# AVANCE: DETECCIÓN DE FRAUDE FINANCIERO MEDIANTE MODELOS DE APRENDIZAJE AUTOMÁTICO

Por:

Juan Pablo Gómez López

## Resumen

En el presente informe se describe brevemente los que se tiene del proyecto hasta el momento. Se realizó la exploración de datos (EDA) de los datos de entrenamiento de la competición IEEE-CIS Fraud Detection con el apoyo de las discusiones sobre EDA [1, 2, 3, 4] del foro planteadas en la misma plataforma. Los principales problemas que se tuvieron fueron respecto al uso de la memoria RAM dentro de colab. Queda pendiente el EDA para el dataset de prueba y generar un primer modelo como primera iteración.

## Descripción de avances

Antes de comenzar a describir los resultados de la exploración de los datos, recordemos brevemente el contexto del proyecto. El objetivo es predecir la probabilidad de que una transacción en línea sea clasificada como fraudulenta. Nos dan cuatro .csv: train_transaction, train_identity, test_transaction y test_identity; unos están asociados directamente a las transacciones y otros asociados a la identidad de los clientes.

En esta discusión de kaggle se da la descripción detallada del significado de cada característica del dataset. Es importante anotar que muchas de las características fueron enmascaradas ("engineered") por los host de la competición por motivos de protección de los datos; lo anterior hace más difícil el proceso de exploración de los datos, pues tenemos nombres de características genéricas que no dan información alguna.

Comenzada la carga de todos los datos y algunas sentencias básicas de exploración de datos en colab se encontró el primer problema: **no había suficiente memoria RAM** disponible en el notebook de colab al ejecutar algunos comandos.

Al principio, sabiendo que el dataset es altamente desbalanceado (se encontró que sólo el 3.50% de los datos son etiquetados como fraude, el resto no lo son) se pensaba aplicarle a los dataset de entrenamiento y de prueba un *random undersampling*, para obtener nuevos dataset balanceados y de mucho menor tamaño para poder hacer exploración inicial.

Sin embargo, se encontró una función que lograba disminuir significativamente el uso de memoria RAM al cargar los datos en memoria dentro de colab [1]. A continuación se muestra el código.

```python
## Function to reduce the DF size
def reduce_mem_usage(df, verbose=True):
  numerics = ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']
  start_mem = df.memory_usage().sum() / 1024**2
  for col in df.columns:
      col_type = df[col].dtypes
      if col_type in numerics:
          c_min = df[col].min()
          c_max = df[col].max()
          if str(col_type)[:3] == 'int':
              if c_min > np.iinfo(np.int8).min and c_max < np.iinfo(np.int8).max:
                  df[col] = df[col].astype(np.int8)
              elif c_min > np.iinfo(np.int16).min and c_max < np.iinfo(np.int16).max:
                  df[col] = df[col].astype(np.int16)
              elif c_min > np.iinfo(np.int32).min and c_max < np.iinfo(np.int32).max:
                  df[col] = df[col].astype(np.int32)
              elif c_min > np.iinfo(np.int64).min and c_max < np.iinfo(np.int64).max:
                  df[col] = df[col].astype(np.int64)
          else:
              if c_min > np.finfo(np.float16).min and c_max < np.finfo(np.float16).max:
                  df[col] = df[col].astype(np.float16)
              elif c_min > np.finfo(np.float32).min and c_max < np.finfo(np.float32).max:
                  df[col] = df[col].astype(np.float32)
              else:
                  df[col] = df[col].astype(np.float64)
  end_mem = df.memory_usage().sum() / 1024**2
  if verbose:
    print('Mem. usage decreased to {:5.2f} Mb ({:.1f}% reduction)'
          .format(end_mem, 100 * (start_mem - end_mem) / start_mem))
  return df
```

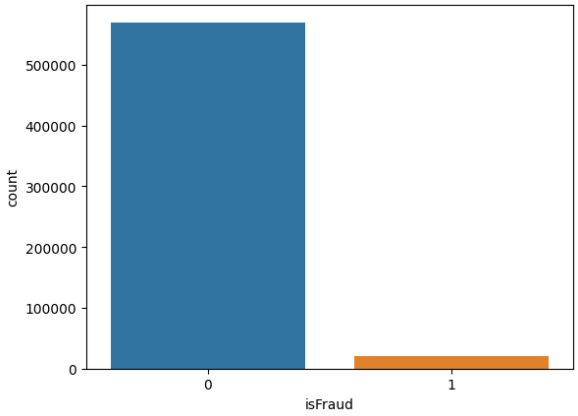
Ejemplo de uso:

```python
df_trans = pd.read_csv("data/train_transaction.csv")
df_trans = reduce_mem_usage(df_trans)
df_trans.head()
```

```
Mem. usage decreased to 542.35 Mb (69.4% reduction)
```

A parte de la carga de datos, se debe tener mucho cuidad con los métodos que se invocan, especialmente los que implican hacer una copia interna a los datos, que eran los que terminaban por tumbar la sesión del colab, perdiendo así los resultados.

Con el inconveniente de la memoria resuelto, se muestra a continuación algunas gráficas mostrando información relevante del problema. En la siguiente imagen se muestra la distribución de muestras que son fraude respecto a las que no son, en el dataset de entrenamiento. Allí se demuestra que efectivamente el dataset está desbalanceado, como se mencionó anteriormente.




Con la ayuda de un método para construir una tabla resumen [2], se puede obtener información de los valores del dataset de entrenamiento. A continuación se muestran algunos resultados relevantes.

| | Nombre | dtypes | % NaN | Unique | Primer Valor | Segundo Valor | Tercer Valor |
|---|---|---|---|---|---|---|---|
| 0 | TransactionID | int32 | 0.0 | 590540 | 2987000 | 2987001 | 2987002 |
| 1 | isFraud | int8 | 0.0 | 2 | 0 | 0 | 0 |
| 2 | TransactionDT | int32 | 0.0 | 573349 | 86400 | 86401 | 86469 |
| 3 | TransactionAmt | float16 | 0.0 | 8195 | 68.5 | 29.0 | 59.0 |
| 4 | ProductCD | object | 0.0 | 5 | W | W | W |

| | Nombre | dtypes | % NaN | Unique | Primer Valor | Segundo Valor | Tercer Valor |
|---|---|---|---|---|---|---|---|
| 4 | ProductCD | object | 0.0 | 5 | W | W | W |
| 17 | C1 | float16 | 0.0 | 1495 | 1.0 | 1.0 | 1.0 |
| 18 | C2 | float16 | 0.0 | 1167 | 1.0 | 1.0 | 1.0 |
| 19 | C3 | float16 | 0.0 | 27 | 0.0 | 0.0 | 0.0 |

| | Nombre | dtypes | % NaN | Unique | Primer Valor | Segundo Valor | Tercer Valor |
|---|---|---|---|---|---|---|---|
| 31 | D1 | float16 | 0.214888 | 641 | 14.0 | 0.0 | 0.0 |
| 32 | D2 | float16 | 47.549192 | 641 | NaN | NaN | NaN |
| 33 | D3 | float16 | 44.514851 | 649 | 13.0 | NaN | NaN |
| 34 | D4 | float16 | 28.604667 | 808 | NaN | 0.0 | 0.0 |
| 35 | D5 | float16 | 52.467403 | 688 | NaN | NaN | NaN |
| 36 | D6 | float16 | 87.606767 | 829 | NaN | NaN | NaN |
| 37 | D7 | float16 | 93.409930 | 597 | NaN | NaN | NaN |

La ventaja de tener la tabla resumen es que podemos ver el porcentaje de valores nulos, la cantidad de valores únicos, el tipo de dato y los tres primeros valores (las tres primeras filas), según el nombre de cada columna. De estas tablas se observa que hay columnas o que no tienen datos nulos, o que tienen un porcentaje considerable (o casi todos) de estos.

## Conclusiones

- Se debe hacer una selección de las características más relevantes que podrían ser relevantes para la generación de un modelo, ya que se tienen en total 394 columnas y muchas de ellas sin ninguna descripción que den contexto explícito de lo que significan.
- El reto principalmente será en la ingeniería de características, ya que al leer los resultados de los ganadores de la competición resultados de los ganadores de la competición , se observa mucho trabajo en esta parte, además de introducir técnicas y conceptos no vistos en el curso hasta el momento.

## Propuestas para desarrollo futuro

Se planea hacer lo siguiente para generar un primer modelo:

1. Eliminar las columnas que tengan más del 50% de los datos como NaN, tanto como para el dataset de entrenamiento como el de prueba.
2. Realizar random undersampling en el dataset de entrenamiento.
3. Aplicar one hot encoding a las variables categóricas
4. Entrenar un modelo de regresión logística y quizás un SVC
5. Observar resultados

Dependiendo de los resultados, se continuará procesando el dataset hasta encontrar un modelo aceptable.

## Referencias de las discusiones


[1] Kaggle, «IEEE Fraud Detection - First Look and EDA,» 23 04 2023. [En línea]. Available: https://www.kaggle.com/code/robikscube/ieee-fraud-detection-first-look-and-eda/notebook#Train-vs-Test-are-Time-Series-Split.

[2] Kaggle, «Extensive EDA and Modeling XGB Hyperopt,» 23 04 2023. [En línea]. Available: https://www.kaggle.com/code/kabure/extensive-eda-and-modeling-xgb-hyperopt/notebook#Target-Distribution.

[3] Kaggle, «EDA and models,» 23 04 2023. [En línea]. Available: https://www.kaggle.com/code/artgor/eda-and-models/notebook#Data-loading-and-overview.

[4] Kaggle, «IEEE Transaction columns Reference,» 23 04 2023. [En línea]. Available: https://www.kaggle.com/code/alijs1/ieee-transaction-columns-reference/notebook.