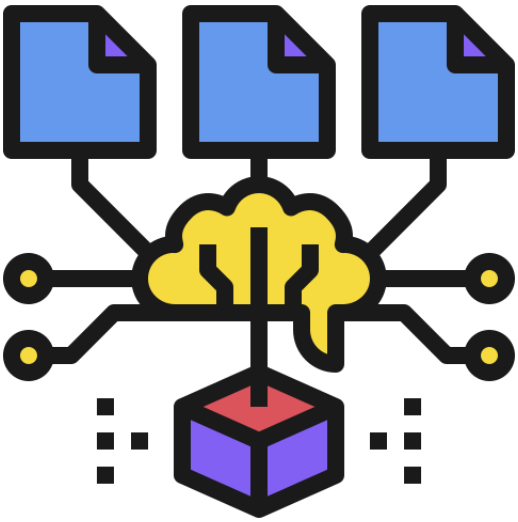
Actividad Modelado de Datos

Modelado de Datos sobre CSV a través de RapidMiner



Sistemas Inteligentes - M31

Grado de Ingeniería Informática

Curso 2023-2024

Identificador de Alumno

Diego Rodríguez Sanz - 22167749

Github – [Tarea-Modeling]

Director de Proyecto

Christian Vladimi Sucuzhanay Arevalo

Fecha: 21/11/2023

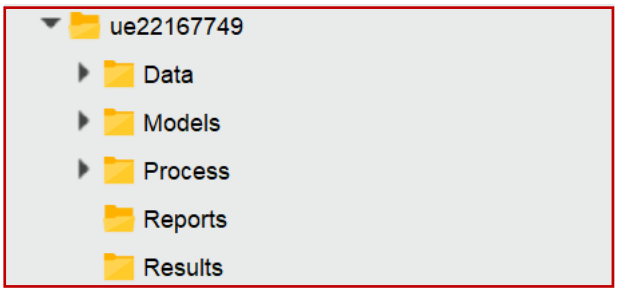
Índice del Documento

Capítulo 1. Elaboración de Subtareas

A lo largo de dicho apartado del documento, se irán presentando todas las subtareas pertinentes para el análisis y modelado de los datos proporcionados, de la manera más profesional posible y abordando a su vez todos y cada uno de los apartados pertinentes.

1.1 Organización del Entorno de Trabajo RapidMiner

Para la elaboración de dicha tarea, se ha procedido a crear en nuestro "LocalRepository", todas y cada una de sus carpetas, además de inicializar el correspondiente repositorio local con la herramienta "Git":



Como se puede observar, se ha creado una carpeta principal que almacena toda la información (Datos, Modelos, Procesos, Reportes y Resultados) con el expediente correspondiente.

Tras ello, se procedió a inicializar el repositorio local en dicho directorio con los comandos que se presentan a continuación:

```
PS C:\Users\diego\Documents\RapidMiner\Local Repository\ue22167749> git init
Initialized empty Git repository in C:/Users/diego/Documents/RapidMiner/Local Repository/ue22167749/.git/
PS C:\Users\diego\Documents\RapidMiner\Local Repository\ue22167749> git remote add origin https://github.com/DiegoK
36/Tarea-Modeling-Sistemas-Inteligentes
PS C:\Users\diego\Documents\RapidMiner\Local Repository\ue22167749> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Data/
        Process/

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\diego\Documents\RapidMiner\Local Repository\ue22167749>
```

A su vez, se revisó que todo funcionó correctamente comprobando el estado del repositorio con el comando "git status".

Tras ello, se inicializaron todos los archivos sobre la rama main para su posterior subida en GitHub tras terminar la actividad al completo:

1.2 Guardado y Almacenamiento del Dataset

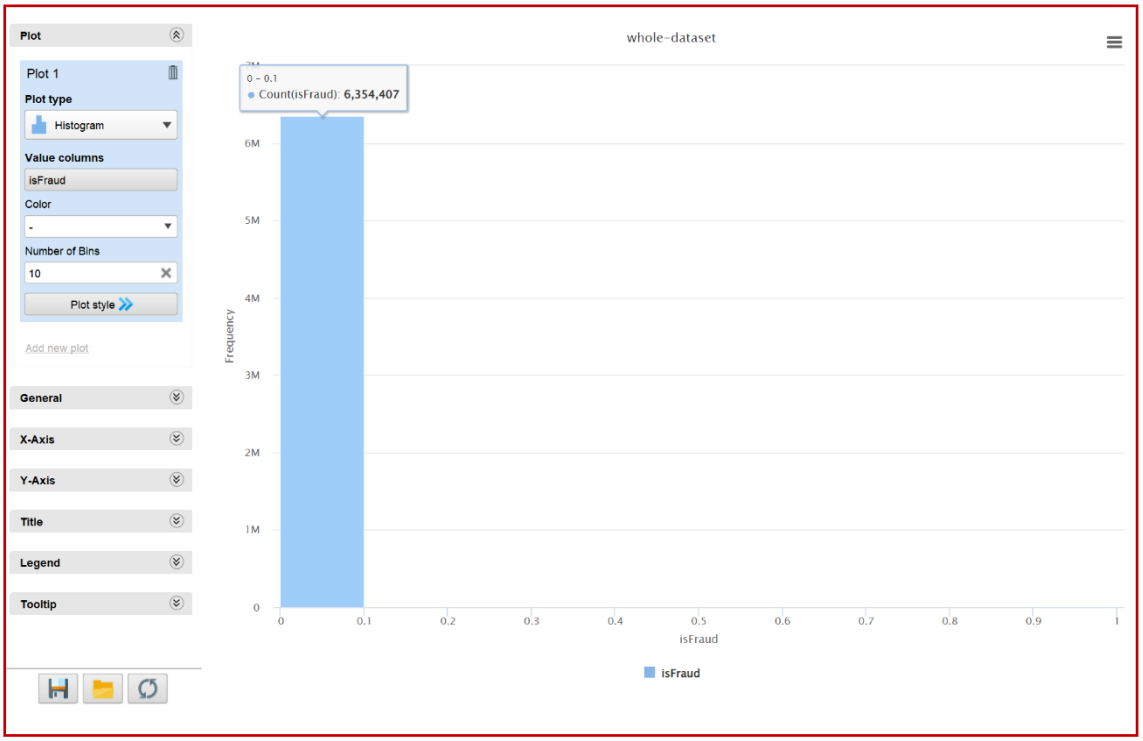
Tras la descarga del Dataset correspondiente, lo guardamos sobre la carpeta "Data" y procedemos al almacenamiento de dichos datos mediante los operadores de RapidMiner de la siguiente manera:

A través de dichos operadores, accedemos a la ruta del CSV con el operador "Read CSV" y lo abrimos, para luego almacenarlo en forma de conjunto de datos mediante el operador "Store" del RapidMiner.

Tras realizarlo, la información queda almacenada en "whole-dataset" para poder operar más adelante al completo con dichos datos.

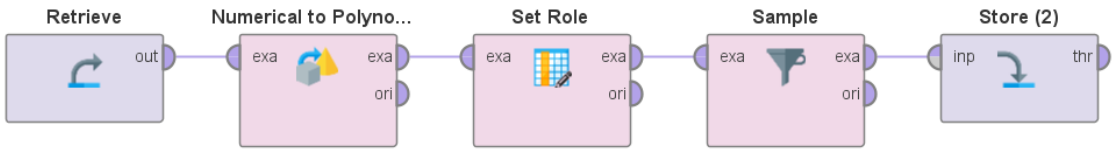
1.3 Balanceado de la Variable "isFraud"

A continuación, se muestra como la variable "isFraud" se encuentra muy desbalanceada respecto al resto:



Debido a ello, debemos balancearla y generar un nuevo conjunto de datos (Dataset) mucho más balanceado, como se muestra a continuación.

Para ello, utilizamos los siguientes operadores de RapidMiner:



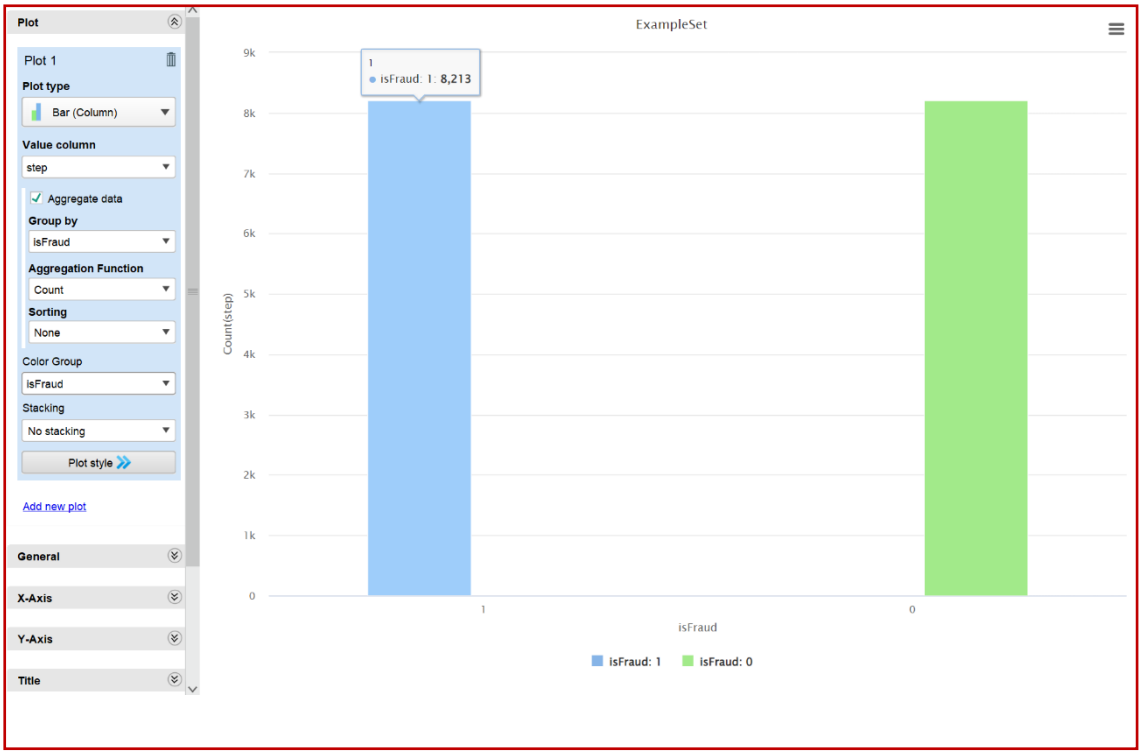
A continuación, se describe el uso que se hace de cada uno de los operadores:

- **Retrieve** → Permite acceder a toda la información almacenada en "**whole-dataset**".
- **Numerical to Polynomial** → Convierte el formato numérico de dicha variable "**isFraud**" en uno legible para nuestro "**Sample**".
- **Set Role** → Permite establecer nuestra variable "isFraud" como label para que más adelante lo pueda procesar el operador "**Sample**"
- **Sample** → En dicho operador, establecemos el atributo en **0 y 1** para la correcta visualización de este al comprobar el balanceamiento:

class	size
0	8213
1	8213

- **Store** → Almacena el conjunto de datos bajo el nombre "balanced-dataset".

Con todo ello, obtenemos el valor balanceado y la siguiente gráfica de valor:

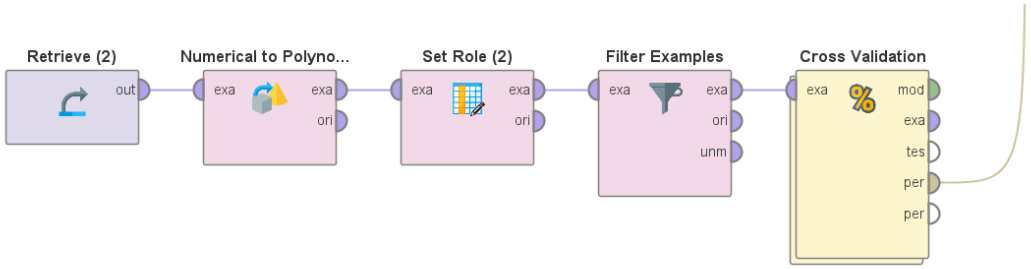


Con ello, completamos las tareas 3 y 4, ya que ya tenemos el Dataset balanceado.

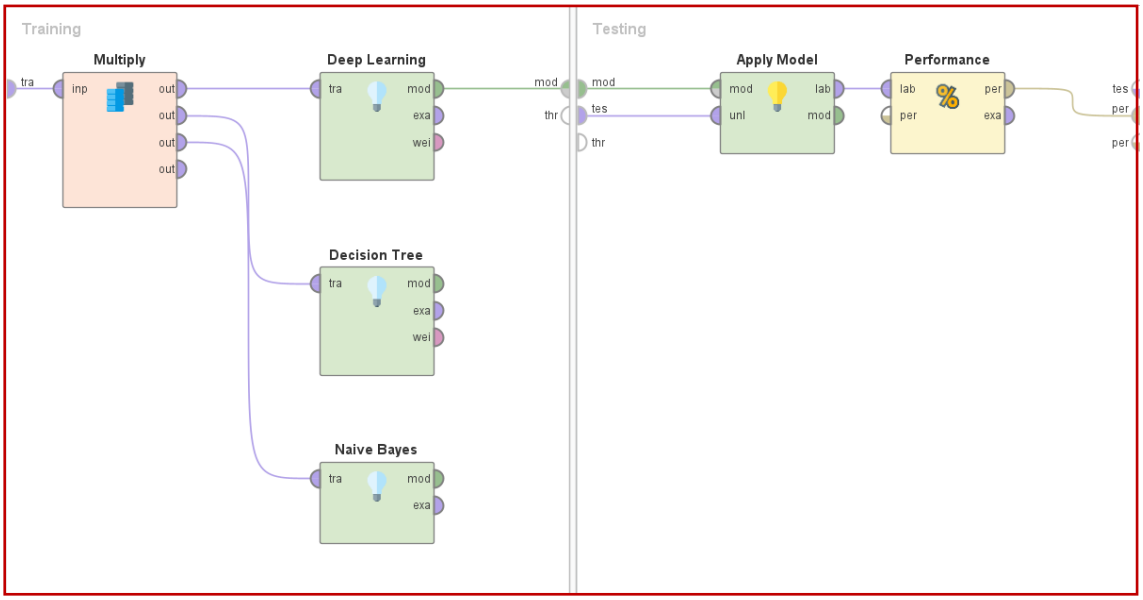
Como se puede observar, ya contenemos el conjunto de datos balanceado para realizar nuestro modelado.

1.4 Modelado Final del Dataset y subida a Github

Finalmente, utilizamos “Cross-Validation” para obtener el mejor modelo para este conjunto de datos (Dataset), con los siguientes operadores:



Utilizamos los operadores mencionados anteriormente, añadiendo “Filter Examples” para limpiar posibles valores innecesarios y “Cross Validation” para probar cada uno de los modelos de la siguiente manera:



Primero lo probamos con el modelo “Deep Learning” y con los operadores “Apply Model” y “Performance” lo aplicamos sobre el Dataset y comprobamos como de eficiente es para él.

Finalmente, de entre los tres modelos que probé: “Deep Learning”, “Árbol de Decisiones” y “Redes Bayesianas” realizando lo mencionado anteriormente, obtengo que finalmente el modelo que más se adapta en este caso es:

“DEEP LEARNING”

Con esto, damos por finalizada la actividad y procedemos a la correspondiente subida al repositorio de GitHub mediante un “Push”:

```
PS C:\Users\diego\Documents\RapidMiner\Local Repository\ue22167749> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 20 threads
Compressing objects: 100% (8/8), done.
error: RPC failed; HTTP 408 curl 22 The requested URL returned error: 408
send-pack: unexpected disconnect while reading sideband packet
Writing objects: 100% (10/10), 395.25 MiB | 7.46 MiB/s, done.
Total 10 (delta 1), reused 0 (delta 0), pack-reused 0
fatal: the remote end hung up unexpectedly
Everything up-to-date
```

Repositorio → [Tarea-Modeling]

Con todo ello, ya tenemos la actividad de modelado realizada al completo.