El cliente de SPSS Modeler incluye la opción "Reemplazar modelo anterior", que reemplaza automáticamente los modelos que tengan el mismo nombre (por ejemplo, cuando un script se itera a través de un bucle para producir un modelo diferente cada vez). Sin embargo, esta opción no está disponible cuando el mismo script se ejecuta en IBM SPSS Collaboration and Deployment Services. Puede evitar esta situación cambiando el nombre del modelo generado en cada iteración para evitar referencias ambiguas a los modelos o borrando el modelo actual (por ejemplo, añadiendo una instrucción `clear generated palette`) antes del final del bucle.

Nombres de tipo de resultados

La siguiente tabla indica los tipos de objetos de resultados y los nodos que los crean. Para obtener una lista completa de los formatos de exportación disponibles para cada tipo de objeto de salida, consulte la descripción de las propiedades del nodo que crea el tipo de salida, disponible en Propiedades comunes de nodos Gráfico y en Propiedades de los nodos de resultados.

Tabla 259. Tipos de objeto de salida y los nodos que los crean.

| Tipo de objeto de resultado | Nodo |
|-----------------------------|--------------------|
| analysisoutput | Análisis |
| collectionoutput | Colección |
| dataauditoutput | Auditoría de datos |
| distributionoutput | Distribución |
| evaluationoutput | Evaluación |
| histogramoutput | Histograma |
| matrixoutput | Matriz |
| meansoutput | Medias |
| multiplotoutput | G. múltiple |

Tabla 259. Tipos de objeto de salida y los nodos que los crean (continuación).

| Tipo de objeto de resultado | Nodo |
|-----------------------------|---|
| plotoutput | Gráfico |
| qualityoutput | Calidad |
| reportdocumentoutput | Este tipo de objeto no es de un nodo, es un resultado creado por un informe de proyecto |
| reportoutput | Informe |
| statisticsprocedureoutput | Resultado de Estadísticas |
| statisticsoutput | Estadísticos |
| tableoutput | Tabla |
| timeplotoutput | Gráfico de tiempo |
| weboutput | Malla |

Apéndice B. Migración desde scripts de herencia a scripts Python

Visión general de la migración de scripts de herencia

Esta sección proporciona un resumen de las diferencias entre el script de Python y el script de herencia en IBM SPSS Modeler y proporciona información acerca de cómo migrar los scripts de herencia a scripts Python. En esta sección encontrar una lista de los comandos de herencia de SPSS Modeler estándar y los comandos Python equivalentes.

Diferencias generales

Una gran parte del diseño de los scripts de herencia se debe a los scripts de comandos del sistema operativo. Los scripts de herencia están orientados a líneas y, aunque existen algunas estructuras de bloque, por ejemplo `if...then...else...endif` y `for...endfor`, generalmente la indentación no es importante.

En los scripts Python, la indentación es importante y las líneas que pertenecen al mismo bloque lógico se deben indentar en el mismo nivel.

Nota: Debe prestar atención cuando copie y pegue el código Python. En el editor, una línea que se ha indentado utilizando pestañas puede parecer la misma que una línea que se ha indentado utilizando espacios. Sin embargo, el script Python generará un error porque no se considera que la indentación de las líneas sea la misma.

El contexto de los scripts

El contexto de script define el entorno en el que se ejecuta un script como, por ejemplo, la ruta o Supernodo que ejecuta el script. En los scripts heredados el contexto es implícito, lo que significa que, por ejemplo, se asume que toda referencia a un nodo de una ruta está dentro de la ruta que ejecuta el script.

En los scripts Python el contexto de script se proporciona de forma explícita mediante el módulo `modeler.script`. Por ejemplo, un script Python de ruta puede acceder a la ruta que ejecuta el script mediante el código siguiente:

```
s = modeler.script.stream()
```

A continuación podrán invocarse funciones relacionadas con la ruta a través del objeto devuelto.

Comparativa de comandos y funciones

Los scripts heredados están orientados a comando. Esto significa que cada línea del script suele comenzar con el comando a ejecutar seguido de los parámetros, por ejemplo:

```
connect 'Type':typenode to :filternode  
rename :derivenode as "Compute Total"
```

Python utiliza funciones que suelen invocarse a través de un objeto (módulo, clase u objeto) que define la función, por ejemplo:

```
stream = modeler.script.stream()  
typenode = stream.findByType("type", "Type")  
filternode = stream.findByType("filter", None)  
ruta.link(nodotipo, nodofiltro)  
derive.setLabel("Compute Total")
```

Literales y comentarios

Algunos de los literales y comandos de comentarios que normalmente se utilizan en IBM SPSS Modeler tienen sus equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 260. Correlación de scripts de herencia con scripts Python para literales y comentarios.

| Scripts de herencia | scripts Python |
|---|---|
| Entero, por ejemplo 4 | El mismo |
| Flotante, por ejemplo, 0,003 | El mismo |
| Cadenas entre comillas simples, por ejemplo, 'Hola' | El mismo Nota: Los literales de cadena que contengan caracteres que no sean ASCII deberán tener el prefijo u para garantizar que se representen en Unicode. |
| Cadenas entre comillas dobles, por ejemplo, "Hola de nuevo" | El mismo Nota: Los literales de cadena que contengan caracteres que no sean ASCII deberán tener el prefijo u para garantizar que se representen en Unicode. |
| Cadenas largas, por ejemplo, """Estas es una cadena que abarca varias líneas""" | El mismo |
| Listas, por ejemplo, [1 2 3] | [1, 2, 3] |
| Referencia de variable, por ejemplo, set x = 3 | x = 3 |
| Continuación de línea (\), por ejemplo, set x = [1 2 \ 3 4] | x = [1, 2,\n3, 4] |
| Comentario de bloque, por ejemplo, /* Éste es un comentario largo a través de una línea. */ | """ Este es un comentario largo a través de una línea. """ |
| Comentario de línea, por ejemplo, set x = 3 # make x 3 | x = 3 # make x 3 |
| undef | Ninguno |
| true | True |
| false | False |

Operadores

Algunos de los comandos de operadores que normalmente se utilizan en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 261. Correlación de scripts de herencia con scripts Python para operadores.

| Scripts de herencia | scripts Python |
|---|---|
| NUM1 + NUM2 LIST + ITEM LIST1 + LIST2 | NUM1 + NUM2 LIST.append(ITEM) LIST1.extend(LIST2) |
| NUM1 - NUM2 LIST - ITEM | NUM1 - NUM2 LIST.remove(ITEM) |
| NUM1 * NUM2 | NUM1 * NUM2 |

Tabla 261. Correlación de scripts de herencia con scripts Python para operadores (continuación).

| Scripts de herencia | scripts Python |
|------------------------------------|------------------------------------|
| NUM1 / NUM2 | NUM1 / NUM2 |
| = == | == |
| /= /== | != |
| X ** Y | X ** Y |
| X < Y X <= Y X > Y X >= Y | X < Y X <= Y X > Y X >= Y |
| X div Y X rem Y X mod Y | X // Y X % Y X % Y |
| and or not(EXPR) | and or not EXPR |

Comandos condicionales y de bucle

Algunos comandos condicionales y de bucle utilizados habitualmente en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 262. Correspondencia de scripts de herencia con scripts Python en lo referente a comandos condicionales y de bucle.

| Scripts de herencia | scripts Python |
|--|--|
| for VAR from INT1 to INT2 ... endfor | for VAR in range(INT1, INT2): ... o VAR = INT1 while VAR <= INT2: ... VAR += 1 |
| for VAR in LIST ... endfor | for VAR in LIST: ... |
| for VAR in_fields_to NODE ... endfor | for VAR in NODE.getInputDataModel(): ... |
| for VAR in_fields_at NODE ... endfor | for VAR in NODE.getOutputDataModel(): ... |
| if...then ... elseif...then ... else ... endif | if ...: ... elif ...: ... else: ... |
| with TYPE OBJECT ... endwith | Sin equivalente |

Tabla 262. Correspondencia de scripts de herencia con scripts Python en lo referente a comandos condicionales y de bucle (continuación).

| Scripts de herencia | scripts Python |
|---------------------|---|
| var VAR1 | La declaración de variables no es obligatoria |

Variabes

En los scripts heredados, las variables se declaran antes de ser referenciadas, por ejemplo:

```
var minodo
set minodo = create typenode at 96 96
```

En los scripts Python, las variables se crean la primera vez que se referencian, por ejemplo:

```
minodo = stream.createAt("type", "Type", 96, 96)
```

En los scripts heredados, las referencias a variables deben eliminarse explícitamente mediante el operador `^`, por ejemplo:

```
var minodo
set minodo = create typenode at 96 96
set ^minodo.direction."Age" = Input
```

Al igual que en la mayoría de lenguajes de script, esto no es necesario en los scripts Python, por ejemplo:

```
minodo = stream.createAt("type", "Type", 96, 96)
minodo.setKeyedPropertyValue("direction", "Age", "Input")
```

Tipos modelo, resultado y nodo

En los scripts heredados, a los distintos tipos de objeto (nodo, resultado y modelo) se les suele añadir el tipo al tipo de objeto. Por ejemplo, el nodo Derivar es de tipo `derivenode`:

```
set feature_name_node = create derivenode at 96 96
```

El API de IBM SPSS Modeler en Python no incluye el sufijo `node`, de modo que el nodo Derivar tiene el tipo `derive`, por ejemplo:

```
feature_name_node = stream.createAt("derive", "Feature", 96, 96)
```

La única diferencia en los tipos de nombre entre los scripts Python y los heredados es la ausencia del sufijo de tipo.

Nombres de propiedades

Los nombres de las propiedades son los mismos en scripts heredados y en scripts Python. Por ejemplo, en el nodo Archivo variable, la propiedad que define la ubicación del archivo es `full_filename` en ambos entornos de creación de scripts.

Referencias de nodos

Muchos scripts de herencia utilizan una búsqueda implícita para buscar y acceder al nodo que se ha de modificar. Por ejemplo, los comandos siguientes buscan en la ruta actual un nodo Type con la etiqueta "Type", a continuación, establecen la dirección (o el rol de modelado) del campo "Age" como entrada y el campo "Drug" como destino, esto es, el valor predicho:

```
set 'Type':typenode.direction."Age" = Input
set 'Type':typenode.direction."Drug" = Target
```

En los scripts Python, los objetos de nodo se han de localizar de forma explícita antes de llamar a la función para establecer el valor de propiedad, por ejemplo:

```
typenode = stream.findByType("type", "Type")
typenode.setKeyedPropertyValue("direction", "Age", "Input")
typenode.setKeyedPropertyValue("direction", "Drug", "Target")
```

Nota: En este caso, "Target" debe estar encerrado entre comillas en la cadena.

Los scripts Python pueden utilizar de forma alternativa la enumeración `ModelingRole` del paquete `modeler.api`.

Aunque la versión de los scripts Python puede ser más verbosa, el rendimiento de tiempo de ejecución es mejor ya que la búsqueda del nodo generalmente solo se realiza una vez. En el ejemplo de scripts de herencia, la búsqueda del nodo se realiza para cada comando.

También está soportado buscar nodos por ID (el ID de nodo se puede ver en la pestaña Anotaciones del diálogo del nodo). Por ejemplo, en los scripts de herencia:

```
# id65EMPB9VL87 es el ID de un nodo Type
set @id65EMPB9VL87.direction."Age" = Input
```

El script siguiente muestra el mismo ejemplo en scripts Python:

```
typenode = stream.findByID("id65EMPB9VL87")
typenode.setKeyedPropertyValue("direction", "Age", "Input")
```

Obtener y establecer propiedades

Los scripts de herencia utilizan el comando `set` para asignar un valor. El término que sigue al comando `set` puede ser una definición de propiedad. El script siguiente muestra dos formatos de script posibles para establecer una propiedad:

```
set <referencia de nodo>.<propiedad> = <valor>
set <referencia de nodo>.<propiedad-con claves>.<clave> = <valor>
```

En los scripts Python, se obtiene el mismo resultado utilizando las funciones `setProperty()` y `setKeyedPropertyValue()`, por ejemplo:

```
objeto.setProperty(propiedad, valor)
objeto.setKeyedPropertyValue(propiedad-con claves, clave, valor)
```

En los scripts de herencia, se puede acceder a los valores de las propiedades utilizando el comando `get`, por ejemplo:

```
var n v
set n = get node :filternode
set v = ^n.name
```

En los scripts Python, se obtiene el mismo resultado utilizando la función `getProperty()`, por ejemplo:

```
n = stream.findByType("filter", None)
v = n.getProperty("name")
```

Edición de rutas

En los scripts de herencia, se utiliza el comando `create` para crear un nodo nuevo, por ejemplo:

```
var agg select
set agg = create aggregatenode at 96 96
set select = create selectnode at 164 96
```

En los scripts Python, las rutas tienen varios métodos para crear nodos, por ejemplo:

```
stream = modeler.script.stream()
agg = stream.createAt("aggregate", "Aggregate", 96, 96)
select = stream.createAt("select", "Select", 164, 96)
```

En los scripts de herencia, se utiliza el comando `connect` para crear enlaces entre nodos, por ejemplo:
`connect ^agg to ^select`

En los scripts Python, se utiliza el método `link` para crear enlaces entre nodos, por ejemplo:
`stream.link(agg, select)`

En los scripts de herencia, se utiliza el comando `disconnect` para eliminar enlaces entre nodos, por ejemplo:
`disconnect ^agg from ^select`

En los scripts Python, se utiliza el método `unlink` para eliminar enlaces entre nodos, por ejemplo:
`stream.unlink(agg, select)`

En los scripts de herencia, se utiliza el comando `position` para posicionar los nodos en el lienzo de rutas o entre nodos, por ejemplo:
`position ^agg at 256 256`
`position ^agg between ^myselect and ^mydistinct`

En los scripts Python, se obtiene el mismo resultado utilizando dos métodos separados: `setXYPosition` y `setPositionBetween`. Por ejemplo:
`agg.setXYPosition(256, 256)`
`agg.setPositionBetween(myselect, mydistinct)`

Operaciones de nodo

Algunos de los comandos de operaciones de nodo que normalmente se utilizan en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 263. Correlación de scripts de herencia con scripts Python para operaciones de nodo.

| Scripts de herencia | scripts Python |
|--|--|
| <code>create especificaciónodo at x y</code> | <code>ruta.create(tipo, nombre)</code> <code>ruta.createAt(tipo, nombre, x, y)</code> <code>ruta.createBetween(tipo, nombre, preNode, postNode)</code> <code>ruta.createModelApplier(modelo, nombre)</code> |
| <code>connect desdeNodo to aNodo</code> | <code>ruta.link(desdeNodo, aNodo)</code> |
| <code>delete nodo</code> | <code>ruta.delete(nodo)</code> |
| <code>disable nodo</code> | <code>ruta.setEnabled(nodo, False)</code> |
| <code>enable nodo</code> | <code>ruta.setEnabled(nodo, True)</code> |
| <code>disconnect desdeNodo from aNodo</code> | <code>ruta.unlink(desdeNodo, aNodo)</code> <code>ruta.disconnect(nodo)</code> |
| <code>duplicate nodo</code> | <code>nodo.duplicate()</code> |
| <code>execute nodo</code> | <code>ruta.runSelected(nodos, resultados)</code> <code>ruta.runAll(resultados)</code> |
| <code>flush nodo</code> | <code>nodo.flushCache()</code> |
| <code>position nodo at x y</code> | <code>nodo.setXYPosition(x, y)</code> |
| <code>position nodo between nodo1 and nodo2</code> | <code>nodo.setPositionBetween(nodo1, nodo2)</code> |
| <code>rename nodo as nombre</code> | <code>nodo.setLabel(nombre)</code> |

Bucle

En los scripts de herencia, hay dos opciones de bucle principales a las que se da soporte:

- Bucles de *Valor contado*, en los que una variable de índice se mueve entre dos límites de entero.
- Bucles de *secuencia* que avanzan en bucle por una secuencia de valores, enlazando el valor actual con la variable de bucle.

El script siguiente es un ejemplo de un bucle de valor contado en un script de herencia:

```
for i from 1 to 10
  println ^i
endfor
```

El script siguiente es un ejemplo de un bucle de secuencia en un script de herencia:

```
var items
set items = [a b c d]

for i in items
  println ^i
endfor
```

También existen otros tipos de bucles que se pueden utilizar:

- Iteración por los modelos de la paleta de modelos o por los resultados de la paleta de resultados.
- Iteración por los campos de entrada o salida de un nodo.

Los scripts Python también dan soporte a diferentes tipos de bucles. El script siguiente es un ejemplo de un bucle de valor contado en un script Python:

```
i = 1
while i <= 10:
  print i
  i += 1
```

El script siguiente es un ejemplo de un bucle de secuencia en un script Python:

```
items = ["a", "b", "c", "d"]
for i in items:
  print i
```

El bucle de secuencia es muy flexible y cuando se combina con los métodos de la API de IBM SPSS Modeler puede dar soporte a la mayoría de los casos de uso scripts de herencia. El siguiente ejemplo muestra cómo utilizar un bucle de secuencia en scripts Python para iterar por los campos de salida de un nodo:

```
node = modeler.script.stream().findByType("filter", None)
for column in node.getOutputDataModel().columnIterator():
  print column.getColumnName()
```

Ejecución de rutas

Durante la ejecución de la ruta, el modelo o los objetos de resultados que se generan se añaden a uno de los gestores de objeto. En el script existente, el script debe localizar los objetos creados desde el gestor de objeto, o acceder al resultado generado más recientemente desde el nodo que ha generado el resultado.

La ejecución de rutas en Python es diferente, ya que cualquier objeto de modelo o resultados que genere la ejecución se devuelve una lista que se pasa a la función de ejecución. Esto hace que resulte más sencillo acceder a los resultados de la ejecución de la ruta.

Los scripts de herencia dan soporte a tres comandos de ejecución de ruta:

- `execute_all` ejecuta todos nodos terminales ejecutables en la ruta.

- `execute_script` ejecuta el script de ruta independientemente del valor de la ejecución del script.
- `execute_nodo` ejecuta el nodo especificado.

Los scripts Python dan soporte a un conjunto de funciones similares:

- `ruta.runAll(lista-resultados)` ejecuta todos los nodos terminales ejecutables de la ruta.
- `ruta.runScript(lista-resultados)` ejecuta el script de ruta independientemente del valor de la ejecución del script.
- `ruta.runSelected(matriz-nodos, lista-resultados)` ejecuta el conjunto de nodos especificados en el orden en que se suministran.
- `nodo.run(lista-resultados)` ejecuta el nodo especificado.

En los scripts de herencia, la ejecución de ruta se puede finalizar con el comando `exit` con un código de entero opcional, por ejemplo:

```
exit 1
```

En los scripts Python, se puede obtener el mismo resultado con el script siguiente:

```
modeler.script.exit(1)
```

Acceso a objetos mediante el sistema de archivos y el repositorio

En los scripts heredados se puede abrir una ruta, un modelo o un resultado existentes mediante el comando `open`, por ejemplo:

```
var s
set s = open stream "c:/my streams/modeling.str"
```

En los scripts Python, existe la clase `TaskRunner`, accesible desde la sesión, que puede utilizarse para realizar tareas similares, por ejemplo:

```
taskrunner = modeler.script.session().getTaskRunner()
s = taskrunner.openStreamFromFile("c:/my streams/modeling.str", True)
```

Para guardar un objeto en los scripts heredados, puede utilizarse el comando `save`, por ejemplo:

```
save stream s as "c:/my streams/new_modeling.str"
```

El enfoque de un script Python consiste en utilizar la clase `TaskRunner`, por ejemplo:

```
taskrunner.saveStreamToFile(s, "c:/my streams/new_modeling.str")
```

Las operaciones basadas en un Repositorio de IBM SPSS Collaboration and Deployment Services se soportan en los scripts heredados mediante los comandos `retrieve` y `store`, por ejemplo:

```
var s
set s = retrieve stream "/my repository folder/my_stream.str"
store stream ^s as "/my repository folder/my_stream_copy.str"
```

En los scripts Python, se accede a la funcionalidad equivalente a través del objeto `Repository` asociado a la sesión, por ejemplo:

```
session = modeler.script.session()
repo = session.getRepository()
s = repo.retrieveStream("/my repository folder/my_stream.str", None, None, True)
repo.storeStream(s, "/my repository folder/my_stream_copy.str", None)
```

Nota: El acceso al repositorio exige que la sesión se haya configurado con una conexión de repositorio válida.

Operaciones de ruta

Algunos comandos de operación de ruta que normalmente se utilizan en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 264. Correlación de scripts de herencia con scripts Python para operaciones de ruta.

| Scripts de herencia | scripts Python |
|---|--|
| create stream <i>NOMBREARCHIVO_PREDETERMINADO</i> | <i>ejecutortareas.createStream(nombre, autoConectar, autoGestionar)</i> |
| close stream | <i>ruta.close()</i> |
| clear stream | <i>ruta.clear()</i> |
| get stream <i>ruta</i> | Sin equivalente |
| load stream <i>vía de acceso</i> | Sin equivalente |
| open stream <i>vía de acceso</i> | <i>ejecutortareas.openStreamFromFile(vía de acceso, autoGestionar)</i> |
| save <i>ruta</i> as <i>vía de acceso</i> | <i>ejecutortareas.saveStreamToFile(ruta, vía de acceso)</i> |
| retrieve stream <i>vía de acceso</i> | <i>repositorio.retrieveStream(vía de acceso, versión, etiqueta, autoGestionar)</i> |
| store <i>ruta</i> as <i>vía de acceso</i> | <i>repositorio.storeStream(ruta, vía de acceso, etiqueta)</i> |

Operaciones de modelo

Algunos de los comandos de operación de modelo que normalmente se utilizan en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 265. Correlación de scripts de herencia con scripts Python para operaciones de modelo.

| Scripts de herencia | scripts Python |
|---|---|
| open model <i>vía de acceso</i> | <i>ejecutortareas.openModelFromFile(vía de acceso, autoGestionar)</i> |
| save <i>modelo</i> as <i>vía de acceso</i> | <i>ejecutortareas.saveModelToFile(modelo, vía de acceso)</i> |
| retrieve model <i>vía de acceso</i> | <i>repositorio.retrieveModel(vía de acceso, versión, etiqueta, autoGestionar)</i> |
| store <i>modelo</i> as <i>vía de acceso</i> | <i>repositorio.storeModel(modelo, vía de acceso, etiqueta)</i> |

Operaciones de resultado de documento

Algunos de los comandos de operaciones de resultado de documento que normalmente se utilizan en IBM SPSS Modeler tienen sus comandos equivalentes en los scripts Python. Esto puede ayudarle a convertir los scripts de SPSS Modeler de herencia existentes en scripts Python para utilizarlos en IBM SPSS Modeler 17.

Tabla 266. Correlación de scripts de herencia con scripts Python para operaciones de resultado de documento.

| Scripts de herencia | scripts Python |
|---|--|
| open output <i>vía de acceso</i> | <i>ejecutortareas.openDocumentFromFile(vía de acceso, autoGestionar)</i> |
| save <i>resultado</i> as <i>vía de acceso</i> | <i>ejecutortareas.saveDocumentToFile(resultado, vía de acceso)</i> |

Tabla 266. Correlación de scripts de herencia con scripts Python para operaciones de resultado de documento (continuación).

| Scripts de herencia | scripts Python |
|---|--|
| retrieve output <i>vía de acceso</i> | <code>repositorio.retrieveDocument(vía de acceso, versión, etiqueta, autoGestionar)</code> |
| store resultado as <i>vía de acceso</i> | <code>repositorio.storeDocument(resultado, vía de acceso, etiqueta)</code> |

Otras diferencias entre scripts heredados y scripts Python

Los scripts heredados soportan la manipulación de proyectos de IBM SPSS Modeler. Los scripts Python no soportan esto actualmente.

Los scripts heredados proporcionan cierto soporte de carga de objetos de *estado* (combinaciones de rutas y modelos). Los objetos de estado han caído en desuso desde IBM SPSS Modeler 8.0. Los scripts Python no soportan objetos de estado.

Los scripts Python proporcionan las siguientes funciones adicionales no disponibles en los scripts heredados:

- Definiciones de clase y función.
- Manejo de errores.
- Soporte más sofisticado de entrada/salida.
- Módulos externos y de terceros.

Avisos

Esta información se ha desarrollado para productos y servicios ofrecidos en los EE.UU. Este material puede estar disponible en IBM en otros idiomas. Sin embargo, es posible que deba ser propietario de una copia del producto o de la versión del producto en dicho idioma para acceder a él.

Es posible que IBM no ofrezca los productos, servicios o características que se tratan en este documento en otros países. El representante local de IBM le puede informar sobre los productos y servicios que están actualmente disponibles en su localidad. Cualquier referencia a un producto, programa o servicio de IBM no pretende afirmar ni implicar que solamente se pueda utilizar ese producto, programa o servicio de IBM. En su lugar, se puede utilizar cualquier producto, programa o servicio funcionalmente equivalente que no infrinja los derechos de propiedad intelectual de IBM. Sin embargo, es responsabilidad del usuario evaluar y comprobar el funcionamiento de todo producto, programa o servicio que no sea de IBM.

IBM puede tener patentes o solicitudes de patente en tramitación que cubran la materia descrita en este documento. Este documento no le otorga ninguna licencia para estas patentes. Puede enviar preguntas acerca de las licencias, por escrito, a:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
EE.UU.*

Para consultas sobre licencias relacionadas con información de doble byte (DBCS), póngase en contacto con el departamento de propiedad intelectual de IBM de su país o envíe sus consultas, por escrito, a:

*Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokio 103-8510, Japón*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROPORCIONA ESTA PUBLICACIÓN "TAL CUAL", SIN GARANTÍAS DE NINGUNA CLASE, NI EXPLÍCITAS NI IMPLÍCITAS, INCLUYENDO, PERO SIN LIMITARSE A, LAS GARANTÍAS IMPLÍCITAS DE NO VULNERACIÓN, COMERCIALIZACIÓN O ADECUACIÓN A UN PROPÓSITO DETERMINADO. Algunas jurisdicciones no permiten la renuncia a las garantías explícitas o implícitas en determinadas transacciones; por lo tanto, es posible que esta declaración no sea aplicable a su caso.

Esta información puede incluir imprecisiones técnicas o errores tipográficos. Periódicamente, se efectúan cambios en la información aquí y estos cambios se incorporarán en nuevas ediciones de la publicación. IBM puede realizar en cualquier momento mejoras o cambios en los productos o programas descritos en esta publicación sin previo aviso.

Las referencias hechas en esta publicación a sitios web que no son de IBM se proporcionan sólo para la comodidad del usuario y no constituyen de modo alguno un aval de esos sitios web. La información de esos sitios web no forma parte de la información de este producto de IBM y la utilización de esos sitios web se realiza bajo la responsabilidad del usuario.

IBM puede utilizar o distribuir la información que se le proporcione del modo que considere adecuado sin incurrir por ello en ninguna obligación con el remitente.

Los titulares de licencias de este programa que deseen tener información sobre el mismo con el fin de permitir: (i) el intercambio de información entre programas creados independientemente y otros programas (incluido este) y (ii) el uso mutuo de la información que se ha intercambiado, deberán ponerse en contacto con:

*IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
EE.UU.*

Esta información estará disponible, bajo las condiciones adecuadas, incluyendo en algunos casos el pago de una cuota.

El programa bajo licencia que se describe en este documento y todo el material bajo licencia disponible los proporciona IBM bajo los términos de las Condiciones Generales de IBM, Acuerdo Internacional de Programas Bajo Licencia de IBM o cualquier acuerdo equivalente entre las partes.

Los ejemplos de datos de rendimiento y de clientes citados se presentan solamente a efectos ilustrativos. Los resultados reales de rendimiento pueden variar en función de las configuraciones específicas y condiciones de operación.

La información relacionada con productos no IBM se ha obtenido de los proveedores de esos productos, de sus anuncios publicados o de otras fuentes disponibles públicamente. IBM no ha probado esos productos y no puede confirmar la exactitud del rendimiento, la compatibilidad ni ninguna otra afirmación relacionada con productos no IBM. Las preguntas sobre las posibilidades de productos que no son de IBM deben dirigirse a los proveedores de esos productos.

Las declaraciones sobre el futuro rumbo o intención de IBM están sujetas a cambio o retirada sin previo aviso y representan únicamente metas y objetivos.

Esta información contiene ejemplos de datos e informes utilizados en operaciones comerciales diarias. Para ilustrarlos lo máximo posible, los ejemplos incluyen los nombres de las personas, empresas, marcas y productos. Todos estos nombres son ficticios y cualquier parecido con personas o empresas comerciales reales es pura coincidencia.

Marcas comerciales

IBM, el logotipo de IBM e `ibm.com` son marcas registradas o marcas comerciales de International Business Machines Corp., registradas en muchas jurisdicciones en todo el mundo. Otros nombres de productos y servicios podrían ser marcas registradas de IBM u otras compañías. En Internet hay disponible una lista actualizada de las marcas registradas de IBM, en "Copyright and trademark information", en www.ibm.com/legal/copytrade.shtml.

Adobe, el logotipo Adobe, PostScript y el logotipo PostScript son marcas registradas o marcas comerciales de Adobe Systems Incorporated en Estados Unidos y/o otros países.

Intel, el logotipo de Intel, Intel Inside, el logotipo de Intel Inside, Intel Centrino, el logotipo de Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium y Pentium son marcas comerciales o marcas registradas de Intel Corporation o sus filiales en Estados Unidos y otros países.

Linux es una marca registrada de Linus Torvalds en Estados Unidos, otros países o ambos.

Microsoft, Windows, Windows NT, y el logotipo de Windows son marcas comerciales de Microsoft Corporation en Estados Unidos, otros países o ambos.

UNIX es una marca registrada de The Open Group en Estados Unidos y otros países.

Java y todas las marcas comerciales y los logotipos basados en Java son marcas comerciales o registradas de Oracle y/o sus afiliados.

Términos y condiciones para la documentación del producto

Los permisos para utilizar estas publicaciones se otorgan de acuerdo con los términos y condiciones siguientes.

Aplicabilidad

Estos términos y condiciones son adicionales a los términos de uso del sitio web de IBM.

Uso personal

Estas publicaciones se pueden reproducir para uso personal no comercial siempre que se conserven todos los avisos de propiedad. No puede distribuir, visualizar ni realizar trabajos derivados de estas publicaciones, ni de partes de las mismas, sin el consentimiento expreso de IBM.

Uso comercial

Puede reproducir, distribuir y visualizar estas publicaciones únicamente dentro de la empresa a condición de que se conserven todos los avisos de propiedad. No puede realizar trabajos derivados de estas publicaciones, ni de partes de las mismas, ni reproducirlas, distribuirlas o visualizarlas fuera de su empresa sin el consentimiento expreso de IBM.

Derechos

Excepto de la forma explícitamente otorgada en este permiso, no se otorga ningún permiso, licencia ni derecho, ni explícito ni implícito, sobre las publicaciones ni a ninguna otra información, datos, software u otra propiedad intelectual contenida en ellas.

IBM se reserva el derecho de retirar los permisos aquí otorgados siempre que, a su discreción, el uso de las publicaciones sea perjudicial para su interés o cuando, según determine IBM, las instrucciones anteriores no se sigan correctamente.

No puede descargar, exportar ni volver a exportar esta información si no es cumpliendo totalmente todas las leyes y regulaciones aplicables, incluyendo las leyes y regulaciones de exportación de los Estados Unidos.

IBM NO GARANTIZA EL CONTENIDO DE ESTAS PUBLICACIONES. LAS PUBLICACIONES SE PROPORCIONAN "TAL CUAL" Y SIN GARANTÍA DE NINGUNA CLASE, NI EXPLÍCITA NI IMPLÍCITA, INCLUYENDO PERO SIN LIMITARSE A LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN, NO VULNERACIÓN E IDONEIDAD PARA UN FIN DETERMINADO.

Índice

A

- acceder a los resultados de la ejecución de ruta
 - modelo de contenido de tabla 54
 - modelo de contenido JSON 57
 - modelo de contenido XML 55
- acceder a resultados de ejecución de la ruta 53, 58
- adición de atributos 24
- API de creación de scripts
 - acceso a objetos generados 41
 - buscar 37
 - ejemplo 37
 - introducción 37
 - manejo de errores 42
 - metadatos 38
 - obtener un directorio 37
 - parámetros de ruta 43
 - parámetros de sesión 43
 - parámetros de Supernodos 43
 - scripts autónomos 47
 - valores globales 46
 - varias rutas 47
- applyextension, propiedades 270
- applyocsvm, propiedades 276
- applyxgboostlinearnode, propiedades 281
- applyxgboosttreenode, propiedades 281
- Árbol de decisión de MS
 - propiedades de scripts de nodos 283, 285
- argumentos
 - archivo de comandos 68
 - conexión con el servidor 66
 - conexión de IBM SPSS Collaboration and Deployment Services Repository 67
 - Conexión del repositorio de IBM SPSS Analytic Server 67
 - sistema 64
- atravesar los nodos 33

B

- bloques de código 19
- bucles
 - uso en scripts 49
- bucles en rutas 6, 7
- buscar nodos 29

C

- cadena 17
 - cambio entre minúsculas y mayúsculas 49
- campos
 - desactivación de los scripts 161
- caracteres que no son ASCII 22
- clave de iteración
 - bucle en scripts 8

CLEM

- scripts 1
- Clúster de secuencias de MS
 - propiedades de scripts de nodos 285
- comando clear generated palette 53
- comando de conjunto múltiple 69
- comando for 49
- comando retrieve 50
- comando store 50
- comentarios 18
- comprobación de errores
 - scripts 52
- configuración de propiedades 30
- contraseñas
 - adición a scripts 52
 - codificadas 66
- contraseñas codificadas
 - adición a scripts 52
- cplexnode, propiedades 129
- creación de nodos 31, 33
- crear una clase 24

D

- definición de atributos 24
- definir métodos 24
- definir una clase 24
- derive_stbnode
 - propiedades 109
- Diagramas 27

E

- ejecución condicional de rutas 6, 11
- Ejecución de rutas 27
- ejecución de scripts 12
- ejemplos 20
- exportModelToFile 41
- extensionexportnode, propiedades 333
- extensionimportnode, propiedades 89
- extensionmodelnode, propiedades 203
- extensionoutputnode, propiedades 310
- extensionprocessnode, propiedades 112

F

- función lowertoupper 49
- funciones
 - bucle 359
 - comentarios 358
 - condicionales 359
 - literales 358
 - operaciones de modelo 365
 - operaciones de nodo 362
 - operaciones de resultado de documento 365
 - operaciones de ruta 365
 - operadores 358
 - referencias de objeto 358
- funciones de cadena 49

G

- GLE, modelos
 - propiedades de scripts de nodos 216, 273

I

- IBM SPSS Modeler
 - ejecución desde la línea de comandos 63
- identificadores 19
- inheritance 25
- instrucciones 19
- interrupción de scripts 12

J

- Jython 15

L

- línea de comandos
 - ejecutar IBM SPSS Modeler 63
 - lista de argumentos 64, 66, 67
 - parámetros 65
 - scripts 52
 - varios argumentos 68
- listas 16

M

- marcas
 - argumentos de la línea de comandos 63
 - combinación de varias marcas 68
- métodos matemáticos 21
- migrar
 - acceder a objetos 364
 - borrar rutas, salida y gestores de modelos 34
 - bucle 363
 - comandos 357
 - conceptos básicos 357
 - configuración de propiedades 361
 - contexto de los scripts 357
 - diferencias generales 357
 - editar rutas 361
 - ejecución de rutas 363
 - funciones 357
 - nombres de propiedad 360
 - obtener propiedades 361
 - referencias de nodos 360
 - repositorio 364
 - sistema de archivos 364
 - tipos de modelos 360
 - tipos de nodo 360
 - tipos de salida 360
 - variables 360
 - varios 366

| | | | | | |
|--|----------|--|---------------|---|---------------|
| modelado de bases de datos | 283 | Modelos de clasificador automático | | modelos de secuencias | |
| modelo de contenido de tabla | 54 | propiedades de scripts de nodos | 266 | propiedades de scripts de nodos | 242, 278 |
| modelo de contenido JSON | 57 | modelos de detección de anomalías | | modelos de selección de características | |
| modelo de contenido XML | 55 | propiedades de scripts de nodos | 179, 265 | aplicación | 5 |
| modelos | | Modelos de IBM SPSS Statistics | | propiedades de scripts de nodos | 207, 272 |
| nombres de scripts | 353, 355 | propiedades de scripts de nodos | 342 | scripts | 5 |
| modelos apriori | | modelos de K-medias | | modelos de serie temporal | |
| propiedades de scripts de nodos | 181, 265 | propiedades de scripts de nodos | 221, 273 | propiedades de scripts de nodos | 253, 279 |
| Modelos Apriori de Oracle | | modelos de K-medias de Netezza | | modelos de serie temporal de modalidad continua | |
| propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 122 |
| modelos autonuméricos | | Modelos de K-medias de Oracle | | modelos de series temporales | |
| propiedades de scripts de nodos | 188 | propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 253, 257, 279 |
| Modelos autonuméricos | | modelos de la máquina de vectores de soporte | | Modelos de series temporales de Netezza | |
| propiedades de scripts de nodos | 267 | propiedades de scripts de nodos | 248, 278 | propiedades de scripts de nodos | 294 |
| Modelos bayesianos adaptativos de Oracle | | modelos de la máquina de vectores de soporte lineal | | modelos del vecino más próximo | |
| propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 231, 275 | propiedades de scripts de nodos | 222 |
| modelos bietápicos | | modelos de listas de decisiones | | modelos discriminantes | |
| propiedades de scripts de nodos | 261, 280 | propiedades de scripts de nodos | 200, 269 | propiedades de scripts de nodos | 202, 269 |
| modelos bietápicos AS | | Modelos de máquinas de vectores de soporte de Oracle | | modelos generados | |
| propiedades de scripts de nodos | 262, 280 | propiedades de scripts de nodos | 287, 293 | nombres de scripts | 353, 355 |
| modelos C5.0 | | modelos de Microsoft | | Modelos GLMM | |
| propiedades de scripts de nodos | 192, 267 | propiedades de scripts de nodos | 283, 285 | propiedades de scripts de nodos | 212, 272 |
| modelos CARMA | | Modelos de Netezza | | modelos KNN | |
| propiedades de scripts de nodos | 193, 268 | propiedades de scripts de nodos | 294 | propiedades de scripts de nodos | 274 |
| modelos causales temporales | | Modelos de NMF de Oracle | | Modelos KNN de Netezza | |
| propiedades de scripts de nodos | 249 | propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 294, 305 |
| modelos CHAID | | modelos de Oracle | | modelos kohonen | |
| propiedades de scripts de nodos | 197, 268 | propiedades de scripts de nodos | 287 | propiedades de scripts de nodos | 223 |
| Modelos de agrupación en clústeres divisivo de Netezza | | modelos de red bayesiana | | modelos Kohonen | |
| propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 190 | propiedades de scripts de nodos | 274 |
| modelos de árbol aleatorio | | Modelos de red bayesiana de Netezza | | modelos lineales | |
| propiedades de scripts de nodos | 238, 277 | propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 225, 274 |
| modelos de árbol C&R | | modelos de red neuronal | | modelos lineales generalizados | |
| propiedades de scripts de nodos | 194, 268 | propiedades de scripts de nodos | 232, 275 | propiedades de scripts de nodos | 209, 272 |
| Modelos de árbol de decisión de Oracle | | Modelos de redes bayesianas | | Modelos lineales generalizados de Netezza | |
| propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 267 | propiedades de scripts de nodos | 294 |
| modelos de árboles de decisión de Netezza | | Modelos de regresión de Cox | | Modelos lineales generalizados de Oracle | |
| propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 199, 269 | propiedades de scripts de nodos | 287 |
| Modelos de árboles de regresión de Netezza | | modelos de regresión lineal | | modelos linear-AS | |
| propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 240, 277, 278 | propiedades de scripts de nodos | 226, 274 |
| modelos de Autoclúster | | Modelos de regresión lineal de Netezza | | modelos LSVM | |
| propiedades de scripts de nodos | 267 | propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 231 |
| Modelos de bayesiano ingenuo de Netezza | | modelos de regresión logística | | Modelos Oracle AI | |
| propiedades de scripts de nodos | 294, 305 | propiedades de scripts de nodos | 227, 275 | propiedades de scripts de nodos | 287 |
| Modelos de bayesiano ingenuo de Oracle | | modelos de respuesta de autoaprendizaje | | Modelos para LMD de Oracle | |
| propiedades de scripts de nodos | 287, 293 | propiedades de scripts de nodos | 243, 278 | propiedades de scripts de nodos | 287, 293 |

| | | | | | |
|---------------------------------|--------|---------------------------------------|-----|--------------------------------------|--------|
| modelos PCA/Factorial | | nodo de Modelo de extensión | | nodo Generación de análisis de serie | |
| propiedades de scripts de nodos | 206, | propiedades de scripts de nodos | 203 | temporal | |
| 271 | | Nodo de origen de Excel | | propiedades | 126 |
| modelos QUEST | | propiedades | 88 | nodo Gráfico de tiempo | |
| propiedades de scripts de nodos | 236, | Nodo de origen de IBM SPSS Statistics | | propiedades | 176 |
| 276 | | Collection | | Nodo graphboard | |
| modelos SLRM | | propiedades | 341 | propiedades | 166 |
| propiedades de scripts de nodos | 243, | Nodo de origen de importación de TWC | | nodo histogram | |
| 278 | | propiedades | 99 | propiedades | 168 |
| modelos SVM | | nodo de origen Geospacial | | nodo history | |
| propiedades de scripts de nodos | 248 | propiedades | 94 | propiedades | 143 |
| modelos tcm | | nodo de origen Recopilación de datos | | nodo Informe | |
| propiedades de scripts de nodos | 279 | propiedades | 86 | propiedades | 314 |
| modelos Tree-AS | | Nodo de origen SAS | | nodo Intervalos de tiempo | |
| propiedades de scripts de nodos | 259, | propiedades | 94 | propiedades | 150 |
| 280 | | nodo de origen Vista de datos | | nodo Intervalos de tiempo AS | |
| modificar rutas | 31, 33 | propiedades | 104 | propiedades | 135 |
| | | Nodo de origen XML | | nodo Malla | |
| | | propiedades | 104 | propiedades | 177 |
| | | nodo de predicción espaciotemporal | | nodo Malla direccional | |
| | | propiedades | 244 | propiedades | 177 |
| | | nodo de Resultados de extensión | | Nodo Marcas | |
| | | propiedades | 310 | propiedades | 149 |
| | | Nodo de salida de IBM SPSS Statistics | | nodo matrix | |
| | | Collection | | propiedades | 311 |
| | | propiedades | 342 | nodo Medias | |
| | | nodo de simulación de ajuste | | propiedades | 313 |
| | | propiedades | 317 | nodo origen Analytic Server | |
| | | nodo de simulación de evaluación | | propiedades | 81 |
| | | propiedades | 316 | nodo Origen de IBM Cognos | |
| | | nodo de transformación | | propiedades | 82 |
| | | propiedades | 322 | nodo Origen de IBM Cognos TM1 | |
| | | nodo de Transformación de extensión | | propiedades | 97, 98 |
| | | propiedades | 112 | nodo partition | |
| | | Nodo de transformación de IBM SPSS | | propiedades | 144 |
| | | Statistics Collection | | nodo plot | |
| | | propiedades | 341 | propiedades | 174 |
| | | nodo de transformación R | | nodo R Build | |
| | | propiedades | 117 | propiedades de scripts de nodos | 191 |
| | | nodo de visualización de mapas | | nodo Reclasificar | |
| | | propiedades | 169 | propiedades | 145 |
| | | nodo derive | | nodo Reestructurar | |
| | | propiedades | 138 | propiedades | 147 |
| | | nodo distinct | | nodo Reglas de asociación | |
| | | propiedades | 111 | propiedades | 182 |
| | | nodo distribution | | nodo Rellenar | |
| | | propiedades | 163 | propiedades | 141 |
| | | nodo ensemble | | nodo Reordenar | |
| | | propiedades | 140 | propiedades | 146 |
| | | nodo Estadísticos | | nodo Reorg. campos | |
| | | propiedades | 318 | propiedades | 146 |
| | | nodo evaluation | | nodo Reprojection | |
| | | propiedades | 164 | propiedades | 146 |
| | | nodo Exportar base de datos | | nodo Routput | |
| | | propiedades | 328 | propiedades | 315 |
| | | Nodo Exportar SAS | | nodo sample | |
| | | propiedades | 335 | propiedades | 118 |
| | | nodo filter | | nodo Seleccionar | |
| | | propiedades | 142 | propiedades | 119 |
| | | nodo fixedfile | | nodo Sim Eval | |
| | | propiedades | 92 | propiedades | 316 |
| | | nodo flatfilenode | | nodo Sim Fit | |
| | | propiedades | 334 | propiedades | 317 |
| | | nodo Fundir | | nodo Sim Gen | |
| | | propiedades | 114 | propiedades | 95 |
| | | nodo G. múltiple | | nodo Simulación de generación | |
| | | propiedades | 173 | propiedades | 95 |

N

| | | | | | |
|---|----------|--|--|--|--|
| Nodo Adición de RFM | | | | | |
| propiedades | 115 | | | | |
| nodo aggregate | | | | | |
| propiedades | 107 | | | | |
| Nodo Agrupación en clústeres automática | | | | | |
| propiedades de scripts de nodos | 187 | | | | |
| Nodo Análisis de RFM | | | | | |
| propiedades | 147 | | | | |
| Nodo analysis | | | | | |
| propiedades | 307 | | | | |
| nodo anonymize | | | | | |
| propiedades | 131 | | | | |
| nodo append | | | | | |
| propiedades | 107 | | | | |
| nodo Archivo var. | | | | | |
| propiedades | 100 | | | | |
| Nodo Auditoría de datos | | | | | |
| propiedades | 308 | | | | |
| nodo balance | | | | | |
| propiedades | 108 | | | | |
| nodo Base de datos | | | | | |
| propiedades | 84 | | | | |
| nodo binning | | | | | |
| propiedades | 135 | | | | |
| nodo Cajas-Espacio-Tiempo | | | | | |
| propiedades | 109, 120 | | | | |
| Nodo Clasificador automático | | | | | |
| propiedades de scripts de nodos | 185 | | | | |
| nodo Colección | | | | | |
| propiedades | 162 | | | | |
| nodo de datos de usuario | | | | | |
| propiedades | 100 | | | | |
| nodo de Exportación de extensión | | | | | |
| propiedades | 333 | | | | |
| Nodo de exportación de IBM SPSS | | | | | |
| Statistics Collection | | | | | |
| propiedades | 343 | | | | |
| nodo de exportación de recopilación de | | | | | |
| datos | | | | | |
| propiedades | 332 | | | | |
| Nodo de exportación Excel | | | | | |
| propiedades | 332 | | | | |
| nodo de exportación XML | | | | | |
| propiedades | 339 | | | | |
| nodo de importación de extensión | | | | | |
| propiedades | 89 | | | | |

- nodo sort
 - propiedades 120
- nodo STP
 - propiedades 244
- nodo Tabla
 - propiedades 319
- nodo Tipo
 - propiedades 155
- nodo Transponer
 - propiedades 154
- nodo Val. globales
 - propiedades 316
- nodos
 - desenlazar nodos 31
 - enlazar nodos 31
 - importación 33
 - información 34
 - recorrido en bucle en scripts 49
 - referencia de nombres 353
 - suprimir 33
 - volver a poner 33
- nodos de exportación
 - propiedades de scripts de nodos 325
- nodos de gráficos
 - propiedades de los scripts 161
- nodos de modelado
 - propiedades de scripts de nodos 179
- nodos de origen
 - propiedades 77
- nodos de resultados
 - propiedades de los scripts 307
- nombres de campos
 - cambio entre minúsculas y mayúsculas 49
- nugget
 - propiedades de scripts de nodos 265
- nugget de nodo de reglas de asociación
 - propiedades 266
- nugget de nodo STP
 - propiedades 278
- nuggets de modelo
 - nombres de scripts 353, 355
 - propiedades de scripts de nodos 265

O

- O-clúster de Oracle
 - propiedades de scripts de nodos 287, 293
- objetos de resultados
 - nombres de scripts 355
- objetos del modelo
 - nombres de scripts 353, 355
- ocsvmnode, propiedades 348
- operaciones 16
- Optimización de CPLEX, nodo
 - propiedades 129
- orden de ejecución
 - modificación con scripts 49
- orden de ejecución de rutas
 - modificación con scripts 49
- orientado a objetos 23

P

- palabra clave generada 53
- parámetros 5, 69, 71, 73
 - scripts 16
 - Supernodos 351
- parámetros de intervalo 5, 69, 71
- pasar argumentos 20
- preparación automática de datos
 - propiedades 132
- properties autodataprepnode 132
- propiedad stream.nodes 49
- propiedades
 - nodos de modelado de bases de datos 283
 - nodos Filtrar 69
 - ruta 73
 - scripts 69, 71, 179, 265, 325
 - scripts comunes 71
 - Supernodos 351
 - propiedades applyrandomtrees 277
 - propiedades applyts 279
 - propiedades associationrulesnode 182
 - propiedades de aggregatenode 107
 - propiedades de analysisnode 307
 - propiedades de
 - anomalydetectionnode 179
 - propiedades de anonymizenode 131
 - propiedades de appendnode 107
 - propiedades de
 - applyanomalydetectionnode 265
 - propiedades de applyapriorinode 265
 - propiedades de
 - applyassociationrulesnode 266
 - propiedades de
 - applyautoclassifiernode 266
 - propiedades de
 - applyautoclusternode 267
 - propiedades de
 - applyautonumericnode 267
 - propiedades de applybayesnetnode 267
 - propiedades de applyc50node 267
 - propiedades de applycarmanode 268
 - propiedades de applycartnode 268
 - propiedades de applychaidnode 268
 - propiedades de applycoxregnode 269
 - propiedades de
 - applydecisionlistnode 269
 - propiedades de
 - applydiscriminantnode 269
 - propiedades de applyfactornode 271
 - propiedades de
 - applyfeatureselectionnode 272
 - propiedades de
 - applygeneralizedlinearnode 272
 - Propiedades de applygle 273
 - Propiedades de applyglmnode 272
 - propiedades de applykmeansnode 273
 - propiedades de applyknnnode 274
 - propiedades de applykohonennode 274
 - propiedades de applylinearnode 274
 - Propiedades de applylinearnode 274
 - propiedades de applylogregnode 275
 - propiedades de applysvmnode 275
 - propiedades de
 - applysmlogisticnode 285
 - propiedades de
 - applysmneuralnetworknode 285
 - propiedades de
 - applysmregressionnode 285
 - propiedades de
 - applysmsequenceclusternode 285
 - propiedades de
 - applysmtimeseriesnode 285
 - propiedades de
 - applysmstreenode 285
 - propiedades de
 - applynetzezbayesnode 305
 - propiedades de
 - applynetzezadectreenode 305
 - propiedades de
 - applynetzezadivclusternode 305
 - propiedades de
 - applynetzezakmeansnode 305
 - propiedades de
 - applynetzezaknnode 305
 - propiedades de
 - applynetzezalineressionnode 305
 - propiedades de
 - applynetzezanaivebayesnode 305
 - propiedades de
 - applynetzezapcanode 305
 - propiedades de
 - applynetzezaregtreenode 305
 - propiedades de
 - applyneuralnetnode 275
 - propiedades de
 - applyneuralnetworknode 276
 - propiedades de applyoraabnnode 293
 - propiedades de
 - applyoradecisiontreenode 293
 - propiedades de
 - applyorakmeansnode 293
 - propiedades de applyoranbnode 293
 - propiedades de applyoranmfnode 293
 - propiedades de
 - applyoraoclusternode 293
 - propiedades de applyorasvmnode 293
 - propiedades de applyquestnode 276
 - propiedades de applyr 277
 - propiedades de
 - applyregressionnode 278
 - propiedades de
 - applyselflearningnode 278
 - propiedades de applysequencenode 278
 - propiedades de applystpnnode 278
 - propiedades de applysvmnode 278
 - Propiedades de applytcmmode 279
 - propiedades de
 - applytimeseriesnode 279
 - propiedades de applytreeas 280
 - propiedades de applytwostepAS 280
 - propiedades de applytwostepnode 280
 - propiedades de apriorinode 181
 - propiedades de asexport 325
 - propiedades de asimport 81
 - propiedades de astimeintervalnode 135
 - propiedades de autoclassifiernode 185
 - propiedades de autonumericnode 188
 - propiedades de balancenode 108
 - propiedades de bayesnet 190
 - propiedades de binningnode 135
 - propiedades de buildr 191
 - propiedades de c50node 192
 - propiedades de carmanode 193
 - propiedades de cartnode 194
 - propiedades de chaidnode 197

- propiedades de collectionnode 162
- propiedades de coxregnode 199
- propiedades de dataauditnode 308
- propiedades de databaseexportnode 328
- propiedades de databasenode 84
- propiedades de
 - datacollectionexportnode 332
- propiedades de
 - datacollectionimportnode 86
- propiedades de dataviewimport 104
- propiedades de decisionlist 200
- propiedades de derivenode 138
- propiedades de directwebnode 177
- propiedades de discriminantnode 202
- propiedades de distinctnode 111
- propiedades de distributionnode 163
- propiedades de ensemblenode 140
- propiedades de evaluationnode 164
- Propiedades de excelexportnode 332
- propiedades de excelimportnode 88
- propiedades de factornode 206
- propiedades de featureselectionnode 5, 207
- propiedades de fillernode 141
- propiedades de filternode 142
- propiedades de fixedfilenode 92
- propiedades de flatfilenode 334
- propiedades de genlinnode 209
- Propiedades de gle 216
- Propiedades de glmnode 212
- Propiedades de graphboardnode 166
- propiedades de histogramnode 168
- propiedades de historynode 143
- propiedades de kmeansnode 221
- propiedades de knnnode 222
- propiedades de kohonenode 223
- Propiedades de logregnode 227
- propiedades de lsvmnode 231
- Propiedades de mapvisualization 169
- propiedades de matrixnode 311
- propiedades de meansnode 313
- propiedades de mergenode 114
- propiedades de msassocnode 283
- propiedades de msbayesnode 283
- propiedades de msclusternode 283
- propiedades de mslogisticnode 283
- propiedades de
 - msneuralnetworknode 283
- propiedades de msregressionnode 283
- propiedades de
 - mssequenceclusternode 283
- propiedades de mstimeseriesnode 283
- propiedades de mstreenode 283
- propiedades de multiplotnode 173
- Propiedades de netezabayesnode 294
- Propiedades de netezadectreenode 294
- Propiedades de
 - netezadivclusternode 294
- Propiedades de netezzaglmnode 294
- Propiedades de netezzakmeansnode 294
- Propiedades de netezzaknnode 294
- Propiedades de
 - netezalineregressionnode 294
- propiedades de
 - netezanaivebayesnode 294
- Propiedades de netezzapcanode 294
- Propiedades de netezaregtreenode 294

- Propiedades de
 - netezatimeseriesnode 294
- propiedades de neuralnetnode 232
- Propiedades de neuralnetwork 235
- propiedades de nodo de agrupación en clústeres automática 187
- propiedades de
 - numericpredictornode 188
- propiedades de oraabnnode 287
- Propiedades de oraainode 287
- Propiedades de oraapriorinode 287
- Propiedades de oradecisiontreenode 287
- Propiedades de oraglmnode 287
- Propiedades de orakmeansnode 287
- propiedades de oramdlnode 287
- propiedades de oranbnode 287
- Propiedades de oranmfnode 287
- Propiedades de oraoclusternode 287
- propiedades de orasvmnode 287
- propiedades de outputfilenode 334
- propiedades de partitionnode 144
- propiedades de plotnode 174
- propiedades de questnode 236
- propiedades de reclassifynode 145
- Propiedades de regressionnode 240
- propiedades de reordernode 146
- propiedades de reportnode 314
- propiedades de reprojectnode 146
- propiedades de restructurenode 147
- propiedades de rfmanalysisnode 147
- propiedades de routputnode 315
- propiedades de Rprocessnode 117
- propiedades de samplenode 118
- propiedades de sasexportnode 335
- propiedades de sasimportnode 94
- propiedades de scripts de nodos 283
 - nodos de exportación 325
 - nodos de modelado 179
 - nuggets de modelo 265
- propiedades de selectnode 119
- propiedades de sequencenode 242
- propiedades de setglobalsnode 316
- propiedades de settoflagnode 149
- propiedades de simevalnode 316
- propiedades de simfitnode 317
- propiedades de simgenode 95
- propiedades de slrmnode 243
- propiedades de sortnode 120
- Propiedades de spacetimeboxes 120
- propiedades de statisticsexportnode 343
- propiedades de statisticsimportnode 5, 341
- propiedades de statisticsmodelnode 342
- propiedades de statisticsnode 318
- propiedades de statisticsoutputnode 342
- propiedades de
 - statisticstransformnode 341
- propiedades de stpnode 244
- propiedades de svmnode 248
- propiedades de tablenode 319
- propiedades de tcmlnode 249
- propiedades de timeintervalnode 150
- propiedades de timeplotnode 176
- propiedades de timeseriesnode 257
- propiedades de transformnode 322
- propiedades de transposenode 154
- propiedades de treeas 259

- propiedades de twostepAS 262
- propiedades de twostepnode 261
- propiedades de typenode 5, 155
- propiedades de userinputnode 100
- propiedades de variablefilenode 100
- propiedades de webnode 177
- propiedades de xmlexportnode 339
- propiedades de xmlimportnode 104
- propiedades del nodo
 - Cajas-Espacio-Tiempo 109
- propiedades del nodo cognosimport 82
- propiedades del nodo gsdata_import 94
- propiedades del nodo tm1import 98
- propiedades del nodo
 - tm1odataimport 97
- propiedades estructuradas 69
- propiedades lineales 225
- propiedades linear-AS 226
- propiedades randomtrees 238
- propiedades rfmaggregatenode 115
- propiedades streamingtimeseries 122
- propiedades streamingts 126
- propiedades ts 253
- Python 15
 - scripts 16
- Python, modelos
 - propiedades de scripts de nodos 276, 281

R

- Red neuronal de MS
 - propiedades de scripts de nodos 283, 285
- redes neuronales
 - propiedades de scripts de nodos 235, 276
- referencia a nodos 29
 - buscar nodos 29
 - configuración de propiedades 30
- Regresión lineal de MS
 - propiedades de scripts de nodos 283, 285
- Regresión logística de MS
 - propiedades de scripts de nodos 283, 285
- Repositorio de IBM SPSS Analytic Server Repository
 - argumentos de la línea de comandos 67
- Repositorio de IBM SPSS Collaboration and Deployment Services
 - argumentos de la línea de comandos 67
 - scripts 50
- resultados de ejecución de la ruta,
 - acceder 53, 58
 - modelo de contenido de tabla 54
 - modelo de contenido JSON 57
 - modelo de contenido XML 55
- rutas
 - bucle 6, 7
 - comando de conjunto múltiple 69
 - ejecución 27
 - ejecución condicional 6, 11
 - modificándose 31
 - propiedades 73

rutas (*continuación*)
scripts 1, 27

S

script
abreviaturas utilizadas 70
bucles visuales 6, 7
clave de iteración 8
comprobación de errores 52
conceptos básicos 1, 15
ejecución condicional 6, 11
en Supernodos 5
interfaz de usuario 1, 4, 5
nodos de resultados 307
orden de ejecución de rutas 49
propiedades comunes 71
scripts de herencia 358, 359, 362, 365
scripts Python 358, 359, 362, 365
selección de campos 10
sintaxis 16, 17, 18, 19, 20, 21, 22, 23,
24, 25
variable de iteración 9

scripts
almacenamiento 1
bucle 6, 7
clave de iteración 8
compatibilidad con versiones
anteriores 53
contexto 28
desde la línea de comandos 52
Diagramas 27
ejecución 12
ejecución condicional 6, 11
importación desde archivos de
texto 1
interrupción 12
modelos de selección de
características 5
nodos de gráficos 161
rutas 1, 27
rutas de supernodo 27
scripts autónomos 1, 27
Scripts de Supernodo 1, 27
selección de campos 10
variable de iteración 9

scripts autónomos 1, 4, 27

seguridad
contraseñas codificadas 52, 66

Series temporales de MS
propiedades de scripts de nodos 285

servidor
argumentos de la línea de
comandos 66

sistema
argumentos de la línea de
comandos 64

sistema de coordenadas de reproyección
propiedades 146

SMOTE, nodo
propiedades 345

smotenode, propiedades 345

supernodo 69

Supernodo
ruta 27

Supernodos
configuración de propiedades 351

Supernodos (*continuación*)
parámetros 351
propiedades 351
rutas 27
scripts 1, 5, 6, 27, 351

SVM de una clase, nodo
propiedades 348

T

twcimport, propiedades de nodo 99

V

variable de iteración
bucle en scripts 9

variables
scripts 16

variables ocultas 25

X

XGBoost Linear, nodo
propiedades 347

XGBoost Tree, nodo
propiedades 345

xgboostlinearnode, propiedades 347

xgboosttreenode, propiedades 345



Impreso en España