SEGUNDA ENTREGA DEL PROYECTO

Por:

Ricardo Osorio Castro

CC: 1036785264

Materia:

Modelos I

Profesor

Raul Ramos Pollan

Facultad de Ingeniería

Universidad de Antioquia - Colombia

2023

1. **Recordemos el problema**

   El objetivo del modelo es predecir la probabilidad de que un cliente no pague el saldo de su tarjeta de crédito en el futuro en función de su perfil de cliente mensual. La variable binaria objetivo se calcula observando una ventana de rendimiento de 18 meses después del último extracto de la tarjeta de crédito, y si el cliente no paga el monto adeudado en 120 días después de la última fecha del extracto, se considera un evento de incumplimiento.

   https://www.kaggle.com/competitions/amex-default-prediction/data

2. **Preprocesamiento de Datos**

   Para comenzar con el procesamiento de datos nos encontramos con el primer gran problema: La cantidad de datos es enorme.
   Tiene 924621 número de muestras y 396 columnas de todo el dataset.
   En la competencia podemos ver que el tamaño total del dataset es de 50.31GB.

   Esto supuso un reto enorme, ya que con el tamaño de tales archivos usar Google Colab era imposible, no solo por su dimensión, que ya era bastante el tiempo tomado para subir solo el archivo de train con 16,39GB, sino además de la cantidad de RAM requerida para hacer cualquier procesamiento con estos datos.

   En vista a esto, encontramos en la sección “Discussion” de kaggle que este era un problema que ya se habían planteado antes, y ya había encontrado una solución: Feather.

   Una compresión de datos 9 veces mayor con Feather

   Feather es un formato de archivo binario ligero y eficiente para el almacenamiento y la transferencia de datos tabulares, especialmente en el contexto de la ciencia de datos y el análisis de datos. Algunos aspectos sobre Feather son:

   Eficiencia de Almacenamiento: Feather está diseñado para ser un formato de archivo eficiente en términos de almacenamiento. Al comprimir los datos y aprovechar estructuras de datos en memoria como DataFrames, Feather puede reducir significativamente el espacio necesario para almacenar datos tabulares.

   Velocidad de Lectura/Escritura: Feather es rápido tanto para la lectura como para la escritura de datos, lo que lo convierte en una opción eficaz para transferir datos entre diferentes plataformas y herramientas de análisis de datos.

   Datos Tabulares: Feather se utiliza comúnmente para datos tabulares, como conjuntos de datos estructurados que se asemejan a una tabla de base de datos o un DataFrame en pandas.

   Con base a esto, la usuaria de kaggle: Ruchi Bhatia, ha creado versiones en formato feather y parquet de los archivos csv y los recopiló en un conjunto de datos.

Data	Size of csv file	Size of parquet file	Size of feather file
train_data	16.4 GB	6.7 GB	**1.8 GB**
test_data	33.8 GB	13.7 GB	**3.6 GB**

Para el proceso que realizó para hacer este cambio fue el siguiente:

Columnas numéricas -> float16 y
Columnas categóricas -> categoría

Con esto, los archivos feather eran mucho más “trabajables” que los originales, por esto decidimos usar estos archivos para el entrenamiento.
Puede acceder a ellos a través del siguiente enlace:
https://www.kaggle.com/datasets/ruchi798/parquet-files-amexdefault-prediction/data/train_data.ftr?select=train_data.ftr

3. **Limpieza de datos**

Ya con los datos a disposición, después de verlos y hacer una tabla describiendo algunos de sus valores, proseguimos a ver el número de datos faltantes o “Nan” en el dataset, y debido a que algunas columnas se veían demasiados Nan hicimos un resumen con las columnas para las cuales su porcentaje de datos con Nan eran mayor al 90% y el resultado fue el siguiente:

```
Columnas con más del 90% de NaN:
D_49     0.901376
D_73     0.989902
R_9      0.943499
B_29     0.931046
D_87     0.999301
D_88     0.998915
D_106    0.902133
D_108    0.994768
D_110    0.994335
D_111    0.994335
B_39     0.993920
B_42     0.987078
D_132    0.901911
D_134    0.964801
D_135    0.964801
D_136    0.964801
D_137    0.964801
D_138    0.964801
```

Un pequeño problema

Cuando estábamos por seguir con el proceso, notamos que al dataset le faltaba la columna “target”, la cual era la principal, ya que está es a la que el modelo debe predecir.
Después nos dimos cuenta que la columna target estaba en otro archivo: train_labels.csv , lo cual al parecer es una práctica muy común, pero no lo sabíamos. Lo siguiente fue unir ambos archivos para ya tener el dataset completo. Fin del problema.

Continuando

Ya con el dataset completo, realizamos algunos gráficos para ver la relación entre las columnas con más Nan y la variable target, para darnos una idea de su comportamiento.

Teniendo conocimiento de esto, nuestra primera opción es borrar estas columnas ya que sería difícil rellenarlas con tan poca información disponible de las mismas, pero en vista a nuestro pobre conocimiento en el tema, decidimos esperar y preguntar a personas con más saberes acerca de esto (cómo el profesor o el monitor) para tomar una decisión definitiva.

4. **Codificación de variables categóricas**

Para nuestro caso tenemos 13 variables categóricas, tomadas por nosotros, pero según la competencia solo son 11.

```
['customer_ID', 'S_2', 'D_63', 'D_64', 'D_66', 'D_68', 'B_30', 'B_38', 'D_114', 'D_116', 'D_117', 'D_120', 'D_126']
```

Lo que sucede es que estamos tomando el ID del cliente y la fecha del registro cómo variables categóricas. Pero en el momento no sabemos cómo solucionar esto, entonces es otra tarea pendiente.

Para las siguientes variables si pudimos observar que eran categóricas, con los siguientes valores:

```
D_63 ['CL' 'CO' 'CR' 'XL' 'XM' 'XZ']
D_64 ['-1' 'O' 'R' 'U']
D_66 [0. 1.]
D_68 [0. 1. 2. 3. 4. 5. 6.]
B_30 [0. 1. 2.]
B_38 [1. 2. 3. 4. 5. 6. 7.]
D_114 [0. 1.]
D_116 [0. 1.]
D_117 [-1.  1.  2.  3.  4.  5.  6.]
D_120 [0. 1.]
D_126 [-1.  0.  1.]
```

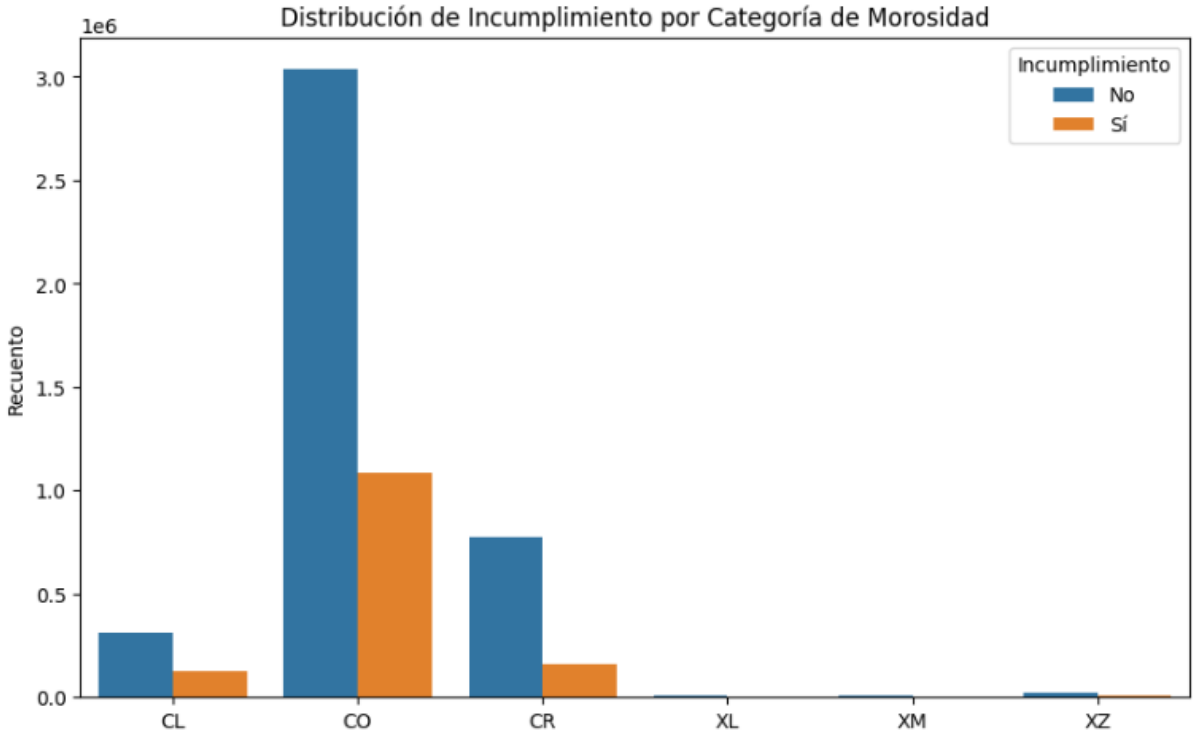
Para referenciar, esto es lo que significan las letras antes del número:
D_* = Delinquency variables
S_* = Spend variables
P_* = Payment variables
B_* = Balance variables
R_* = Risk variables

Debido a que el problema es del sector financiero, nuestro conocimiento es muy limitado al respecto, por lo que por el momento solo realizamos la codificación de una variable: D_63

Podemos observar que la variable D_63 contiene unos valores un tanto extraños, pues aqui esta su posible definición:

A menudo, estos códigos se utilizan para indicar diferentes estados de morosidad o incumplimiento de un préstamo o deuda. Algunos ejemplos comunes de significados en el contexto financiero podrían ser:
-'CL': Collection (en inglés), que indica que la cuenta se encuentra en proceso de cobro.
-'CO': Charged Off (en inglés), que significa que la deuda se considera incobrable.
-'CR': Current (en inglés), que indica que la cuenta está al día y no hay morosidad.
-'XL', 'XM', 'XZ': Estos códigos pueden ser específicos de un sistema de codificación particular y no es posible determinar su significado sin más información.

Con estos datos realizamos un gráfico para ver la relación entre estas y la variable target:



Cómo observamos que las tres últimas columnas apenas y aparecen en el dataset las unimos en una y pasamos a hacer la codificación one-hot.

5. Tareas pendientes

Tenemos claro que el camino es largo por recorrer, pero aún no podemos comenzar el entrenamiento ya que no hemos acabado por completo con la limpieza de los datos. Para esto tenemos las siguientes tareas a priori:

1. Reducir aún más el tamaño del dataset: Aun con el formato feather, el tamaño sigue siendo bastante grande, por lo que que podríamos cambiar el tipo de algunas variables a un formato más pequeño (Customer_ID, S_2,...), cambiar el formato del archivo, usar Chunks, etc.
2. Comenzar a probar modelos: Debemos comenzar lo más rápido posible con el ensayo y error para encontrar el modelo que mejor se adapte a nuestro caso.