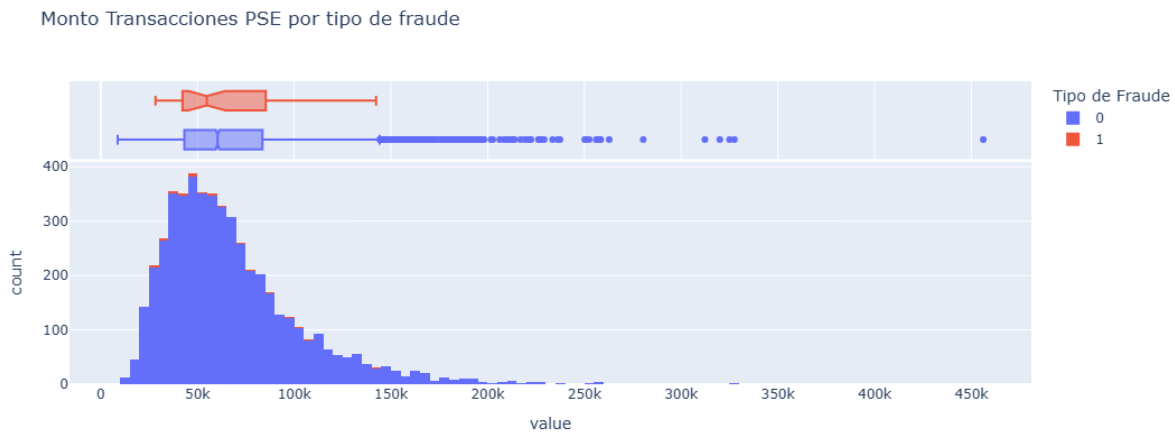
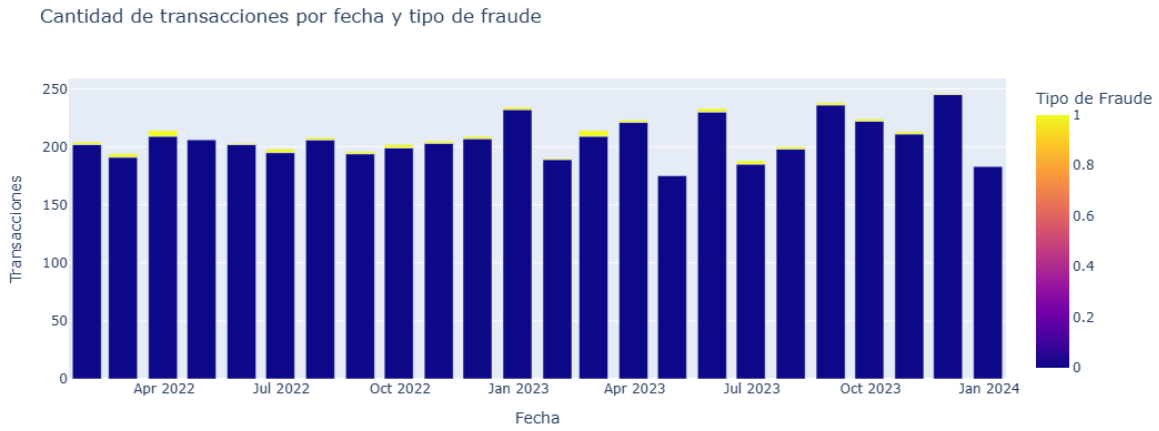
FRAUDE TRANSACCIONAL

Para identificar los perfiles de clientes con transacciones o comportamientos fraudulentos dentro de la aplicación, se realiza un análisis exploratorio de datos; con la finalidad de poder encontrar patrones y tendencias estudiando las distribuciones de las variables numéricas y categóricas.

Teniendo en cuenta lo anterior, con el análisis descriptivo y exploratorio de los datos, se encuentra que la mayoría de las transacciones por PSE en la plataforma tienen valores por debajo de los 145K, pero se observa en el grafico de la distribución de los montos de transacciones fraudulentas, que estas se efectuaron con montos bajos, ya que ninguna de las transacciones fraudes superó los 145K. Es decir que las transacciones fraude no se efectuaron con montos atípicos a la distribución normal del conjunto de datos.



Al estudiar los patrones de fraude es importante identificar si durante el transcurso de los dos años estudiados hay alguna concentración de fraude en el tiempo o hay una tendencia fraudulenta al estudiar el fraude por mes.



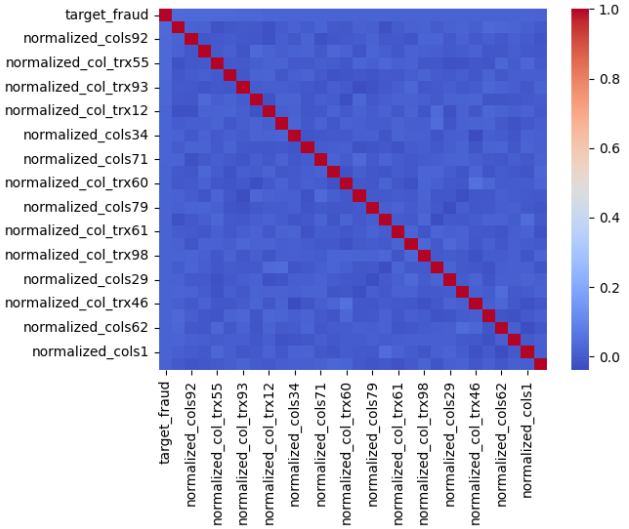
En la anterior grafica se puede observar que no hay un patrón claro o una tendencia marcada en el fraude por mes. Lo que, si se encontró, es una disminución de un 15% aproximadamente, entre el número de casos de fraude que se obtuvieron en el año 2023 con respecto al año inmediatamente anterior.

Tratamiento del desbalance de clases en variable objetivo.

Debido a que estamos frente a un caso de fraude en transacciones bancarias, es más usual encontrar muchas más transacciones no fraude que fraudes. En este caso no es la excepción, para entrenar posteriormente un modelo de aprendizaje supervisado es importante manejar el desbalance de las clases. Aquí se emplea un sobre muestreo empleando el algoritmo **BorderlineSMOTE** el cual se centra en generar muestras sintéticas cerca del límite de decisión entre las clases minoritaria y mayoritaria; enfocándose en las instancias más difíciles de clasificar. Al utilizar este método, la clase minoritaria tendrá un tamaño igual al **5%** de la clase mayoritaria. Pasando de un **99%-1%** a un **95%-5%** en el desbalance de las clases. No se agregan más muestras sintéticas para evitar generar ruido en el conjunto de datos, lo que puede conllevar a un sobre ajuste de los modelos a entrenar.

Elección de las variables predictoras

Para determinar y seleccionar las variables independientes o predictoras, se realiza un análisis multivariado sobre las características, donde se compara el grado de relación lineal que tienen las variables independientes con la variable objetivo **target_fraud**.



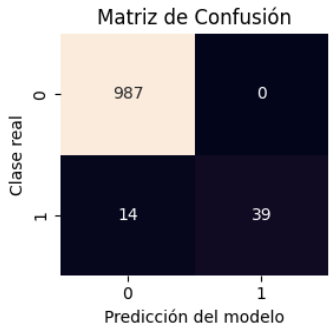
Se puede observar que, de las variables independientes que más correlación tienen con la variable objetivo ninguna tiene un coeficiente de correlación superior al 0.4, esto indica que hay una relación débil positiva o negativa con las variables independientes o no hay relación. Por esta razón se emplea un algoritmo seleccionador de características donde se toman las 20 variables más determinantes para construir el clasificador, dentro de estas variables se encuentran **TransactionValue_PSE** y **mes_transaction** (mes de la de transacción).

Selección de modelo de clasificación.

Se construyen seis modelos bases que son empleados en la solución de problemas de clasificación binaria, los cuales son un algoritmo de vecinos más cercanos, regresión logística, random forest, xgboost y un lightgbm. Los resultados del rendimiento de estos algoritmos se comparan empleando las métricas f1-score y el recall debido a que las clases están desbalanceadas.

Modelo	F1-Score	Recall
Lightgbm	0.860	0.755
Xgoost	0.860	0.755
Random Forest	0.848	0.736
Naive bayes	0.571	0.491
KNN	0.517	0.434
Logistic Regression	0.301	0.208

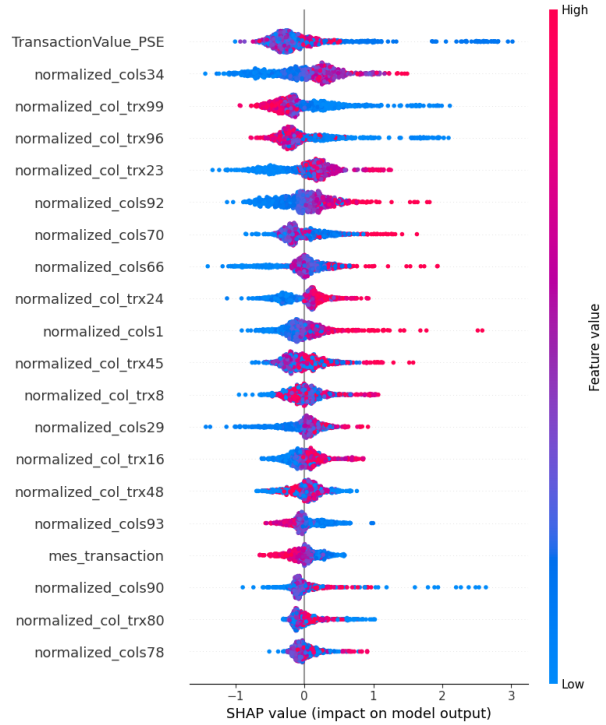
Los modelos que mejor capturan los casos de fraude y que los clasifica con una buena precisión son el Lightgbm y el Xgoost. Los cuales entrenar una serie de modelos débiles donde cada uno intenta corregir los errores del anterior. Además, son modelos que tienen parámetro de balanceo de clases que ayudan a mejorar el rendimiento de los modelos. Se emplea como modelo final el **Lightgbm** por ser un modelo más ligero computacionalmente para realizar optimización de hiperparámetros, con el objetivo de mejorar el rendimiento del modelo y buscar parámetros que ayuden a minimizar el sobreajuste.



Con los resultados de la matriz se puede decir que el modelo es bueno evitando falsos positivos, ya que no penaliza buenas transacciones como fraude. Sin embargo, aún deja pasar algunas transacciones fraudulentas (14 falsos negativos), lo que significa que no detecta todos los fraudes reales.

Con la optimización de hiperparámetros del modelo **Lightgbm**, realizando una búsqueda con una técnica bayesiana se mejoran los resultados iniciales del modelo, obteniendo un f1-score macro de **92%** y un recall macro de **87%**, lo que indica que el modelo encuentra de buena forma ambas clases con una gran precisión.

Interpretabilidad del modelo



Se puede observar que la variable más importante para el modelo son los montos de transacción por PSE, el grafico nos indica que cuando se tienen montos de transacciones bajos hay más probabilidad de tener una transacción fraudulenta en la aplicación. Luego en orden de importancia se encuentran el resto de las variables normalizadas.

Conclusión

Se logra construir un algoritmo capaz de identificar transacciones fraudes con una buena confiabilidad y nula penalización de aquellas transacciones no fraudes. En la prevención de fraude, se debe seguir refinando el modelo, ya que es preferible reducir aún más los falsos negativos para evitar pérdidas económicas, debido a que no se están friccionando las buenas transacciones. Es un algoritmo que logra generalizar o predecir casos de fraudes, es decir, que se logra tener un modelo el cual no está sobre ajustado.