Convenio Plus TI

DETECTAR FRAUDES EN TRANSACCIONES CON MONTO INUSUALMENTE ALTO PARA EL CLIENTE

Marco Ramirez

Resumen

El presente trabajo se centra en la detección de transacciones fraudulentas de alto monto, atípicos para cada cliente. Se empleó un pipeline que inicia con un análisis exploratorio de datos (EDA) para caracterizar la distribución de montos y el balance de clases, seguido de ingeniería de características, donde se crearon variables temporales, Z-score del monto respecto al historial de cada cliente y el indicador de primera compra en un comercio. Posteriormente, se entrenó un modelo inicial de LightGBM aplicando un split temporal, undersampling del 30 % en no fraudes y evaluación con métricas tradicionales (AUC-ROC y F1). Se definieron métricas personalizadas — penalización de falsos positivos y ponderación por monto — para orientar el modelo al objetivo de minimizar falsos positivos sin sacrificar la detección. La optimización de hiperparámetros con Optuna mejoró el rendimiento. En la evaluación sobre el último trimestre, el modelo alcanzó un AUC de 0.99955 y demostró una capacidad de detección elevada con riesgo controlado de falsos positivos. La métrica personalizada 'balanced_f1' ofreció el mejor equilibrio entre precisión y recall, por lo que se seleccionó como métrica final para guiar las decisiones futuras.

Descripción del Dataset

1.8M transacciones simuladas (2019-2020), variable objetivo is_fraud. Split temporal: prueba en el último trimestre de 2020.

Metodología

1. EDA

Exploración de montos y balance de clases. Identificación de fraudes concentrados en horario nocturno. Se identificó un fuerte desbalance entre clases, con menos del 1 % de transacciones fraudulentas. Los montos se distribuyeron en cuatro rangos por percentiles, y se observó que los fraudes se concentraban mayormente en el rango más alto y durante horarios nocturnos (00:00–06:00).

2. Ingeniería de Variables

Se generaron variables temporales como is_night, is_weekend, y is_business_hours a partir de la fecha de transacción. También se construyó un **Z-score por cliente** para identificar montos atípicos respecto a su historial. Se incluyeron indicadores de comportamiento como si era la **primera compra en un comercio**, la **distancia cliente–comercio**, y contadores de visitas en distintos periodos (día, mes, año). Estas variables permitieron capturar patrones de familiaridad, comportamiento y anomalías por cliente.

3. Modelo Base con LightGBM

Se entrenó un modelo inicial con LightGBM utilizando un **split temporal**: datos antes de octubre 2020 para entrenamiento y el último trimestre para prueba. Para manejar el desbalance, se aplicó **undersampling del 30 % en transacciones legítimas**, conservando todos los fraudes. El modelo se evaluó con **AUC-ROC y F1-Score**, sirviendo como base para comparar mejoras posteriores con métricas personalizadas y optimización por hiperparámetros.

4. Definición de Métricas Personalizadas

Se definieron tres métricas personalizadas para adaptar el modelo al objetivo de minimizar falsos positivos en transacciones de alto valor:

- **fp_penalty**: penaliza la proporción de falsos positivos,
- **amt_weighted_fp**: da mayor peso a errores en montos altos, y
- **balanced_f1**: modifica la F1 clásica para mejorar el equilibrio entre precisión y recall.

Tras comparar su desempeño, **balanced_f1 (0.8356)** fue seleccionada por su mejor compromiso entre detección y control de alertas falsas.

5. Optimización de Hiperparámetros

Para mejorar el rendimiento del modelo, se utilizó **Optuna** como herramienta de búsqueda bayesiana. Se ejecutaron **50 iteraciones** probando combinaciones de hiperparámetros críticos de LightGBM, como learning_rate, num_leaves y scale_pos_weight, evaluando con AUC-ROC. Este proceso permitió encontrar una configuración más robusta, optimizando la detección de fraudes con un menor riesgo de falsos positivos.

Descripción de la implementación práctica

El desarrollo se realizó en **Python 3.11** dentro de un **Jupyter Notebook**, utilizando pandas, numpy, scikit-learn, LightGBM, Optuna, seaborn y matplotlib. El flujo incluyó: carga y exploración de datos, ingeniería de variables (temporales y de comportamiento), preparación del conjunto, división temporal, entrenamiento del modelo base, definición de métricas personalizadas, optimización de hiperparámetros con Optuna y evaluación final. El proyecto completo se encuentra documentado en el repositorio: https://github.com/mvrcentes/ClientAwareFraudDetect

Resultados

El modelo entrenado alcanzó un AUC de 0.99955 y un F1-Score de 0.8356, mostrando excelente capacidad para distinguir fraudes con bajo nivel de falsos positivos.

Uno de los hallazgos más relevantes fue que el 80 % de los fraudes ocurrieron en horario nocturno (00:00–06:00), lo que validó la importancia de la variable is_night.

Se evaluaron tres métricas personalizadas para guiar la decisión final:

- **fp_penalty (1.1235)**: enfocada en minimizar falsos positivos.
- **amt_weighted_fp (1.1545)**: penaliza errores en transacciones de alto valor.
- **balanced_f1 (0.8356)**: logró el mejor equilibrio entre precisión y recall.

Se eligió balanced_f1 como métrica principal por ofrecer un desempeño más estable y práctico en escenarios reales, donde es necesario maximizar la detección sin generar alertas excesivas. Esta métrica permite comparar iteraciones futuras y comunicar resultados de forma clara a los stakeholders.

Conclusión

El modelo desarrollado demostró una alta capacidad para detectar fraudes de alto monto, combinando ingeniería de variables contextuales con aprendizaje automático y métricas personalizadas. La métrica balanced_f1 permitió lograr un equilibrio efectivo entre precisión y recall, validando la utilidad del enfoque para reducir riesgos sin generar una sobrecarga de alertas. Este sistema sienta una base sólida para una detección más inteligente, adaptable y centrada en el comportamiento del cliente.