PREVENCIÓN DE FRAUDE
CON MARCHINE LEARNING

Alumnos

Nicolás Mendoza

Gastón Guevara

Tutor

Juan Ignacio Demaestri

INTRODUCCIÓN

El principal riesgo a los que están sometidas las entidades financieras son los ataques de fraudes electrónicos. La empresas y entidades bancarias necesitan que sus usuarios gestionen operaciones y transacciones por internet eleva el potencial riesgo que dichos procesos sean manipulados por personas con malas intenciones, poniendo en peligro la integridad de los datos de los usuarios. Mucho dinero es absorbido en pérdidas cada año por las entidades financieras debido a transacciones fraudulentas.

El fraude electrónico es también llamado delito informático. Es una actividad indebida basada en la manipulación fraudulenta de elementos informáticos y sistemas de comunicación, para obtener un beneficio no autorizado. Existen diferente tipo de fraude electrónico:

- Phishing: es la obtención fraudulenta de información crítica de los clientes por medio de una página web similar a la auténtica
- Malware: “Malicious Software”, todo programa o código malicioso cuyo objetivo es dañar el sistema operativo o causar un mal funcionamiento
- Keyloggers: pequeños programas maliciosos utilizados para capturar cualquier tipo de actividad realizada, de esta manera se podría conocer todo lo digitado por la víctima
- Spyware: conocido como software espía, para este ataque se utiliza un software que es instalado sin autorización en la máquina de la víctima para monitorear las actividades que esta realiza desde un acceso remoto
- Spam: envío de correo no deseado donde los correos electrónicos no solicitados, con frecuencia son maliciosos y en ocasiones los delincuentes pretenden ser instituciones o compañías financieras solicitando información personal
- Hacking: obtienen acceso no autorizado a grandes cantidades de datos confidenciales con el objetivo de robar información, causar daños monetarios y de reputación a la entidad objetivo

El fraude contra el sistema financiero constituye un gran problema para los bancos ya que ocasionan pérdida económica, pérdida de imagen y desconfianza de los clientes. La prevención y detección de fraude tiene el deber de cuidar el procesamiento de transacciones legitimas, evitando bloquear incorrectamente operaciones o tarjetas. Sólo un bajo porcentaje de transacciones son fraudulentas y pocos operadores son los que revisan por los analistas de fraudes.

Por este motivo es que las entidades financieras requieren contar con herramientas informáticas que permitan identificar fraude dentro del gran volumen de registros de transacciones. Para ello se evalúan patrones de comportamiento que no sean usuales o que correspondiente a actividades potencialmente fraudulentas.

Con este trabajo buscamos implementar un modelo que, según un conjunto de datos de prueba, permita predecir exitosamente la mayor cantidad de casos de transacciones fraudulentas posible, con un mínimo porcentaje de falsos negativos. El modelo fue planteado utilizando datos de transacciones de autorización de tarjetas de crédito pero puede ser fácilmente adaptado para otro tipo de transacciones financieras.

MACHINE LEARNING

Es una rama de la inteligencia artificial, basada en la idea que los sistemas pueden aprender de datos, identificar patrones y tomar decisiones con una mínima intervención humana. Siendo así un método de análisis de datos que automatiza la construcción de "modelos analíticos", que surgió en los años 50.

La principal característica de este tipo de algoritmo es que son capaces de reajustarse automáticamente para mejorar su rendimiento en función del numero de aciertos y fallos producidos en un proceso de entrenamiento previo a su aplicación y durante la ejecución en tiempo real del mismo.

Machine Learning (ML) es una disciplina científica que maneja sistemas inteligentes, es decir que aprenden automáticamente al identificar ciertos patrones presentes en los datos. Para este aprendizaje ML usa algoritmos que se encargan de revisar datos mediante ejemplos o instrucciones predefinidas para así predecir comportamientos futuros permitiendo además la incorporación de información adicional y reajustar el resultado. ML maneja conocimiento inductivo obteniendo un enunciado general en base a enunciados que describen casos particulares (Mohri, Rostamizadeh, & Talwalkar, 2018).

El surgimiento del interés en el aprendizaje basado en máquinas se debe a:

- los volúmenes y variedad de datos disponibles
- el procesamiento computacional más económico y poderoso
- el almacenaje de dato asequible y mucho menos costoso
- y, todo hacen posible producir modelos de manera rápida y automática que pueda analizar datos grandes y complejos

Existen diferentes etapas, la primera es definir el objetivo. Es vital entender el problema a resolver y comprender la data que tenderemos a disposición. La segunda etapa es la de recolección de datos (data acquisition), de dónde y cómo conseguiremos la información. Como tercera etapa está la preparación de la data (se llama data wrangling): limpieza o formateo de datos que tiene como objetivo manipular y convertir la data en forma que produzcan mejores resultados. Algunas acciones pueden ser la categorización de los valores de las variables, normalizar datos numéricos o escalarlos para que sean compatibles.

La cuarta etapa es la selección del algoritmo más adecuado en relación con el problema que deseamos resolver, es decir el tipo de aprendizaje a implementar. Los algoritmos de ML se clasifican generalmente en:

- Aprendizaje supervisado: Su objetivo principal es predecir respuestas que habrá en el futuro, gracias al entrenamiento del algoritmo con datos conocidos del pasado (históricos). Pudiendo ser problemas de clasificación (necesitan predecir la clase más probable de un elemento en función de un conjunto de variables de entrada, siendo la variable taget del tipo categórica) o de regresión (predicen valores numéricos, no categóricos, por lo cual la variable target es del tipo cuantitativa).

Son algoritmos entrenados utilizando ejemplos etiquetados, aprende comparando su resultado real con resultados correctos para encontrar errores. La salida de este algoritmo es conocida. Algunos ejemplos son predicciones, optimización de procesos, diagnósticos, detección de fraude y otros.

- Aprendizaje no supervisado: Son algoritmos que infieren patrones de un conjunto de datos sin referencia a resultados conocidos o etiquetados (es decir que sólo conocemos la variable x). Por lo cual, estos algoritmos permiten realizar tareas de procesamiento de datos más complejos en comparación al aprendizaje supervisado.
  Sólo se dan los datos finales (inputs) a la máquina para que encuentre patrones interesantes a partir de esos datos. Todas las variables tienen la misma importancia, el objetivo es encontrar interdependencia entre las variables porque no hay una variable objetivos, o de salida, ni variables que ayuden a predecir la variable objetivo.
  Dentro de los más populares están los de clustering o agrupamiento, reglas de asociación y algoritmos de reducción de la dimensionalidad, entre otros.

- Aprendizaje por refuerzo: Se utiliza con frecuencia para robótica, juegos y navegación, siendo un algoritmo que descubre a través de ensayo y error qué acciones producen las mayores recompensas. Tiene tres componentes principales: el agente (el que aprende o toma decisiones), el entorno (todo con lo que interactúa el agente) y acciones (lo que el agente puede hacer).

La quinta etapa es la de entrenar el modelo. El proceso de entrenamiento de un modelo de ML consiste en proporcionarle al modelo datos de entrenamiento de los cuales pueda aprender (training y test). La sexta etapa es la validación del modelo; ya con el modelo entrenado se debe validar los resultados obtenidos según métricas utilizada para los diferentes tipos de algoritmos.

Y, por último, la séptima etapa es el deployment o despliegue del modelo, el cual consiste en la implementación en producción de nuestro modelo.

SISTEMA DE DETECCIÓN DE FRAUDES

Estos sistemas se basan en operaciones manuales realizadas por personal de la entidad financiera denominadas investigadores de fraude y automáticas mediante algoritmos que funcionan en tiempo real y casi tiempo real (las operaciones en tiempo real tienen lugar antes de que se autorice el pago, mientras que las operaciones casi en tiempo real se ejecutan después de que se produjo el pago).

Con este trabajo proponemos un modelo de ML que pueda identificar transacciones fraudulentas y no fraudulentas, para luego alimentar al mismo modelo con datos nuevos, y así tomar acción sobre compras peligrosas. Es decir, que la propuesta planteada se implementa para las operaciones cuya autorización ha sido emitida.

Primero se buscó una base de datos en donde haya información relevante como para crear un modelo de aprendizaje automático supervisado aplicado a prevenir fraudes con tarjetas de crédito, se encontró en kaggle una en donde había 1.296.674 filas y 23 columnas.

Para proceder a la construcción de un correcto modelo de aprendizaje automático supervisado se deben seguir los siguientes pasos:

1. Realizar el proceso de feature engineering (ingeniería de características) para transformar los datos históricos en un conjunto de características y etiqueta de clasificación para ser utilizado en un algoritmo de aprendizaje automático supervisado.
2. Dividir el conjunto de datos disponible en dos partes, una para construir el modelo y otra para probar el modelo.
3. Construir el modelo con las características y etiquetas de entrenamiento.
4. Probar el modelo con el conjunto de datos de prueba para obtener predicciones y luego comparar las predicciones obtenidas del modelo con las etiquetas del conjunto de pruebas.
5. Ajustar el modelo hasta obtener un índice de precisión deseado.

La ingeniería de características o feature engineering hace referencia al proceso de transformar datos sin procesar en entradas para un algoritmo de aprendizaje automático. La ingeniería de características depende en gran medida del tipo de caso de uso y las posibles fuentes de datos. Para el caso de fraudes en tarjetas de crédito las características (atributos) de cada transacción se pueden dividir en:

- Atributos de la transacción: fecha y hora de la transacción, el comercio y su categoría, monto de las operaciones, ciudad de origen y país de origen.
- Atributos de la tarjeta y su dueño: número y tipo de tarjeta, edad, nacionalidad y género del dueño, y si es adicional.
- Atributos del histórico de transacciones: monto mínimo, monto máximo, sumatoria de montos, cantidad de transacciones, promedio de los montos, monto mínimo y máximo por cada categoría del comercio, suma de los montos y cantidad por categoría del comercio.

Para la implementación se pensó que los eventos de las transacciones sean entregados a través del sistema de mensajería. Se procesa y verifica las transacciones en busca de características fraudulentas utilizando el modelo implementado y asigna una probabilidad de fraude. Luego se almacenan las transacciones enriquecidas con los datos agregados y almacena la tabla con el top N de las transacciones con mayor índice de probabilidad de fraude. Esta tabla es luego tomada por el sistema monitor que muestra los datos al personal encargado de comunicarse con el dueño de la tarjeta y etiquetar la transacción como genuina o fraudulenta que a posterior será utilizado en el siguiente periodo de entrenamiento como feedback.

La solución propuesta en este trabajo describe un sistema de detección de fraudes donde la herramienta permite la interacción con los investigadores de fraude. El papel de los investigadores de fraude es centrarse en las transacciones más sospechosas y contactar a los titulares de tarjetas. Para esto el sistema automático recibe una retroalimentación (legítimo o fraude) de solo un pequeño subconjunto de transacciones que activaron una alerta, para el resto de las transacciones no se reciben comentarios a menos que el titular de la tarjeta informe un fraude. Esto significa que se puede suponer que las transacciones no alertadas son genuinas solo después de un tiempo.

ANÁLISIS Y RESULTADOS

El dataset conseguido en Kaggle.com (link) cuenta con más de un millón doscientas mil filas y 24 columnas. Por el volumen de datos, se crearon dos bases del mismo archivo, uno tomando 600.000 filas y otro de test con 200.000. Para ello se utilizó Pandas:

```
df_train = df1.iloc[:600000,:]df_test = df1.iloc[600001:800000,:]
```

Una vez definida la base, comenzamos con la construcción de nuestra notebook. Primero importamos las librerías que utilizaremos, y se importa el archivo .csv

```
df = pd.read_csv('trainf.csv')
df.head(7)
```

Comenzamos a revisar la data, y nos encontramos que quedaron 600000 filas, y 24 columnas. Con el método *.info* revisamos el tipo de dato y confirmamos que no existen nulls.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 600000 entries, 0 to 599999
Data columns (total 24 columns):
 #   Column                 Non-Null Count   Dtype  
---  ------                 --------------   -----  
 0   Unnamed: 0             600000 non-null  int64  
 1   Unnamed: 0.1           600000 non-null  int64  
 2   trans_date_trans_time  600000 non-null  object 
 3   cc_num                 600000 non-null  int64  
 4   merchant               600000 non-null  object 
 5   category               600000 non-null  object 
 6   amt                    600000 non-null  float64
 7   first                  600000 non-null  object 
 8   last                   600000 non-null  object 
 9   gender                 600000 non-null  object 
 10  street                 600000 non-null  object 
 11  city                   600000 non-null  object 
 12  state                  600000 non-null  object 
 13  zip                    600000 non-null  float64
 14  lat                    600000 non-null  float64
 15  long                   600000 non-null  float64
 16  city_pop               600000 non-null  float64
 17  job                    600000 non-null  object 
 18  dob                    600000 non-null  object 
 19  trans_num              600000 non-null  object 
 20  unix_time              600000 non-null  float64
 21  merch_lat              600000 non-null  float64
 22  merch_long             600000 non-null  float64
 23  is_fraud               600000 non-null  float64
dtypes: float64(9), int64(3), object(12)
memory usage: 109.9+ MB
```

Para evitar sobrecargar el análisis con datos que no serán de utilidad, eliminamos algunas columnas y luego revisamos con qué nos quedamos

```
df.columns
```

```
Index(['Unnamed: 0.1', 'trans_date_trans_time', 'merchant', 'category', 'amt',
       'gender', 'city', 'state', 'zip', 'lat', 'long', 'city_pop', 'job',
       'dob', 'merch_lat', 'merch_long', 'is_fraud'],
```

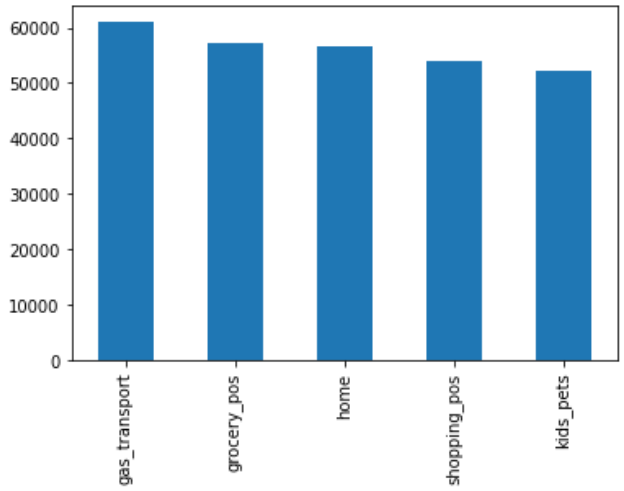
Y, mediante el *.describe*, revisamos los Principales Estadísticos

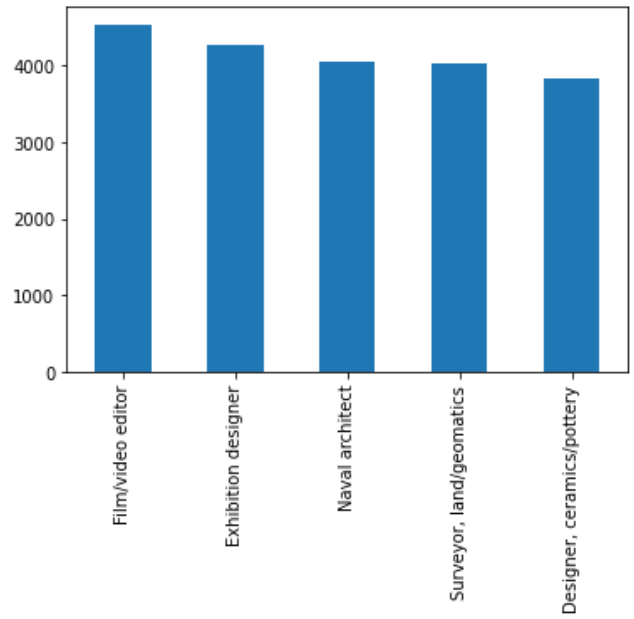
	count	mean	std	min	25%	50%	75%	max
Unnamed: 0.1	600000.0	299999.500000	173205.225094	0.000000	149999.750000	299999.500000	449999.250000	5.999990e+05
amt	600000.0	70.277534	158.332110	1.000000	9.640000	47.390000	83.070000	2.508694e+04
zip	600000.0	48765.593803	26908.885784	1257.000000	26041.000000	48154.000000	72011.000000	9.978300e+04
lat	600000.0	38.537569	5.074403	20.027100	34.668900	39.354300	41.894800	6.669330e+01
long	600000.0	-90.210855	13.768362	-165.672300	-96.790900	-87.458100	-80.138100	-6.795030e+01
city_pop	600000.0	89110.560567	302990.782526	23.000000	743.000000	2456.000000	20478.000000	2.906700e+06
merch_lat	600000.0	38.537173	5.108151	19.029798	34.741897	39.366208	41.955407	6.751027e+01
merch_long	600000.0	-90.211297	13.780443	-166.671242	-96.878002	-87.410123	-80.212446	-6.695654e+01
is_fraud	600000.0	0.005863	0.076348	0.000000	0.000000	0.000000	0.000000	1.000000e+00

Por medio de *Pandas Profilling* revisamos los datos con más detalle, y habiendo conseguido verificar la data importante, comenzamos con los primeros análisis.

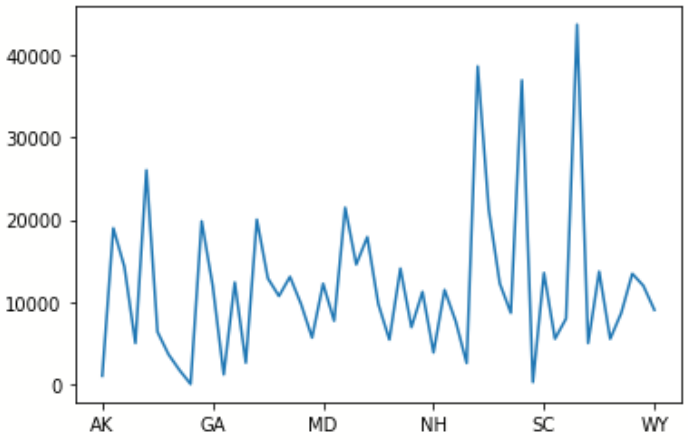
ANÁLISIS UNIVARIADO

Nos parece importante conocer información sobre dos variables 'category' y 'job, para saber qué tipo de datos son (top 5):





También revisamos otras variables como 'state' y 'amt', para conocer su densidad, utilizando otro tipo de gráficos: líneas e histograma.



DISTRIBUCION DE FRECUENCIAS

Nos encargamos de conocer la frecuencia absoluta y acumulada de la variable 'category', para lo cual creamos un DataFrame con la información. Primero conseguimos los valores de las frecuencias absolutas, para luego crear una función que en una lista vacia guarde las frecuencias acumuladas y vaya iterando la lista para ir sumándolas.

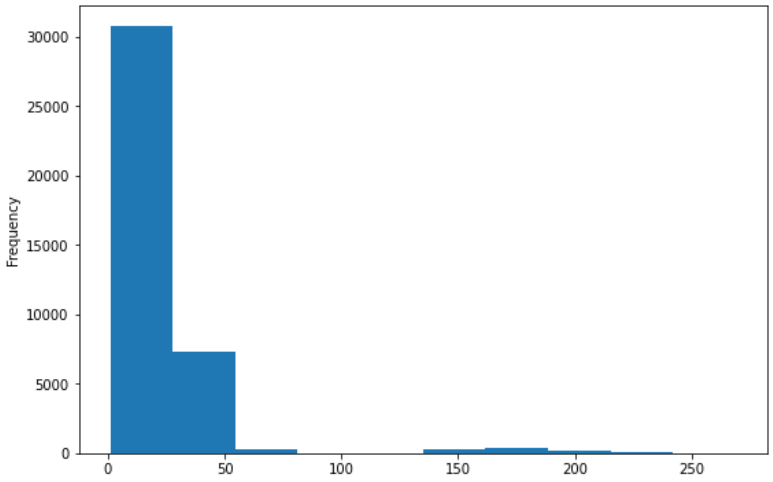
```
acum = []
valor_acum = 0

for i in frec_abs:
    valor_acum =+ i
    acum.append(valor_acum)
```

Y, por último, calculamos la frecuencia relativa (%) y la acumulando. Quedándonos la siguiente información:

	Frec_abs	frec_abs_acum	frec_rel_%	frec_rel_%_acum
gas_transport	61062	61062	10.177000	10.177000
grocery_pos	57254	57254	9.542333	9.542333
home	56735	56735	9.455833	9.455833
shopping_pos	54012	54012	9.002000	9.002000
kids_pets	52304	52304	8.717333	8.717333
shopping_net	44943	44943	7.490500	7.490500
entertainment	43542	43542	7.257000	7.257000
food_dining	42500	42500	7.083333	7.083333
personal_care	42118	42118	7.019667	7.019667
health_fitness	39824	39824	6.637333	6.637333
misc_pos	36731	36731	6.121833	6.121833
misc_net	29260	29260	4.876667	4.876667
grocery_net	20918	20918	3.486333	3.486333
travel	18797	18797	3.132833	3.132833

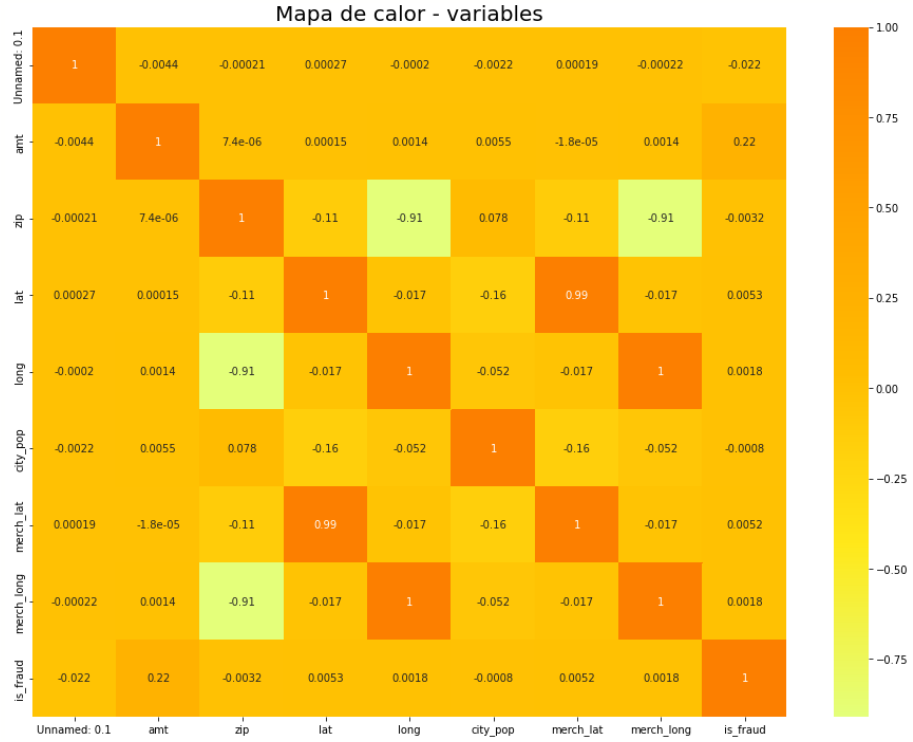
ANÁLISIS BIVARIADO

Al contar con tanta información y al ser transacciones de diferentes montos (‘amt’), la variable cuenta con valores concentrados en por debajo de los 30USD pero tiene otro muy altos. En el gráfico del análisis univariado de ‘amt’ de puede ver claramente:



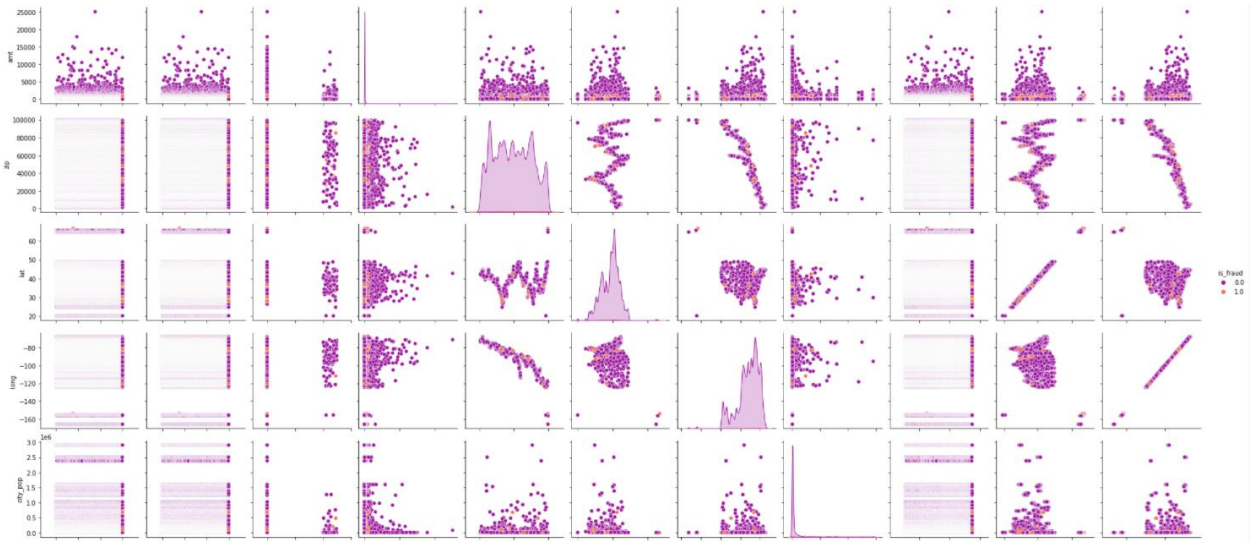
Como primer análisis bivariado se seleccionaron las variables ‘gender’ y ‘amt’, para revisar si existe alguna correlación, y luego se agregó la variable que determina si la transacción fue fraudulenta o no, habiendo quedado un resultado como este:

Y, además, se incluyó un mapa de calor para poder ver la correlación de datos entre las variables en análisis:



ANÁLISIS MULTIVARIADO

Dado que nuestro dataset cuanta con pocas variables numéricas, sólo realizamos un Pairplot para dimensionar la correlación de las variables, diferenciando los casos en el que la transacción fue fraude y cuáles no, habiendo quedado algo así:



Realizamos otros tipos de análisis multivariados que nos ayudaron a la selección de modelo, del tipo supervisado, más conveniente para el conjunto de datos y nuestro objetivo.

PREPARACIÓN DE LA DATA E IMPLEMENTACION DE MODELO

Por medio del método .get_dummies asignamos valores numéricos a columnas categóricas como 'gender' y 'category'. Para luego combinar las tablas y quitamos las columnas convertidas.

Realizamos el Split de la base de entrenamiento y test, asignando también la Y (nuestra variable objetivo). Entrenamos varios modelos, y comparamos los resultados.

Ajustamos algunos hiperparámetros con el objetivo de mejorar los modelos:

```
#Hiperparametros

#Random Forest
n_estimators = [10, 100]#, 1000,10000
max_features = ['sqrt', 'log2']
rf_grid = dict(n_estimators=n_estimators,max_features=max_features)

#Logistic Regression
solvers = ['newton-cg']#, 'lbfgs', 'liblinear']
penalty = ['l2']
c_values = [ 1.0, 0.1, 0.01]#100, 10,
lr_grid = dict(solver=solvers,penalty=penalty,C=c_values)

#LGB
class_weight = [None,'balanced']
boosting_type = ['gbdt']# 'goss', 'dart'
num_leaves = [30,50] #list(range(30, 150)),
learning_rate = list(np.logspace(np.log(0.005), np.log(0.2), base = np.exp(1), num = 10)) #1000
lgg_grid = dict(class_weight=class_weight, boosting_type=boosting_type, num_leaves=num_leaves, learning_rate =learning_rate)
```

Para, por último, utilizar GridSearch y RandomSearch para comparar resultados:

```
#GRID SEARCH
models = [rf,lr,lgg]
grids = [rf_grid,lr_grid,lgg_grid]
col = 0

for ind in range(0,len(models)):
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3,random_state=1)
    grid_search = GridSearchCV(estimator=models[col],
                  param_grid=grids[col], n_jobs=-1, cv=cv,
                  scoring='accuracy',error_score=0)
    grid_clf_acc = grid_search.fit(x_train, y_train)
    resul.iloc[1,col] = grid_clf_acc.score(x_test,y_test)
    col += 1

resul.head()
```

Case	RndForest	LogReg	LGB
Standard	0.995939	0.993450	0.996317
GridSearch	0.995917	0.993439	0.996428
RandomSearch	0.000000	0.000000	0.000000

```
[ ] #RANDOM SEARCH

    from scipy.stats import randint as sp_randint
    from sklearn.model_selection import RandomizedSearchCV
    col = 0
    for ind in range(0,len(models)):
        cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3,
                                     random_state=1)
        n_iter_search = 3
        random_search = RandomizedSearchCV(models[col],
        param_distributions=grids[col],n_iter=n_iter_search, cv=cv)
        random_search.fit(x_train,y_train)
        resul.iloc[2,col] = random_search.score(x_test,y_test)
        col += 1

    resul.head()
```

```
c:\Users\nicus\AppData\Local\Programs\Python\Python310\lib\site-packages\scipy\op
  warn(msg, LineSearchWarning)
c:\Users\nicus\AppData\Local\Programs\Python\Python310\lib\site-packages\scipy\op
  warn('The line search algorithm did not converge', LineSearchWarning)
c:\Users\nicus\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\
  warnings.warn("Line Search failed")
```

Case	RndForest	LogReg	LGB
Standard	0.995939	0.993450	0.996317
GridSearch	0.995917	0.993439	0.996428
RandomSearch	0.995983	0.993439	0.996428

CONCLUSIÓN

La detección de eventos fraudulentos es una tarea particularmente desafiante y compleja, al ser eventos raros y al estar en constante evolución son difíciles de modelar. El aumento del volumen de transacciones que ocurren todos los días exige el uso de herramientas automáticas para apoyar la detección y los recursos humanos destinados a apoyar esta tarea deben estar enfocados a investigar los casos más sospechosos.

La tecnología y Big Data permite superar estos problemas, en este caso el modelo más conveniente para nuestro proyecto es el Random Forest, según los resultados conseguidos.

Como trabajo futuro, se planifica que los resultados del modelo planteado sean revisados por los investigadores expertos en fraude para obtener sus observaciones y mejorar las predicciones del modelo. Por otro lado, es conveniente aumentar los datasets para nuevos experimentos.