CREACION DE UN MODELO PREENTRENADO QUE DETECTE TRANSACCIONES FRAUDULENTAS DE UN NEGOCIO TIPO ECOMMERCE

Metodología KDD (*Knowledge Discovery in Database*)

Metodología desarrollada dentro un proceso iterativo que explora grandes cantidades de datos para determinar las relaciones entre estos, con la finalidad de encontrar información útil dentro de los repositorios de datos de una empresa.

EL proceso lo podremos describir en 6 pasos:

1.- Selección de datos

Consiste en determinar los datos a utilizar, como también las fuentes de información de donde se realiza la extracción de los datos.

Fuente de los datos: Utilizamos un repositorio de kaggle donde estos datos tienen más de 20 millones de transacciones generadas a partir de una simulación de mundo virtual multiagente realizada por IBM.

"Los datos aquí casi no tienen ofuscación y se proporcionan en un archivo CSV cuyo esquema se describe en la primera fila. Los datos cubren 2000 consumidores (sintéticos) residentes en los Estados Unidos, pero que viajan por el mundo. Los datos también cubren décadas de compras e incluyen múltiples tarjetas de muchos de los consumidores." (Kaggle.com)

2.- Preprocesamiento.

Herramientas utilizadas: Python, SQL Server, Power BI, Visual Studio 2022

Donde se realiza todos los procesos de limpieza de los datos ya extraídos. Generalmente es aquí donde algunos datos sufren transformaciones para que su posterior tratamiento sea más fácil de procesar, esta etapa es también llamada ETL.

2.1.- Extracción: La base de datos seleccionada contaba un número muy grande de transacciones por lo que se extrajo una muestra aleatoria de 2 millones de transacciones con un script de Python, de las cuales apenas 0.13% son transacciones fraudulentas, luego se seleccionan las variables críticas para un nuevo análisis.

https://www.kaggle.com/datasets/ealtman2019/credit-card-transactions/data

Descripción de los datos Extraídos

Descripción de la tabla de datos MuestraTransacciones		
Columna	**Tipo de dato**	**Descripción**
User	Int	ID de los usuarios registrados
Card	Int	Número de tarjetas que posee un usuario
Year	Int	Año de Transacción (2006-2020)
Month	Int	Mes de la Transacción (1-12)
Day	Int	Día de la Transacción (1-31)
Time	Time	Hora de la transacción

Amount	Money	Monto de la transacción
Use_Chip	VarChar(50)	Tipo de transacción
MerchantName	VarChar(50)	ID de tienda o comercio
Merchan_City	VarChar(50)	Ciudad de ubicación del comercio
Merchant_State	VarChar(50)	Siglas del Estado ubicación del comercio
Zip	VarChar(50)	Código ZIP de la ubicación del comercio
MCC	VarChar(50)	Código categoría de mercado
Errors	VarChar(50)	Tipo de error detectado en la transacción
Is_Fraud	VarChar(50)	Si la transacción es un fraude o no
TransaccionID	Int	ID de las transacciones registradas (Primary key)

Tabla 1 -descripción de la tabla MuestraTransacciones

Descripción de la tabla Tarjetas		
Columna	**Tipo de dato**	**Descripción**
User	Int	ID de los usuarios registrados
Card_index	int	Index de las tarjetas que posee un usuario
BancoCredito	VarChar(50)	Nombre de empresa crediticia
TipoTarjeta	VarChar(50)	Tipo de tarjeta
NumeroTarjeta	Bigint	Número de tarjeta (Primay Key)
Expires	VarChar(50)	Fecha de expiración de la tarjeta
CVV	Int	Código CVV del reverso de la tarjeta
Has_Chip	VarChar(50)	Tipo de transacción
Cards_Issued	Int	Número de tarjetas emitidas
LimiteCredito	Money	Límite de crédito en tarjeta
Acct_Open_Date	VarChar(50)	Fecha apertura de cuenta
Year_PIN_last_Changued	Int	Fecha del último cambio del PIN
Card_on_Dark_web	VarChar(50)	SI la tarjeta se encuentra reportada en la DarkWeb

Tabla 2 - Descripción tabla Tarjetas

Descripción de la tabla Usuarios		
Columna	**Tipo de dato**	**Descripción**
Person	VarChar(50)	Nombre del Cliente
Current_Age	Int	Edad del cliente
Retirement_Age	Int	Edad de retiro del cliente
Brith_Year	Int	Año de nacimiento
Brith_Month	Int	Mes de nacimiento
Brith_Day	Int	Día de Nacimiento
Gender	VarChar(50)	Genero
Dirección	VarChar(50)	Dirección
Apartment	VarChar(50)	Numero de apartamento
Ciudad	VarChar(50)	Ciudad de residencia
State	VarChar(50)	Estado de residencia
Zipcode	Int	Código ZIP Dirección cliente
Latitude	Float	Latitud GPS
Longitude	Float	Longitud GPS
IngresoPerCapita	Money	Ingreso Per Capita
IngresoAnual	Money	Total Ingreso anual

DebitoTotal	Money	Total de débitos anuales
FICO_Score	Int	Calificación FICO de score crediticio
NumeroTarjetasCredito	Int	Cantidad de tarjetas que posee el usuario
UserID	Int	ID de los usuarios registrados (Primay Key)

Tabla 3 - Descripción tabal Usuarios

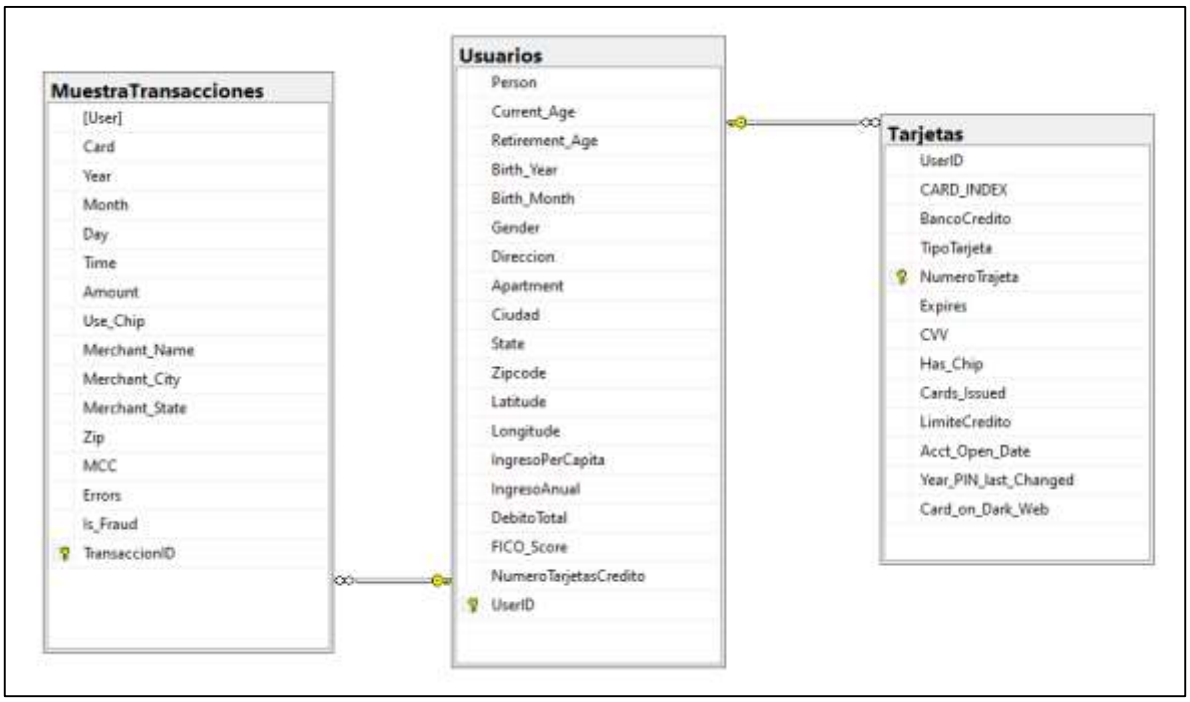


Ilustración 1-Diagrama de base datos original

2.2.- Transformación: Los datos son transformados en formato que se pueda cargar en la base de datos, la observación, sondeo y cambio del tipo de datos de cada columna y el procesamiento de datos nulos y erróneos usando SQL Server.

2.3.- Carga de los datos:

los datos son replicados en un nuevo almacén de datos FraudulentECoommerce agrupados en 4 tablas Dimensionales (DIM_USUARIO_DEST, DIM_TARJETA_DEST, DIM_TIEMPO_DEST, DIM_COMERCIO_DEST) y una tabla de Hechos (H_TRANSACCION_DEST), ya se puede hacer diversas gráficas para visualizar tendencias y relaciones entre variables, el modelo ETL permite hacer consultas más complejas y con más detalle, como los que se presentan más adelante.

Descripción de los datos cargados

Descripción de la tabla DIM_USUARIO_DEST		
Columna	**Tipo de dato**	**Descripción**
UserID	Int	ID de los usuarios registrados (Primay Key)
Person	VarChar(50)	Nombre del Cliente
Gender	VarChar(50)	Genero
Direccion	VarChar(50)	Dirección
Ciudad	VarChar(50)	Ciudad de residencia
IngresoPerCapita	Money	Ingreso Per Capita
IngresoAnual	Money	Total Ingreso anual
DebitoTotal	Money	Total de débitos anuales
FICO_Score	Int	Calificación FICO de score crediticio

Tabla 4 - Descripción de la tabla Dimensional Usuario

Descripción de la tabla DIM_TARJETA_DEST		
Columna	**Tipo de dato**	**Descripción**
Número de tarjeta	Bigint	Número de tarjeta (Primay Key)
BancoCredito	VarChar(50)	Nombre de empresa crediticia
TipoTarjeta	VarChar(50)	Tipo de tarjeta
Has_Chip	VarChar(50)	Tipo de transacción
LimiteCredito	Money	Límite de crédito en tarjeta

Tabla 5 - Descripción de la tabla dimensional Tarjeta

Descripción de la tabla DIM_TIEMPO_DEST		
Columna	**Tipo de dato**	**Descripción**
TiempoID	Int	Índice de los registros de tiempo (Primay Key)
Year	Int	Año de Transacción (2006-2020)
Month	Int	Mes de la Transacción (1-12)
Day	Int	Día de la Transacción (1-31)
Time	Time	Hora de la transacción
Dia_Semana	VarChar(20)	Día de la semana registrada en la transacción

Tabla 6 - Descripción de la tabla dimensional Tiempo

Descripción de la tabla DIM_COMERCIO_DEST		
Columna	**Tipo de dato**	**Descripción**
ComercioID	Int	Índice de los Comercio registrados (Primary Key)
MerchantName	VarChar(50)	ID de tienda o comercio
Merchan_City	VarChar(50)	Ciudad de ubicación del comercio
Merchant_State	VarChar(50)	Siglas del Estado ubicación del comercio
Zip	VarChar(50)	Código ZIP de la ubicación del comercio

Tabla 7 - Descripción de la tabla dimensiona Comercio

Descripción de la tabla H_TRANSACCION_DEST		
Columna	**Tipo de dato**	**Descripción**
UserID	Int	ID de los usuarios registrados (Primay Key)
Número de tarjeta	Bigint	Número de tarjeta (Primay Key)
TiempoID	Int	Índice de los registros de tiempo (Primay Key)

ComercioID	Int	Índice de los Comercio registrados (Primary Key)
Amount	Money	Monto de la transacción
Errors	VarChar(50)	Tipo de error detectado en la transacción
Is_Fraud	VarChar(50)	Si la transacción es un fraude o no

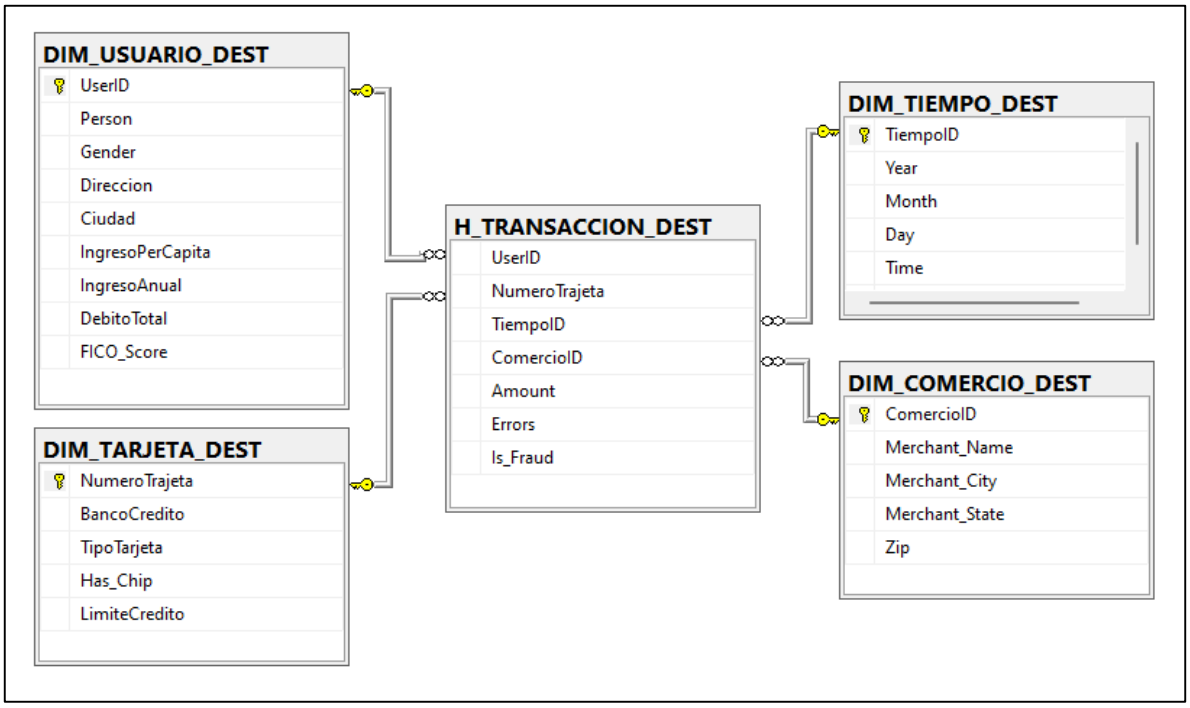


Ilustración 2- Modelo Estrella de la base de datos FraudulentECommece

Adicionalmente en este paso se hizo un chequeo exploratorio de algunas variables y sus incidencias en relación a la variable objetivo

Según el muestreo se puede observar relaciones muy evidentes en relación a los meses, días y picos de actividad en ciertas horas del día, datos que pueden servir para optimizar recursos en el monitoreo de transacciones en los tiempos determinados.

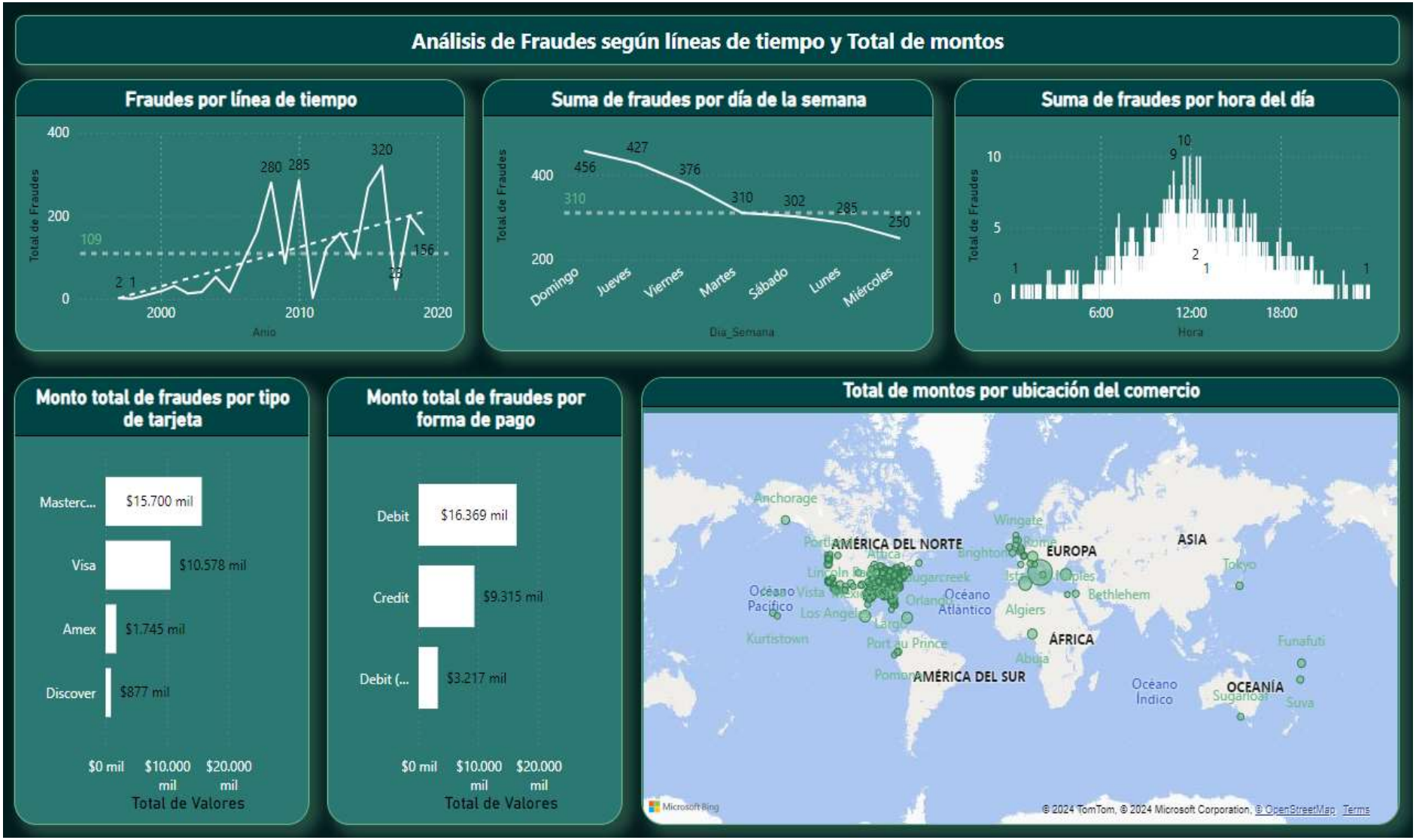


Ilustración 3 - Análisis Gráficos, diseños en Power BI

3.- Transformación

Herramientas utilizadas: Python, Google Colab

todos los cambios que no se realizaron en el paso anterior son llevados a cabo en esta etapa, donde el dato se prepara finalmente para poder ser procesado por algún algoritmo de minería de datos. Las operaciones más comunes realizadas son de agregación de datos o normalización.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1999899 entries, 0 to 1999898
Data columns (total 22 columns):
 #   Column            Dtype
---  ------            -----
 0   Amount            int64
 1   Errors            object
 2   Is_Fraud          int64
 3   Gender            object
 4   Direccion         object
 5   Ciudad            object
 6   IngresoPerCapita  int64
 7   IngresoAnual      int64
 8   DebitoTotal       int64
 9   FICO_Score        int64
 10  Year              int64
 11  Month             int64
 12  Day               int64
 13  Dia_Semana        object
 14  Time              object
 15  BancoCredito      object
 16  TipoTarjeta       object
 17  LimiteCredito     int64
 18  Merchant_Name     int64
 19  Merchant_City     object
 20  Merchant_State    object
 21  Zip               int64
dtypes: int64(12), object(10)
memory usage: 335.7+ MB
```

Ilustración 4 - Información de las variables utilizadas en la tabla de datos

3.1.- Eliminación de columnas innecesarias:

Las columnas que no tomaremos en cuenta para el entrenamiento del modelo de predicción serán en primera instancia los datos geográficos y Gender, ya que, aunque en el análisis exploratorio se detectó lugares donde existe mayor incidencia de actividades fraudulentas, no creemos que será relevante para identificar su legitimidad, además de la columna día por ser un numero ordinal entre 1 y 31 que podría causar confusión en el entrenamiento

Columnas eliminadas	Tipo de dato
Dirección	Object
Day	Int
Merchant_Name	Object
Merchant_city	Object
Merchant_State	Object
Zip	Int
Ciudad	Object
Gender	Object

Tabla 8 - Columnas eliminadas del Data Frame

3.2.- transformación de los datos

Procedemos con la transformación de datos de las variables categóricas de tipo object que no van a ser reconocidas en el entrenamiento como son: Errors, Dia_Semana, Time, BancoCredito, TipoTarjeta. Las transformaciones serán de tipo Encoding OHE (One Hot Encodig), lo que hace es crear nuevas columnas iguales a cada categoría de las variables a transformar dándoles valores binarios, esto además de darle valores numéricos a las variables también aumenta la dimensión de la Data Frame.

Columnas Transformadas	Tipo de dato	Tipo Transformación	Columnas creadas
Errors	Object	OHE	SIN ERROR, Insufficient Balance, Technical Glitch, Bad PIN, Bad Card Number, Bad CVV, Bad Expiration, Bad Zipcode
BancoCredito	Object	OHE	Amex, Mastercard, Visa, Discover
TipoTarjeta	Object	OHE	Credit, Debit, Debit (Prepaid)
Dia_Semana	Object	OHE	Miércoles, Domingo, Martes, Lunes, Sábado, Jueves, Viernes
Time	Object	Label_encoder	Time

Tabla 9 - Columnas transformadas

4.- Minería de datos

corresponde al modelo propiamente tal, en donde los algoritmos son aplicados con el objetivo de encontrar las relaciones que puedan existir

Para el caso probamos algunos algoritmos, para comparar y ver cual se adapta mejor a las propiedades de los datos y al desbalance que existe entre las clases de la columna Es_Fraude

4.1.- Regresión Logística:

Modelo de análisis de regresión utilizada en estadística y aprendizaje automático para predecir la probabilidad de que un evento ocurra, como está basado en la función sigmoide es utilizada principalmente para problemas de clasificación binaria, lo que nos viene bien para nuestro problema

Los resultados muestran que el modelo tiene dificultades para identificar correctamente la clase minoritaria (fraude). Esto es evidente por la baja precisión y F1-Score en la clase 1. Aunque el Sensibilidad para la clase 1 no es muy bajo (0.55), la precisión es extremadamente baja, lo que significa que hay muchos falsos positivos.

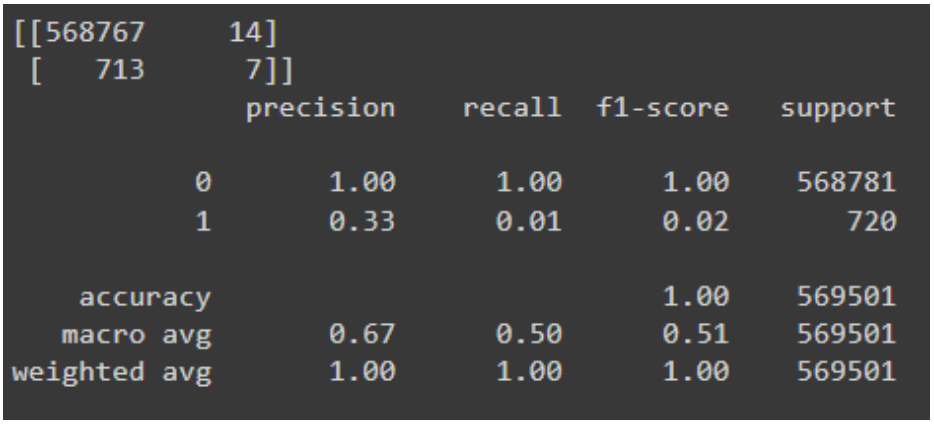
```
[[437263 131518]
 [   326    394]]
              precision    recall  f1-score   support

           0       1.00      0.77      0.87    568781
           1       0.00      0.55      0.01       720

    accuracy                           0.77    569501
   macro avg       0.50      0.66      0.44    569501
weighted avg       1.00      0.77      0.87    569501
```

Ilustración 5- Evaluación del Modelo de Regresión Logística

4.2.- Random forest:

Es un método de aprendizaje supervisado utilizado tanto para clasificación como para regresión. Se basa en la construcción de múltiples árboles de decisión durante el entrenamiento y la combinación de sus resultados para mejorar la precisión y controlar el sobreajuste.

El modelo está funcionando casi perfectamente para la clase mayoritaria (No fraude), con una precisión y recall muy altos para esa clase. Sin embargo, está fallando significativamente en la clase minoritaria (Fraude):



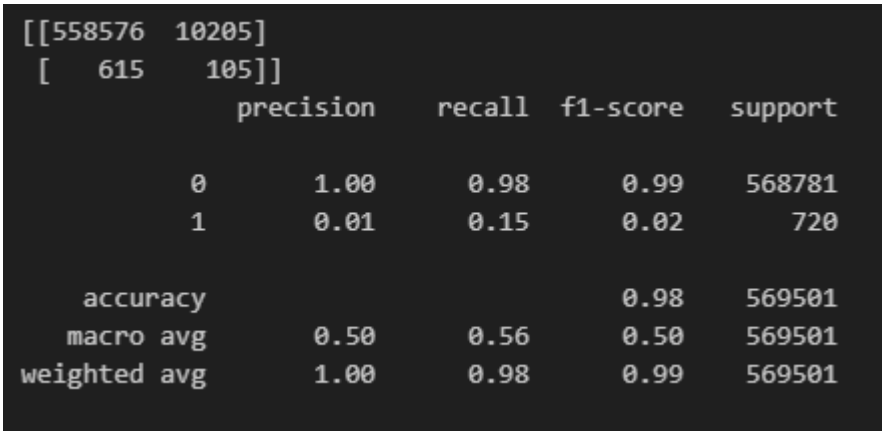
```
[[568767     14]
 [   713      7]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    568781
           1       0.33      0.01      0.02       720

    accuracy                           1.00    569501
   macro avg       0.67      0.50      0.51    569501
weighted avg       1.00      1.00      1.00    569501
```

Ilustración 6- Evaluación Modelo Random Forest

4.3.- Naive Bayes:

Es un clasificador probabilístico basado en el teorema de Bayes con una suposición fuerte de independencia de las características. A pesar de esta suposición, Naive Bayes ha demostrado ser muy efectivo para ciertas tareas de clasificación

La precisión y el recall para la clase fraude es extremadamente baja (0.01) y (0.15) respectivamente, lo que indica que el modelo está prediciendo incorrectamente muchos casos de fraude como no fraude, la precisión 0.98 es engañosa por el desbalanceo de las clases.



```
[[558576  10205]
 [   615    105]]
              precision    recall  f1-score   support

           0       1.00      0.98      0.99    568781
           1       0.01      0.15      0.02       720

    accuracy                           0.98    569501
   macro avg       0.50      0.56      0.50    569501
weighted avg       1.00      0.98      0.99    569501
```

Ilustración 7 - Evaluación del modelo Naive Bayes

4.4.- XGBoost:

Es un algoritmo de aprendizaje supervisado basado en árboles de decisión que es conocido por su eficiencia, flexibilidad y capacidad para manejar grandes conjuntos de datos. Es especialmente útil para problemas de clasificación y regresión, incluyendo aquellos con conjuntos de datos desbalanceados.

Las métricas ponderadas reflejan principalmente el rendimiento de la clase mayoritaria debido a la disparidad en el número de ejemplos entre las dos clases. Por lo tanto, es importante

considerar métricas como el recall y el F1-score para la clase minoritaria para evaluar correctamente el modelo, sin embargo y pese a la disparidad el modelo puede identificar más del 50% de los casos de fraude

```
[[533505  35276]
 [   335    385]]
              precision    recall  f1-score   support

           0       1.00      0.94      0.97    568781
           1       0.01      0.53      0.02       720

    accuracy                           0.94    569501
   macro avg       0.51      0.74      0.49    569501
weighted avg       1.00      0.94      0.97    569501
```

Ilustración 8 - Evaluación del modelo XGBoost

4.5.- XGBoost + Hiperparámetros.

XGBoost hasta ahora es el algoritmo que mejores resultados nos ha dado, para intentar refinar un poco la efectividad del modelo aplicamos variaciones de Hiperparámetros, para ello utilizamos RandomizedSearchCV que prueba aleatoriamente un conjunto de parámetros predefinidos dentro de un rango especifico, luego se entrena el modelo con las variables ajustadas que mejor resultados se obtuvo

Parametros:

```
subsample= 0.7
scale_pos_weight= ratio
n_estimators= 200
max_depth= 6
learning_rate= 0.22749999999999998
colsample_bytree= 1
use_label_encoder=False
eval_metric='logloss'
```

Aunque las métricas generales mejoraron un poco, el problema sigue siendo la clase minoritaria, la sensibilidad bajó al 39% por lo que la detección de transacciones fraudulentas va a ser menos eficiente que el entrenamiento anterior que fue del 53%

```
[[553072  15709]
 [   438    282]]
              precision    recall  f1-score   support

           0       1.00      0.97      0.99    568781
           1       0.02      0.39      0.03       720

    accuracy                           0.97    569501
   macro avg       0.51      0.68      0.51    569501
weighted avg       1.00      0.97      0.98    569501
```

Ilustración 9- Evaluación del modelo XKBoost + Hiperparámetros modificados

5.- Interpretación

finalmente, se identifican los patrones encontrados en la etapa anterior que puedan ser de utilidad. Todo esto bajo métricas de evaluación que deben ser consistente al objetivo de búsqueda.

Algoritmo	**Métricas Generales**						
	Error	Precisión		Sensibilidad		Exactitud	
		0	1	0	1	0	1
Regresión Logística	23.15%	1.00	0.00	0.77	0.55	0.87	0.01
		0.50		0.66		0.44	
Random Forest	0.12%	1.00	0.33	1.00	0.01	1.00	0.02
		0.64		0.50		0.51	
Naive Bayes	1.90%	1.00	0.01	0.98	0.15	0.99	0.02
		0.50		0.56		0.50	
XGBoost	6.25%	1	0.01	0.94	0.53	0.97	0.02
		0.51		0.74		0.49	
XGBoost + Hiperparametros	2.83%	1	0.02	0.97	0.39	0.99	0.03
		0.87		0.68		0.97	

Tabla 10 - Métricas de los algoritmos implementados

Para problemas de detección de fraudes, donde la clase de fraude es la minoritaria y es crítica detectarla correctamente, es importante centrarse en mejorar la **sensibilidad (recall)** y la **precisión** de la clase minoritaria (fraude), incluso si esto reduce la precisión general o la exactitud.

Para nuestro problema en particular el algoritmo que mejores estadísticas nos ofrece es XGBoost, sacrificamos un poco la precisión, pero la detección efectiva de la clase minoritaria es la que nos interesa con la aplicación de este modelo podemos asegurar al menos la detección del 53% de las transacciones fraudulentas.

6.- Evaluación

Para poner a prueba el modelo tenemos un conjunto de datos que no se han utilizado en el conjunto de entrenamiento ni en los test, de esta manera podemos observar cómo se comporta el modelo, este conjunto de datos de prueba contiene 10 transacciones 5 fraudulentas y 5 que no lo son, los resultados de cada modelo se adjuntan en la tabla 10

Algoritmo	**Clase 0 (No Fraude)**		**Clase 1 (No Fraude)**		**Aciertos Datos de Prueba**
	Verdaderos Negativos	Falsos positivos	Falsos Negativos	Verdaderos positivos	
Regresión Logística	437263	131518	326	394	6/10
Random Forest	568767	14	713	7	5/10
Naive Bayes	558576	10205	615	105	5/10
XGBoost	533505	35276	335	385	8/10
XGBoost + Hiperparametros	553072	15709	438	282	8/10

Regresión Logística y **Naive Bayes** tienen dificultades para identificar correctamente los fraudes, con una alta cantidad de FP o FN.

Random Forest muestra una exactitud muy alta pero no detecta casi fraudes, lo cual es problemático.

XGBoost, especialmente con ajuste de hiperparámetros, muestra el mejor rendimiento en términos de detección de fraudes, El ajuste de hiperparámetros mejora la precisión del modelo en términos de reducir los FP, aunque a expensas de un leve aumento en los FN. Este modelo muestra un buen compromiso entre precisión y sensibilidad.

Tomando en cuenta el desbalance de los datos, el modelo recomendado para implementar es el XGBoost hiperparatrizado