UNIVERSIDAD DE MÁLAGA

E.T.S. INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE MÁLAGA

Graduado en Ingeniería del Software

Detección de Anomalías en Transacciones Financieras aplicando Inteligencia Artificial para Evitar el Fraude

Anomaly Detection in Financial Transactions applying Artificial Intelligence for Fraud Prevention

Realizado por
Pablo León Vázquez

Tutorizado por
Jamal Toutouh El Alamin

Departamento
Lenguajes y Ciencias de la Computación

MÁLAGA, junio de 2025

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
Graduado en Ingeniería del Software

Detección de Anomalías en Transacciones Financieras aplicando Inteligencia Artificial para Evitar el Fraude

Anomaly Detection in Financial Transactions applying Artificial Intelligence for Fraud Prevention

Realizado por
Pablo León Vázquez

Tutorizado por
Jamal Toutouh El Alamin

Departamento
Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, JUNIO 2025

Fecha defensa: Julio de 2025

Resumen

En la actualidad, las instituciones financieras juegan un papel crucial en la economía global, gestionando un inmerso volumen de transacciones cada día. Cualquier anomalía en estos procesos puede tener consecuencias graves tanto para los usuarios como para las propias entidades, lo que hace imprescindible garantizar la seguridad en este sector.

Dado que el fraude financiero representa una amenaza constante, es fundamental que estas organizaciones adopten tecnologías innovadoras que les permitan optimizar sus sistemas de detección y prevención. En este contexto, el uso de Machine Learning ha demostrado ser una herramienta clave, ya que permite identificar patrones anómalos en las transacciones con gran precisión y en tiempo real, mejorando significativamente la capacidad de respuesta ante posibles fraudes.

Este documento detalla el proceso completo, desde la recopilación y preprocesamiento de datos, aplicando técnicas de ingeniería de características, como la transformación de fechas o el cálculo de distancias, con el objetivo de mejorar la detección de anomalías. Además, se han explorado distintos enfoques según el tipo de modelo, ajustando el peso de las clases en modelos supervisados para corregir el desbalance de datos y utilizando reducción de dimensionalidad (PCA) en modelos no supervisados. Finalmente, se ha desarrollado una API en FastAPI capaz de analizar y predecir transacciones en tiempo real y almacenar los resultados para su posterior consulta y evaluación.

Palabras clave: Machine Learning, transacciones financieras, fraude, inteligencia artificial, predicción.

Abstract

Currently, financial institutions play a crucial role in the global economy, managing an immense volume of transactions daily. Any anomaly in these processes can have severe consequences for both users and the institutions themselves, making it essential to ensure security in this sector.

Since financial fraud represents a constant threat, it is imperative for these organizations to adopt innovative technologies that enable them to optimize their detection and prevention systems. In this context, the use of Machine Learning has proven to be a key tool, as it allows for the identification of anomalous transaction patterns with high precision and in real time, significantly enhancing the ability to respond to potential fraud.

This document outlines the complete process, from data collection and preprocessing to the application of feature engineering techniques, such as date transformation and distance calculation, aimed at improving anomaly detection. Additionally, various approaches have been explored depending on the model type, adjusting class weights in supervised models to address data imbalance and employing dimensionality reduction (PCA) in unsupervised models. Finally, an API has been developed using FastAPI, capable of analyzing and predicting transactions in real time while storing the results for subsequent consultation and evaluation.

Keywords: Machine Learning, financial transactions, fraud, artificial intelligence, prediction.

Índice

Índice de figuras

Índice de cuadros

1 Introducción

1.1. Motivación

La **Inteligencia Artificial (IA)** se define como una rama de la informática orientada al desarrollo de sistemas capaces de ejecutar tareas que, tradicionalmente, requieren de la inteligencia humana, tales como el aprendizaje, el razonamiento y la percepción [21]. En el marco de la IA, el ***Machine Learning (ML)*** constituye una subdisciplina que emplea algoritmos para identificar patrones y realizar predicciones a partir de conjuntos de datos, los cuales pueden estar compuestos por números, textos o imágenes [11]. El ***Machine Learning*** se utiliza para anticipar comportamientos y respaldar la toma de decisiones fundamentadas en patrones extraídos de grandes volúmenes de información, lo que resulta especialmente relevante en ámbitos como la **detección de fraude**.

El **fraude** se entiende habitualmente como un acto o una serie de actos deliberados, ejecutados por individuos mediante el uso de engaños y astucias, recurriendo tanto a la sugerencia de falsedad como a la omisión de la verdad [1]. Existen diversas tipologías de fraude, si bien en este trabajo el foco se sitúa en las transacciones fraudulentas.

Una **transacción financiera** se define como un proceso en el que se produce un intercambio de activos financieros -como dinero, valores, títulos, bienes o servicios- entre dos partes, con el objetivo de cumplir un acuerdo financiero [13].

Por su parte, una **transacción fraudulenta** corresponde a una actividad no autorizada o ilícita que implica el uso de instrumentos de pago o sistemas financieros, con la finalidad de obtener dinero, productos o servicios sin el consentimiento o la autorización adecuada del titular de la cuenta [6].

A comienzos de la década de 2000, la adopción de internet en el comercio minorista comenzó a generalizarse, lo que propició el crecimiento del comercio electrónico. Empresas co-

mo *Amazon* y *eBay* facilitaron la compraventa de productos en línea, promoviendo el uso de tarjetas de crédito y débito como principal método de pago. No obstante, la seguridad de las transacciones representaba una preocupación relevante, lo que impulsó el desarrollo de protocolos seguros y eficientes, como ***3D Secure*** [25], introducido por *Visa* y *Mastercard* en 2001 con el objetivo de reducir el fraude en las compras en línea.

En 2007, la Unión Europea promulgó la **Directiva de Servicios de Pago** (*PSD1*), que favoreció la creación de un mercado de pagos más integrado y seguro [16]. Posteriormente, en 2015, la ***PSD2*** reforzó la seguridad mediante la implementación de la **Autenticación Reforzada del Cliente** (*SCA*), exigiendo a las entidades financieras la verificación de la identidad del usuario a través de, al menos, dos factores de autenticación [19].

La aplicación de todas estas normativas permitió que, a pesar del auge en las compras en línea, se redujera el volumen de transacciones fraudulentas, como se observa en la Figura 1.

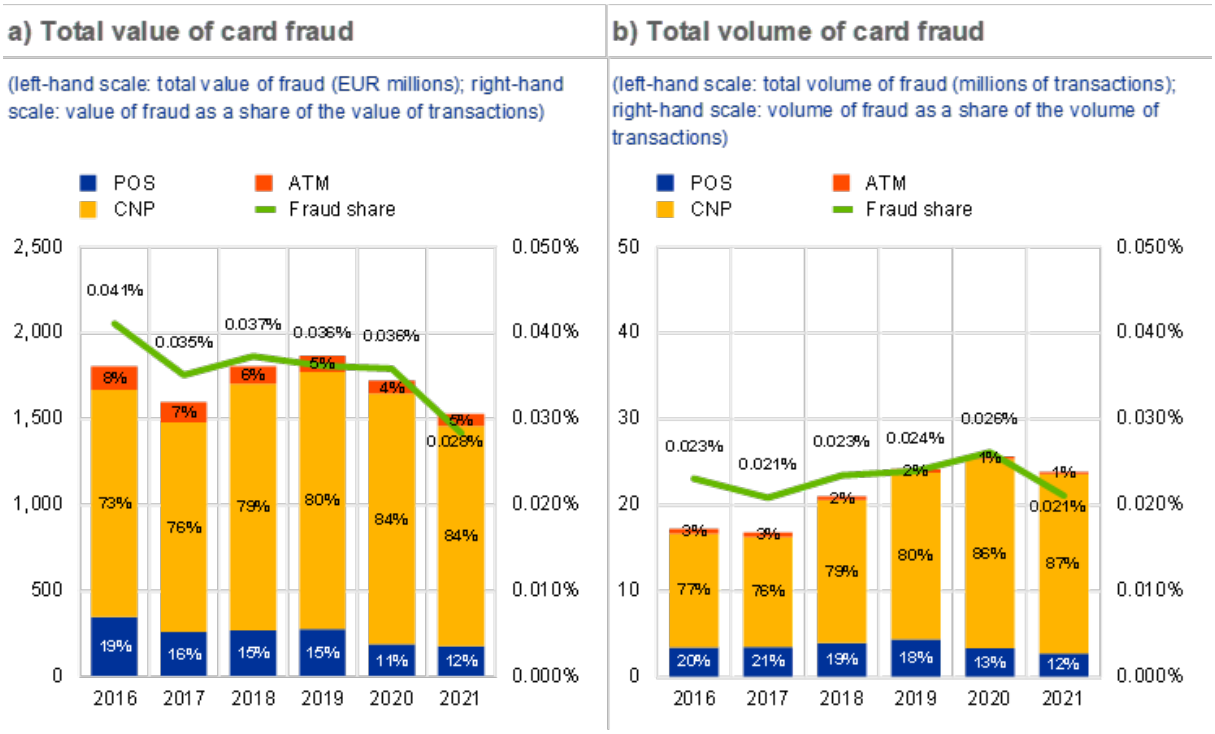

Figura 1: Volumen de fraude en compras en línea entre 2016 y 2021 [7]

Tras la pandemia de *COVID-19* en 2020, se produjo una aceleración en la adopción de los pagos digitales motivada por la necesidad de minimizar el contacto físico durante las transac-

ciones. En este contexto, numerosas empresas implementaron sistemas de pago en línea y los consumidores optaron, de manera creciente, por soluciones digitales para realizar compras de forma segura. Esta tendencia hacia la digitalización derivó en un ligero incremento del fraude en los pagos con tarjeta entre principios de 2022 y mediados de 2023, como se observa en la columna central *"Card payments (issuers)"* de la Figura 2.

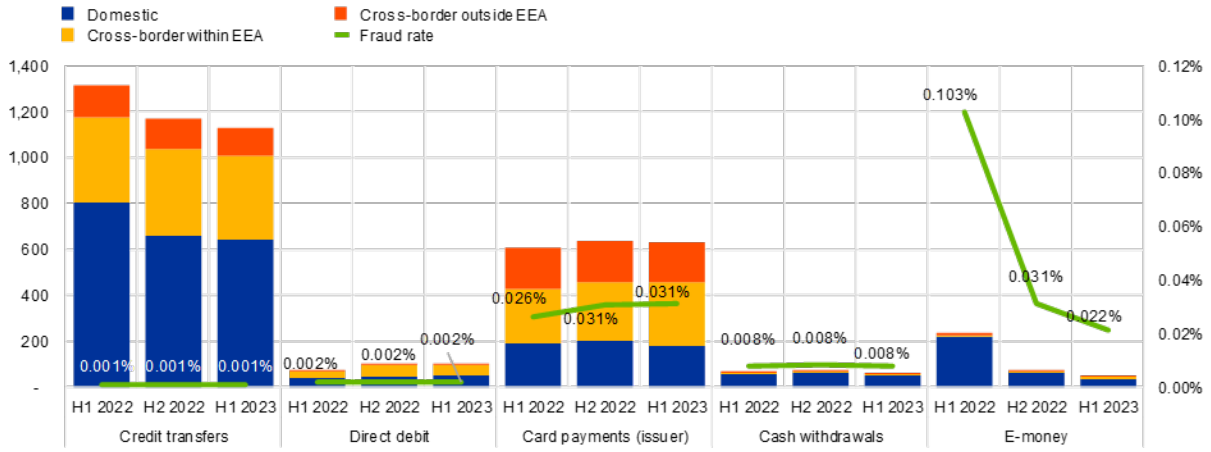

Figura 2: Volumen de fraude en distintos tipos de pagos entre 2022 y 2023 [8]

A partir de estos datos, se pone de manifiesto la necesidad urgente de incorporar tecnologías innovadoras y avanzadas, como el ***Machine Learning***, con el fin de garantizar la seguridad de los datos de los usuarios.

1.2. Objetivos

El objetivo principal de este trabajo consiste en diseñar e implementar un sistema que, mediante la transformación de datos y la aplicación de técnicas de *Machine Learning*, permita identificar en tiempo real la probabilidad de que una transacción sea fraudulenta.

En primer lugar, se procederá a la investigación de diferentes tipos de modelos de *Machine Learning*, realizando varias iteraciones sobre cada uno de ellos con el fin de determinar el más eficiente. Una vez seleccionado el modelo óptimo, se construirá un sistema basado en dicho modelo, capaz de procesar transacciones entrantes y generar una predicción asociada a cada una. Tras la predicción, la transacción y su probabilidad de fraude se almacenarán en una base de datos, permitiendo su consulta posterior por parte de la empresa que haya realizado

la predicción.

1.3. Metodología

La selección de una metodología adecuada constituye un aspecto esencial en el desarrollo de software, ya que determina la organización del trabajo y la gestión de las tareas. Un enfoque estructurado facilita tanto la planificación como la ejecución del proyecto, contribuyendo a mejorar la calidad del producto final.

Existen diversos enfoques metodológicos en el desarrollo de software, entre los que destacan las **metodologías tradicionales** y las **metodologías ágiles**. Las metodologías tradicionales se caracterizan por una planificación detallada y una documentación exhaustiva previa al inicio de la implementación, resultando adecuadas para proyectos con requisitos bien definidos y poco susceptibles de cambio a lo largo de su desarrollo.

En el presente proyecto se ha optado por una **metodología ágil**, basada en un desarrollo iterativo e incremental. Este enfoque permite que los requisitos y las soluciones evolucionen conforme avanza el trabajo, adaptándose a las necesidades emergentes. Además, dado que se ha realizado un trabajo de investigación para evaluar de manera competitiva el rendimiento de distintos modelos, se ha considerado apropiada la adopción de este tipo de metodología. Para ello, se han desarrollado ***sprints* de aproximadamente tres semanas** de duración, en los que se han investigado, planificado y desarrollado diferentes funcionalidades del sistema. Asimismo, se han realizado reuniones periódicas con el tutor del proyecto, con una frecuencia aproximada de mes y medio, para revisar los avances, discutir posibles mejoras y ajustar la planificación según el progreso obtenido.

La utilización de esta metodología ha proporcionado una mayor flexibilidad durante el desarrollo del proyecto, facilitando la integración de cambios y mejoras conforme se identificaban nuevas necesidades. Este modo de trabajo ha resultado especialmente útil para asegurar que el sistema final cumpla con los objetivos establecidos y se adapte de manera eficiente a los requisitos del problema tratado.

1.4. Estructura del documento

El presente documento se estructura en varios capítulos organizados de manera lógica para reflejar el desarrollo completo del proyecto. En primer lugar, se introduce el contexto y el estado del arte, donde se analiza la problemática abordada, así como las principales soluciones y avances existentes en el ámbito de estudio. Posteriormente, se describen las tecnologías utilizadas y la implementación del sistema, detallando las herramientas, librerías y entornos empleados. A continuación, el capítulo 4 recoge los pasos seguidos durante el proceso experimental, la descripción del conjunto de datos utilizado, las técnicas de ingeniería de características aplicadas, los procedimientos de búsqueda de parámetros óptimos y las condiciones de experimentación. Seguidamente, en el capítulo 5 se presentarán y se discutirán los resultados obtenidos, diferenciando entre los modelos supervisados y no supervisados, analizando las distintas iteraciones realizadas y valorando su evolución. Finalmente, el trabajo concluye con un capítulo de 6, donde se recogen las principales aportaciones del proyecto y se proponen posibles mejoras o investigaciones a desarrollar. El documento se completa con un anexo que incluye el manual de instalación del sistema, facilitando así su correcta implementación y ejecución.

2
Contexto

En este capítulo se presentará un estudio del estado del arte relacionado con la detección de fraudes en transacciones financieras mediante inteligencia artificial, más concretamente con *Machine Learning*. Para ello, se comenzará definiendo los distintos tipos de fraude en el ámbito financiero. Posteriormente, se abordarán los conceptos fundamentales de inteligencia artificial y *Machine Learning*, explorando en cómo estas tecnologías han revolucionado la detección y prevención de actividades fraudulentas.

Asimismo, se analizarán los modelos de *Machine Learning* utilizados para dicha detección, detallando sus características, ventajas y limitaciones.

2.1. Fraudes en transacciones financieras

De acuerdo con *Stripe*, los fraudes en transacciones financieras se clasifican en diversas categorías, entre las cuales destacan el fraude con tarjetas de crédito, el robo de identidad y el fraude con cheques [6]. A continuación, se describirán los tipos más relevantes para este estudio.

2.1.1. Fraude con tarjetas de crédito

Los fraudes con tarjetas de crédito implican la utilización no autorizada de una tarjeta de crédito o de sus datos para realizar compras, retirar efectivo o emplear los fondos de manera ilícita sin el consentimiento del titular. Entre sus variantes más frecuentes se encuentran:

1. **Fraude de tarjeta no presente (CNP)**: Este tipo de fraude es uno de los que abordaremos en este proyecto, ya que se practica en transacciones en línea o por teléfono cuando no es necesario tener la tarjeta física.

2. ***Skimming***: Los estafadores son capaces de robar los datos de la tarjeta con dispositivos instalados en cajeros automáticos o en terminales de puntos de venta.

3. **Tarjetas falsificadas**: Estas son tarjetas falsas que utilizan los datos previamente robados de otra tarjeta.

2.1.2. Robo de identidad

Los robos de identidad consisten en el proceso mediante el cual un individuo se apropia ilegítimamente de la identidad de un tercero con el objetivo de acceder a sus cuentas bancarias, solicitar créditos o realizar transacciones sin autorización. Los estafadores suelen obtener información personal sensible a través de técnicas como el *phishing* o el *malware*, permitiéndoles suplantar a la víctima y ejecutar actividades fraudulentas en su nombre.

2.1.3. Pagos fraudulentos en línea

Los pagos fraudulentos en línea se refieren a transacciones efectuadas sin la autorización del titular mediante plataformas digitales o sistemas de pago electrónico. Estas actividades ilícitas incluyen el acceso no consentido a cuentas de usuarios, el uso de técnicas de ingeniería social para obtener datos financieros o la explotación de vulnerabilidades en sistemas de pago. Entre los métodos más comunes destacan el *phishing*, la clonación de tarjetas virtuales y la interceptación de credenciales durante transacciones en línea.

2.2. *Machine Learning*

Tras definir los conceptos fundamentales del *Machine Learning*, resulta esencial profundizar en sus principios básicos. Esta disciplina se fundamenta en la capacidad de los sistemas para aprender patrones a partir de conjuntos de datos mediante algoritmos específicos. En términos generales, estos algoritmos se clasifican principalmente en tres categorías: aprendizaje supervisado, no supervisado y por refuerzo. En el contexto de este trabajo, el análisis se centra en modelos supervisados y no supervisados, los cuales se compararán para determinar cuál presenta un mejor rendimiento ante el problema planteado y el conjunto de datos seleccionado.

2.2.1. Funcionamiento

El *Machine Learning* se fundamenta en algoritmos que analizan datos, identifican patrones y generan modelos predictivos. El proceso general, representado en la Figura 3, sigue las siguientes etapas:

1. **Recolección de datos**: Etapa inicial en la que se recopilan datos relevantes de diversas fuentes. En este trabajo, se ha utilizado un conjunto de datos público previamente clasificado [17]. La calidad y la cantidad de los datos resultan críticas para el rendimiento del modelo.

2. **Preprocesamiento**: Una vez obtenidos los datos, se procede a analizar, limpiar y transformar los datos para garantizar su idoneidad. Este proceso incluye la eliminación de valores atípicos, el tratamiento de datos faltantes, la normalización de escalas y la codificación de variables categóricas en formatos numéricos.

3. **Selección del modelo**: Fase en la que se elige el algoritmo más adecuado según la naturaleza del problema y las características de los datos. En este estudio, se evaluarán diferentes tipos de modelos para determinar cuál ofrece el mejor desempeño en el contexto específico.

4. **Entrenamiento**: Etapa donde el modelo aprende a reconocer patrones y realizar predicciones utilizando los datos de entrenamiento. El objetivo principal radica en ajustar los parámetros del modelo para evitar tanto el sobreajuste como el subajuste.

5. **Evaluación**: Fase de validación en la que se prueba el modelo con datos no utilizados durante el entrenamiento. Para medir su eficacia, se utilizan indicadores como precisión, exactitud, exhaustividad y puntuación *F1*. En problemas de clasificación supervisada, adicionalmente se emplea la curva *AUC-ROC*, que analiza la relación entre la tasa de verdaderos positivos y falsos positivos, ofreciendo una evaluación integral de la capacidad discriminativa del modelo.

6. **Optimización**: Etapa final donde, basándose en los resultados de la evaluación, se implementan ajustes para optimizar su precisión. Esto puede involucrar la modificación

de hiperparámetros, el reequilibrado de datos o la aplicación de técnicas avanzadas de ingeniería de características.

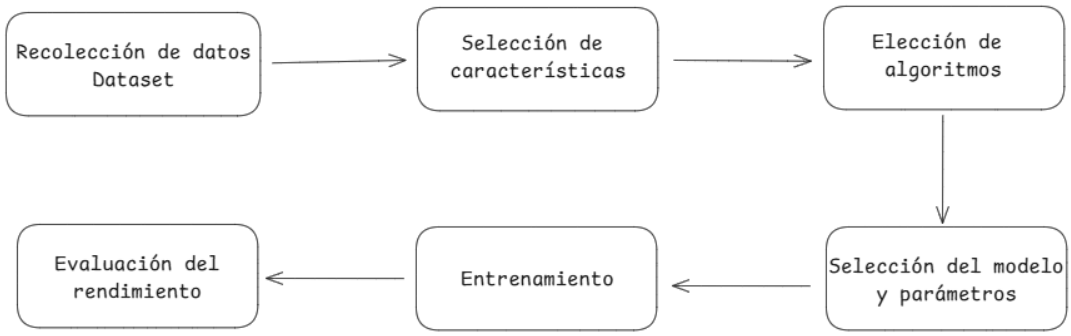

Figura 3: Esquema funcionamiento *Machine Learning* genérico [5]

2.2.2. Tipos de modelos aplicados al fraude

En la detección de fraude en transacciones financieras, los modelos de *Machine Learning* pueden clasificarse en dos enfoques principales: **aprendizaje supervisado** y **aprendizaje no supervisado**.

Aprendizaje supervisado

El aprendizaje supervisado se fundamenta en la utilización de datos previamente etiquetados, es decir, en transacciones que han sido clasificadas como fraude o no fraude. Este enfoque permite que los modelos aprendan a identificar patrones específicos asociados al fraude (véase la Figura 4).

La principal ventaja del aprendizaje supervisado radica en su capacidad para generar predicciones precisas cuando se dispone de un conjunto de datos bien estructurado y representativo. Sin embargo, su eficacia depende en gran medida de la calidad y cantidad de datos etiquetados disponibles. En el ámbito financiero, donde las técnicas de fraude evolucionan de manera constante, los modelos supervisados pueden presentar limitaciones al no ser capaces de detectar nuevas modalidades de fraude que no hayan sido observadas previamente. Además, otra limitación relevante es la escasez de datos etiquetados como fraude.

Ventajas:

1. Alta precisión en la detección de fraude con suficientes datos etiquetados.
2. Permite interpretar las decisiones tomadas a través de técnicas como árboles de decisión.
3. Se puede actualizar con nuevos datos de fraude para mejorar la precisión.

Desventajas

1. Requiere de un gran volumen de datos etiquetados, lo que puede ser costoso y lento de obtener.
2. Puede tener dificultades para detectar fraudes novedosos o no registrados en el historial.
3. La calidad del modelo depende en gran medida de la representatividad de los datos de entrenamiento.

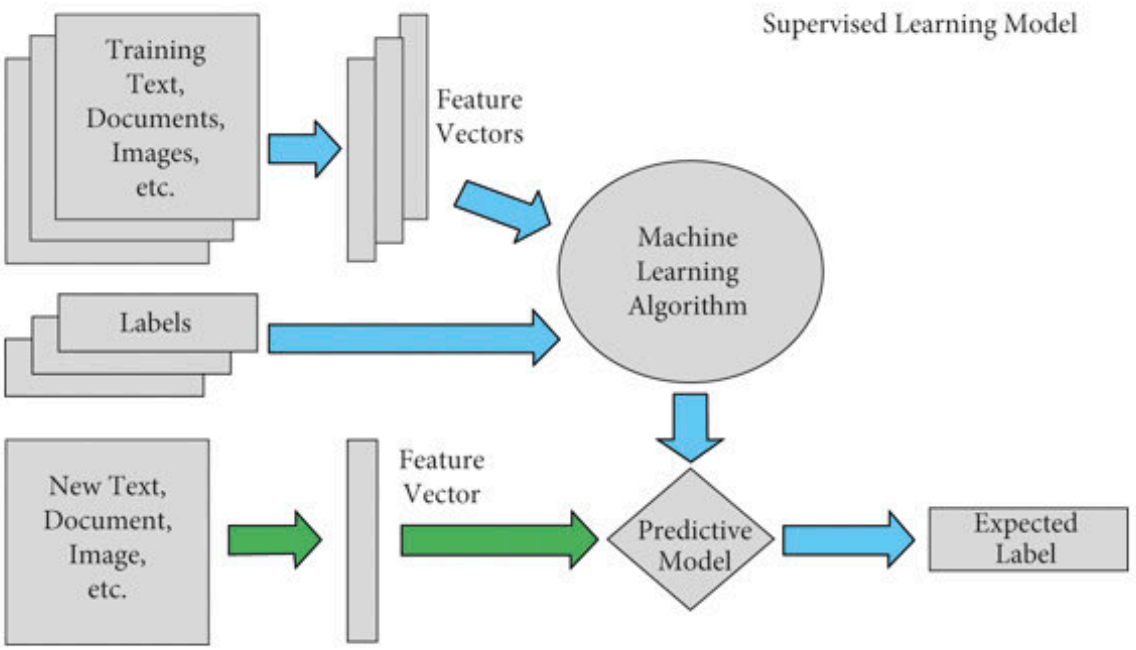

Figura 4: Esquema funcionamiento modelo supervisado [5]

Aprendizaje no supervisado

El aprendizaje no supervisado se centra en el análisis de transacciones sin etiquetas previas, con el objetivo de identificar anomalías y patrones sospechosos. En este enfoque, los modelos no tienen información previa sobre qué transacciones son fraudulentas, sino que buscan diferencias significativas en los datos que puedan indicar un comportamiento inusual (véase la Figura 5).

Ventajas:

1. No requiere de datos ya etiquetados, reduciendo en costos y tiempo de preparación.
2. Detecta fraudes novedosos y no registrados previamente.
3. Se adapta a cambios en los patrones de fraude en tiempo real.

Desventajas:

1. Puede generar falsos positivos si no se ajusta correctamente.
2. Es más difícil de interpretar las decisiones del modelo.
3. Puede requerir de ajustes constantes para evitar detectar transacciones legítimas como sospechosas.

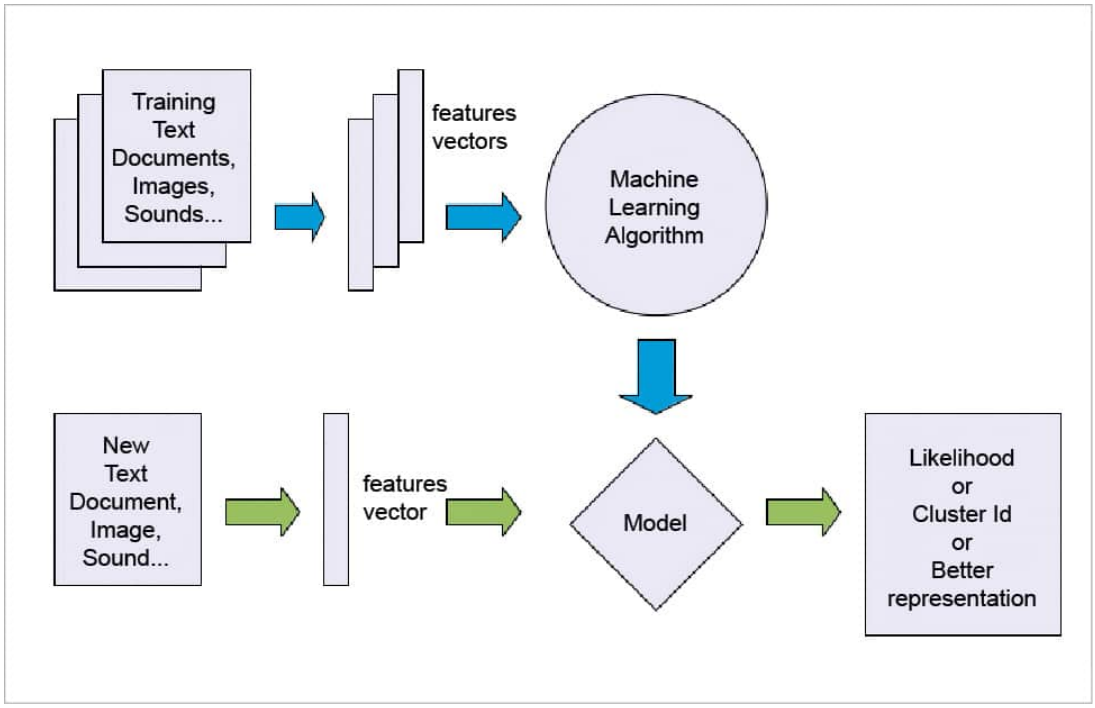

Figura 5: Esquema funcionamiento modelo no supervisado [5]

2.2.3. Métricas para evaluar los modelos supervisados

Se describirán las principales métricas utilizadas para evaluar el rendimiento de los modelos con aprendizaje supervisado en tareas de clasificación, como es el caso de la detección de anomalías en transacciones financieras. Las métricas permiten medir de manera objetiva la capacidad de los modelos para realizar predicciones correctas y evaluar su desempeño en función de los datos disponibles.

Precisión (*accuracy*)

La precisión o exactitud es una de las métricas más comunes en problemas de clasificación, y se define como la proporción de predicciones correctas sobre el total de predicciones realizadas. Esta métrica es útil cuando las clases están balanceadas, ya que mide la capacidad del modelo para predecir correctamente la clase. En nuestro caso no va a ser tan útil debido al desbalance en el conjunto de datos. Al ser la clase objetivo menor del 1 %, si se predice como si no fuesen fraudes, obtendría una precisión muy alta de más del 99 %, lo que no es correcto.

AUC-ROC

El *AUC-ROC* (Área bajo la curva *ROC*) es una métrica que evalúa la capacidad de un modelo para discriminar entre las clases positivas y negativas. La curva *ROC* se genera a partir de diferentes umbrales de clasificación, y el área bajo la curva (*AUC*) indica qué tan bien el modelo es capaz de distinguir entre las clases. Un valor cerca de 1 sugiere un modelo con una gran capacidad de discriminación, mientras que un valor cercano a 0,5 indica un modelo que no distingue bien entre las clases.

Recall

El *recall* o tasa de recuperación mide la proporción de casos positivos correctamente identificados por el modelo. Es particularmente útil cuando el costo de pasar por alto un caso positivo (falso negativo) es alto, como en la detección de fraudes, donde los fraudes no detectados pueden tener consecuencias graves.

Puntuación *F1* (*F1-score*)

La puntuación *F1* es la media armónica entre la precisión (*accuracy*) y el *recall*. Es una métrica más equilibrada que la precisión cuando hay un desbalance entre las clases, ya que pondera tanto la capacidad del modelo para identificar los positivos como para evitar los falsos positivos. Una puntuación *F1* alta indica un buen rendimiento en la predicción de las clases

positivas.

Matriz de confusión

La matriz de confusión es una herramienta visual que permite analizar el desempeño de un modelo de clasificación mostrando: los verdaderos positivos, los falsos positivos, los verdaderos negativos y los falsos negativos. A partir de ella, se pueden calcular otras métricas como la precisión, *recall* y la puntuación *F1*, proporcionando una visión más completa de cómo el modelo realiza sus predicciones.

2.2.4. Métricas para evaluar los modelos no supervisados

Ahora se explorarán las métricas utilizadas para evaluar el rendimiento de los modelos con aprendizaje no supervisado, los cuales no tienen etiquetas de clase y su objetivo principal es identificar patrones anómalos o inusuales en los datos. Para la evaluación de este tipo de modelos, se emplean diferentes enfoques que se centran en las características estadísticas y la distribución de las predicciones.

Distribución de puntajes de anomalía

Una de las métricas clave en los modelos no supervisados es el análisis de la distribución de los puntajes de anomalía, que mide la intensidad de la "anomalía"de cada observación en el conjunto de datos. Los modelos asignan un puntaje a cada muestra, indicando qué tan probable es que sea una anomalía. La distribución de estos puntajes ayuda a determinar un umbral adecuado para clasificar las observaciones como normales o anómalas, y puede proporcionar información valiosa sobre la naturaleza de las anomalías detectadas.

Evaluación de predicciones con reducción de dimensionalidad (*PCA*)

Otra métrica importante en modelos no supervisados es la evaluación de las predicciones a través de técnicas de reducción de dimensionalidad, como el Análisis de Componentes Principales (*PCA*). *PCA* puede ser utilizado para reducir las dimensiones de los datos y visualizar las predicciones de anomalías en un espacio de menor dimensión. La distribución de las predicciones proyectadas sobre los componentes principales puede ofrecer una representación visual del comportamiento de los datos, ayudando a evaluar cómo los puntos de datos anómalos se distribuyen en relación con los puntos normales. Las técnicas de visualización, como los gráficos de dispersión de *PCA*, pueden ser útiles para identificar patrones y evaluar la efectividad del modelo en la detección de anomalías.

2.3. Modelos aplicados en el trabajo

En este trabajo se quiere probar la capacidad de diferentes modelos de *Machine Learning* cuando abordan el problema de la detección de fraude. A continuación, se presentan los modelos empleados.

2.3.1. *Random Forest*

Random Forest es un modelo basado en múltiples árboles de decisión que combinan sus predicciones para mejorar la precisión y reducir el sobreajuste. Funciona construyendo múltiples árboles con subconjuntos de datos y variables aleatorias, lo que lo hace robusto frente a datos ruidosos (véase la Figura 6).

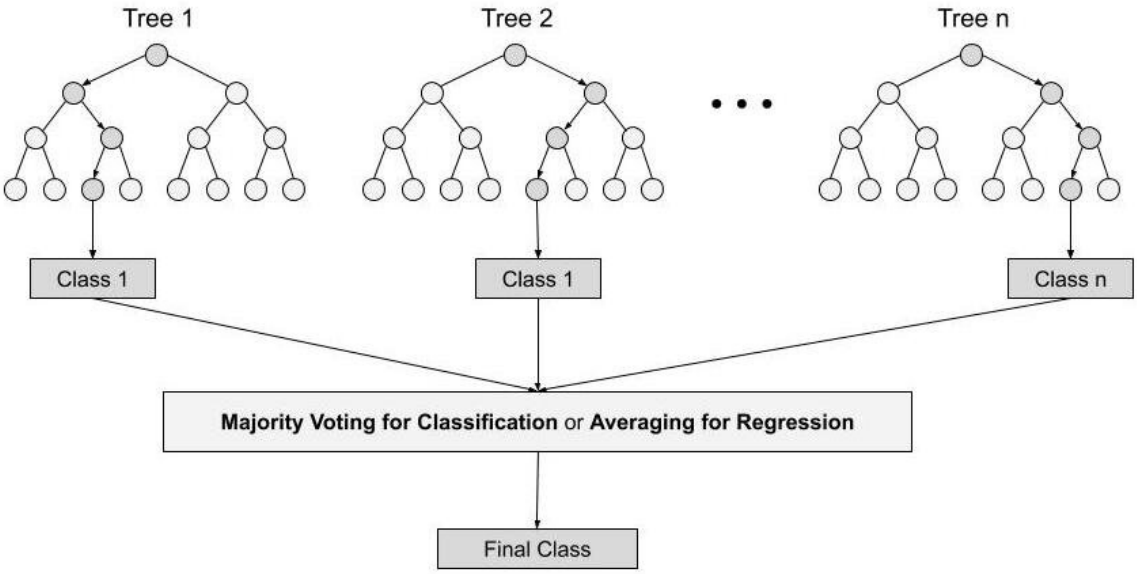

Figura 6: *Random Forest* [22]

2.3.2. *LightGBM*

LightGBM es un algoritmo de *boosting* basado en árboles de decisión que optimiza el uso de gradientes para mejorar la velocidad y precisión en grandes volúmenes de datos. A diferencia de otros enfoques como *XGBoost*, que crecen por nivel, *LightGBM* utiliza una expansión por hoja (*leaf-wise*) como se puede apreciar en la Figura 7. Gracias a esto, le permite reducir significativamente el tiempo de entrenamiento sin perder capacidad predictiva. Esta estrategia

es especialmente útil en conjuntos de datos grandes como el de este proyecto, ya que permite manejar eficientemente el desbalance de clases y mejorar la detección de fraudes.

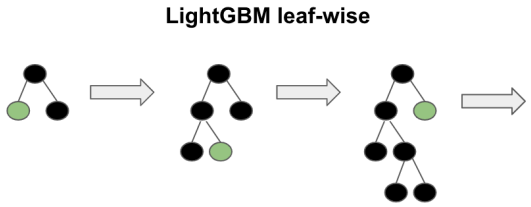

Figura 7: *Leaf Wise LightGBM* [26]

2.3.3. *Isolation Forest*

Isolation Forest es un modelo basado en la idea de que las anomalías son más fáciles de aislar que las instancias normales. Construye múltiples árboles de decisión aleatorios para separar las transacciones y detectar aquellas que se comportan de manera inusual (véase la Figura 8).

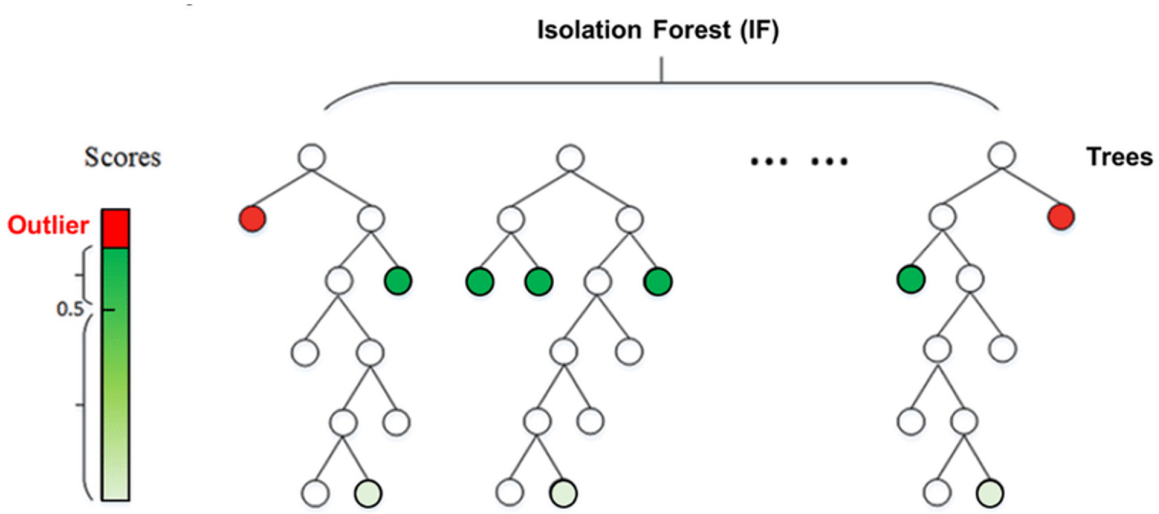

Figura 8: *Isolation Forest* [10]

2.3.4. *One Class SVM*

One Class SVM es un modelo de aprendizaje no supervisado que identifica anomalías utilizando una función de margen máximo para diferenciar las transacciones normales de las fraudulentas. Es útil en escenarios donde la cantidad de datos fraudulentos es muy baja en comparación con las transacciones legítimas. La Figura 9 representa cómo se crea el hiperplano para la clasificación en el SVM.

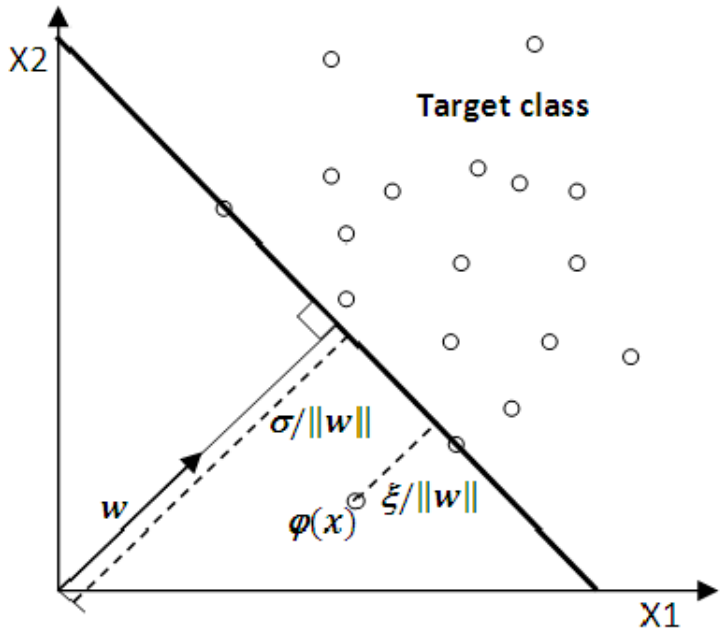

Figura 9: *One Class SVM* [3]

3

Tecnologías usadas e Implementación del sistema

Para el desarrollo de este proyecto, la selección de tecnologías adecuadas constituye un aspecto fundamental para garantizar la construcción de un sistema eficiente y escalable. En primer lugar, se detallarán el lenguaje de programación seleccionado y las principales dependencias empleadas, explicando su papel dentro del desarrollo del sistema. Posteriormente, se abordará el proceso de implementación del sistema, describiendo la estructura adoptada y las decisiones técnicas tomadas en relación con la organización del código, la integración del modelo de *Machine Learning* con la *API* y la gestión de la base de datos, con el objetivo de asegurar un funcionamiento óptimo y confiable.

3.1. Tecnologías utilizadas

En la implementación de este sistema se han seleccionado diversas tecnologías, cada una elegida estratégicamente por sus capacidades específicas y su adecuación a los requisitos del proyecto. A continuación, se detallan las herramientas fundamentales que conforman la arquitectura tecnológica de la solución desarrollada.

3.1.1. *Python*

"*Python* es un lenguaje interpretado, interactivo y orientado a objetos. Incorpora módulos, excepciones, tipado dinámico, tipos de datos de muy alto nivel y clases. *Python* combina un poder destacado con una sintaxis muy clara. Tiene interfaces a muchas llamadas de sistema y bibliotecas, así como a varios sistemas de ventana, y es extensible en C o C++-[18]. Con el

tiempo, *Python* se ha consolidado como una opción sólida en inteligencia artificial, gracias a su simplicidad y a una amplia variedad de bibliotecas especializadas. Su versatilidad y el respaldo de una comunidad activa lo han convertido en una elección adecuada para este trabajo.

3.1.2. *MySQL*

"*MySQL* es un sistema *open source* de administración de bases de datos que es desarrollado y soportado por *Oracle*-[2]. Destaca por fiabilidad, rendimiento y facilidad de integración con diversas aplicaciones. Su robustez y flexibilidad lo hacen una elección idónea para el almacenamiento y la gestión de datos en este proyecto.

3.1.3. *Scikit-Learn*

Scikit-learn "es una biblioteca para aprendizaje automático de software libre para el lenguaje de programación *Python*. Incluye varios algoritmos de clasificación, regresión y análisis de grupos entre los cuales están máquinas de vectores de soporte, bosques aleatorios, *Gradient boosting*, *K-means* y *DBSCAN*. Está diseñada para interoperar con las bibliotecas numéricas y científicas NumPy y SciPy-[23]. Scikit-learn se ha consolidado como una herramienta fundamental en el desarrollo de modelos de aprendizaje automático debido a su facilidad de uso, eficiencia y compatibilidad con otras bibliotecas del ecosistema científico de *Python*. Su amplia colección de algoritmos y su enfoque en la simplicidad hacen que la implementación de los modelos sea rápida y efectiva, lo que la hace especialmente adecuada en este trabajo.

3.1.4. *Pandas*

"La biblioteca de software de código abierto *Pandas* está diseñada específicamente para la manipulación y el análisis de datos en el lenguaje *Python*. Es potente, flexible y fácil de usar-[15]. Gracias a su eficiencia en el manejo de estructuras como *DataFrames* y *Series* y su capacidad para procesar grandes volúmenes de información de manera intuitiva, la hacen ideal para la preparación y exploración de los datos en este trabajo.

3.1.5. *Numpy*

"*Numpy* es una biblioteca de *Python* muy popular que se utiliza principalmente para realizar cálculos matemáticos y científicos-[14]. Destaca por su eficiencia en el manejo de *arrays*

multidimensionales y su amplia colección de funciones matemáticas. Su optimización en operaciones numéricas la hace ideal para el procesamiento de datos de este trabajo.

3.1.6. *Matplotlib*

"*Matplolib* es una librería *Python open source* que permite crear visualizaciones de datos-[12]. Proporciona herramientas versátiles para generar gráficos claros y personalizados. Además, su integración con otras bibliotecas científicas la hace especialmente útil para analizar y representar visualmente los resultados obtenidos en este trabajo.

3.1.7. *FastAPI*

"*FastAPI* es un *framework* web moderno, rápido (de alto rendimiento), para construir *APIs* con *Python* basado en las anotaciones de tipos estándar de *Python*-[9]. Cabe destacar su alto rendimiento y su facilidad de uso, junto con su optimización para aplicaciones asíncronas, lo que la convierte en una opción ideal para la implementación de la *API* en este trabajo.

3.1.8. *SQLAlchemy*

"*SQLAlchemy* es el kit de herramientas *SQL* de *Python* y el mapeador relacional de objetos que ofrece a los desarrolladores de aplicaciones la máxima potencia y flexibilidad de *SQL*. Esta proporciona un conjunto completo de patrones de persistencia conocidos a nivel empresarial, diseñados para un acceso a bases de datos eficiente y de alto rendimiento, adaptados a un lenguaje de dominio simple y *Pythonic*-[24].

3.1.9. *PyCharm*

"*PyCharm* es un entorno de desarrollo integrado (*IDE*) utilizado en programación informática, concretamente para el lenguaje de programación *Python*. Proporciona análisis de código, un depurador gráfico, un probador de unidades integrado, integración con sistemas de control de versiones, y soporta el desarrollo web, así como la ciencia de datos-[20].

3.1.10. *GitHub*

"*GitHub* es una plataforma basada en la nube donde puedes almacenar, compartir y trabajar junto con otros usuarios para escribir código-[4]. En este trabajo, nos permitirá tener un control de versiones, además de poder compartir el código con terceras personas para la supervisión durante la realización del trabajo.

3.2. Implementación del sistema

En esta sección se presenta una introducción al desarrollo práctico del sistema de detección de fraude en tiempo real. Se describe el objetivo principal que guía el diseño del sistema, se detalla la arquitectura general que sustenta su funcionamiento y se muestra el flujo del sistema que permite la detección eficiente de transacciones fraudulentas. Esta implementación constituye el núcleo del proyecto, integrando los conocimientos teóricos previamente expuestos en una solución práctica y escalable para entidades financieras.

3.2.1. Objetivo del sistema

El objetivo principal del sistema consiste en detectar transacciones financieras fraudulentas en tiempo real. Su funcionalidad central radica en analizar cada transacción inmediatamente después de su generación y determinar, en cuestión de milisegundos, si debe autorizarse o bloquearse. Para alcanzar este propósito, se propone como solución un *middleware* implementado mediante una *API REST*. Esta *API* se integrará en tiempo real con una base de datos destinada al almacenamiento de toda la información institucional y transaccional, permitiendo su consulta posterior por parte de la entidad financiera correspondiente.

3.2.2. Arquitectura general del sistema

El sistema de detección de fraude en tiempo real se diseña como un *middleware* para instituciones financieras, evaluando el riesgo asociado a cada transacción recibida y generando una predicción sobre su legitimidad. La arquitectura, representada en la Figura 10, se estructura en los siguientes componentes:

1. **Entidad financiera**: Corresponde a la institución bancaria o proveedora de servicios

que envía transacciones para su análisis. La comunicación se establece mediante solicitudes a la *API REST*.

2. ***API REST***: Constituye el componente central del sistema, actuando como interfaz entre la entidad financiera y el modelo de *Machine Learning*. Sus funciones incluyen:

    - Verificación de autenticidad mediante consulta al servicio de autenticación
    - Extracción de características requeridas por el modelo predictivo
    - Gestión de la comunicación bidireccional con el modelo y la base de datos
    - Retorno de resultados a la entidad solicitante

3. **Servicio de autenticación**: Valida el registro y autorización de la entidad financiera en el sistema. Las solicitudes de entidades no registradas son rechazadas antes del procesamiento.

4. **Modelo de *Machine Learning***: Componente en formato *PKL* que, mediante patrones aprendidos durante su entrenamiento, calcula la probabilidad de fraude asociada a cada transacción recibida.

5. **Base de datos**: Almacena de manera estructurada todas las transacciones procesadas junto con sus predicciones asociadas, facilitando su auditoría y análisis retrospectivo.

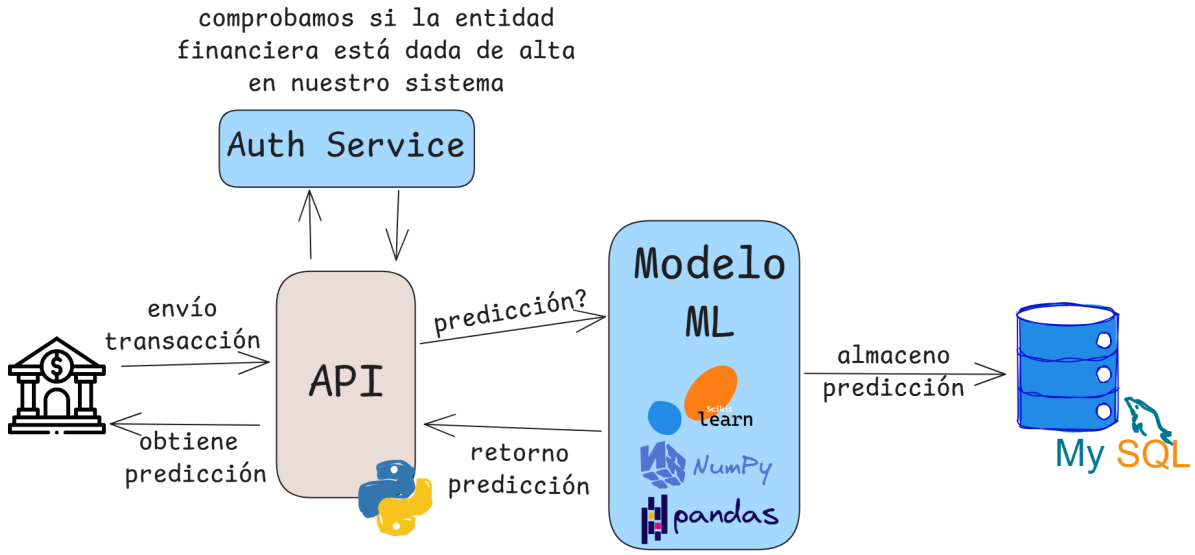

Figura 10: Arquitectura general del sistema

3.2.3. Flujo del sistema

Caso entidad financiera autorizada (Figura 11):

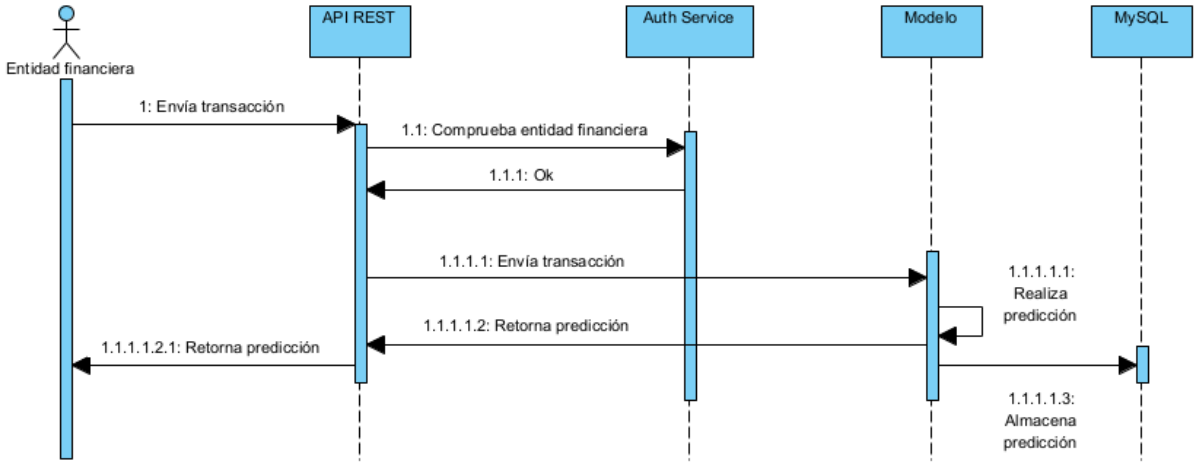

Figura 11: Diagrama de secuencia entidad autorizada

Caso entidad financiera no autorizada (Figura 12):

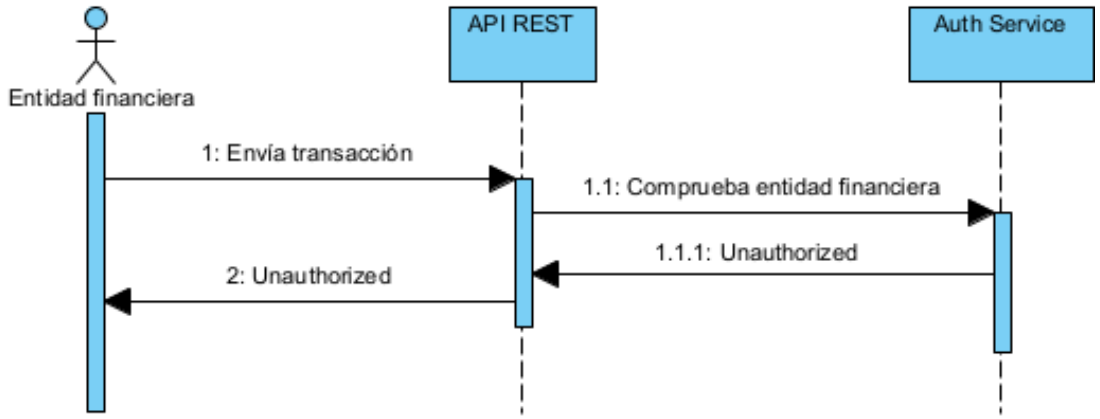

Figura 12: Diagrama de secuencia entidad no autorizada

4

Diseño experimental

En este capítulo se expone el proceso experimental desarrollado para la creación de los modelos de *Machine Learning* implementados. Se detallan meticulosamente las etapas preparatorias al entrenamiento de los modelos, incluyendo la descripción del conjunto de datos utilizado, las transformaciones aplicadas mediante técnicas de ingeniería de características y los métodos empleados para la selección óptima de parámetros, como la búsqueda en rejilla.

Asimismo, se presenta la estructura de las diferentes iteraciones realizadas, resaltando las mejoras incrementales incorporadas en cada fase. Adicionalmente, se especifican las plataformas tecnológicas utilizadas durante la experimentación y se destaca la participación de un experto en el ámbito, cuyas aportaciones resultaron cruciales para fundamentar las decisiones relativas a la selección y evaluación de los modelos.

Este enfoque metodológico riguroso ha permitido optimizar el rendimiento de los modelos y garantizar la validez de los resultados obtenidos en condiciones realistas de operación.

4.1. Pasos seguidos

Los pasos descritos a continuación representan el flujo de trabajo completo desde la obtención de los datos en bruto hasta la preparación final para el entrenamiento de los algoritmos de *Machine Learning*.

1. **Carga de datos**: Consiste en la carga inicial del conjunto de datos, compuesto por 1,3 millones de registros transaccionales.

2. **Análisis exploratorio de datos (*EDA*)**: Proceso orientado a comprender la estructura y características fundamentales de los datos. Incluye la identificación de tipos de variables, distribución de valores y detección de valores faltantes o inconsistentes.

3. **Manejo de datos incompletos y selección de características**: Etapa dedicada al tratamiento de valores faltantes mediante estrategias como el relleno de datos con la media o mediana. Paralelamente, se eliminan columnas irrelevantes para el modelo, como identificadores únicos, que podrían introducir ruido en las predicciones.

4. **Codificación de variables categóricas**: Transformación de variables no numéricas (ej. categoría de una venta) a formato numérico. Se implementó la técnica de codificación por frecuencia (*frequency encoding*), que sustituye cada categoría por su frecuencia relativa en el conjunto de datos.

5. **Balance de clases**: Dada la desproporción entre transacciones legítimas (99 %) y fraudulentas (1 %), se aplicó un balanceo mediante ajuste de pesos en los algoritmos. Este método asigna mayor penalización a los errores en la clase minoritaria, evitando así el sesgo predictivo hacia la clase mayoritaria.

6. **Ingeniería de características**: Aplicación de transformaciones avanzadas en las características para extraer patrones relevantes. Se incluyó la creación de características temporales, agregación de comportamientos históricos por cliente y normalización de magnitudes financieras.

7. **Separación de variables predictoras y objetivo**: División del conjunto de datos en dos componentes: X (variables independientes) e y (variable objetivo binaria que indica fraude).

8. **Partición entrenamiento-prueba y validación cruzada**: División estratificada del *dataset* en conjuntos de entrenamiento (80 %) y prueba (20 %), preservando la proporción original de clases. Adicionalmente, se implementó validación cruzada estratificada con 5 pliegues (*folds*), técnica que divide los datos en múltiples subconjuntos (*folds*) manteniendo la distribución de clases, permitiendo una evaluación robusta del rendimiento generalizado del modelo.

4.2. Conjunto de datos

Para el desarrollo de este proyecto, se ha utilizado un conjunto de datos con transacciones financieras ya clasificadas [17]. El *dataset* consta de 1.296.675 registros y 24 columnas,

conteniendo información detallada de cada transacción realizada con tarjetas de crédito.

4.2.1. Estructura del *dataset*

Las 24 columnas del conjunto de datos contienen diferentes tipos de información que se almacenan cada vez que se produce una transacción financiera (véase la Figura 13). Esta información se organiza en tres categorías principales que abarcan todos los aspectos relevantes para la detección de fraude: los detalles específicos de la transacción, la información del cliente que la realiza y los datos del establecimiento comercial donde se efectúa.

En esta primera categoría se comprenden todos los datos relacionados con la transacción.

- **Datos de la transacción**:

  - ***trans_num***: Identificador único de la transacción, almacenado como cadena de texto.
  - ***Unnamed 0***: Es el id autoincremental de la fila del conjunto de datos, representado como número entero.
  - ***unix_time***: Representación en formato *UNIX* de la fecha en la que se realizó la transacción.
  - ***category***: Categoría de la transacción, definida como variable categórica de texto.
  - ***amt***: Cantidad de la transacción, registrada como número decimal de punto flotante.
  - ***is_fraud***: Variable objetivo binaria, indica si la transacción es fraudulenta (1) o no (0), codificada como número entero.
  - ***trans_date_trans_time***: Fecha y hora de la transacción en formato "*YYYY-mm-dd HH:mm:ss*", almacenada como cadena de texto.

  Esta segunda categoría comprende toda la información personal y demográfica del titular de la tarjeta, elementos fundamentales para establecer patrones de comportamiento y detectar anomalías en el uso de la tarjeta.

- **Datos del cliente**:

- ***cc_num***: Número de la tarjeta de crédito asociado al cliente, representada como número entero.
- ***first***, ***last***: Nombre y apellido del titular, almacenados como cadenas de texto.
- ***gender***: Género del cliente, definido como variable categórica de texto.
- ***street***, ***city***, ***state***: Dirección postal del cliente, registrada como cadenas de texto.
- ***zip***: Código postal del cliente, almacenado como número entero.
- ***lat***, ***long***: Coordenadas geográficas del cliente, expresadas como números decimales de punto flotante.
- ***city_pop***: Número de habitantes de la ciudad donde reside el cliente, representada como número entero.
- ***job***: Profesión del cliente, definida como variable categórica de texto.
- ***dob***: Fecha de nacimiento del cliente, almacenada como cadena de texto.

Finalmente, la tercera categoría incluye los datos del establecimiento comercial donde se realiza la transacción, información crucial para analizar la coherencia geográfica y detectar posibles inconsistencias en los patrones de compra.

- **Datos del comercio**:

  - ***merchant***: Nombre del comercio, registrado como cadena de texto.
  - ***merch_lat***, ***merch_long***: Coordenadas geográficas del comercio, expresadas como números decimales de punto flotante.
  - ***merch_zipcode***: Código postal del comercio, almacenado como número decimal de punto flotante.

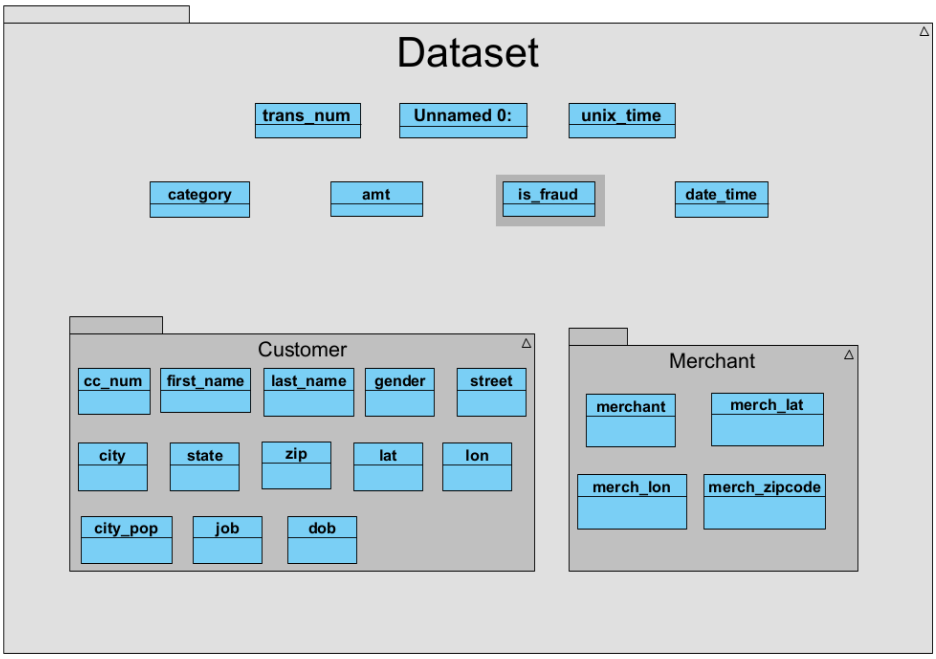

Figura 13: Estructura del conjunto de datos

4.3. Ingeniería de características (*feature engineering*)

Para mejorar la capacidad predictiva del modelo y obtener patrones relevantes en los datos, se llevó a cabo un proceso de ingeniería de características basado en la transformación y creación de nuevas variables a partir de las existentes. Este apartado detalla cada una de las transformaciones implementadas:

4.3.1. Cálculo de la edad del cliente

A partir de la columna *dob* (fecha de nacimiento), se generó la variable *age*, que representa la edad actual del cliente.

4.3.2. Distancias entre cliente y comercio

Se crearon dos variables de distancia que miden la separación geográfica entre el cliente y el comercio:

- ***distance_euclidean_between_cust_merch***: Almacena la distancia euclídea entre las coordenadas del cliente y las del comercio.

- ***distance_manhattan_between_cust_merch***: Almacena la distancia de *Manhattan* entre las mismas coordenadas.

Ambas medidas permiten detectar transacciones sospechosas que ocurren en lugares físicamente distantes a la ubicación habitual del cliente.

4.3.3. Tiempo desde la última transacción

La variable *seconds_since_last_transaction* mide el tiempo (en segundos) transcurrido desde la última transacción realizada con la misma tarjeta de crédito (*cc_num*). Este valor se calcula agrupando todas las transacciones por tarjeta y ordenando por fecha (*trans_date_trans_time*). Valores muy bajos podrían indicar actividad sospechosa, como los ataques de fraude automatizados.

4.3.4. Desviación estándar de las últimas transacciones

Representa la desviación estándar normalizada de la cantidad (*amt*) de las últimas 30 transacciones realizadas por un cliente.

Esta variable permite detectar variaciones atípicas en el comportamiento del gasto de un cliente. Un valor alto podría indicar un cambio repentino en el patrón de gasto, lo cual es una señal común en casos de fraude.

4.3.5. Velocidad desde la última transacción

Se calculó la velocidad en kilómetros por hora (*speed_kmh*) que implicaría una transacción en función del tiempo transcurrido y la distancia recorrida entre dos ubicaciones consecutivas (de un mismo cliente).

Valores anormalmente altos pueden indicar una imposibilidad física de que el mismo cliente haya realizado dos transacciones tan separadas geográficamente en un intervalo tan breve.

4.4. Búsqueda de parámetros

Con el objetivo de optimizar el rendimiento de los modelos, se ha implementado un proceso de optimización de hiperparámetros mediante validación cruzada anidada. Esta técnica permi-

te evaluar el desempeño del modelo de manera rigurosa, minimizando el riesgo de sobreajuste durante la fase de ajuste paramétrico.

4.4.1. Validación cruzada anidada

La validación cruzada anidada se estructura en dos niveles jerárquicos:

- **Nivel externo**: Se aplicó una validación cruzada estratificada de 5 particiones para evaluar el rendimiento general del modelo en cada configuración paramétrica.
- **Nivel interno**: Dentro de cada partición externa, se ejecutó una validación cruzada interna de 3 particiones combinada con búsqueda en rejilla (*Grid Search*), con el fin de seleccionar la combinación óptima de hiperparámetros.

Este enfoque garantiza la independencia estadística entre el proceso de ajuste paramétrico y la evaluación final, eliminando posibles sesgos derivados de la contaminación de datos durante la optimización.

4.4.2. Espacio de búsqueda

Para cada modelo, se definió un conjunto de posibles valores para sus hiperparámetros, que se fue explorando de forma sistemática. Este espacio de búsqueda fue adaptado en función de las características específicas de cada algoritmo, incluyendo parámetros como la tasa de aprendizaje, la profundidad máxima, el número de estimadores, entre otros.

4.4.3. Métrica de evaluación

El indicador seleccionado para guiar la búsqueda fue el área bajo la curva *ROC* (*AUC-ROC*). Esta métrica resulta particularmente adecuada para problemas con clases desbalanceadas, ya que cuantifica la capacidad del modelo para discriminar entre transacciones legítimas y fraudulentas independientemente de los umbrales de clasificación. Su elección garantiza una evaluación equilibrada del rendimiento en contextos donde la clase minoritaria (fraude) posee mayor relevancia operativa.

4.4.4. Proceso de entrenamiento

El proceso completo de búsqueda de hiperparámetros constó de los siguientes pasos:

1. Inicialización del modelo con parámetros por defecto.
2. Ejecución de la búsqueda en rejilla sobre el conjunto de hiperparámetros definido, utilizando la validación interna.
3. Evaluación del rendimiento medio de la mejor configuración encontrada sobre los pliegues de la validación externa.
4. Entrenamiento final del modelo con los mejores hiperparámetros sobre todo el conjunto de entrenamiento.
5. Evaluación sobre el conjunto de pruebas y visualización de los resultados mediante representaciones gráficas, como *boxplots* de la tasa de falsos negativos (*False Negative Rate*), para comparar el comportamiento de las distintas configuraciones.

4.4.5. Tiempo de entrenamiento

Durante este proceso, también se registró el tiempo total del entrenamiento de cada modelo, lo que permitió evaluar no solo su rendimiento predictivo, sino también su eficiencia computacional.

4.5. Plataforma de experimentación

Para el desarrollo, entrenamiento y evaluación de los modelos de detección de fraude, se ha utilizado un entorno de programación local, sin necesidad de acceso a servidores externos ni entornos en la nube.

4.5.1. Entorno de desarrollo

La implementación del proyecto se ha llevado a cabo utilizando ***Python*** como lenguaje de programación en el entorno de desarrollo integrado (*IDE*) ***PyCharm***, que ha facilitado la organización del código, la depuración y la ejecución de los experimentos.

4.5.2. *Hardware* utilizado

Todos los experimentos se han ejecutado en un equipo personal, concretamente en un ***HP Envy Laptop 14-eb1002ns***, con las siguientes especificaciones:

- **Procesador**: Intel Core i7-11300H a 3,4GHz (1 *socket*, 4 núcleos físicos, 8 hilos lógicos)
- **Memoria RAM**: 16GB DDR4 a 3200MT/s.
- **Sistema operativo**: *Windows* 11.

Este *hardware* ha resultado suficiente para realizar las tareas de carga de datos, procesamiento de datos, ingeniería de características y entrenamiento de modelos. En ciertos momentos, especialmente durante la búsqueda en rejilla de hiperparámetros, el tiempo de ejecución se ha visto aumentado debido a la carga computacional.

4.6. Iteraciones seguidas en modelos supervisados

En el desarrollo de los modelos supervisados, se ha seguido un total de cuatro iteraciones con el objetivo de encontrar qué combinación de características mejora el rendimiento del modelo. En los no supervisados, se ha escogido la mejor combinación de características de los supervisados.

Cada iteración se ha basado en los resultados obtenidos en la anterior, aplicando ajustes tanto en la selección como en la transformación de las características.

En todas las iteraciones se han comparado dos versiones para cada modelo:

- Una versión sin ingeniería de características (*NO FE*), usando únicamente las características originales del conjunto de datos.
- Una versión con ingeniería de características (*FE*), que incluye las características creadas a partir de otras, para enriquecer la información proporcionada al modelo. A continuación, se describen las decisiones tomadas en cada una de estas iteraciones:

4.6.1. Primera iteración

Esta primera aproximación al problema establece las bases metodológicas de la solución, explorando diferentes enfoques de codificación de variables categóricas y evaluando el impacto de la ingeniería de características en el rendimiento del modelo.

- Se parte de un total de 25 columnas.

- En cuanto a la codificación de las variables categóricas, se aplica tanto **codificación por frecuencia** como **codificación *one-hot***. La desventaja de esta última es que se generan tantas columnas como valores distintos tenga la columna a codificar, en este caso solo se ha aplicado a la columna *category* lo que ha generado 14 columnas adicionales.

- En la versión con ingeniería de características (***FE***), se incluyen las siguientes variables generadas:

  - **Desviación estándar** del importe de transacciones recientes.
  - **Segundos desde la última transacción** del mismo cliente.
  - **Distancia euclídea** entre el cliente y el comercio.

- En la versión sin ingeniería de características (***NO FE***), se usan únicamente las siguientes columnas:

  - *amt*, *merchant_frequency_encoded*, *city_frequency_encoded*, *city_pop*, *cc_num*, *job_frequency_encoded*, *state_frequency_encoded*, además de las 14 columnas generadas por la aplicación de la codificación *one-hot* a la columna *category*.

4.6.2. Segunda iteración

Basándose en los resultados obtenidos en la primera iteración, se implementan mejoras específicas orientadas a refinar las variables de ingeniería de características, así como de tratar de encontrar la mejor combinación de variables para la versión sin ingeniería de características.

- Se reemplaza la **desviación estándar** por la **desviación estándar normalizada**, con el objetivo de evitar que los importes absolutos afecten desproporcionadamente al cálculo (por ejemplo, se considera más relevante una duplicación relativa que una diferencia absoluta).

- Se incorpora una nueva variable en la versión de ingeniería de características (***FE***): la **velocidad entre transacciones consecutivas** (calculada a partir de la distancia geográfica y el tiempo transcurrido.

- Se introduce un análisis más robusto del modelo utilizando ***boxplots* de la tasa de falsos negativos *FNR (False Negative Rate)***, para seleccionar el modelo más adecuado según los criterios de los expertos.

- En cuanto a la evaluación de la importancia de las variables, se añade a los métodos existentes la importancia de las características (*feature importance*, la importancia por permutación *permutation importance*) y el algoritmo de **eliminación recursiva de características** (*recursive feature elimination*).

- La variable *category* pasa de estar codificada con la codificación *one-hot* a ser transformada mediante la codificación por frecuencia, reduciendo el número de columnas.

4.6.3. Tercera iteración

Al observar que el rendimiento del modelo no mejoraba con respecto a la primera iteración (e incluso disminuía ligeramente), se decide revertir algunos cambios.

- Se **elimina la variable de velocidad** entre transacciones del modelo con ingeniería de características *FE*.

- La variable *category* vuelve a ser representada mediante la codificación *one-hot*, reintroduciendo las 14 columnas adicionales.

- Se elimina una columna auxiliar generada durante el cálculo de la desviación estándar normalizada.

- Se reduce el parámetro *min_periods* de la ventana móvil de 5 a **4 transacciones**, con el objetivo de diminuir el impacto de ruido inicial cuando aún no hay suficientes transacciones en el historial de un cliente.

4.6.4. Cuarta iteración (definitiva)

Esta iteración final consolida las mejores prácticas identificadas en las fases anteriores, priorizando la eficiencia computacional sin comprometer el rendimiento predictivo del modelo.

- En ambas versiones se vuelve a codificar *category* con la codificación por frecuencia.

- En la versión con ingeniería de características (*FE*) se elimina la **distancia entre cliente y vendedor**, ya que no aportaba mejora significativa en el rendimiento.
- La versión sin ingeniería de características (*NO FE*) queda compuesta por 8 columnas.
- La versión con ingeniería de características (*FE*) queda compuesta por 12 columnas.
- Esta última configuración se adopta como la final por ofrecer un rendimiento predictivo comparable al anterior, además de ser más **eficiente computacionalmente**, al reducir considerablemente el número de atributos.

4.7. Iteraciones seguidas en los modelos no supervisados

En base a la investigación realizada en los modelos supervisados, se ha optado por escoger los mismos atributos que se obtienen del mejor modelo escogido. Por lo que solo se ha realizado una única iteración con el objetivo de ver cómo se comportan los modelos no supervisados.

4.7.1. Primera iteración

- La versión sin ingeniería de características queda compuesta por 8 columnas.
- La versión con ingeniería de características queda compuesta por 12 columnas.

4.8. Contactar con los expertos

Durante el desarrollo del proyecto, surgió una cuestión clave: ¿cómo determinar cuál es el mejor modelo a utilizar? Si bien las métricas tradicionales como la precisión o el área bajo la curva *ROC* pueden servir de guía, resulta evidente que en un contexto como el de la detección de fraude financiera se necesita una mirada más profesional y cercana al sector para establecer el criterio más adecuado. En concreto, la duda giraba en torno a qué tipo de error era más crítico evitar: los falsos positivos (transacciones legítimas clasificadas como fraude) o los falsos negativos (transacciones fraudulentas que no se detectan).
Con esta inquietud en mente, se intentó contactar con diversos expertos del sector financiero. Se enviaron múltiples mensajes tanto por correo electrónico como a través de la red social *LinkedIn*, sin obtener respuesta por parte de los destinatarios. Sin embargo, gracias a un contacto

directo, fue posible trasladar esta consulta a una persona que había ocupado un alto cargo en una entidad financiera.

La respuesta que se obtuvo, aunque transmitida de manera informal a través de un amigo, ofreció una perspectiva valiosa y realista del problema:

"*Obviamente él priorizaría la seguridad de las cuentas bancarias e intentaría detectar el máximo de fraudes posibles, porque con el dinero de la gente no se juega pero tampoco hay que pasarse. Hay que intentar que el banco no parezca una caja fuerte inaccesible. Me explicó que es una cuestión muy complicada y uno de los grandes retos de los bancos: maximizar la seguridad sin perjudicar demasiado la experiencia del usuario. Me dijo que, en general, la prioridad suele ser la seguridad, es decir, minimizar los falsos negativos, pero sin dejar de lado los falsos positivos.*"

Esta respuesta ayudó a entender que, aunque el objetivo principal debe ser minimizar los falsos negativos (*FNR*), no se puede ignorar el impacto de los falsos positivos (*FPR*), ya que afectarían negativamente a la experiencia del usuario. Un bloqueo injustificado de la tarjeta, especialmente en situaciones sensibles como durante un viaje, puede resultar extremadamente molesto y hasta derivar en la pérdida del cliente por un cambio de entidad bancaria.

Por ello, en las distintas iteraciones del proyecto se ha optado por representar los resultados de los modelos mediante *boxplots* centrados en los falsos negativos, pero también se ha aplicado un análisis cualitativo de los falsos positivos para lograr un equilibrio razonable.

5

Análisis experimental

Este capítulo está dedicado al análisis y discusión de los resultados obtenidos durante el desarrollo experimental de los modelos. Se abordará en primer lugar el desempeño de los modelos supervisados, estructurado en función de las distintas iteraciones que han sido seguidas con el objetivo de reflejar la evolución de cada cambio aplicado. A continuación, se analizarán los resultados de los modelos no supervisados, evaluando su comportamiento frente a la tarea de detección de anomalías. Posteriormente, se realizará una comparativa entre ambos enfoques para identificar sus fortalezas y debilidades en el contexto del problema. Finalmente, se expondrán las principales limitaciones encontradas durante el proceso.

5.1. Análisis de los resultados de los modelos supervisados

En este apartado presentaremos un análisis detallado de los resultados obtenidos con los modelos supervisados implementados en el proyecto. Para cada uno de los modelos empleados, se exponen las métricas de evaluación más relevantes.

5.1.1. Primera iteración

Random Forest con ingeniería de características

A partir de los resultados obtenidos, podemos observar que el modelo *Random Forest* con técnicas de ingeniería de características ha mostrado un rendimiento notable en la detección de transacciones fraudulentas. A continuación, se mostrarán unas métricas de rendimiento.

- **Precisión**: 0,980, indicando un alto nivel de acierto general. Aunque puede engañar ya que la proporción de la clase a buscar es bastante baja.

- ***AUC-ROC***: 0,965, demostrando una excelente capacidad de discriminación entre clases.

- **Parámetros óptimos**: Compensación por desbalance activada, profundidad máxima de 10 y con 200 estimadores.

El modelo presenta un comportamiento asimétrico entre las clases debido al desbalance natural del conjunto de datos:

- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.

- **Clase minoritaria (fraude)**: Precisión: 0,260, *Recall*: 0,950, *F1-score*: 0,400.

Es particularmente destacable el alto *recall* (0,950) en la detección de fraudes, lo que significa que el modelo identifica correctamente el 95 % de las transacciones fraudulentas. Sin embargo, la precisión relativamente baja (0,260) indica una proporción considerable de falsos positivos.

Los gráficos de importancia de características revelan hallazgos significativos sobre los patrones de fraude (Figura 14).

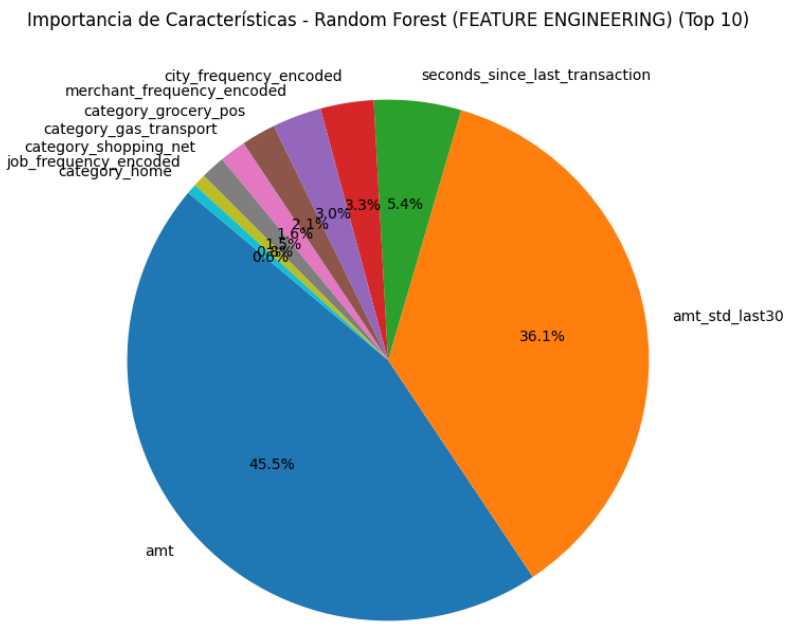

Figura 14: Importancia características *Random Forest* con ingeniería de características 1 iteración

En la Figura 14 se puede apreciar lo siguiente:

- La cantidad de dinero de la transacción *"amt"* es el factor más determinante, representando aproximadamente el 45,5 % de la importancia total.
- La desviación estándar de las cantidades en los últimos días *("amt_std_last30")* constituye el 36,1 %.
- El tiempo transcurrido desde la última transacción *("seconds_since_last_transaction")* aporta un 5,4 %.

***Random Forest* sin ingeniería de características**

El modelo sin técnicas de ingeniería de características también muestra un resultado significativo en la detección de fraude, pero no tan preciso como su versión de ingeniería de características.

- **Precisión**: 0,989, indicando un alto nivel de acierto.

- ***AUC-ROC***: 0,926, demostrando una buena capacidad discriminativa, aunque inferior a su modelo con ingeniería de características.

- **Parámetros óptimos**: Compensación por desbalance activada, profundidad máxima de 10 y con 300 estimadores.

- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,990, *F1-score*: 0,990.

- **Clase minoritaria (fraude)**: Precisión: 0,330, *Recall*: 0,860, *F1-score*: 0,470.

Destaca el buen *recall* (0,860) en la detección de fraudes, aunque con una precisión moderada (0,330), lo que indica una proporción considerable de falsos positivos, aunque menos que en su versión de ingeniería de características.

En la versión con ingeniería de características, 2 de las 3 características más importantes eran sacadas por ingeniería, por lo que en esta versión asume gran protagonismo la variable de la cantidad de dinero transferido *("amt")* que representa aproximadamente el 77 % de la importancia total (Figura 15).

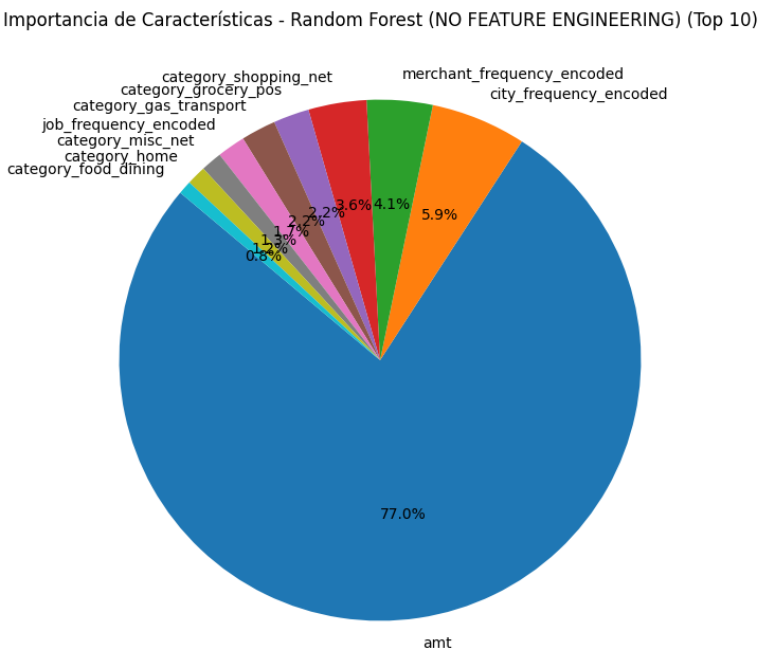

Figura 15: Importancia características *Random Forest* sin ingeniería de características 1 iteración

***LightGBM* con ingeniería de características**

El modelo *LightGBM* implementado con ingeniería de características presenta un resultado sobresaliente en la detección de fraude. A continuación, se detallará el análisis de su rendimiento.

- **Precisión**: 0,983 indicando una alta precisión global.
- ***AUC-ROC***: 0,979 demostrando una capacidad superior al modelo *Random Forest.*
- **Configuración óptima**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, profundidad máxima de 10 y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,250, *Recall*: 0,980, *F1-score*: 0,400.

Destaca especialmente el elevado *recall* (0,980) en la detección de transacciones fraudulentas, significando que el modelo capta prácticamente todos los casos de fraude, aunque con una precisión moderada (0,250) que sugiere un considerable número de falsos positivos.

En cuanto a la importancia de las variables, es muy parecido al modelo de *Random Forest*, a continuación se puede apreciar cuáles son las más importantes.

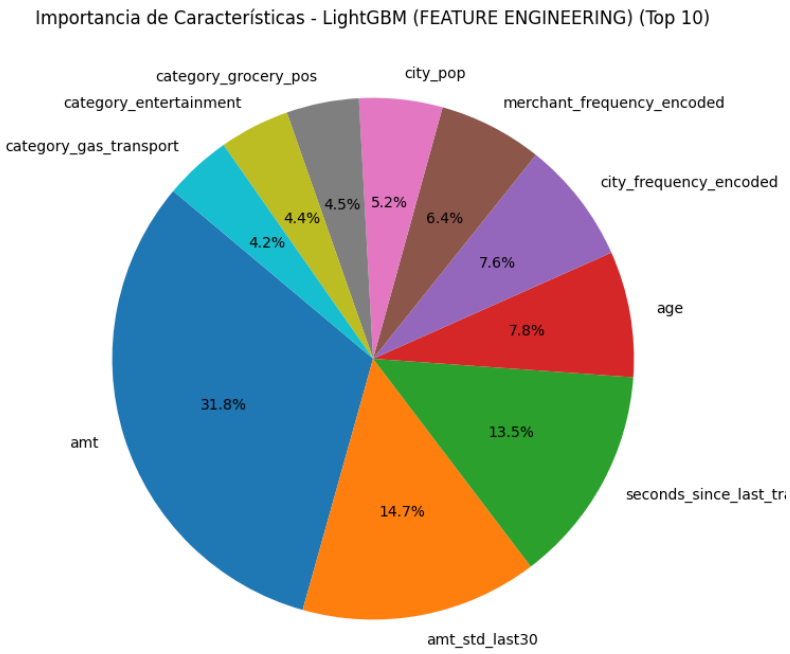

Figura 16: Importancia características *LightGBM* con ingeniería de características 1 iteración

En la Figura 16 se puede apreciar lo siguiente:

- La cantidad de dinero transferido *("amt")* sigue siendo la característica dominante, pero con una importancia relativa de 31,8 %, considerablemente menor que en modelos sin ingeniería de características, o en el modelo con ingeniería de características de *Random Forest*.

- La desviación estándar de la cantidad en los últimos 30 días *("amt_std_last30")* representa un 14,7 %.

- El tiempo desde la última transacción *("seconds_since_last_transaction")* contribuye al modelo con una importancia del 13,5 %.

- La edad del cliente *("age")* aporta un 7,8 %.

LightGBM sin ingeniería de características

Este modelo de *LightGBM* sin ingeniería de características presenta unos resultados similares al modelo de *Random Forest* con ingeniería de características. A continuación se detallará el análisis de su rendimiento.

- **Precisión**: 0,9630 indicando un alto nivel de acierto global.

- ***AUC-ROC***: 0,9630 demostrando una buena capacidad, pero no tan alta como en su versión con ingeniería de características.

- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, profundidad máxima de 10 y con 300 estimadores.

- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,960, *F1-score*: 0,980.

- **Clase minoritaria (fraude)**: Precisión: 0,130, *Recall*: 0,960, *F1-score*: 0,230.

Destaca el alto *recall* como en los anteriores modelos en sus distintas versiones (0,960), pero con una precisión muy baja (0,130) indicando una cantidad considerablemente superior de falsos positivos comparado con los otros modelos.

En cuanto a la importancia de las variables, es distinto al modelo de *Random Forest* en su versión sin ingeniería de características, ya que predominaba en más de un 75 % la cantidad de dinero transferido. En este caso, está más distribuida la importancia de las características.

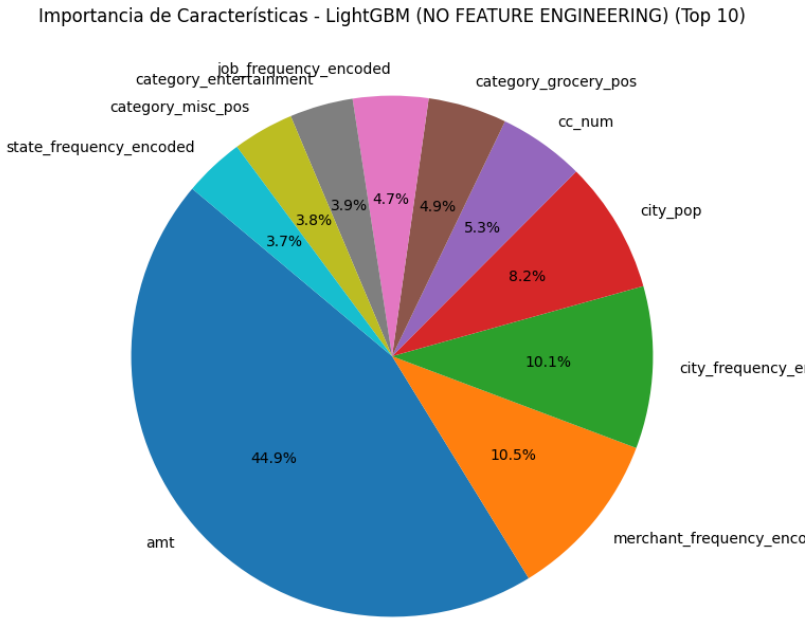

Figura 17: Importancia características *LightGBM* sin ingeniería de características 1 iteración

En la Figura 17 se puede apreciar lo siguiente:

- La cantidad de dinero transferido *("amt")* domina con un 44,9 % de la importancia total.
- La codificación por frecuencia del vendedor (10,5 %) y ciudad (10,1 %) siguen a la cantidad como los siguientes factores más relevantes.
- La población de la ciudad (8,2 %) y el número de tarjeta de crédito (5,3 %) completan el top 5.

Conclusión de esta primera iteración

Tras finalizar esta primera iteración, podemos concluir que el modelo *LightGBM* en su versión con ingeniería de características es el mejor modelo por varios factores (véase también el Cuadro 1).

- Tiene el área bajo la curva *ROC* más alta.

- Tiene el menor número de falsos negativos, con tan solo 37 fraudes que no ha detectado.
- A pesar de que no tenga el menor número de falsos positivos, se considera el mejor modelo por una ponderación de distintos valores

Modelo	**Precisión**	***AUC-ROC***	**Precisión fraude**	***Recall* fraude**	***F1-score* fraude**
Random Forest FE	0,980	0,965	0,260	0,950	0,400
Random Forest no FE	0,989	0,926	0,330	0,860	0,470
LightGBM FE	0,983	0,979	0,250	0,980	0,400
LightGBM no FE	0,963	0,963	0,130	0,960	0,230

Cuadro 1: Resultados modelos supervisados iteración 1.

Tras esto, buscamos seguir mejorando las métricas obtenidas, por lo que abordamos la segunda iteración.

5.1.2. Segunda iteración

***Random Forest* con ingeniería de características**

En este caso, ha empeorado levemente con respecto a la anterior iteración. Se ha podido observar un aumento notable en el número de falsos negativos, y un leve aumento en el número de falsos positivos. A pesar de ello, tampoco ha obtenido unos resultados deplorables.

- **Precisión**: 0,983 mostrando un alto nivel de acierto global.
- ***AUC-ROC***: 0,951, es un buen resultado pero inferior que en la primera iteración.
- **Parámetros óptimos**: Compensación por desbalance activada, profundida máxima de 10 y con 200 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,240, *Recall*: 0,920, *F1-score*: 0,390.

Como se mencionó antes, se ha obtenido una precisión algo más baja que en la primera iteración (0,240, comparado con los 0,260 obtenidos anteriormente).

Con respecto a la importancia de las características, obtenemos resultados similares aumentando la importancia de la cantidad de dinero de la transacción *("amt")* pasando de una importancia de 45,5 % a un 56,5 %. Disminuye la importancia de la desviación estándar normalizada pasa a tener una importancia del 18 % y el tiempo entre transacciones a un 6,8 %. Recalcar que la variable velocidad entre transacciones, añadida en esta iteración, no aparece ni en el top 10 de características más importantes (véase la Figura 18).

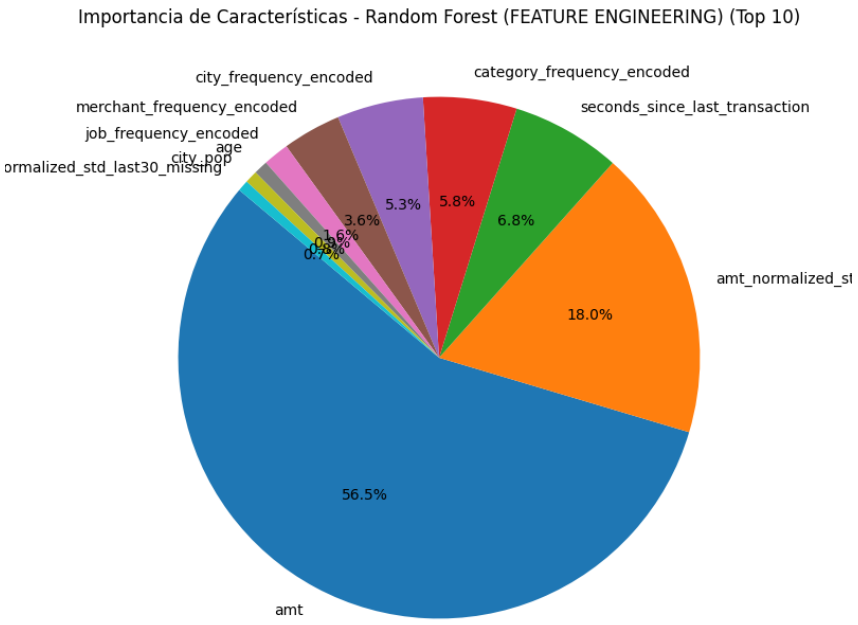

Figura 18: Importancia características *Random Forest* con ingeniería de características 2 iteración

Random Forest sin ingeniería de características

En este caso, se ha notado una breve mejora muy poco significativa en el modelo. Se han disminuido levemente los falsos negativos, y se han aumentado levemente los falsos positivos.

- **Precisión**: 0,988 indicando un alto nivel de acierto global.
- ***AUC-ROC***: 0,931, ha aumentado con respecto a la anterior iteración.

- **Parámetros óptimos**: Compensación por desbalance activada, profundidad máxima de 10 y con 100 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,990, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,310, *Recall*: 0,870, *F1-score*: 0,460.

Se sigue apreciando una precisión moderada, lo que indica una proporción considerable de falsos positivos algo superior a la obtenida en la iteración anterior.

En cuanto a la importancia de las variables, pasa lo mismo que en la anterior iteración y es que 2 de las 3 características más importantes en la versión de ingeniería de características eran obtenidas, por lo que en este caso gana gran protagonismo la cantidad de dinero transferido, como se puede apreciar en la Figura 19.

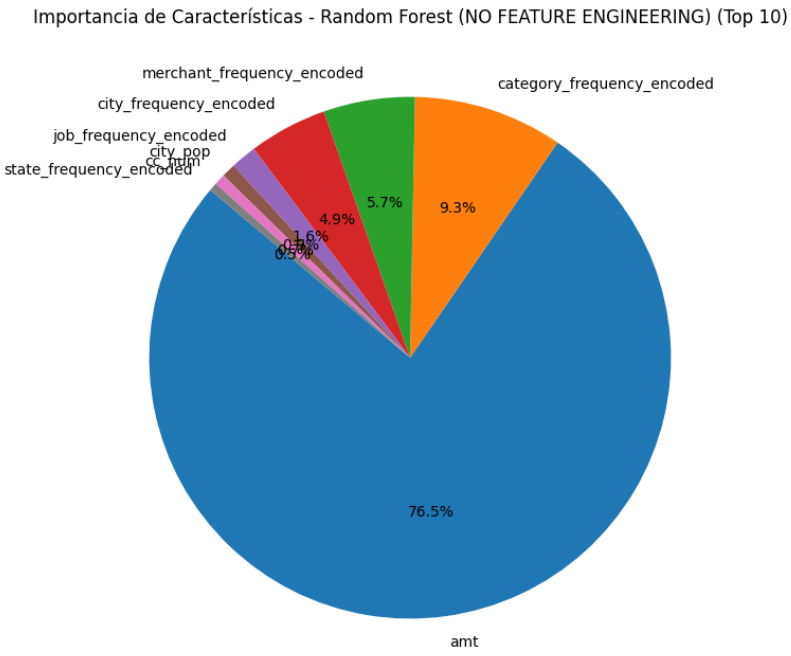

Figura 19: Importancia características *Random Forest* sin ingeniería de características 2 iteración

LightGBM con ingeniería de características

En este caso, se ha apreciado una muy leve caída en el rendimiento del modelo con respecto a su anterior iteración. Se ha podido observar un aumento leve tanto en los falsos positivos como en los falsos negativos. A continuación, se pueden apreciar las métricas.

- **Precisión**: 0,980 indicando una alta precisión global.
- ***AUC-ROC***: 0,974 demostrando un rendimiento espléndido, pero algo inferior a su anterior iteración.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión 0,230, *Recall*: 0,970, *F1-score*: 0,370.

Como hemos comentado anteriormente, se aprecia una disminución de la precisión y de la puntuación *F1* en la clase minoritaria debido al aumento en los falsos positivos y negativos.

En cuanto a la importancia de las variables, es similar a su anterior iteración, pero se ha metido en el top una nueva característica obtenida en esta nueva iteración. En la Figura 20 se puede apreciar lo siguiente:

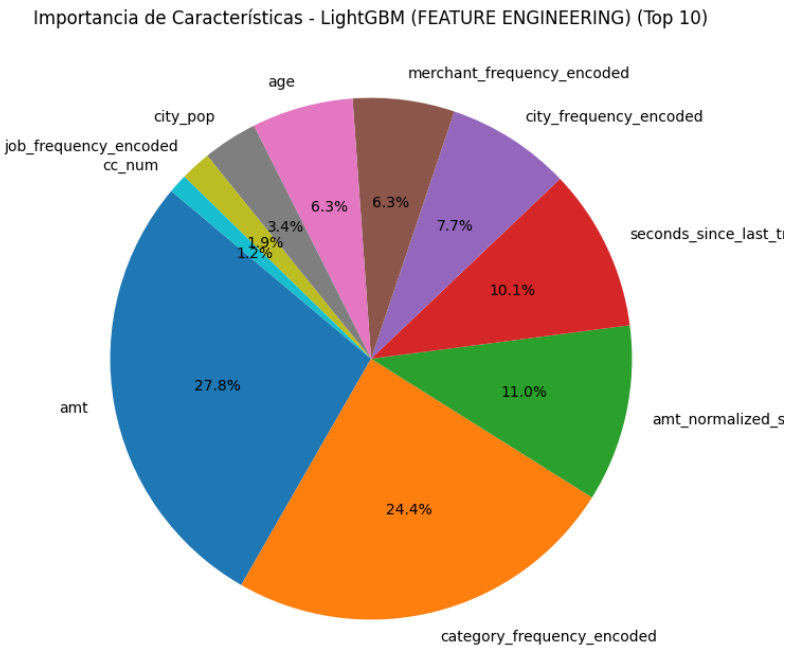

Figura 20: Importancia características *LightGBM* con ingeniería de características 2 iteración

- La cantidad de dinero en la transacción *("amt")* sigue liderando como en todas las versiones anteriores de los modelos representando un 27,8 % de la importancia total.

- La codificación de la categoría por frecuencia (cambio añadido en esta iteración) aparece como top 2 característica más importante con un 24,4 %.

- La desviación estándar normalizada aporta un 11, %.

- El tiempo desde la última transacción constituye en un 10,1 %.

***LightGBM* sin ingeniería de características**

Este modelo ofrece un rendimiento levemente superior a su anterior iteración, aunque es capaz de reducir un poco los falsos negativos, ha aumentado levemente los falsos positivos. A continuación, se pueden apreciar las métricas.

- **Precisión**: 0,963 indicando un alto nivel de acierto global.

- ***AUC-ROC***: 0,967 superando los 0,963 de su anterior iteración.

- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, sin límite de profundidad y con 300 estimadores.

- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,960, *F1-score*: 0,980.

- **Clase minoritaria (fraude)**: Precisión: 0,130, *Recall*: 0,970, *F1-score*: 0,230.

Resultados similares a su anterior iteración. En cuanto a la importancia de las variables, como en la versión con ingeniería de características, se incorpora una nueva variable obtenida en esta iteración que es la codificación de la categoría por frecuencia, contribuyendo con un 30,2 %. Sigue dominando la cantidad de dinero transferido representando un 35,7 % (Figura 21).

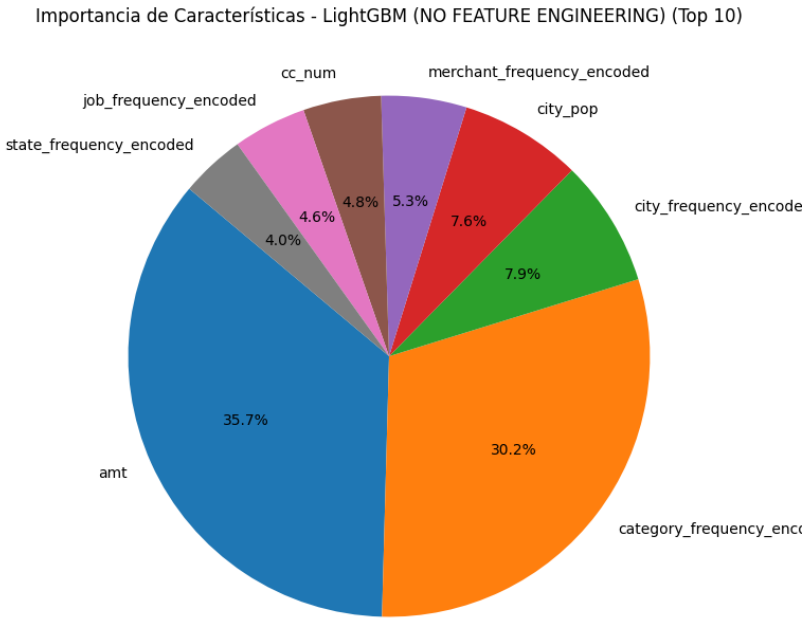

Figura 21: Importancia características *LightGBM* sin ingeniería de características 2 iteración

Conclusión de esta segunda iteración

Tras finalizar esta segunda iteración, podemos concluir lo mismo que en la anterior iteración, que el modelo *LightGBM* en su versión con ingeniería de características es el mejor modelo. A pesar de ello, cabe recalcar que ha obtenido un rendimiento levemente inferior al obtenido en la primera iteración. Ha sufrido un aumento de falsos positivos y negativos, como puede apreciarse en el Cuadro 2.

Modelo	**Precisión**	***AUC-ROC***	**Precisión fraude**	***Recall* fraude**	***F1-score* fraude**
Random Forest FE	0,983	0,951	0,240	0,920	0,390
Random Forest no FE	0,988	0,931	0,310	0,870	0,460
LightGBM FE	0,980	0,974	0,230	0,970	0,370
LightGBM no FE	0,963	0,967	0,130	0,970	0,230

Cuadro 2: Resultados modelos supervisados iteración 2.

5.1.3. Tercera iteración

Random Forest con ingeniería de características

En este caso, he mejorado levemente con respecto a la anterior iteración. Se ha podido observar una disminución leve en el número de falsos positivos y negativos. Se muestran las métricas a continuación.

- **Precisión**: 0,984 mostrando un alto nivel de acierto global.
- ***AUC-ROC***: 0,954 es un buen resultado, algo superior en comparación a los 0,9511 obtenidos en la anterior iteración.
- **Parámetros óptimos**: Compensación por desbalance activada, profundidad máxima de 10 y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,260, *Recall*: 0,920, *F1-score*: 0,410.

Como se mencionó antes, se puede apreciar un leve aumento tanto en la precisión como en la puntuación *F1* de la clase minoritaria debido a la reducción leve de los falsos positivos y negativos.

Con respecto a la importancia de las características, obtenemos resultados similares en comparación con la iteración anterior. Sigue predominando con un 57,6 % de importancia la cantidad de dinero transferido. Mantiene en un 18,8 % la desviación estándar normalizada y el tiempo entre transacciones se mantiene también en un 6,8 %. Resultado casi idéntico a la anterior iteración (Figura 22).

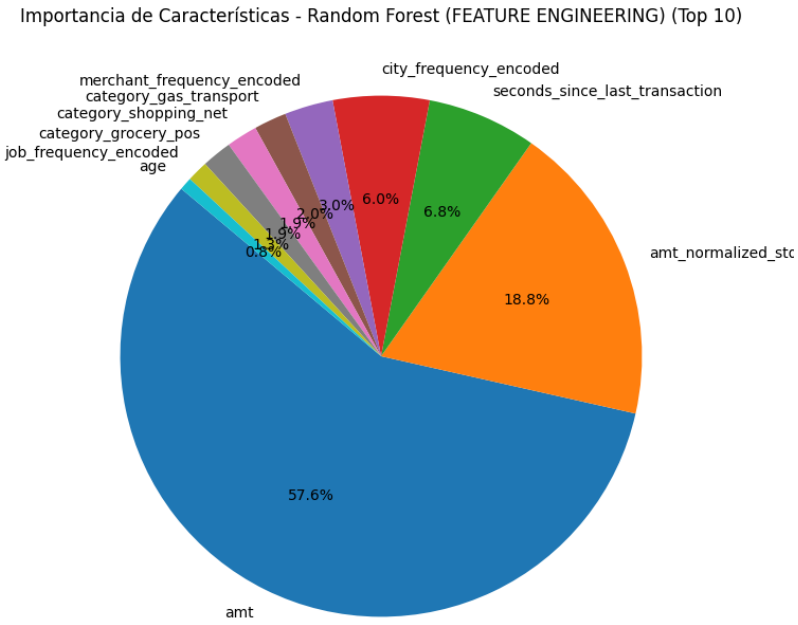

Figura 22: Importancia características *Random Forest* con ingeniería de características 3 iteración

***Random Forest* sin ingeniería de características**

En este caso, se ha observado un menor rendimiento con respecto a la anterior iteración. A pesar de que ha disminuido levemente el número de falsos positivos, se ha apreciado un aumento de los falsos negativos.

- **Precisión**:0,989 indicando un alto nivel de acierto global.
- ***AUC-ROC***: 0,926, ha disminuido con respecto a la anterior iteración.
- **Parámetros óptimos**: Compensación por desbalance activada, profundidad máxima de 10 y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,990, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,330, *Recall*: 0,860, *F1-score*: 0,470.

En cuanto a la importancia de las características, le pasa como en su anterior iteración, el 77 % de la importancia total se le atribuye a la cantidad de dinero transferido (Figura 23).

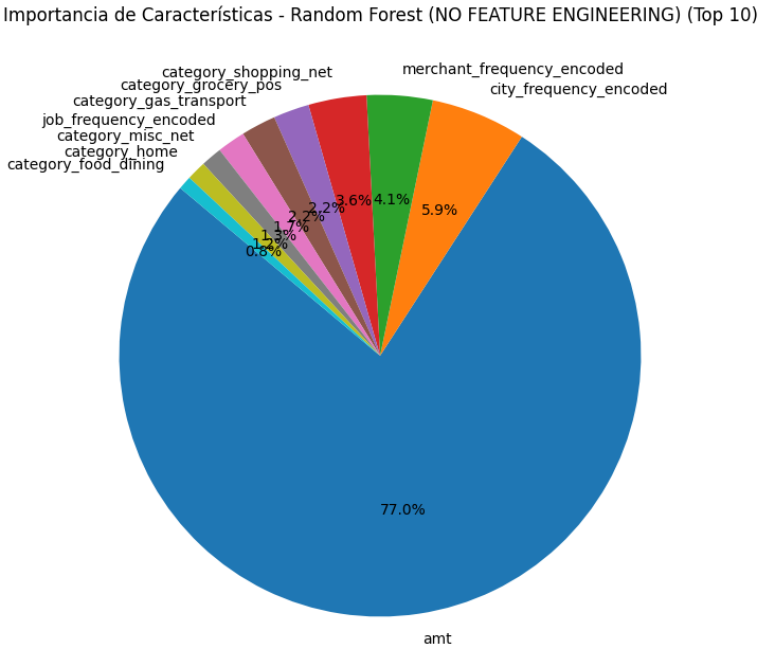

Figura 23: Importancia características *Random Forest* sin ingeniería de características 3 iteración

***LightGBM* con ingeniería de características**

En este caso, los resultados son prácticamente idénticos con respecto a la anterior iteración. El aumento ha sido insignificativo. A continuación, se pueden apreciar las métricas.

- **Precisión**: 0,9804 indicando una alta precisión global.
- ***AUC-ROC***: 0,975, se aprecia un aumento insignificativo con respecto a los 0,974 de la anterior iteración.
- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, sin profundidad máxima y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.

- **Clase minoritaria (fraude)**: Precisión: 0,220, *Recall*: 0,970, *F1-score*: 0,360.

Mostrando resultados casi idénticos a la anterior iteración.

Con respecto a la importancia de las características, se puede apreciar que como en todos los modelos en las anteriores iteraciones sigue dominando la cantidad de dinero transferido 30,7 %, le sigue la desviación estándar normalizada obteniendo un 14,6 % de importancia en el modelo y el tiempo transcurrido entre transferencias con un 13,6 %, como se puede apreciar en la Figura 24.

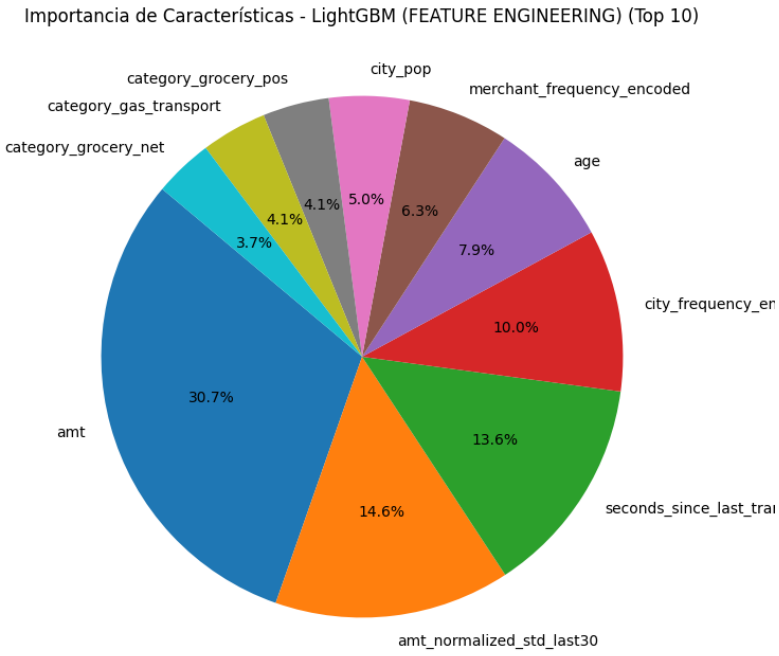

Figura 24: Importancia características *LightGBM* con ingeniería de características 3 iteración

***LightGBM* sin ingeniería de características**

En este caso se ha podido observar una disminución leve del rendimiento, a pesar de su bajada de falsos positivos, ha sufrido un aumento de los falsos negativos.

- **Precisión**: 0,963 indicando un alto nivel de acierto global.
- ***AUC-ROC***: 0,963, volviendo a obtener la misma métrica que en la primera iteración y algo menor a la segunda iteración.

- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, profundidad máxima de 10 y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión 1,000, *Recall*: 0,960, *F1-score*: 0,980.
- **Clase minoritaria (fraude)**: Precisión: 0,130, *Recall*: 0,960, *F1-score*: 0,230.

En cuanto a la importancia de las características, en esta iteración se ha cambiado el modo de codificación de la categoría, por lo que gana más responsabilidad aún la cantidad de dinero transferido con una importancia total de un 44,9 % con respecto al 35,7 % que tenía en la anterior iteración. Les sigue la codificación por frecuencia tanto del vendedor con un 10,5 % como de la ciudad con un 10,1 % (Figura 25).

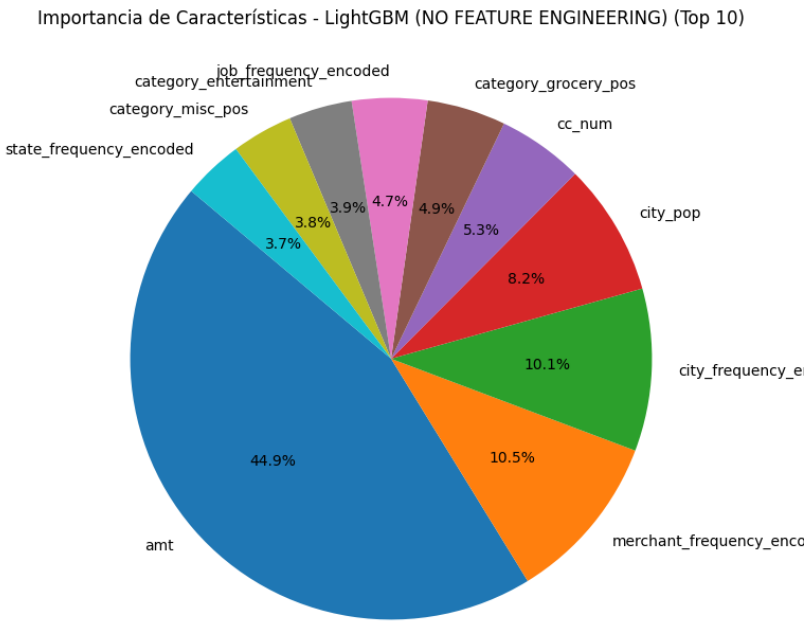

Figura 25: Importancia características *LightGBM* sin ingeniería de características 3 iteración

Conclusión de esta tercera iteración

Tras finalizar esta tercera iteración, podemos concluir lo mismo que en las dos anteriores iteraciones: que el modelo *LightGBM* en su versión con ingeniería de características es el que

demuestra un mejor desempeño en la predicción de fraudes, como podemos observar en el Cuadro 3.

Modelo	**Precisión**	***AUC-ROC***	**Precisión fraude**	***Recall* fraude**	***F1-score* fraude**
Random Forest FE	0,984	0,954	0,260	0,920	0,410
Random Forest no FE	0,989	0,926	0,330	0,860	0,470
LightGBM FE	0,980	0,975	0,220	0,970	0,360
LightGBM no FE	0,963	0,963	0,130	0,960	0,230

Cuadro 3: Resultados modelos supervisados iteración 3.

5.1.4. Cuarta iteración

***Random Forest* con ingeniería de características**

En este caso, ha mejorado muy levemente con respecto a la anterior iteración. A pesar del aumento de falsos positivos, se ha podido observar una disminución de los falsos negativos.

- **Precisión**: 0,983 mostrando un alto nivel de acierto global.
- ***AUC-ROC***: 0,956, sigue incrementando poco a poco con respecto a las anteriores iteraciones.
- **Parámetros óptimos**: Ponderación de clases balanceada, profundidad máxima de 10 y con 100 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,250, *Recall*: 0,930, *F1-score*: 0,390.

Valores muy similares en comparación con la iteración anterior.

Con respecto a la importancia de las características, obtenemos resultados similares en comparación con la iteración anterior. Sigue predominando con un 58,8 % de importancia total la cantidad de dinero transferido. Disminuye levemente en comparación con la iteración anterior de un 18,8 % a un 16,8 % la desviación estándar normalizada y el tiempo entre transacciones también disminuye a un 5,7 % de la importancia total (Figura 26).

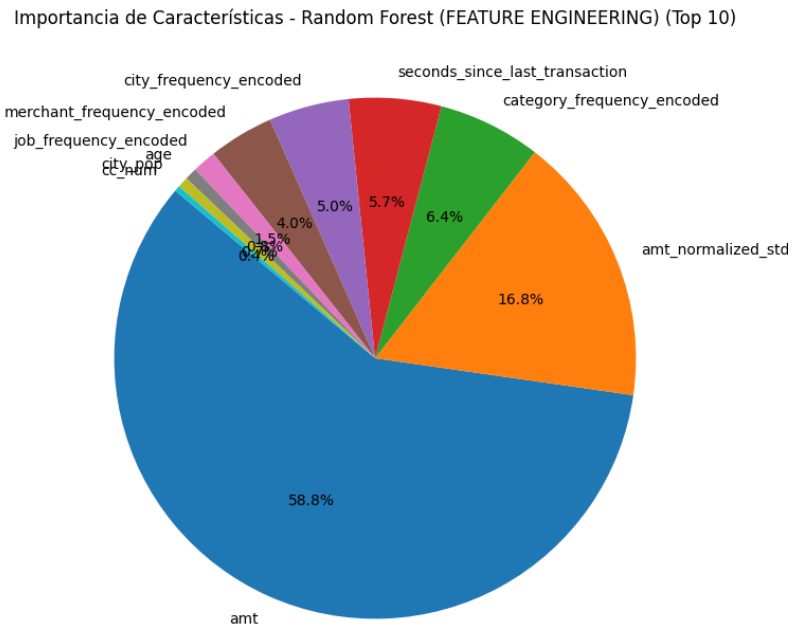

Figura 26: Importancia características *Random Forest* con ingeniería de características 4 iteración

***Random forest* sin ingeniería de características**

En este caso ha ocurrido como en su versión con ingeniería de características, se ha podido observar una mejora leve en el rendimiento del modelo. A pesar del aumento de falsos positivos, se ha podido observar una disminución en el número de falsos negativos.

- **Precisión**: 0,988 indicando un alto nivel de acierto global.
- ***AUC-ROC***: 0,931, se puede apreciar un leve aumento en comparación a su anterior iteración (0,926).
- **Parámetros óptimos**: Ponderación de clases balanceadas, profundidad máxima de 10 y con 100 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,990, *F1-score*: 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,310, *Recall*: 0,870, *F1-score*: 0,460.

En cuanto a la importancia de las características, sigue pasando como en su anterior iteración. Sigue dominando la cantidad de dinero transferido con una importancia total de 76,5 %. El cambio realizado con respecto a la anterior iteración, la codificación por frecuencia de la categoría se posiciona en el top 2 con una importancia total de un 9,3 %. Tras este, le sigue la codificación por frecuencia tanto de la ciudad como del vendedor con un 5,7 % y 4,9 % respectivamente (Figura 27).

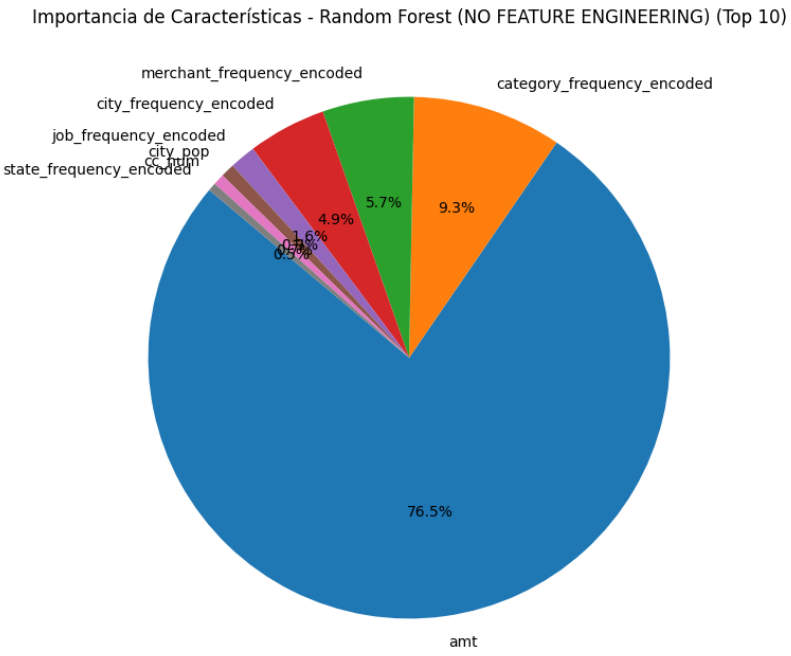

Figura 27: Importancia características *Random Forest* sin ingeniería de características 4 iteración

LightGBM con ingeniería de características

En este caso, los resultados son prácticamente idénticos con respecto a la anterior iteración. La disminución ha sido insignificante.

- **Precisión**: 0,9806 indicando una alta precisión global.
- ***AUC-ROC***: 0,974, se aprecia una disminución insignificativa con respecto a los 0,975 de la anterior iteración e idéntica a la segunda iteración con 0,974.

- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, profundidad máxima de 10 y con 300 estimadores.
- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,980, *F1-score* 0,990.
- **Clase minoritaria (fraude)**: Precisión: 0,230, *Recall*: 0,970, *F1-score*: 0,370.

Mostrando resultados casi idénticos en comparación con la anterior iteración.

Con respecto a la importancia de las características, se alza en el top 2 el cambio con respecto a la anterior iteración que es la codificación por frecuencia de la categoría, obteniendo una importancia total de un 23,5 %. Sigue dominando con un 27,2 % de importancia total la cantidad de dinero transferido y en el top 3 y 4 obtenemos la desviación estándar normalizada y el tiempo transcurrido desde la última transacción con un 11,4 % y un 10,3 % respectivamente (Figura 28).

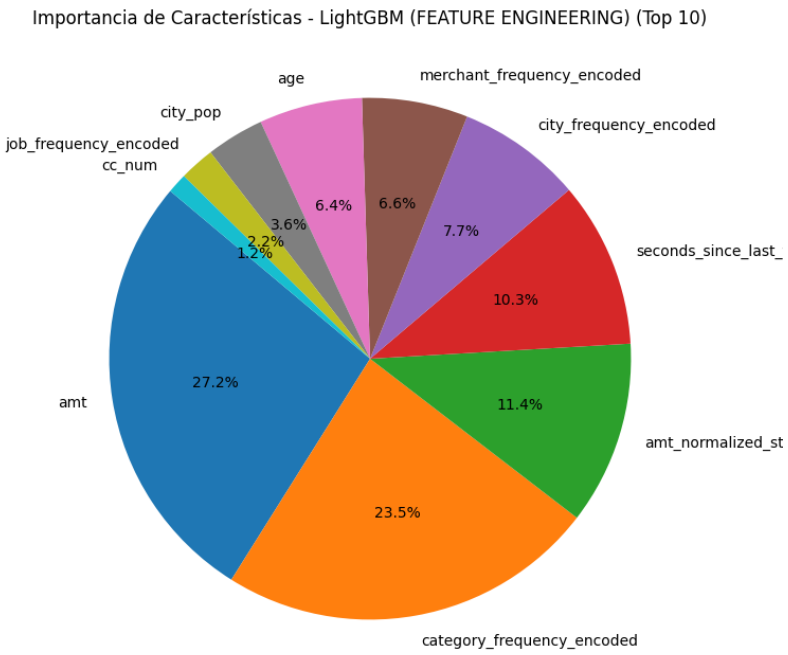

Figura 28: Importancia características *LightGBM* con ingeniería de características 4 iteración

LightGBM sin ingeniería de características

En este caso se ha podido observar un aumento leve en el rendimiento del modelo. A pesar del leve aumento de los falsos positivos, se puede apreciar una reducción del número de falsos negativos.

- **Precisión**: 0,963 indicando un alto nivel de acierto global.

- ***AUC-ROC***: 0,967, superior a los 0,963 de la anterior iteración.

- **Parámetros óptimos**: Compensación por desbalance activada, tasa de aprendizaje de 0,01, sin límite de profundidad y ccon 300 estimadores.

- **Clase mayoritaria (no fraude)**: Precisión: 1,000, *Recall*: 0,960, *F1-score*: 0,980.

- **Clase minoritaria (fraude)**: Precisión: 0,130, *Recall*: 0,970, *F1-score*: 0,230.

En cuanto a la importancia de las características, en esta iteración le ha pasado como a su versión con ingeniería de características. En el top 2 se ha alzado la codificación de la categoría por frecuencia (cambio realizado en esta iteración) con un 30,2 % de la importancia total. Sigue dominando la cantidad de dinero transferido con un 35,7 % y la codificación por frecuencia de la ciudad y la población de la ciudad aportan un 7,9 % y un 7,6 % de la importancia total respectivamente (Figura 29).

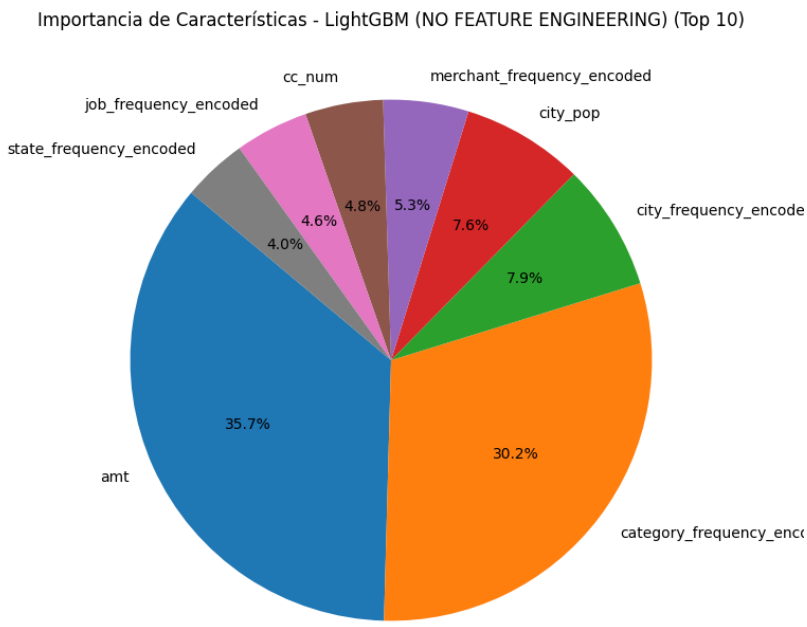

Figura 29: Importancia características *LightGBM* sin ingeniería de características 4 iteración

Conclusión de esta cuarta iteración

Tras finalizar esta cuarta iteración podemos concluir lo mismo que en las anteriores iteraciones, que el modelo *LightGBM* en su versión con ingeniería de características es el que demuestra un mejor desempeño en la predicción de fraudes (véase el Cuadro 4). A pesar de que en su versión sin ingeniería de características sea el que obtenga un menor número de falsos negativos (fraudes que no son detectados por el modelo) con un total de 44 en comparación con los 49 que obtiene la versión con ingeniería de características, lo elegimos porque reduce en la mitad el número de falsos positivos 4980 en comparación con los 9509 de la versión sin ingeniería de características. Por lo que, basándonos en el criterio del experto contactado, optamos por la versión con ingeniería de características, ya que obtiene un buen desempeño, y a pesar de que no ha predicho bien 5 fraudes, está mejorando en el doble la experiencia del usuario, ya que reduce en la mitad los falsos positivos.

Modelo	Precisión	*AUC-ROC*	Precisión fraude	*Recall* fraude	*F1-score* fraude
Random Forest FE	0,983	0,956	0,250	0,930	0,390
Random Forest no FE	0,988	0,931	0,310	0,870	0,460
LightGBM FE	0,981	0,974	0,230	0,970	0,370
LightGBM no FE	0,963	0,967	0,130	0,970	0,230

Cuadro 4: Resultados modelos supervisados iteración 4.

5.2. Análisis de los resultados de los modelos no supervisados

En este apartado presentaremos un análisis detallado de los resultados obtenidos con los modelos no supervisados implementados en el proyecto. Para cada uno de los modelos empleados, se exponen las métricas de evaluación más relevantes. Para las métricas, son diferentes en comparación con los modelos supervisados ya que no disponemos de la clase a predecir (fraude en este caso), por lo que de métricas nos basaremos en la distribución de puntajes de anomalías y la visualización de los resultados en la reducción *PCA*.

5.2.1. Primera iteración

Isolation Forest con ingeniería de características

En este caso, los resultados a simple vista no reflejan una buena detección de anomalías debido a que en la distribución de los puntajes de anomalía hay bastante frecuencia en los valores altos y poco en los bajos. Además, para corroborar, en la visualización con *PCA* de las predicciones, las anomalías están bastante mezcladas con las transacciones legítimas, por lo que no están concentradas en una región concreta (Figura 30 y Figura 31).

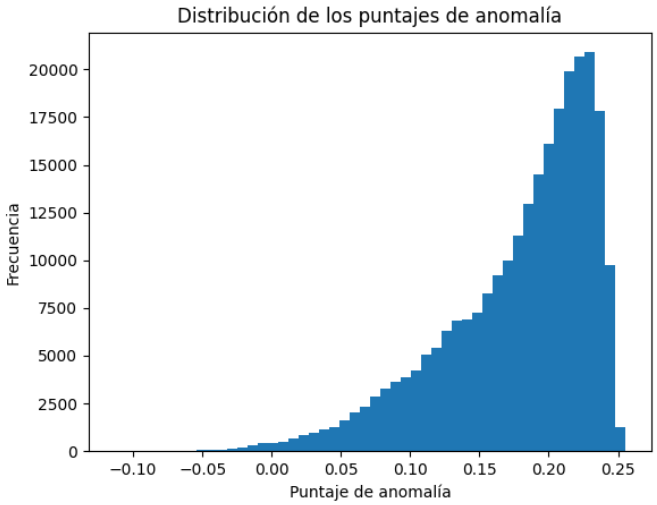

Figura 30: Distribución puntajes anomalía *Isolation Forest* con ingeniería de características

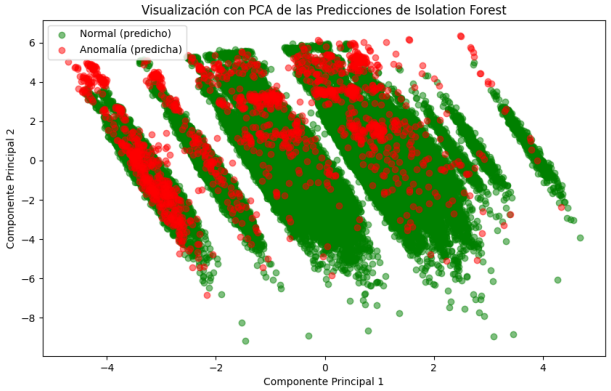

Figura 31: Resultados *PCA Isolation Forest* con ingeniería de características

Ante dichos resultados, y aprovechando que en el conjunto de datos tenemos la columna fraude (aunque estemos probando modelos no supervisados, quise comprobar el rendimiento real del modelo con las mismas métricas que en los modelos supervisados) y los resultados fueron nefastos.

Cabe recalcar la baja área bajo la curva *ROC* con un valor de 0,200. Recalcar también la alta cantidad de falsos negativos (fraudes no detectados) 1438 en comparación con los 49 que conseguimos con el modelo final de *Lightgbm* en su cuarta iteración con ingeniería de características.

***Isolation Forest* sin ingeniería de características**

En este caso, pasa igual que en su versión con ingeniería de características. En la distribución de puntajes de anomalías sigue habiendo una alta frecuencia en los valores más altos, cosa que el modelo considera que muchas de las transacciones tienen cierto grado de anomalía. En el caso de los resultados con *PCA* se siguen agrupando a lo largo de todas las regiones (Figura 32 y Figura 33).

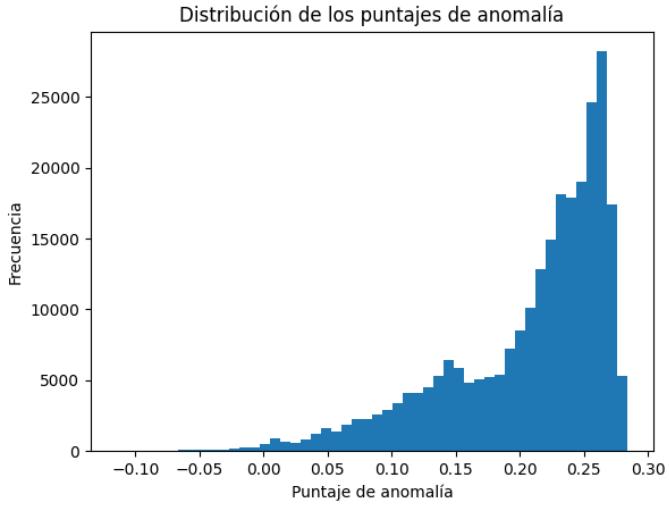

Figura 32: Distribución puntajes anomalía *Isolation Forest* sin ingeniería de características

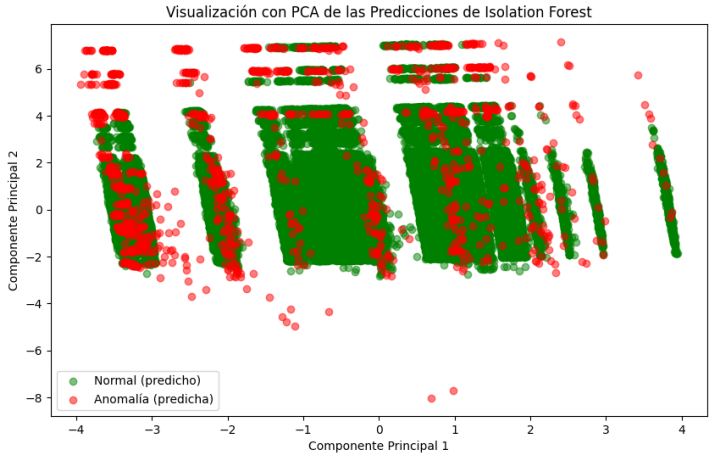

Figura 33: Resultados *PCA Isolation Forest* sin ingeniería características

Como hicimos en su versión con ingeniería de características, ante dichos resultados se optó por comparar con su columna a predecir para ver el rendimiento real del modelo. Como

se mencionó en su versión con ingeniería de características, cabe recalcar la baja área bajo la curva *ROC* con valor 0,171, aún menor que en su otra versión que obtuvo 0,200.

One Class SVM con ingeniería de características

En este caso, y ante los pésimos resultados obtenidos en el modelo *Isolation Forest* se ha optado por aumentar el número de anomalías que el modelo ha de predecir, a ver si así aumenta el número de aciertos. Tras esto, hemos obtenido una grata sorpresa y es que se sigue observando exactamente lo mismo. En la distribución de puntajes de anomalía sigue habiendo una alta frecuencia en los valores altos, a lo mismo que en los resultados de la reducción *PCA* siguen distribuyéndose las predichas como anomalía a lo largo de la región. Se puede apreciar en la Figura 34 y Figura 35.

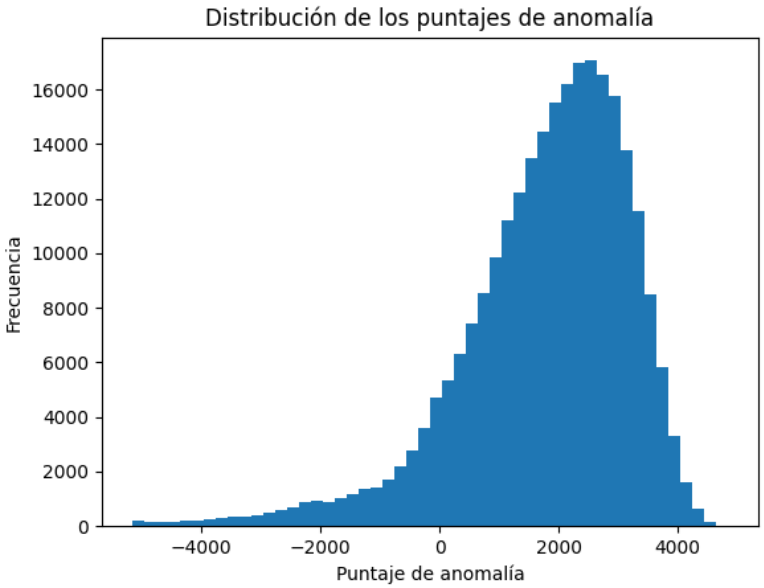

Figura 34: Distribución puntajes anomalía *One Class SVM* con ingeniería de características

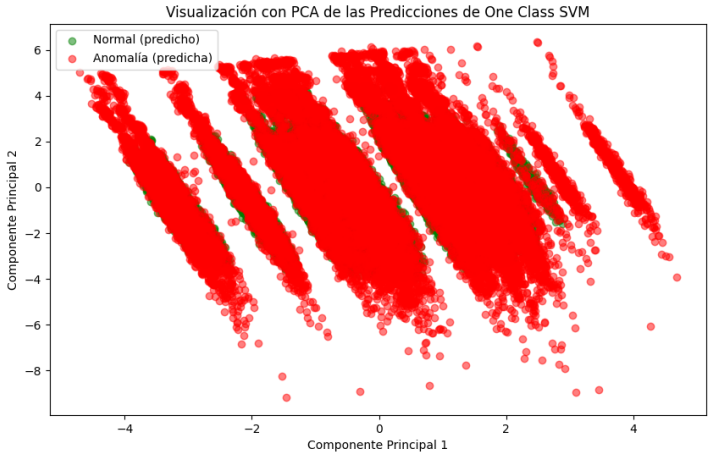

Figura 35: Resultados *PCA One Class SVM* con ingeniería de características

Como hicimos en ambas versiones del modelo *Isolation Forest*, ante dichos resultados y más, habiendo aumentado el número de anomalías a predecir, se optó por comparar con su columna a predecir para ver el rendimiento real del modelo. Se puede apreciar que del número total de fraudes a predecir que son 1501, solo ha sido capaz de predecir bien 878, fallando en un total de 25730 predicciones. Obteniendo así un *recall* en la clase minoritaria (fraude) de 0,580.

One Class SVM sin ingeniería de características

En este caso, se puede predecir cuáles van a ser los resultados viendo los modelos y sus versiones anteriores en este tipo de modelos (no supervisados). Se sigue apreciando una alta frecuencia en valores altos en la distribución de puntajes de anomalía, además de que en los resultados de la reducción *PCA* siguen distribuyéndose a lo largo de la región y no en regiones concretas. Se puede apreciar en la Figura 36 y Figura 37.

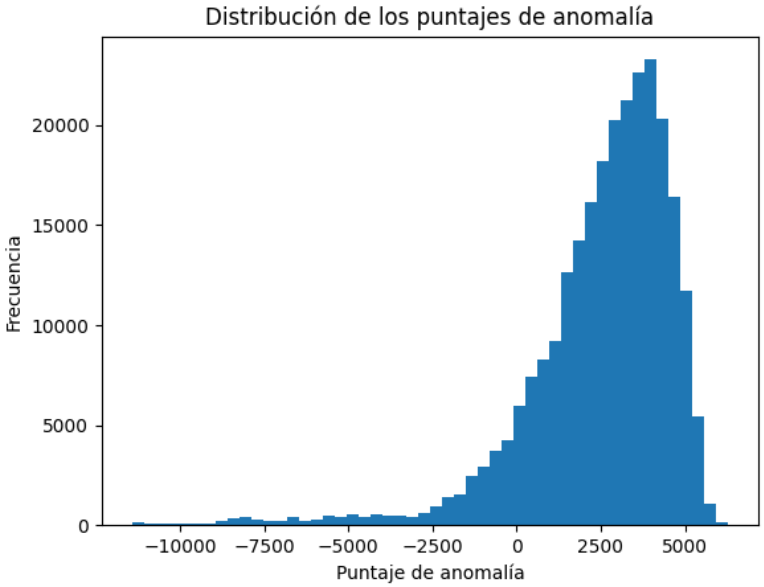

Figura 36: Distribución puntajes anomalía *One Class SVM* sin ingeniería de características

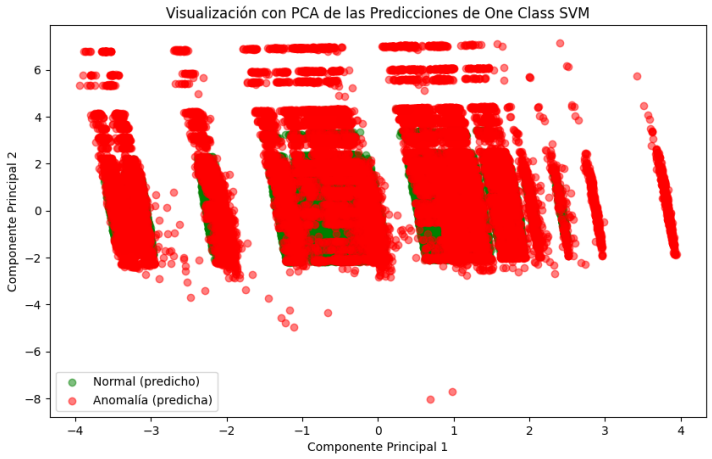

Figura 37: Resultados *PCA One Class SVM* sin ingeniería de características

Como se han hecho en los anteriores modelos, ante dichos resultados se optó por comparar con su columna a predecir para ver el rendimiento real del modelo. Se puede apreciar que en esta versión ha aumentado levemente el rendimiento, ha predicho bien 916 fraudes en comparación con los 878 en su versión con ingeniería de características, dejando así un *recall* en la clase minoritaria (fraude) de 0,610. Siguen siendo resultados muy bajos.

Conclusión de esta primera iteración

Los modelos no supervisados, en general, no han sido capaces de estar a la altura como los modelos supervisados. Han arrojado resultados pésimos para un área tan crítica en la que un mínimo error puede costar mucho dinero (véase el Cuadro 5). Esto no significa necesariamente que el modelo funcione mal, sino que la naturaleza de las anomalías financieras es compleja y multidimensional, y no bidimensional, como es en el caso de la reducción *PCA* que hemos aplicado.

Modelo	***AUC-ROC***	***Recall* fraude**
Isolation Forest FE	0,200	0,040
Isolation Forest no FE	0,171	0,090
One Class SVM FE	0,176	0,580
One Class SVM no FE	0,148	0,610

Cuadro 5: Resultados modelos no supervisados única iteración.

5.3. Conclusión final sobre los resultados

El proceso experimental desarrollado a través de cuatro iteraciones supervisadas y una iteración no supervisada revela patrones consistentes en el rendimiento de los algoritmos de detección de fraude. Los modelos no supervisados (*Isolation Forest* y *One Class SVM*) demostraron limitaciones significativas para este dominio específico, con valores como el área bajo la curva *ROC* que no superaron 0,200 y tasas de *recall* para fraudes extremadamente bajas, confirmando que la naturaleza compleja y multidimensional de las anomalías financieras requiere enfoques supervisados que puedan aprovechar el conocimiento previo sobre patrones fraudulentos.

Tras analizar exhaustivamente los resultados de todas las iteraciones, el modelo ***LightGBM*** con ingeniería de características en su cuarta iteración emerge como la mejor opción para integrar en la *API* de detección de fraudes, al combinar un *recall* excepcional (97 %), capaz de identificar prácticamente todos los casos de fraude, con un área bajo la curva *ROC* excepcional (0,974) que refleja su robustez discriminativa, además de una eficiencia computacional 8 veces superior a *Random Forest* (1701 segundos de entrenamiento vs. 14235 segundos). Además, a pesar de que en su primera iteración hubiese dado levemente un mejor rendimiento, se ha conseguido un rendimiento similar bajando de más de 20 variables a tan solo 12.

A diferencia de los modelos sin ingeniería de características, que dependían en exceso de la cantidad de dinero transferida (hasta 77 % de importancia), este modelo distribuye la capacidad predictiva entre variables temporales, contextuales y demográficas (27 % cantidad de dinero transferido, 24 % la categoría, 14 % desviación estándar de la cantidad de dinero transferido), capturando patrones complejos de fraude.

Por lo que la elección final recae en el modelo ***LightGBM*** con ingeniería de características, por su equilibrio entre precisión operativa, velocidad de respuesta y adaptabilidad a escenarios reales.

5.4. Debilidades

Las principales debilidades identificadas en los modelos no supervisados para la detección de fraude van más allá de su bajo rendimiento en el área bajo la curva *ROC*. Este radica en su **falta de explicabilidad**, operando como "cajas negras"que no permiten justificar las decisiones tomadas. Este problema impide que los analistas comprendan por qué una transacción fue marcada como fraudulenta, dificultando la validación de resultados y la mejora iterativa del modelo.

En el contexto financiero, donde cada decisión debe ser justificable ante reguladores y clientes, esta **ausencia de interpretabilidad** representa una barrera crítica para su adopción en entornos de producción, independientemente de sus capacidades en técnicas de detección de patrones de fraude. A diferencia de los modelos supervisados que permitieron identificar claramente las variables más influyentes, los modelos no supervisados mantienen ocultas las relaciones causales, limitando severamente su utilidad práctica en sistemas de detección de fraudes en entornos financieros, donde cada decisión tomada puede influir críticamente en la experiencia de usuario y en el dinero de las personas.

6

Conclusiones y Líneas Futuras

Como resultado de este trabajo se ha desarrollado una *API* funcional capaz de detectar anomalías en transacciones financieras mediante el uso de técnicas de *Machine Learning*, además de llevarse a cabo una investigación exhaustiva sobre distintos tipos de modelos y versiones en sus entrenamientos. Esta investigación ha permitido analizar cuál de ellos se adapta mejor a la naturaleza de los datos financieros disponibles, caracterizados por su alta complejidad y su fuerte desbalance. A través de varias iteraciones experimentales y un proceso continuo de mejora tanto en la selección de características como en la optimización de parámetros, se han conseguido obtener resultados sólidos en los modelos supervisados, mientras que los modelos no supervisados han servido para abrir nuevas líneas de trabajo a pesar de presentar un rendimiento más limitado.

La realización de este proyecto ha supuesto, no solo un acercamiento práctico a la construcción de sistemas inteligentes al sector financiero, sino también una inmersión en el proceso de experimentación y evaluación de modelos de *Machine Learning*, aspecto fundamental para el éxito de este tipo de soluciones. Aunque los resultados obtenidos todavía presentan margen de mejora, el trabajo realizado supone un paso importante en el camino hacia la automatización de la detección de fraude, un campo que incluso hoy en día sigue planteando grandes retos a las entidades financieras.

Respecto a las líneas futuras, sería interesante incorporar técnicas más avanzadas como modelos de aprendizaje profundo o métodos basados en *autoencoders* para mejorar la detección de patrones anómalos más sutiles. Asimismo, resulta fundamental integrar técnicas de explicabilidad que aumenten la confianza de los usuarios finales, como pueden ser personal bancario y auditores, permitiendo así justificar las decisiones algorítmicas en un contexto don-

de la transparencia es crítica. También se plantea ampliar el conjunto de datos con información adicional del comportamiento de los usuarios y desplegar la *API* en un entorno de producción, donde se pueda validar su rendimiento en condiciones reales y seguir evolucionando los modelos a partir de datos en tiempo real.

Referencias

[1] *¿Qué es el fraude? Definición y ejemplos importantes de fraude* [Accedido: 19 de marzo de 2025]. (2025). https://financialcrimeacademy.org/es/que-es-el-fraude-definicion-y-ejemplos-importantes-de-fraude/

[2] *¿Qué Es MySQL? Una Explicación para Principiantes* [Accedido: 30 de marzo de 2025]. (2025). https://kinsta.com/es/base-de-conocimiento/que-es-mysql/

[3] Abdelhamid, D., Babahenini, M., & taleb-ahmed, A. (2011). A fast multi-class SVM learning method for huge databases. *International Journal of Computer Science Issues*, *8*, 544-550.

[4] *Acerca de GitHub y Git* [Accedido: 30 de marzo de 2025]. (2025). https://docs.github.com/es/get-started/start-your-journey/about-github-and-git

[5] Alzubi, J. (2018). Machine Learning from Theory to Algorithms: An Overview. *Journal of Physics: Conference Series*. https://doi.org/10.1088/1742-6596/1142/1/012012

[6] *Aspectos básicos del fraude en las transacciones: motivos y posibles medidas empresariales* [Accedido: 29 de marzo de 2025]. (2023). https://stripe.com/es/resources/more/fraudulent-transactions-101

[7] *Card fraud in Europe declines significantly* [Accedido: 29 de marzo de 2025]. (2023). https://www.ecb.europa.eu/press/pr/date/2023/html/ecb.pr230526~f09bc3c664.es.html

[8] *ECB and EBA publish joint report on payment fraud* [Accedido: 29 de marzo de 2025]. (2024). https://www.ecb.europa.eu/press/pr/date/2024/html/ecb.pr240801~f21cc4a009.es.html

[9] *FastAPI* [Accedido: 30 de marzo de 2025]. (s.f.). https://fastapi.tiangolo.com/es/

[10] .M, D., El-Kilany, A., & Mokhtar, H. (2021). A Hybrid Model for Documents Representation. *International Journal of Advanced Computer Science and Applications*, *12*. https://doi.org/10.14569/IJACSA.2021.0120339

[11] *Machine learning* [Accedido: 19 de marzo de 2025]. (2022). https://www.redhat.com/es/topics/ai/what-is-machine-learning

[12] *Matplotlib: todo lo que tienes que saber sobre la librería Python de Dataviz* [Accedido: 30 de marzo de 2025]. (2022). https://datascientest.com/es/todo-sobre-matplotlib

[13] Morales, F. C. (2024). *Transacción financiera* [Accedido: 29 de marzo de 2025]. https://www.rankia.com/diccionario/bolsa/transaccion-financiera

[14] *NumPy : La biblioteca de Python más utilizada en Data Science* [Accedido: 30 de marzo de 2025]. (2023). https://datascientest.com/es/numpy-la-biblioteca-python

[15] *Pandas : La biblioteca de Python dedicada a la Data Science* [Accedido: 30 de marzo de 2025]. (2022). https://datascientest.com/es/pandas-python

[16] Parliament, E., & of the European Union, C. (2007). *Directive 2007/64/EC on payment services in the internal market* (N.º L 319/1). Official Journal of the European Union. Consultado el 29 de marzo de 2024, desde https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32007L0064

[17] pointe77. (2024). *Credit Card Transaction* [Último acceso: 30 de marzo de 2025]. https://huggingface.co/datasets/pointe77/credit-card-transaction

[18] *Preguntas frecuentes generales sobre Python* [Accedido: 30 de marzo de 2025]. (2024). https://docs.python.org/es/3.10/faq/general.html#what-is-python

[19] *PSD2: Tus compras por Internet, ahroa más seguras* [Accedido: 29 de marzo de 2025]. (s.f.). https://www.caixabank.es/particular/seguridad/psd2-y-exenciones.html

[20] *PyCharm* [Accedido: 30 de marzo de 2025]. (2025). https://es.wikipedia.org/wiki/PyCharm

[21] *Qué es la Inteligencia Artificial* [Accedido: 19 de marzo de 2025]. (2023). https://planderecuperacion.gob.es/noticias/que-es-inteligencia-artificial-ia-prtr

[22] *Random Forest Model* [Accedido: 30 de marzo de 2025]. (s.f.). https://www.ml-science.com/random-forest

[23] *Scikit-learn* [Accedido: 30 de marzo de 2025]. (2024). https://es.wikipedia.org/wiki/Scikit-learn

[24] *SQLAlchemy* [Accedido: 30 de marzo de 2025]. (2025). https://entrenamiento-frameworks-web-python.readthedocs.io/es/latest/leccion2/sqlalchemy.html

[25] *Visa Secure con EMV 3-D Secure* [Accedido: 29 de marzo de 2025]. (2018). https://www.visa.es/gestiona-tu-negocio/herramientas-para-pequenas-empresas/tecnologia-de-pago/visa-secure.html#:~:text=Permite%20el%20intercambio%20de%20datos%20entre%20comercios%20y%20bancos%20emisores,y%20la%20detecci%C3%B3n%20de%20fraudes.

[26] *What is Light GBM?* [Accedido: 30 de marzo de 2025]. (2019). https://datascience.eu/machine-learning/1-what-is-light-gbm/

Apéndice A

Manual de Instalación

Este manual describe paso a paso cómo instalar y ejecutar la API de detección de anomalías en transacciones financieras desarrollada en este trabajo.

El proceso de instalación es sencillo, pero requiere tener instalado Python (recomendado Python 3.10 o superior) y disponer de acceso a internet para descargar las dependencias necesarias. También se necesita tener un gestor de bases de datos MySQL funcionando en local.

En primer lugar, se debe clonar el repositorio del proyecto en la máquina local. Para ello, se abre una terminal y se ejecuta el comando *"git clone https://github.com/pabloleonn/Anomaly-Detection-in-Financial-Transaction—Final-Degree-Project.git"*. Una vez clonado, se debe acceder al directorio del proyecto con *"cd Anomaly-Detection-in-Financial-Transaction—Final-Degree-Project"*

Debido a las limitaciones de espacio de GitHub, el dataset utilizado no se encuentra en el repositorio. Se debe descargar manualmente desde el enlace proporcionado:

"https://drive.google.com/file/d/1_P7JNccEY2ZxCm2cyJZIt-KMxxi3V-Rc/view?usp=sharing"

Una vez descargado, se debe extraer en el directorio raíz del proyecto, de forma que la carpeta *"data/"* contenga el archivo *"credit_card_transactions.csv"*. La estructura final de carpetas deberá ser similar a la mostrada en la Figura 38.

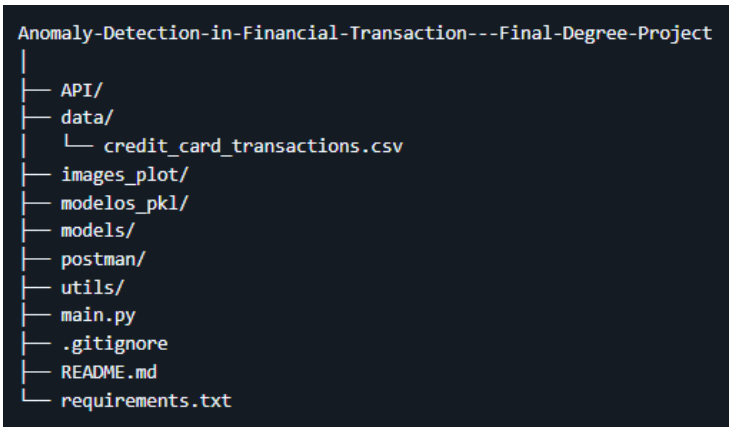

Figura 38: Directorio raíz proyecto

Aunque no es obligatorio, se recomienda crear un entorno virtual para aislar las dependencias del proyecto. Para ello, se puede ejecutar *"python -m venv venv"* y activar el entorno virtual mediante *".\venv\Scripts\activate"* en sistemas Windows.

Una vez activado el entorno, se deben instalar las librerías necesarias mediante el comando *"pip install -r requirements.txt"*, el cuál instalará todas las dependencias requeridas como FastApi, scikit-learn, SQLAlchemy y pymysql.

Para que la API funcione correctamente, es necesario asegurarse de que el puerto 3306 esté disponible para MySQL. En este proyecto se ha utilizado XAMPP como herramienta de gestión del servidor de bases de datos, aunque se puede utilizar cualquier otra alternativa. Para facilitar este proceso, se ha incluido en el repositorio el archivo *"iniciar_puerto_sql.bat"*, el cual, ejecutado con permisos de administrador, liberará el puerto si estuviera ocupado y pondrá en marcha MySQL automáticamente. Para su correcto funcionamiento, en caso de usar XAMPP, es posible configurar la variable *"XAMPP_PATH"* dentro del script apuntando a la ruta donde se tenga instalado XAMPP.

Una vez funcionando el servidor MySQL, es necesario configurar la conexión a la base de datos dentro del proyecto. Para ello, en el archivo *"./API/database.py"*, se debe actualizar la variable *"DATABASE_URL"* siguiendo el formato *"mysql+pymysql://usuario:contraseña@localhost:puerto/nombre*
Por ejemplo: *"mysql+pymysql://root:1234@localhost:3306/financial_db"*, donde *"root"* sería el usuario, *"1234"* la contraseña, y *"financia_db"* el nombre de la base de datos creada previamente en MySQL.

Con todo configurado, ya se puede proceder a lanzar la API. Si se desea, se puede ejecutar

el archivo *"iniciar_venv.bat"* que abrirá automáticamente una consola con el entorno virtual activado. Desde esa consola, simplemente ejecutando *"python main.py"* se pondrá en marcha el servidor de la API. Una vez iniciado, la API estará disponible accediendo a la dirección *"http://localhost:8000"*.

Además, para consultar la documentación de la API de forma gráfica y poder probar sus distintos endpoints, se puede acceder a *Swagger* a través de *"http://localhost:8000/docs"*, donde se presentará de manera automática toda la estructura de rutas y métodos disponibles, tal y como se muestra en la Figura 39.

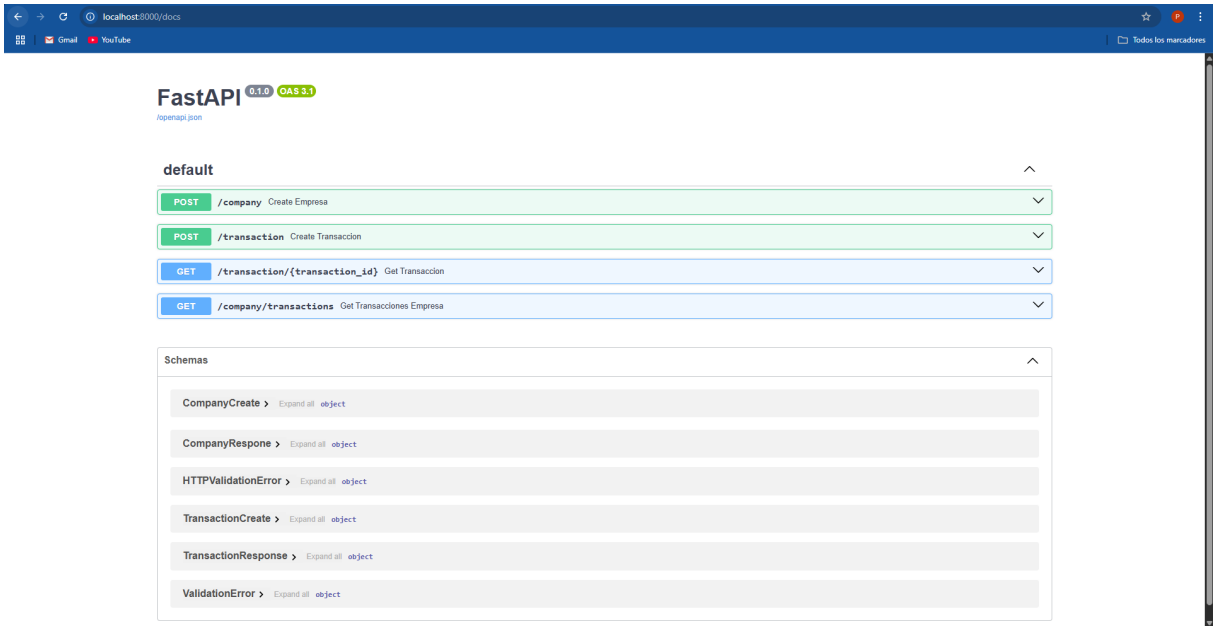

Figura 39: Documentación API en Swagger

UNIVERSIDAD DE MÁLAGA | uma.es

E.T.S de Ingeniería Informática
Bulevar Louis Pasteur, 35
Campus de Teatinos
29071 Málaga

E.T.S. DE INGENIERÍA INFORMÁTICA