CAN YOU DETECT FRAUD FROM CUSTOMER TRANSACTIONS?

PRESENTADO POR:

AURA LUZ MORENO DÍAZ,

CC 43758500, INGENIERÍA INDUSTRIAL

EVELYN ZHARICK SAEZ GALLEGO,

CC 1006776490, INGENIERÍA AMBIENTAL

PRESENTADO A:

RAÚL RAMOS POLLAN

UNIVERSIDAD DE ANTIOQUIA

FACULTAD DE INGENIERIA

2023

PREPROCESAMIENTO DEL DATASET

El presente trabajo hace parte de la segunda entrega para el curso Inteligencia Artificial para las Ciencias y la Ingeniería. Para esto, debemos dar un testimonio acerca de las dificultades y los aciertos que tuvimos luego de ver los videos del curso e investigar en fuentes alternativas para el desarrollo hasta la fecha de nuestro trabajo predictivo.

La mayor parte del tiempo fue invertido en conocer como traer los datos desde Kaggle. Se intentó inicialmente cargar los datos desde google drive pero no era funcional. Se intentó desde un hosting alternativo haciendo el llamado desde colab, pero tampoco funcionó, luego de leer toda la documentación disponible, pudimos crear la API KEY y traer los datos directamente desde la competencia de Kaggle y que funcionara correctamente.

Hemos decidido trabajar con el dataset de la empresa VESTA en la cual se trata de detectar cuando una transacción es fraudulenta. La variable crítica es IsFraud que finalmente se convierte en un booleano que toma valores True o False

El Dataset está compuesto por 4 archivos .CSV (tablas) así:

sample_submission

test_identity

test_transaction

train_identity

train_transaction

De estos, teníamos que elegir con cual trabajaríamos, sin embargo, desde la misma competencia de Kaggle nos indicaban que ambas tablas estaban relacionadas por la clave primaria del código de la transacción, por lo que sabemos desde ya que para el desarrollo final de este trabajo debemos incluir a ambas: **Identity** and **Transactions**.

ANALISIS DE LOS DATOS

TABLA IDENTITY

Las variables en esta tabla son información de identidad:

información de conexión de red (IP, ISP, Proxy, etc.) y firma digital (UA/ navegador/OS/versión, etc.) asociada con las transacciones.

Son recopilados por el sistema de protección contra fraudes de Vesta y los socios de seguridad digital.

(Los nombres de los campos están enmascarados y no se proporcionará el diccionario por pares para la protección de la privacidad y el acuerdo del contrato)

- TransactionID
- id_12 - id_38
- DeviceType
- DeviceInfo

TABLA TRANSACTIONS:

- TransactionDT: timedelta de una fecha y hora de referencia determinada (no una marca de tiempo real). timedelta de una fecha y hora de referencia dada (no una marca de tiempo real). El primer valor de TransactionDT es 86400, que corresponde a la cantidad de segundos en un día (60 * 60 * 24 = 86400), así que creo que la unidad es segundos. Usando esto, sabemos que los datos abarcan 6 meses, ya que el valor máximo es 15811131, que correspondería al día 183”
- TransactionAMT: monto del pago de la transacción en USD
- ProductCD: código de producto, el producto para cada transacción
- card1 - card6: información de la tarjeta de pago, como tipo de tarjeta, categoría de tarjeta, banco emisor, país, etc.
- dirección: dirección addr1 como región de facturación, addr2 como país de facturación
- distancia: distancias entre (no limitadas) la dirección de facturación, la dirección postal, el código postal, la dirección IP, el área telefónica, etc
- P_ y (R__) emaildomain: dominio de correo electrónico del comprador y del destinatario
- C1-C14: conteo, como cuántas direcciones se encuentran asociadas con la tarjeta de pago, etc. El significado real está enmascarado.
- D1-D15: timedelta, como días entre transacciones anteriores, etc.
- M1-M9: coincidencia, como nombres en la tarjeta y dirección, etc.
- Vxxx: características completas diseñadas por Vesta, que incluyen clasificación, conteo y otras relaciones de entidad.

Características categóricas:

ProductCD

card1 - card6

addr1, addr2

P_emaildomain

R_emaildomain

M1 - M9

La tabla más grande corresponde a la de transacciones y es la que tiene información más relevante, por ejemplo, el monto de la transacción la cual podríamos usar para saber el monto total de transacciones que son fraudulentas, cruzándola con la tabla identidad, podríamos conocer desde que navegador se realizan, o cual franquicia es la más vulnerada (Amex, Visa, Mastercard, etc) por monto o por cantidad de repeticiones.

También podríamos determinar si los fraudes se realizaron más desde celulares o desde computadores y desde qué sistema operativo se realizaron.

Cuáles son los usuarios más vulnerados según el correo electrónico que usen, por ejemplo gmail, outlook o correos con dominios privados.

PROCESAMIENTO DE DATOS

Se realiza un preprocesamiento de datos identificando las columnas de ambas tablas. Para esto, se determinan cuales son susceptibles para nuestras métricas.

Tenemos inicialmente dfi para el Data Frame de Identity

```
[ ]   1 #Y la info exacta del DF
      2 dfi.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144233 entries, 0 to 144232
Data columns (total 41 columns):
```

Para lo cual tenemos valores NaN con un porcentaje del 35.6%

```
Existen valores NaN: True
Total de valores NaN: 2104107
El porcentaje de valores NaN: 35.6%
```

Creamos un dfi_sinNan para filtrar todos los datos nulos y tener un dataframe más limpio.

Al tratar de hacer lo mismo con la tabla transacciones, nos damos cuenta que más del 41% son datos nulos y que no siempre coinciden con la clave primaria que es el ID de la transacción, por lo que decidimos dejarla tal cual está.

Luego se unen las dos tablas para dejar solo un Dataframe llamado df

AHORA CONCATENAMOS AMBAS TABLAS CON DATOS LIMPIOS

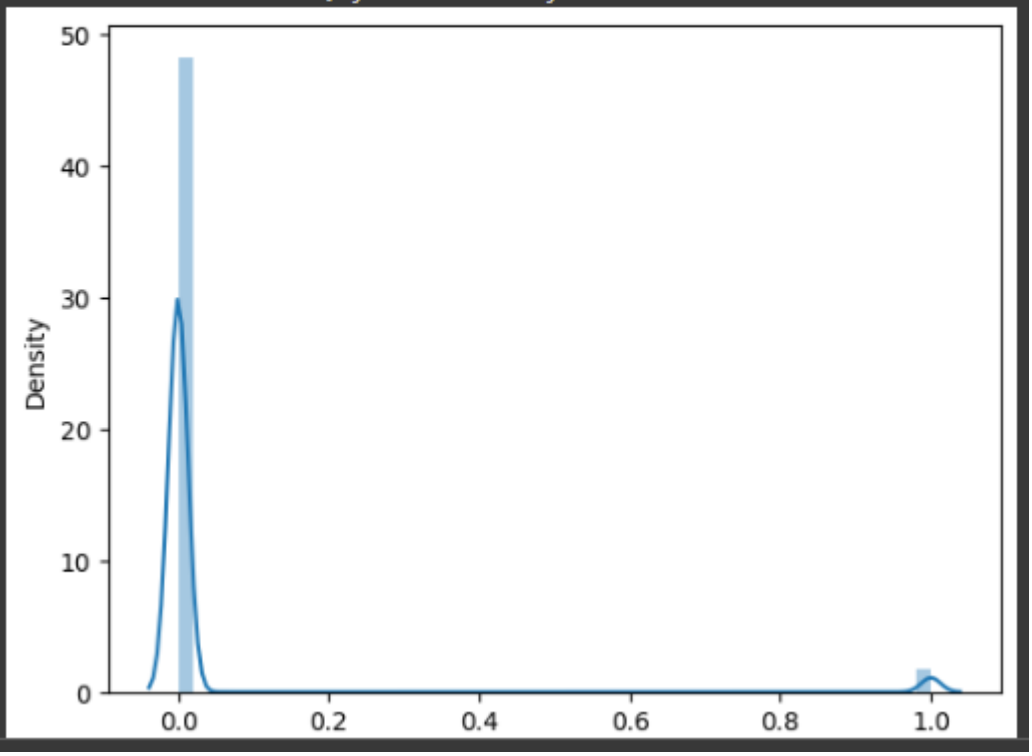
```
[30]  1 #Concatenamos los datos de dfi sin NAN con dft
      2 df = dft.merge(dfi_sinNaN,on = 'TransactionID',how = 'left')
```

ANALISIS DE LOS DATOS

Se analiza la variable objetivo IsFraud

```
[ ]   1 #Distribución de la variable objetivo
      2 sns.distplot(df['isFraud'])
```

```
<Axes: xlabel='isFraud', ylabel='Density'>
```

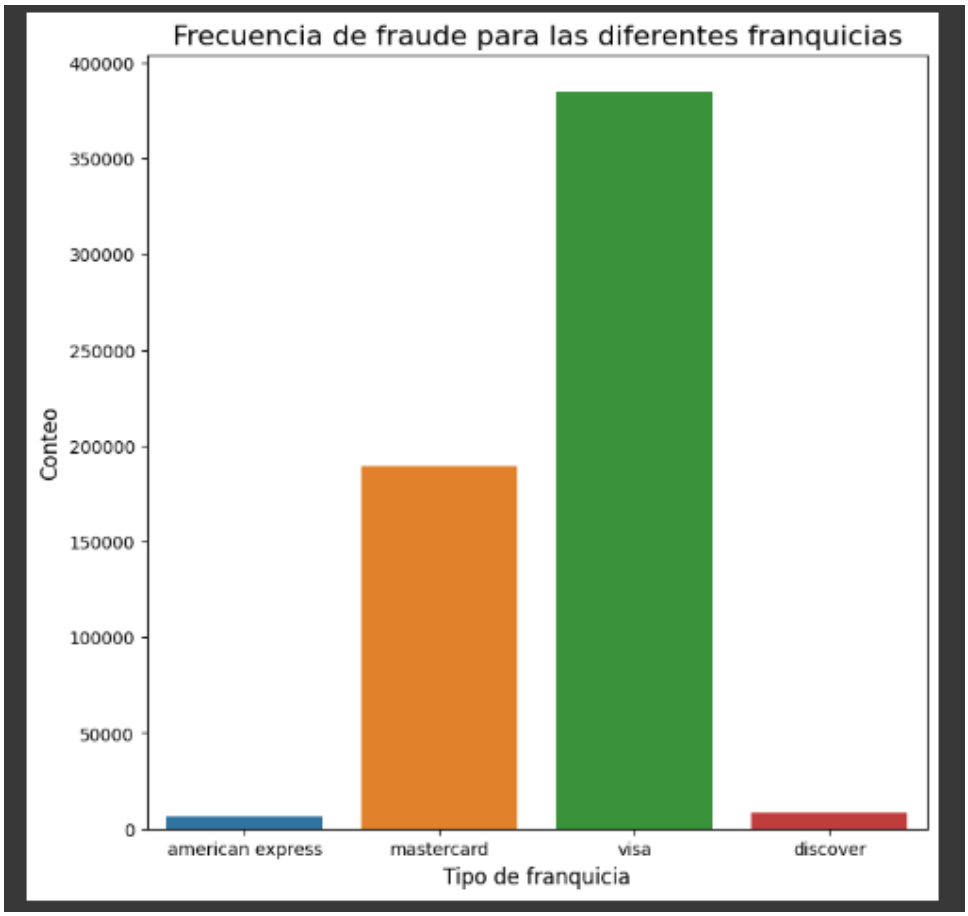


Y se mide su sesgo, donde se podría inferir que es una prueba de bondad y ajuste en la que la distribución obedece a una distribución normal.

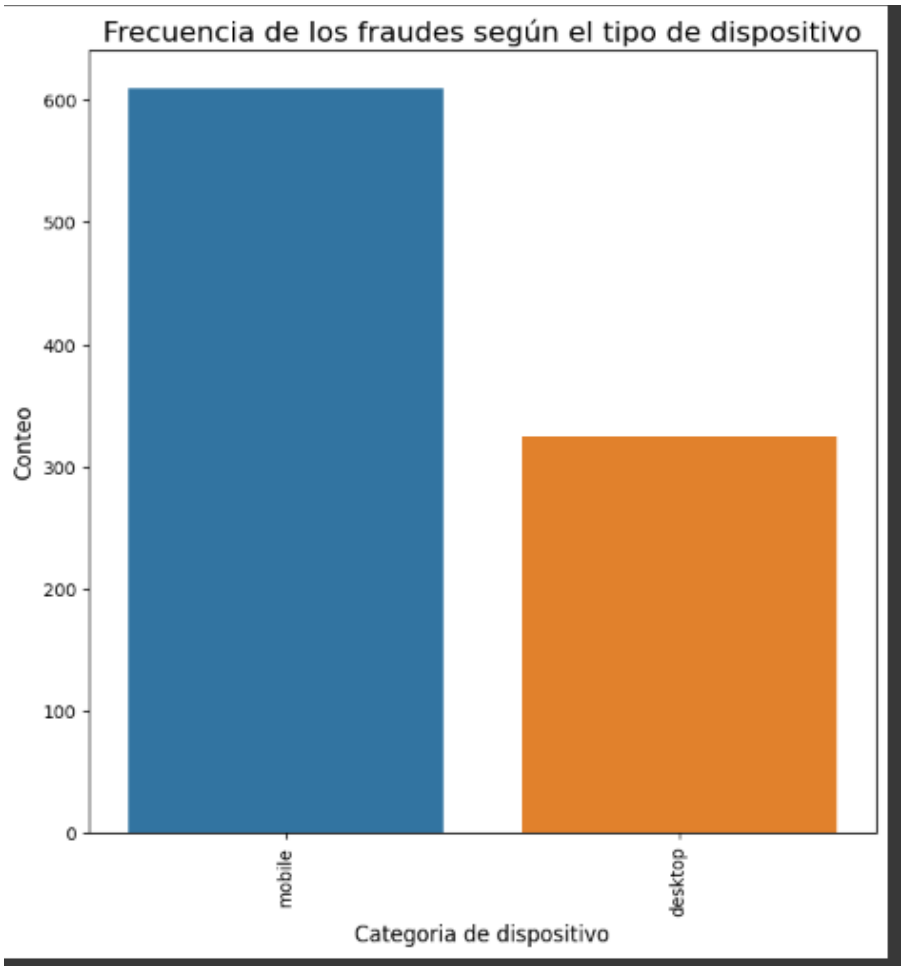
```
[ ]  1 #Ahora mediremos el sesgo de nuestra variable objetivo
     2 print('Skewness de la variable objetivo {:.1f}%'.format(round((df['isFraud'].skew()),1)))

    Skewness de la variable objetivo 5.1%
```

Se realizan algunas aproximaciones para enteder un poco más los datos:



Por ejemplo desde que dispostivo (movil o escritorio) se realizan más fraudes:



METRICAS DE EVALUACIÒN

Como métricas de estudio para la entrega final usaremos accuracy para medir la exactitud del modelo (% de casos en que el modelo ha acertado) y f1_score para combinar la precisión y la exhaustividad en un solo valor se calcula la medida armónica.

Tendremos en cuenta la variable isFraud para saber si una transacción está marcada como fraudulenta o no, bajo que franquicia y se evaluarán otras condiciones.

Nos gustaría contestarnos estas preguntas:

¿Cuántas transacciones realizadas con cada franquicia son fraude?

¿Cuál es el navegador que más se usa en las transacciones marcadas como fraude?

¿Desde que dispositivo se realizan mayores fraudes: desktop o mobile?

¿Cuál es el mayor valor por franquicia donde se ha hecho fraude?

¿Cuál es el acumulado de fraude por cada franquicia?

¿Con cual modalidad se hace más fraude: Débito o crédito?

¿Con cual franquicia se hace más fraude según el tipo de tarjeta: Crédito o débito?

¿Con cuál dominio de correo se hacen más fraudes?

DIFICULTADES

Hemos encontrado que por ser un dataframe de datos bancarios, se vuelve información MUY sensible, haciendo que Vesta no comparta muchos de los datos que ellos usan para detectar fraude, limitando el dataset a solo algunos datos que permitan sacar conclusiones. Nos hubiera gustado datos más abiertos, por ejemplo de ubicación. Si la persona estaba en un lugar diferente a su ubicación normal. O desde que país se hacen más fraudes, pero esta información no está disponible.

BIBLIOGRAFIA