Sistema predictivo de detección de fraude mediante aprendizaje automático

Autor:

Rubén Gómez Matamoros

Dirigido por:

Esteban Herreros Suárez

Grado de Ingeniería Informática

Curs 2024-2025
Tarragona

ESCOLA TÈCNICA SUPERIOR
D'ENGINYERIA
Universitat Rovira i Virgili

Índice

Resumen

Este Trabajo de Fin de Grado se centra en el desarrollo de una solución tecnológica orientada a la detección de comportamientos fraudulentos en entornos digitales. A través de un enfoque práctico y aplicado, el proyecto combina distintas fases de análisis, diseño y validación para construir un sistema funcional que pueda identificar actividades sospechosas en procesos de compra en línea. Durante su desarrollo, se ha simulado un entorno realista que permite poner a prueba el funcionamiento del sistema, evaluando su precisión y utilidad en situaciones controladas. El proyecto refleja una integración de conocimientos adquiridos a lo largo del grado, aplicados a un problema actual de gran relevancia en el ámbito tecnológico y social. El resultado es una propuesta sólida, dirigida a mejorar la seguridad y confianza en los sistemas digitales.

Resum

Aquest Treball de Fi de Grau se centra en el desenvolupament d'una solució tecnològica orientada a la detecció de comportaments fraudulents en entorns digitals. Mitjançant un enfocament pràctic i aplicat, el projecte combina diferents fases d'anàlisi, disseny i validació per construir un sistema funcional que pugui identificar activitats sospitoses en processos de compra en línia. Durant el seu desenvolupament, s'ha simulat un entorn realista que permet posar a prova el funcionament del sistema, avaluant-ne la precisió i utilitat en situacions controlades. El projecte reflecteix una integració de coneixements adquirits al llarg del grau, aplicats a un problema actual de gran rellevància en l'àmbit tecnològic i social. El resultat és una proposta sòlida, dirigida a millorar la seguretat i la confiança en els sistemes digitals.

Abstract

This Final Degree Project focuses on the development of a technological solution aimed at detecting fraudulent behavior in digital environments. Through a practical and applied approach, the project combines different phases of analysis, design, and validation to build a functional system capable of identifying suspicious activities in online purchase processes. During its development, a realistic environment was simulated to test the system's operation, evaluating its accuracy and usefulness in controlled situations. The project reflects an integration of knowledge acquired throughout the degree, applied to a current problem of great relevance in the technological and social fields. The result is a solid proposal designed to improve security and trust in digital systems.

Palabras clave

Aprendizaje supervisado, Base de datos, Detección de fraude, Desbalanceo de clases, Random Forest, Preprocesamiento de datos, Evaluación de modelos, Firebase, Aplicación Android, Machine Learning.

1. Introducción

El crecimiento exponencial del comercio electrónico, las plataformas digitales y los servicios financieros en línea ha incrementado significativamente la exposición de usuarios y organizaciones a actividades fraudulentas. Este fenómeno ha despertado un interés creciente por parte de empresas, instituciones y la comunidad académica en el desarrollo de sistemas inteligentes capaces de detectar, prevenir y mitigar los riesgos asociados al fraude. El fraude financiero, tanto en el ámbito del consumidor como en las relaciones interempresariales, constituye una amenaza real que implica no solo pérdidas económicas sustanciales, sino también una grave afectación a la confianza de los usuarios y a la reputación corporativa.

Las estrategias tradicionales, basadas en reglas estáticas o revisiones manuales, han demostrado ser insuficientes ante patrones de fraude cada vez más complejos y dinámicos. En este contexto, las técnicas de Machine Learning, subcampo de la inteligencia artificial que se centra en desarrollar algoritmos capaces de aprender patrones a partir de datos y realizar predicciones o decisiones (ML) se posicionan como una solución eficaz, al permitir la detección de comportamientos anómalos mediante el análisis de grandes volúmenes de datos y la identificación de patrones no evidentes. Estas técnicas no solo incrementan la capacidad de automatización del sistema, sino que también posibilitan una mejora continua en la precisión, adaptándose a nuevas tácticas empleadas por los defraudadores.

El presente Trabajo de Fin de Grado (TFG) tiene como objetivo el diseño e implementación de un sistema de detección de fraude basado en algoritmos de aprendizaje automático, con un enfoque analítico y aplicado. Para ello, se ha desarrollado una arquitectura compuesta por varios módulos funcionales: generación de una base de datos de prueba simulada, Exploratory Data Analysis, proceso de análisis preliminar de los datos que permite comprender su estructura, detectar patrones, valores atípicos y guiar decisiones de modelado (EDA), implementación de modelos de clasificación y despliegue del sistema en una aplicación funcional con tecnologías de backend modernas como Firebase.

Durante el desarrollo se han considerado diversos tipos de fraude relevantes, tales como el fraude de identidad (uso de tarjetas robadas o técnicas de *card testing*), reembolsos injustificados, falsificación de comprobantes de pago, abuso de programas de fidelización mediante cuentas falsas y fraudes asociados a transferencias internacionales.

El proyecto se ha estructurado mediante una planificación detallada, que abarca desde la preparación del entorno de trabajo y la revisión de herramientas, hasta la validación final del sistema y la elaboración de la documentación técnica. Para abordar el problema habitual del desbalance de clases presente en los datos de fraude, se ha empleado la técnica de sobremuestreo Synthetic Minority Over-sampling Technique, técnica de sobremuestreo utilizada para equilibrar clases en datasets desbalanceados generando nuevas instancias sintéticas de la clase minoritaria (SMOTE), y se han evaluado modelos de aprendizaje supervisado como Random Forest, algoritmo de aprendizaje supervisado basado en la combinación de múltiples árboles de decisión para mejorar la precisión y reducir el sobreajuste (RF). Estos modelos han sido entrenados, ajustados y comparados mediante métricas como precisión, Recall y F1-Score, con el fin de equilibrar la eficacia en la detección de fraudes y la minimización de falsos positivos.

Asimismo, el sistema se ha diseñado con una orientación práctica, integrando todos los componentes en una aplicación funcional que permite simular operaciones, consultar resultados y gestionar datos en tiempo real. Esta aplicación demuestra la viabilidad de incorporar mo-

delos de aprendizaje automático en entornos de uso real y constituye una muestra tangible del potencial de la solución desarrollada.

El presente trabajo busca no solo aplicar de forma integrada los conocimientos adquiridos en el grado en Ingeniería Informática, sino también ofrecer una solución sólida y replicable a uno de los retos más relevantes en los sistemas de información: la detección inteligente del fraude. El enfoque adoptado permite explorar la confluencia entre ciencia de datos, desarrollo de software y ciberseguridad, reflejando el carácter multidisciplinar inherente a los desafíos actuales del entorno digital.

2. Objetivos y Motivaciones

La elección del TFG fue por ingenio propio, dado cierto interés en conocer el ámbito del ML y una problemática actual. A continuación, trataré los objetivos a cumplir del trabajo y el porqué de la temática.

2.1. Objetivo general

El objetivo principal de este Trabajo de Fin de Grado es diseñar y desarrollar un sistema automatizado de detección de fraudes que emplee técnicas avanzadas de aprendizaje automático (ML). Este sistema debe ser capaz de procesar y analizar grandes volúmenes de datos transaccionales con el fin de identificar patrones indicativos de comportamientos fraudulentos. Además, se busca que los resultados obtenidos sean presentados de manera clara y accesible para el usuario final, facilitando la toma de decisiones basadas en la información generada. Para alcanzar este propósito, se plantea una arquitectura integral que englobe desde la adquisición y gestión eficiente de datos hasta la aplicación práctica del modelo en un entorno simulado que emule escenarios reales.

2.2. Objetivos específicos

Una vez tratado el general, es importante diferenciar las partes esenciales del trabajo para así entender los objetivos dentro de cada apartado.

2.2.1. Generación y gestión de una base de datos artificial

Con el fin de disponer de un conjunto de datos robusto y adecuado para el entrenamiento y evaluación del modelo, se desarrollará una base de datos sintética que contenga múltiples variables representativas de las características típicas de las transacciones electrónicas. Esta base de datos incorporará registros tanto de transacciones normales como fraudulentas, reflejando la heterogeneidad y complejidad del comportamiento real. La generación y gestión adecuada de estos datos es fundamental para garantizar la calidad y la representatividad del sistema, así como para permitir una evaluación precisa de la capacidad del modelo para diferenciar entre transacciones legítimas y sospechosas.

2.2.2. Análisis exploratorio y selección de variables

Para comprender en profundidad las características y la distribución de los datos recopilados, se llevará a cabo un análisis exploratorio de datos (EDA). Este proceso permitirá identificar patrones, detectar posibles anomalías o inconsistencias, y evaluar la relevancia de las diferentes variables contenidas en la base de datos. A partir de esta exploración, se seleccionarán las características más significativas para la construcción del modelo de detección, optimizando así su desempeño y reduciendo la dimensionalidad del problema para evitar sobre ajustes o ruido innecesario.

2.2.3. Evaluación del rendimiento del sistema

Finalmente, se procederá a la evaluación rigurosa del modelo mediante métricas específicas del ámbito del aprendizaje automático, tales como la precisión (*precision*), la exhaustividad o sensibilidad (Recall) y la puntuación F1-Score. Estas métricas ofrecen una valoración equilibrada del rendimiento del sistema. Además, se analizarán otros indicadores complementarios para asegurar que el sistema no solo identifica correctamente los fraudes, sino que también minimiza los falsos positivos, contribuyendo así a una solución efectiva y práctica.

2.2.4. Desarrollo de una aplicación móvil

Este objetivo consiste en la creación de una aplicación móvil destinada a dispositivos Android que permita simular procesos de compra en línea, replicando las dinámicas habituales del comercio electrónico. La aplicación cumplirá una doble función: por una parte, ofrecerá una interfaz amigable y funcional que facilite la interacción del usuario con el sistema; por otra, generará datos de transacciones que reflejen comportamientos tanto legítimos como fraudulentos, sirviendo como fuente realista para alimentar la base de datos del sistema de detección.

2.3. Motivación personal

La elección de esta temática para el Trabajo de Fin de Grado nace de un interés profundo y sostenido por la inteligencia artificial y sus aplicaciones en problemas reales de alta relevancia social y económica. A lo largo de la carrera en Ingeniería Informática, la motivación principal ha sido la posibilidad de desarrollar soluciones tecnológicas con un impacto tangible y beneficios directos para los usuarios y las organizaciones. En este sentido, la detección de fraude en entornos digitales se presenta como un desafío crítico para las empresas, especialmente en el comercio electrónico, donde las pérdidas derivadas de actividades fraudulentas pueden ser significativas y afectan tanto a consumidores como a proveedores.

Este proyecto ofrece una oportunidad única para integrar y aplicar conocimientos en áreas tan diversas como el desarrollo de software, la ciencia de datos y la ciberseguridad. A través de esta integración, se aborda un problema complejo que refleja la realidad y los desafíos actuales del entorno digital, aportando una solución práctica que puede contribuir a mejorar la seguridad y confianza en los sistemas de pago electrónicos.

3. Planificación

En la planificación inicial del proyecto, el enfoque principal se estableció en el desarrollo de un sistema de detección de fraude mediante el uso de técnicas avanzadas de ML. Sin embargo, a medida que avanzaba el desarrollo, surgieron diversas dificultades relacionadas con la recopilación, limpieza y preparación de los datos, lo que llevó a reconsiderar parcialmente la estrategia planteada.

Como resultado de este replanteamiento, se optó por ampliar el alcance original del proyecto, incorporando el diseño y desarrollo de una aplicación móvil. Esta decisión supuso un incremento significativo tanto en la complejidad técnica como en la extensión del trabajo requerido. Además, se asignó tiempo adicional a la depuración del código y a la optimización de los estándares de programación, con el propósito de fomentar una planificación más flexible y adaptativa, permitiendo integrar de manera eficiente estas nuevas tareas dentro del marco general del proyecto.

A continuación, se presenta el diagrama de Diagrama de Gantt, en el que se detallan las distintas fases y tareas ejecutadas a lo largo del desarrollo, reflejando la evolución del trabajo y los ajustes realizados en la planificación.

Figura 1: Diagrama de Gantt general del proyecto.

4. Marco Teórico

La detección de fraude en entornos digitales representa un desafío técnico y estratégico de gran relevancia, acentuado por la rápida expansión del comercio electrónico, la digitalización de los servicios financieros y el incremento en la sofisticación de los métodos utilizados por los defraudadores. En este contexto, los enfoques tradicionales, basados en reglas fijas o auditorías manuales, han demostrado ser insuficientes. Se requiere, por tanto, un marco más robusto, flexible y automatizado, como el que ofrecen las técnicas de aprendizaje automático. Este proyecto se apoya en varios pilares teóricos fundamentales: el ML, el tratamiento del desbalance de clases, los algoritmos de ensamblado como RF, y la gestión eficiente de datos mediante plataformas como Firebase.

El aprendizaje automático, o ML, es una rama de la inteligencia artificial que desarrolla algoritmos capaces de aprender patrones a partir de datos y realizar predicciones o decisiones basadas en esa experiencia previa (Géron, 2019). Su aplicación resulta especialmente pertinente en tareas como la detección de fraude, donde los comportamientos fraudulentos pueden manifestarse a través de patrones complejos y no lineales que escapan a las metodologías tradicionales.

En el marco del aprendizaje automático, el enfoque supervisado ha sido el más utilizado en este proyecto. Este tipo de aprendizaje emplea conjuntos de datos etiquetados, donde cada transacción se clasifica como legítima o fraudulenta. A partir de estos datos históricos, los algoritmos construyen modelos capaces de generalizar y predecir el comportamiento de nuevas transacciones. Se han considerado diferentes técnicas supervisadas, entre ellas la regresión logística, los árboles de decisión y, especialmente, los métodos de ensamblado como RF (Breiman, 2001), dada su capacidad para manejar datos heterogéneos y resistir al sobreajuste. Aunque técnicas más complejas como las redes neuronales profundas también pueden emplearse en este contexto, su alto requerimiento de datos etiquetados y recursos computacionales limita su aplicabilidad en proyectos a pequeña o mediana escala.

Complementariamente, el aprendizaje no supervisado ofrece herramientas útiles cuando los datos no están etiquetados. Este enfoque permite identificar agrupaciones, correlaciones ocultas o anomalías sin una guía explícita. En el contexto del fraude, resulta especialmente valioso para detectar nuevas estrategias de ataque que aún no han sido clasificadas. Algoritmos como *K-Means* o *DBSCAN* pueden emplearse para descubrir comportamientos atípicos, mientras que técnicas como el Análisis de Componentes Principales (PCA) o los *autoencoders* permiten reducir la dimensionalidad y detectar desviaciones significativas en los datos (Géron, 2019).

Uno de los principales retos en problemas de detección de fraude es el fuerte desbalance de clases. En la mayoría de los escenarios reales, las transacciones legítimas representan más del 98 % de los registros, lo que provoca que los algoritmos tiendan a favorecer la clase mayoritaria, minimizando la importancia de los casos fraudulentos (Chawla et al., 2002). Este desequilibrio afecta negativamente a la capacidad del modelo para identificar con precisión las instancias más críticas.

Para mitigar este problema, se utilizan técnicas de preprocesamiento como el submuestreo (reducción de la clase mayoritaria) o el sobremuestreo (aumento de la clase minoritaria). Dentro de estas últimas destaca SMOTE, que genera ejemplos sintéticos a partir de interpolaciones entre instancias reales de la clase minoritaria (Chawla et al., 2002). A diferencia de la duplicación directa, SMOTE preserva la diversidad del conjunto de datos y mejora la

capacidad del modelo para generalizar. Esta técnica resulta especialmente útil en contextos donde los falsos negativos tienen un alto costo, como es el caso de la detección de fraude.

El algoritmo RF ha sido seleccionado como modelo principal de clasificación por su capacidad para combinar múltiples árboles de decisión y generar predicciones más estables y precisas (Breiman, 2001). Al entrenar distintos árboles sobre subconjuntos aleatorios de datos y características, este método introduce diversidad entre modelos, lo que incrementa la robustez y disminuye el riesgo de sobreajuste. Además, permite interpretar la importancia relativa de cada variable, lo que resulta valioso en contextos donde es necesario comprender qué atributos están influyendo en la detección del fraude.

Para facilitar el almacenamiento, sincronización y gestión de los datos generados, se ha utilizado la plataforma Firebase. Desarrollada por Google, proporciona servicios backend como bases de datos en tiempo real, autenticación de usuarios, almacenamiento en la nube y funciones de seguridad personalizadas (Firebase, 2024). En este trabajo se ha empleado *Firebase Realtime Database* para almacenar transacciones y actualizar resultados de manera inmediata en la aplicación móvil, y *Firebase Authentication* para gestionar los accesos y mantener la integridad de los datos. Estas características han permitido una integración fluida entre el sistema de detección y el entorno móvil, facilitando la validación de los modelos en un entorno funcional y realista.

La evaluación del sistema se ha basado en métricas específicas que permiten medir el rendimiento en presencia de datos desbalanceados. Dado que la precisión global puede ser engañosa en estos contextos, se han empleado métricas como la precisión (*precision*), la exhaustividad Recall y la puntuación F1-Score (Powers, 2011). Estas medidas permiten valorar no solo cuántos casos fraudulentos se detectan correctamente, sino también cuántas predicciones positivas son realmente fraude, logrando así un equilibrio entre sensibilidad y especificidad. Asimismo, se ha considerado el uso del área bajo la curva ROC (AUC), que ofrece una visión general de la capacidad del modelo para discriminar entre clases en diferentes umbrales de decisión.

En conjunto, estos fundamentos teóricos proporcionan la base necesaria para abordar de forma rigurosa el problema de la detección automática de fraude, integrando técnicas avanzadas de análisis de datos con herramientas tecnológicas modernas en una solución práctica y funcional.

4.1. Ventajas del enfoque basado en Machine Learning frente al procesamiento tradicional de datos

En el ámbito de la detección de fraude, el procesamiento de datos mediante técnicas convencionales —como los métodos estadísticos clásicos o el uso de reglas fijas— ha demostrado ser cada vez más ineficiente ante la sofisticación y el dinamismo de los fraudes modernos. Estas técnicas, basadas generalmente en supuestos lineales, relaciones deterministas y umbrales estáticos, fueron útiles en contextos donde los datos presentaban poca variabilidad y patrones relativamente estables. Sin embargo, en los actuales entornos digitales, donde los comportamientos maliciosos son altamente adaptativos y complejos, estos enfoques resultan claramente insuficientes.

El fraude digital evoluciona constantemente con el propósito explícito de eludir los sistemas de detección existentes. Esto implica que sus patrones no siguen distribuciones estadísticas evidentes ni relaciones simples entre variables. Por ejemplo, un actor malicioso puede simular durante un periodo prolongado el comportamiento de un usuario legítimo, para luego ejecutar una transacción fraudulenta que, observada de forma aislada, podría no resultar sospechosa. Esta clase de patrones ocultos, que emergen únicamente cuando se consideran interacciones no triviales entre múltiples variables, son prácticamente imposibles de detectar mediante sistemas basados en lógica lineal o análisis univariados.

En este contexto, el ML se presenta como un paradigma analítico mucho más adecuado. A diferencia del procesamiento tradicional, los algoritmos de ML no requieren una definición explícita de las relaciones entre variables por parte del analista. Los modelos son capaces de aprender de forma automática a partir de los datos, extrayendo patrones complejos, estructuras latentes, correlaciones no lineales y relaciones de alta dimensionalidad que escapan a la detección humana directa. Esta capacidad de aprendizaje permite a los sistemas basados en ML detectar señales sutiles de fraude que evolucionan en el tiempo, adaptándose de forma dinámica al comportamiento emergente.

Una de las principales ventajas del enfoque basado en aprendizaje automático es su capacidad de mejora continua. Los modelos pueden ser reentrenados de forma periódica con datos nuevos, lo que permite su actualización automática ante la aparición de nuevas formas de fraude. Mientras que los sistemas tradicionales exigen una intervención manual para redefinir reglas, límites o condiciones, los algoritmos de ML integran de manera natural el aprendizaje continuo, mejorando progresivamente su capacidad de generalización sin necesidad de rediseñar el sistema desde cero.

Asimismo, el enfoque basado en ML incorpora herramientas que permiten cuantificar rigurosamente el rendimiento del sistema a través de métricas objetivas como la precisión, el recall, el F1-score o el área bajo la curva ROC (AUC). Esto facilita la evaluación comparativa de distintos modelos, la realización de validaciones cruzadas y la detección de posibles sesgos o errores sistemáticos. Este tipo de análisis empírico es escaso en los sistemas tradicionales, los cuales suelen carecer de mecanismos sistemáticos de evaluación.

Otra ventaja significativa es la posibilidad de aplicar técnicas avanzadas para abordar el desbalance de clases, un problema común en la detección de fraude, donde las transacciones fraudulentas representan una proporción mínima respecto al total. El ML ofrece múltiples estrategias para afrontar este reto, como el sobremuestreo de la clase minoritaria, el submuestreo de la clase mayoritaria o la generación sintética de instancias mediante algoritmos como SMOTE. Estas técnicas permiten mejorar la sensibilidad del modelo sin comprometer

la especificidad, algo que resulta muy difícil de lograr con enfoques tradicionales.

En definitiva, el uso de ML para el tratamiento de datos en la detección de fraude permite afrontar de forma más eficaz la complejidad, variabilidad y naturaleza cambiante del fenómeno. Aporta mejoras significativas en la capacidad de detección, reduce los errores tipo I y II (falsos positivos y falsos negativos), y dota al sistema de una flexibilidad, escalabilidad y capacidad de adaptación imprescindibles en el contexto actual. Por estas razones, el aprendizaje automático no representa únicamente una mejora técnica respecto al enfoque clásico, sino que constituye una herramienta esencial para el diseño de sistemas robustos, sostenibles y verdaderamente eficaces en la lucha contra el fraude digital.

Cuadro 1: Comparativa entre procesamiento tradicional y Machine Learning en detección de fraude

Procesamiento Tradicional	**Machine Learning**
Uso de reglas fijas y umbrales definidos manualmente	Aprendizaje automático a partir de datos sin necesidad de reglas explícitas
Relaciones lineales y supuestos simplistas entre variables	Capacidad para modelar relaciones complejas y no lineales
Difícil adaptación a nuevos patrones de fraude	Reentrenamiento continuo con nuevos datos para adaptarse dinámicamente
Basado en conocimiento experto y lógica determinista	Basado en patrones estadísticos descubiertos automáticamente
Poca o nula gestión del desbalance de clases	Uso de técnicas avanzadas como sobremuestreo, submuestreo o generación sintética de datos
Limitada capacidad de evaluación sistemática	Evaluación empírica con métricas objetivas como precisión, recall, F1-score y AUC
Baja escalabilidad y flexibilidad ante cambios	Alta escalabilidad y sostenibilidad en entornos dinámicos

5. Requisitos

Este apartado describe los requisitos que debe cumplir el sistema propuesto para garantizar su funcionamiento adecuado desde una perspectiva tanto funcional como técnica. Los requisitos se dividen en dos categorías: funcionales y no funcionales. Los primeros se centran en las funcionalidades que el sistema debe ofrecer al usuario, mientras que los segundos abordan aspectos relacionados con el rendimiento, la seguridad, la escalabilidad y la portabilidad del sistema.

5.1. Requisitos funcionales

El sistema debe permitir a los usuarios autenticarse de forma segura mediante un mecanismo de inicio de sesión gestionado por Firebase Authentication. Una vez autenticado, el usuario podrá acceder a diversas funcionalidades, entre ellas: modificar su perfil, registrar nuevas compras, añadir productos a una lista de favoritos, puntuar productos adquiridos y gestionar devoluciones.

Toda la información generada a través del uso de la aplicación será recogida y almacenada para su posterior análisis por el sistema de detección de fraudes. Este sistema ejecuta sus tareas de forma automática y periódica, analizando las transacciones registradas para identificar posibles actividades anómalas. En caso de detectar una compra sospechosa, la aplicación notificará al usuario, proporcionando así una respuesta directa y comprensible ante posibles casos de fraude.

El diagrama de bloques representado en la Figura 2 describe el flujo general del sistema de detección de fraude, desde la recolección de datos en la aplicación móvil hasta el análisis de dichos datos mediante técnicas de aprendizaje automático.

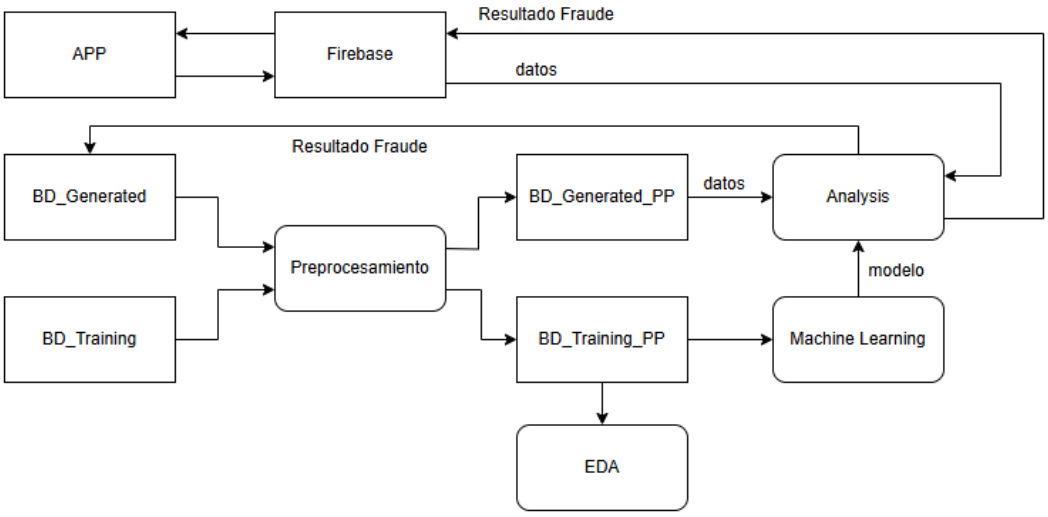

Figura 2: Diagrama general del sistema de detección de fraude.

A continuación, se detallan los distintos componentes del sistema y su función dentro del flujo de trabajo:

- **APP:** Representa la interfaz del usuario final, desde donde se registran datos transaccionales. Además, la aplicación recibe la respuesta del sistema en relación con la detección de fraude.
- **Firebase:** Actúa como plataforma intermedia para la gestión y sincronización de datos en tiempo real. Recibe datos desde la APP y reenvía los resultados del análisis de fraude una vez procesados.
- **BD_Generated y BD_Training:** La base de datos BD_Generated contiene los datos generados para simular otras fuentes de datos, mientras que BD_Training corresponde a un conjunto de datos usados para entrenar los modelos de detección de fraude.
- **Preprocesamiento:** En esta etapa se realiza la limpieza, normalización y transformación de los datos. El objetivo es preparar tanto los datos generados como los de entrenamiento para que sean compatibles y útiles en el modelado.
- **BD_Generated_PP y BD_Training_PP:** Estas bases de datos almacenan los datos preprocesados. La primera corresponde a datos nuevos provenientes de la APP, y la segunda, a los datos históricos de entrenamiento.
- **EDA:** Aplicado sobre BD_Training_PP, permite descubrir patrones relevantes, analizar distribuciones y detectar valores atípicos, contribuyendo a una mejor comprensión del dominio y a una selección adecuada de variables.
- **Machine Learning:** Con los datos de entrenamiento preprocesados, se entrena un modelo capaz de detectar patrones de fraude. Este modelo se emplea posteriormente para clasificar nuevas transacciones.
- **Analysis:** Esta etapa aplica el modelo entrenado sobre los datos nuevos para identificar posibles fraudes. Los resultados se devuelven a Firebase y, en consecuencia, a la APP.

5.2. Requisitos no funcionales

En lo referente a los requisitos no funcionales, el sistema ha sido diseñado bajo principios de eficiencia, escalabilidad, seguridad y portabilidad.

Desde el punto de vista del rendimiento, se ha incorporado la posibilidad de configurar la periodicidad con la que se ejecuta el análisis de fraude. Esta flexibilidad permite adaptar el sistema a diferentes escenarios operativos, optimizando el uso de recursos y mejorando la capacidad de respuesta según las necesidades del entorno.

El diseño modular y escalable del sistema permite gestionar grandes volúmenes de transacciones sin degradar el rendimiento. Esta capacidad se ve reforzada por el uso de Firebase como plataforma backend, cuya arquitectura distribuida y su versión Blaze garantizan un procesamiento eficiente y una expansión progresiva del sistema conforme aumente la demanda.

En materia de seguridad, se ha integrado un sistema de autenticación robusto mediante Firebase Authentication, que admite múltiples métodos de acceso (correo electrónico, número

de teléfono, etc.) y garantiza la protección de los datos mediante el uso de claves seguras y cifrado de extremo a extremo. Además, los datos sensibles gestionados por el sistema se cifran con el algoritmo Fernet, una solución reconocida por su seguridad y eficiencia en aplicaciones distribuidas.

La arquitectura modular también favorece el mantenimiento y evolución del sistema, permitiendo la incorporación de nuevas funcionalidades, la corrección de errores o la adaptación a nuevos requisitos sin comprometer la estabilidad del conjunto. Esta característica resulta esencial en entornos tecnológicos dinámicos, donde los sistemas deben responder de forma ágil a cambios regulatorios, de negocio o tecnológicos.

En cuanto a la disponibilidad, el sistema realiza consultas periódicas a la base de datos para garantizar la actualización continua del análisis. Para contextos que requieran mayor frecuencia de actualización o ejecución en tiempo real, se contempla la posibilidad de desplegar el sistema en servidores dedicados, lo cual incrementaría su capacidad de procesamiento y disponibilidad continua.

La compatibilidad ha sido un factor clave en el diseño del sistema. Gracias a la integración con Firebase, que ofrece soporte multiplataforma, la solución puede ser adaptada fácilmente a otros entornos como iOS o aplicaciones web. Aunque el desarrollo principal se ha llevado a cabo en Android Studio, su migración a frameworks como React Native o Flutter permitiría ampliar el alcance del sistema a otros dispositivos y plataformas.

Por último, la portabilidad del sistema está garantizada mediante el uso de tecnologías ampliamente adoptadas y compatibles. La aplicación móvil desarrollada en Android Studio, combinada con un backend en Python y la base de datos online, conforma una arquitectura flexible y fácilmente desplegable en distintos entornos tecnológicos. Esto facilita su integración en organizaciones con infraestructuras heterogéneas y permite su evolución sin necesidad de reescribir componentes críticos del sistema.

6. Diseño

El diseño del sistema desarrollado se estructura en dos componentes principales: por un lado, la aplicación Android, que permite simular un entorno de compras en línea para la recolección de datos de usuarios y transacciones; y por otro, el sistema de detección de fraude, basado en técnicas de aprendizaje automático y herramientas de backend. Esta arquitectura dual permite integrar de forma eficiente la experiencia de usuario con el procesamiento automático de datos, facilitando tanto la simulación como la detección de comportamientos fraudulentos en un entorno controlado.

A continuación, se detallan los aspectos más relevantes del diseño de la aplicación móvil y del módulo de detección de fraude, destacando las tecnologías utilizadas, las decisiones de implementación y la estructura general del sistema.

6.1. Frontend

Con el objetivo de simular un entorno realista de compras en línea y facilitar la recolección de datos para el análisis de fraude, se ha desarrollado una aplicación móvil para Android. Esta aplicación emula el comportamiento típico de usuarios dentro de una plataforma de comercio electrónico, generando flujos de interacción y datos relevantes para su posterior análisis mediante técnicas de aprendizaje automático.

En cuanto al diseño visual de la aplicación, se ha optado por un enfoque minimalista, con una paleta de colores basada en escalas de negro, gris y blanco, complementada por el uso de azul en los botones para aportar contraste y facilitar la interacción. Esta elección responde tanto a criterios estéticos como funcionales, buscando una interfaz limpia y agradable para el usuario. Además, se ha implementado un modo oscuro (Dark Mode), una característica especialmente valorada por los usuarios por su comodidad visual, especialmente en entornos con poca luz.

La aplicación incorpora un sistema funcional de autenticación mediante correo electrónico y contraseña, lo que permite que cada usuario pueda acceder de forma segura a su perfil personal. Desde esta sección, los usuarios tienen la posibilidad de visualizar y editar sus datos personales, tales como el nombre, la dirección de correo electrónico y la dirección de envío.

En la interfaz principal se presentan productos destacados junto con las distintas categorías disponibles. Al acceder a una categoría específica, se despliega una vista con los productos correspondientes, cada uno de los cuales puede añadirse a favoritos, visualizar reseñas de otros clientes o incluirse en el carrito de compra. Este flujo de navegación reproduce de forma precisa el recorrido de un comprador típico.

El usuario puede completar la transacción en cualquier momento a través del carrito, y también dispone de una sección de historial donde se listan las compras realizadas. Desde esta misma sección es posible consultar los detalles de cada pedido y, si fuera necesario, solicitar una devolución.

En situaciones en las que se detecta un error durante el proceso de compra, la aplicación muestra una ventana emergente informando del incidente. Este mensaje permite al usuario revisar los detalles del intento de transacción una única vez. En caso de que no se realicen correcciones o si el sistema detecta patrones sospechosos que sugieren un posible fraude, la transacción es bloqueada silenciosamente, es decir, sin que el usuario reciba una notificación explícita al respecto.

Gracias a este diseño, la aplicación no solo ofrece una experiencia de usuario coherente con plataformas comerciales reales, sino que también facilita la generación controlada de datos para entrenar, evaluar y validar modelos de detección de fraude.

A continuación adjunto el diagrama general de la aplicación:

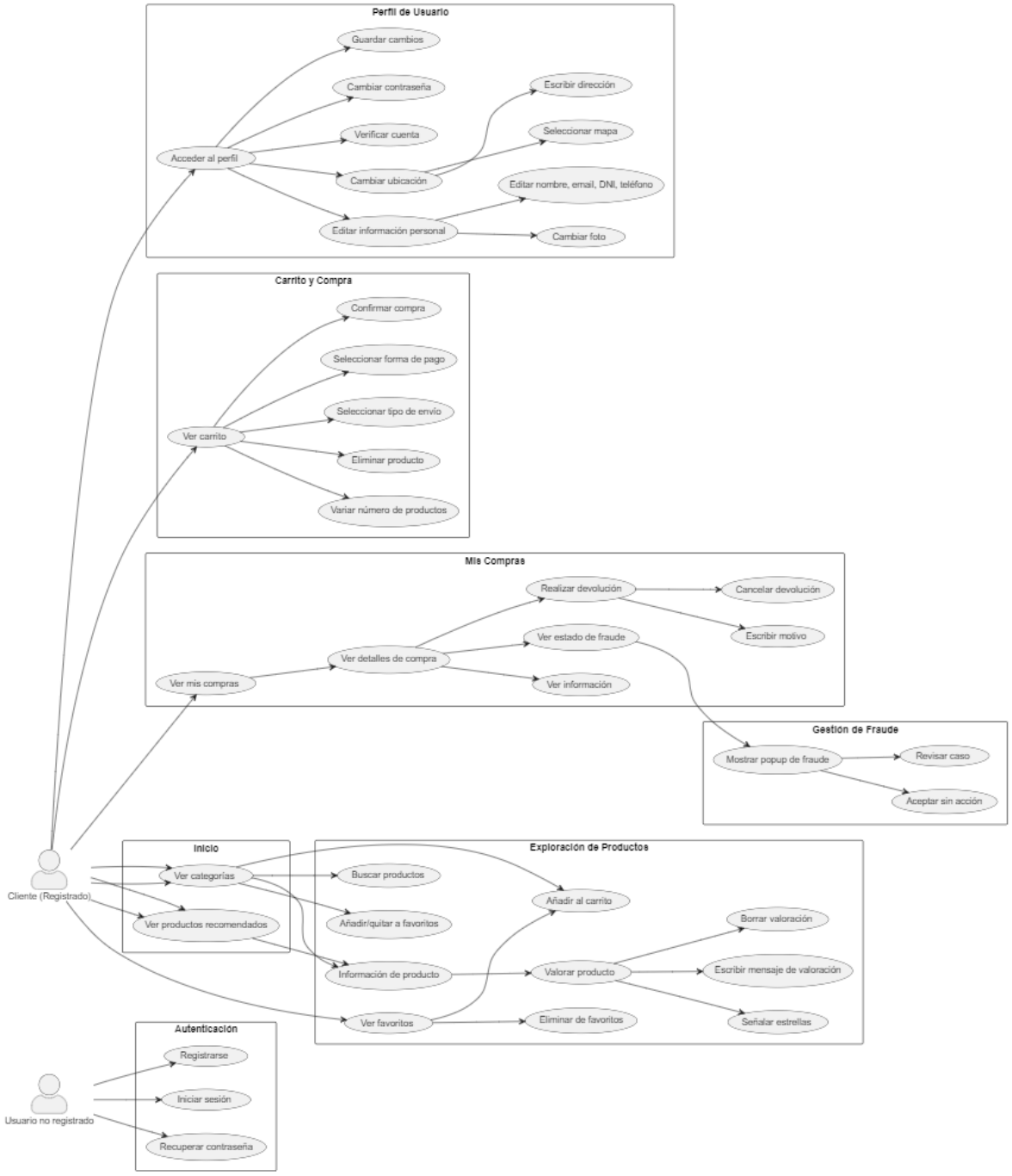

Figura 3: Diagrama general de la app mostrando el flujo de pantallas y funcionalidades.

6.1.1. Diseño de las pestañas

La aplicación Android desarrollada en este proyecto ofrece una experiencia de compra intuitiva, segura y completa, organizada en varias pestañas que cubren todo el ciclo de compra, desde exploración hasta gestión postventa y detección de fraude.

Pantalla Principal y Productos: La pantalla principal muestra información básica del usuario, productos recomendados y categorías navegables mediante tarjetas visuales. La sección de productos permite búsqueda y filtrado, con acceso a detalle ampliado incluyendo imágenes y valoraciones.

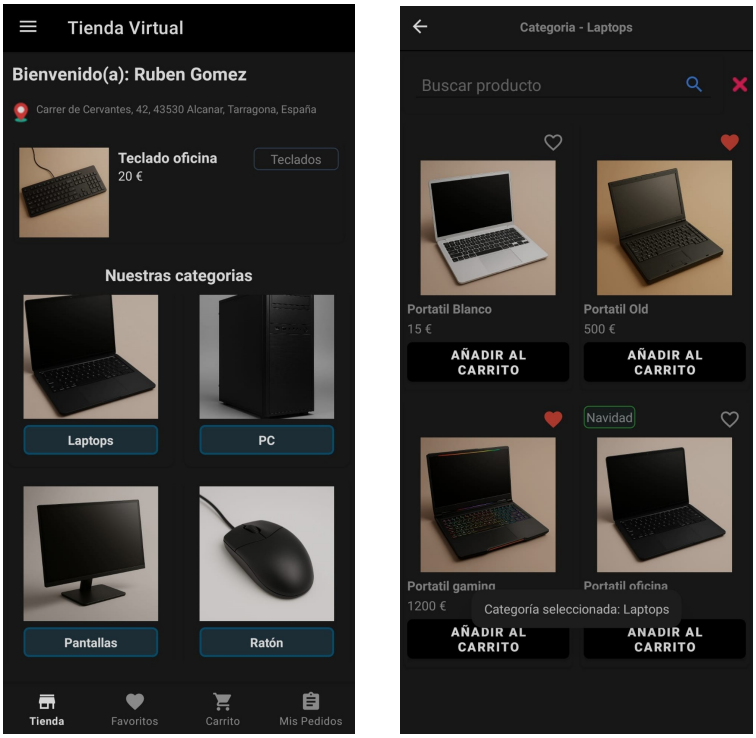

Figura 4: Pantalla principal y listado de productos

Perfil y Gestión de Cuenta: El usuario puede editar su perfil con imagen, datos personales y ubicación integrada con Google Maps. Además, existen funcionalidades para registro, inicio de sesión (incluyendo autenticación Google y teléfono), recuperación y actualización de contraseña, todas implementadas con validación y seguridad.

Interacción con Productos: El usuario puede calificar productos, gestionar favoritos y visualizar el carrito con opciones para modificar cantidades y seleccionar métodos de envío y pago (simulados). La sección de pedidos ofrece historial detallado, con alertas visibles si se detectan fraudes, y permite iniciar devoluciones.

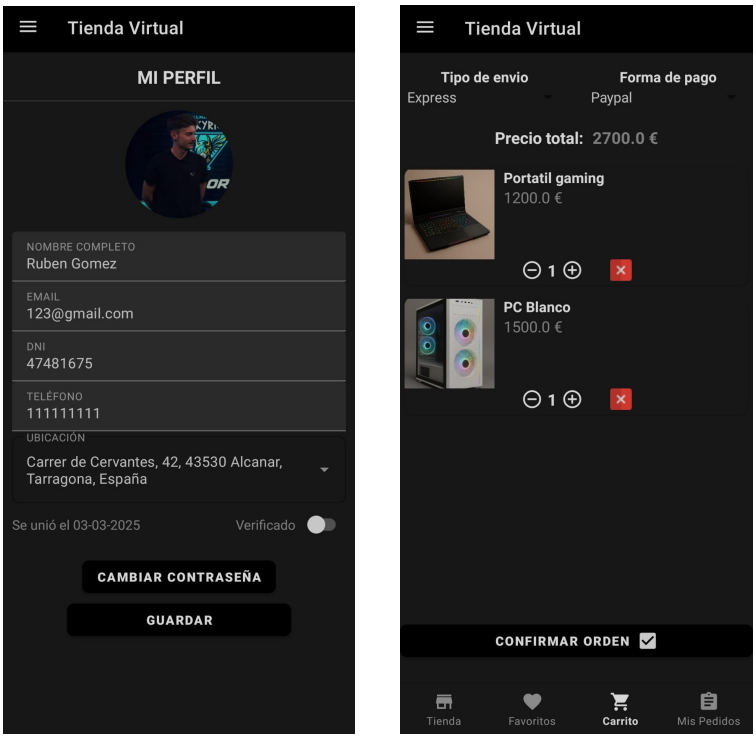

Figura 5: Gestión de perfil y carrito de compra

Detección de Fraude: La aplicación muestra pop-ups alertando sobre pedidos sospechosos y permite al usuario marcar para revisión, integrando el módulo de detección de fraude en el flujo habitual de la experiencia.

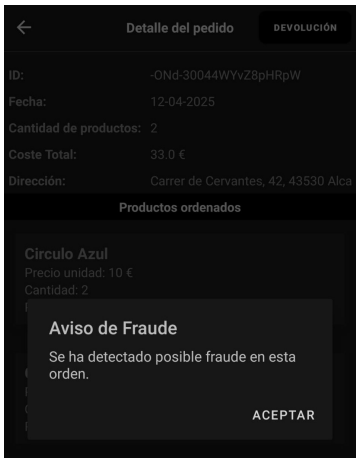

Figura 6: Pop-up de alerta por detección de fraude

Este diseño modular facilita una experiencia de usuario coherente y permite integrar análisis y control de fraudes en el proceso de compra, clave para la seguridad y confianza en la plataforma.

6.2. Backend

6.2.1. Algoritmo machine learning

Para abordar el problema de la detección de fraude en transacciones, se ha diseñado un sistema que combina técnicas avanzadas de ML, estrategias específicas para el tratamiento del desbalance de clases y herramientas modernas para la gestión de información en aplicaciones móviles.

Uno de los retos más relevantes en este ámbito es el fuerte desbalance entre clases: en la mayoría de conjuntos de datos reales, las transacciones legítimas constituyen la inmensa mayoría, mientras que las fraudulentas representan una fracción mínima. Esta desproporción puede llevar a que los modelos de clasificación minimicen la detección de fraudes al favorecer la clase mayoritaria. Para mitigar este efecto, se ha empleado la técnica SMOTE, que permite generar instancias sintéticas de la clase minoritaria a partir de interpolaciones entre observaciones reales cercanas en el espacio de características. A diferencia del sobremuestreo tradicional, SMOTE introduce variabilidad sin añadir ruido artificial, lo que contribuye a mejorar la capacidad del modelo para identificar patrones anómalos.

Una vez equilibrado el conjunto de datos, se ha seleccionado el algoritmo RF como modelo principal de clasificación. Este método de ensamble construye múltiples árboles de decisión sobre subconjuntos aleatorios de datos y variables, combinando sus predicciones mediante votación mayoritaria. Esta estrategia proporciona mayor robustez frente a datos ruidosos o no lineales, y reduce el riesgo de sobreajuste. Además, RF permite calcular la importancia relativa de cada variable, lo cual resulta especialmente útil en contextos donde se requiere interpretabilidad para entender qué factores influyen en la detección de transacciones sospechosas.

La elección de RF frente a otros modelos, como la regresión logística, las Support Vector Machine, algoritmo de aprendizaje supervisado utilizado para clasificación y regresión, que busca encontrar el hiperplano que mejor separa las clases en el espacio de características (SVM) o las redes neuronales profundas, se justifica por su equilibrio entre rendimiento, simplicidad de implementación e interpretabilidad. A diferencia de las SVM, RF gestiona eficientemente grandes volúmenes de datos sin necesidad de transformar complejamente el espacio de entrada. En comparación con las redes neuronales, ofrece resultados competitivos con menores requisitos computacionales y sin necesidad de ajustes intensivos de hiperparámetros.

Complementariamente, el sistema incorpora la plataforma Firebase como backend para la gestión de datos en tiempo real. En concreto, se ha utilizado Firebase Realtime Database, una base de datos NoSQL con estructura JSON, que facilita el almacenamiento, consulta y sincronización de datos en aplicaciones móviles. Su integración con Android permite reflejar instantáneamente los cambios de la base de datos en la interfaz de usuario, lo que mejora la interactividad y la experiencia del usuario. Además, ofrece funcionalidades de autenticación segura y control de acceso, fundamentales para proteger la información sensible tratada por el sistema.

En conjunto, la aplicación de SMOTE para el preprocesamiento de datos, el uso de RF como clasificador principal y la incorporación de Firebase como infraestructura backend conforman una arquitectura técnica robusta, escalable y orientada a la práctica. Este diseño integral permite afrontar el problema de la detección de fraude desde una perspectiva completa, abar-

cando tanto la vertiente analítica como la implementación funcional en un entorno realista.

6.2.2. Funciones del programa

Categoría	Funciones

Conexión y gestión de datos	`fetch_data_from_firebase()`
	`upload_to_firebase()`
	`generate_data_from_firebase(orders)`

Análisis y transformación	`analysis(file)` `transform_data(original)`

Datos personales y contacto	`generate_data(num)`
	`generate_id()`
	`generate_full_name()`
	`generate_mail(nombre_completo)`
	`generate_dni()`
	`generate_phone()`
	`generate_email_verified()`
	`generate_clicks_count()`
	`generate_intents_count()`
	`generate_trust_ratio()`

Dirección y tiempo	`generate_date()`
	`generate_hour()`
	`generate_transaction_time()`
	`generate_ip()`
	`generate_ip_paypal()`
	`generate_address()`

Transacción y producto	`generate_product_count()`
	`generate_total_price(num_products)`
	`generate_discount()`
	`generate_type()`
	`generate_transaction_method()`
	`generate_transaction_number(method)`
	`generate_device()`
	`generate_shipping_type()`

Finanzas	`generate_credit_card()`
	`luhn_checksum(card_number)`
	`generate_iban()`

Comentarios y tracking	`generate_comment(type)`
	`generate_tracking_code()`
	`generate_tracking_url(code)`

Simulación de fraude	`generate_fraud_boolean(fraud_count)` `generate_fraud_count(single_data)`
Simulación de encriptar	`generate_key()`

Categoría	Funciones
	`encrypt_file(file, key, final)` `decrypt_file(file_path, key)`

6.2.3. Distribución ficheros backend

Durante el proceso de creación y tratamiento de los datos, se generan diversas bases de datos que cumplen diferentes funciones dentro del sistema. Algunas de estas bases de datos se almacenan de forma temporal, mientras que otras tienen una importancia relevante y se guardan en formato `.csv` para su uso posterior. A continuación, se describen las principales bases de datos almacenadas y su propósito:

- DBFireBase.csv: Base de datos que contiene los datos originales extraídos de Firebase, actuando como fuente principal de información.
- DBGenerated.csv: Base de datos generada automáticamente, utilizada como fuente alternativa para pruebas y desarrollo, especialmente en el contexto de la aplicación Android.
- DBGeneratedPP.csv: Versión transformada y preprocesada de la base de datos generada, preparada para su análisis posterior. Contiene un volumen moderado de datos y se emplea en tareas de preprocesamiento.
- DBGeneratedEncripted.csv: Base de datos generada que ha sido cifrada para garantizar la seguridad y confidencialidad de la información almacenada.
- DBTraining.csv: Conjunto de datos creado específicamente para el entrenamiento del modelo de aprendizaje automático, caracterizado por su gran tamaño y diversidad de muestras.
- DBTrainingPP.csv: Versión transformada y preprocesada de la base de datos de entrenamiento, lista para su uso en el ajuste y evaluación del modelo predictivo.

Esta estructura organizada y diferenciada de bases de datos permite un manejo eficiente y seguro de la información en las distintas etapas del proyecto, desde la adquisición y preprocesamiento hasta el entrenamiento y validación del modelo.

7. Implementación

La implementación del sistema se ha desarrollado en Python, organizando el código en módulos independientes con el objetivo de favorecer la modularidad, el mantenimiento y la reutilización. Cada módulo está diseñado para abordar una fase específica del flujo de trabajo: generación de bases de datos simuladas, análisis exploratorio de datos y entrenamiento del modelo de detección de fraude. Respecto a la aplicación, se ha desarrollado en Android Studio. A continuación, se describen en detalle los principales componentes desarrollados.

7.1. Frontend

La aplicación Android desarrollada en este proyecto tiene como objetivo proporcionar una experiencia de compra intuitiva, segura y funcional, integrando diversas funcionalidades que abarcan desde la exploración de productos hasta la gestión de pedidos y devoluciones. La interfaz está organizada en varias pestañas principales, que permiten al usuario navegar fácilmente por la tienda, gestionar su perfil, realizar compras y evaluar productos.

Entre las funcionalidades más relevantes se incluyen la pantalla principal con productos destacados y categorías, un buscador para filtrar productos, gestión completa del perfil de usuario con opciones de edición y seguridad, la posibilidad de calificar productos tras la compra, y un carrito de compra que permite modificar cantidades y elegir métodos de envío y pago.

Además, la aplicación incorpora módulos específicos para la gestión de devoluciones, una sección de favoritos para guardar productos de interés y un historial de pedidos donde se notifican alertas de posibles fraudes detectados por el sistema. Este último aspecto es fundamental para el proyecto, ya que integra la funcionalidad de detección de fraude en el flujo de compra, proporcionando avisos y opciones de revisión para los usuarios.

Por último, se contemplan procesos estándar de recuperación y actualización de contraseña, así como registro e inicio de sesión con múltiples métodos de autenticación, garantizando la seguridad y accesibilidad del sistema.

7.2. Generación de Bases de Datos Simuladas

El módulo `GenerateBD.py` permite la creación de bases de datos sintéticas tanto para el entrenamiento del modelo como para la simulación de entornos operativos. A través de una interfaz por consola, el usuario puede seleccionar entre dos modalidades de generación: una orientada a entrenamiento, con tamaño configurable y estructura adaptada al aprendizaje automático; y otra simulada, de mayor volumen y diseñada para emular un entorno más cercano a la producción.

El proceso se inicia con la carga de variables de entorno que definen rutas, nombres de archivos y parámetros de ejecución. Según la opción elegida, se lleva a cabo la creación de los datos, su transformación estructural y el almacenamiento del conjunto final. En el caso de la base simulada, se incluye una columna adicional que indica el estado de la transacción y se aplica un proceso de cifrado al archivo resultante para preservar la confidencialidad de los datos.

La principal diferencia en la creación de las bases de datos radica en el estado asociado a la

variable *fraude*. Para el entrenamiento del modelo es necesario contar con datos etiquetados, es decir, transacciones reales clasificadas como `true` o `false` según hayan sido fraudulentas o no. En cambio, en el caso de la base de datos simulada, las nuevas transacciones se etiquetan inicialmente con el estado `Procesando`, ya que aún no se ha determinado si son fraudulentas.

El objetivo del sistema es que, tras la ejecución del modelo, todas las transacciones marcadas inicialmente como `Procesando` sean clasificadas automáticamente, obteniendo un valor simulado para la variable *fraude*, en función del análisis realizado por el modelo.

Las funciones auxiliares empleadas para la generación, transformación y cifrado están encapsuladas en el módulo `Utils`, lo que refuerza la separación de responsabilidades y mejora la mantenibilidad del código.

Los detalles técnicos completos de este componente se presentan en el Anexo 14.

7.3. Preprocesamiento y transformación de datos

La función encargada del preprocesamiento realiza la preparación inicial del conjunto de datos originales, con el objetivo de adecuarlos a un formato óptimo para el análisis y posterior modelado predictivo.

En primer lugar, se lleva a cabo la carga del conjunto de datos desde un archivo en formato CSV. Se procede a la correcta gestión de valores faltantes, sustituyendo indicadores no numéricos o cadenas especiales por valores nulos reconocidos por las librerías de procesamiento, lo que facilita su tratamiento en etapas posteriores.

A continuación, se asegura la correcta asignación de tipos de datos para cada variable, especialmente para las numéricas, garantizando que estén representadas en formatos adecuados (como enteros o números de punto flotante). Esta conversión es esencial para evitar errores en cálculos posteriores y para que los algoritmos de aprendizaje automático puedan interpretar correctamente la información.

Se generan nuevas variables binarias mediante la transformación y combinación de columnas con información categórica, con el fin de sintetizar características relevantes que puedan aportar valor predictivo al modelo.

De forma paralela, se eliminan aquellas columnas que aportan poca o ninguna información útil para el análisis, o que podrían introducir ruido y complejidad innecesaria. Esto incluye datos como fechas, direcciones o identificadores que no contribuyen directamente al proceso de detección.

Las variables categóricas restantes, que contienen datos nominales, son codificadas numéricamente mediante técnicas de codificación ordinal. Este proceso transforma las categorías en valores enteros, permitiendo que los modelos de aprendizaje automático puedan procesar dichas variables de manera eficiente, incluso cuando se presentan categorías no vistas durante el entrenamiento.

Adicionalmente, se generan variables derivadas a partir de agregaciones estadísticas sobre grupos definidos por identificadores relevantes, como el conteo total o la media de ciertas características por usuario o entidad. Estas nuevas variables capturan patrones de comportamiento que pueden ser indicadores significativos para la detección de irregularidades o fraudes.

Finalmente, la función devuelve el conjunto de datos preprocesado y transformado, preparado para su uso en etapas posteriores, tales como escalado, entrenamiento y predicción con modelos de ML. Este proceso es fundamental para mejorar la calidad de los datos y, en consecuencia, la efectividad y precisión del sistema predictivo implementado.

A continuación adjunto un muestraje del fichero resultado del preprocesamiento:

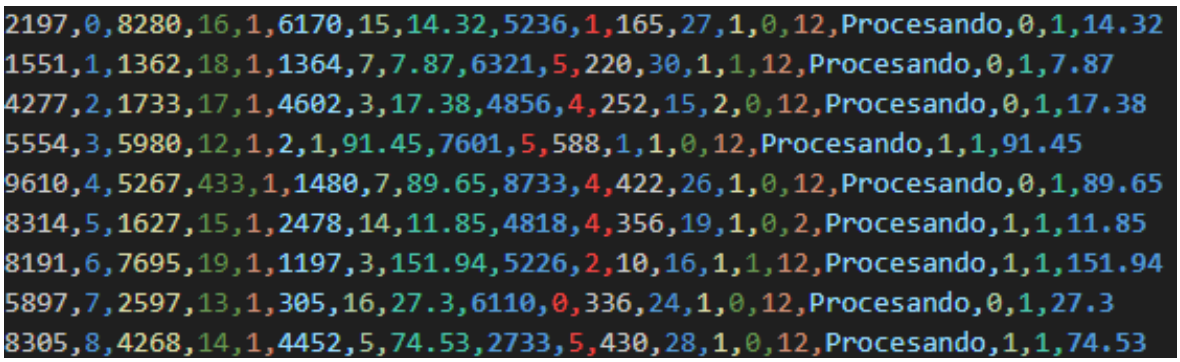

Figura 7: Resultado del preprocesamiento de la DataBase

7.4. Análisis Exploratorio de Datos

El módulo Eda.py realiza un análisis exploratorio de datos EDA sobre el conjunto empleado para entrenar el modelo de detección de fraude. Este análisis tiene como finalidad comprender mejor la estructura de los datos, identificar relaciones significativas entre variables y detectar patrones que puedan ser relevantes para el modelado.

Durante la ejecución, se generan diversas visualizaciones como histogramas, diagramas de caja, mapas de calor de correlación, gráficos de dispersión y conteos categóricos. Estas gráficas se almacenan automáticamente en una carpeta estructurada para facilitar su posterior revisión. Asimismo, el módulo incorpora una barra de progreso que permite al usuario monitorizar en tiempo real el avance del análisis.

Todo el proceso está orientado a servir de apoyo durante la fase de modelado, proporcionando una visión clara del comportamiento de las variables y facilitando la toma de decisiones respecto al preprocesamiento y selección de atributos.

7.5. Modelo de Machine Learning

El módulo correspondiente al modelo de aprendizaje automático implementa un clasificador RF, seleccionado por su capacidad para capturar relaciones no lineales entre variables, su tolerancia al ruido y su buen equilibrio entre rendimiento e interpretabilidad.

El flujo de ejecución comienza con la carga y el preprocesamiento de los datos, que incluye la anonimización mediante la eliminación de identificadores, la normalización de variables numéricas y la partición en conjuntos de entrenamiento y prueba. Para corregir el desbalance de clases característico en problemas de detección de fraude, se emplea la técnica SMOTE, que genera instancias sintéticas de la clase minoritaria a partir de observaciones reales cercanas.

Una vez equilibrado el conjunto de entrenamiento, se procede a entrenar el modelo RF utilizando hiperparámetros previamente ajustados, como el número de árboles y la profundidad máxima. El rendimiento del modelo se evalúa mediante métricas estándar como precisión, Recall, F1-Score, especificidad y matriz de confusión, lo que permite una evaluación detallada de su comportamiento ante ambas clases.

Finalmente, el modelo entrenado se almacena de forma persistente para su posterior uso en la aplicación móvil y en entornos de validación.

Los aspectos técnicos detallados de esta implementación se desarrollan en el Anexo 14.

7.6. Archivo de configuración

El fichero entorno.env contiene las variables de entorno y parámetros de configuración necesarios para la correcta ejecución del proyecto. Este archivo centraliza información sensible y rutas clave, facilitando la gestión segura y modular de la configuración sin necesidad de codificar datos directamente en el código fuente.

Entre las configuraciones destacadas, se incluyen:

- Variables que almacenan el host, usuario, contraseña y nombre de la base de datos. Esto permite conectar la aplicación a la base de datos relacional de forma dinámica y segura.
- Credenciales y URLs para servicios externos: Claves y enlaces de acceso a Firebase, una plataforma de backend en la nube, que facilitan la interacción con bases de datos en tiempo real y otros servicios asociados. También se incluye una clave API para el acceso a modelos GPT, permitiendo la integración de funcionalidades de procesamiento de lenguaje natural.
- Rutas a archivos locales: Se definen las ubicaciones relativas de archivos esenciales para el proyecto, como bases de datos en formato Comma-Separated Values, formato de archivo utilizado para representar datos tabulares mediante texto plano con valores separados por comas (CSV) (por ejemplo, datos originales, preprocesados, cifrados) y modelos entrenados (archivos `.pkl` para el modelo y el escalador). Esto asegura que el código pueda acceder y manipular estos recursos sin necesidad de hardcodear las rutas.
- Variables numéricas que controlan aspectos del procesamiento de datos, como proporciones de fraude en los conjuntos de datos o tamaños de muestras, que permiten ajustar el comportamiento del sistema de manera flexible.

El uso de un archivo .env para almacenar estas variables proporciona una capa de abstracción y seguridad, ya que evita la exposición directa de credenciales y facilita el mantenimiento y la portabilidad del proyecto en diferentes entornos de ejecución.

7.7. Base de datos en Firebase

El diseño de la base de datos en Firebase se modela mediante un diagrama de clases que refleja las entidades principales y sus relaciones, optimizando la organización y gestión de la información para la aplicación.

A continuación, se describen las clases y atributos más relevantes:

- Categoria: representa las categorías de productos con atributos como el identificador único, nombre de la categoría y URL de la imagen asociada.
- Producto: contiene detalles del producto, incluyendo id, nombre, descripción, nota descriptiva, categoría, precio original y con descuento, y un mapa de imágenes relacionadas.
- Imagen: almacena la información de las imágenes vinculadas a los productos, principalmente su identificador y URL.
- Usuario: incluye datos personales y de la autenticación uid, nombres, DNI, email, teléfono, dirección, tipo de usuario, imagen de perfil, ubicación geográfica, verificación y fecha de registro). Además, contiene colecciones como carrito de compras, favoritos y tiempos asociados.
- ProductoCarrito: representa productos en el carrito de un usuario, con cantidad y precios (original, descuento y final).
- Favorito: referencia productos marcados como favoritos por un usuario.
- Tiempo: registra datos temporales vinculados a las compras o actividades del usuario.
- Orden: detalla una orden de compra, con atributos como identificador, coste, fecha, hora, tipo y forma de envío, método de pago, indicadores de fraude y revisión, texto de devolución, usuario y productos asociados.
- UsuarioOrden y ProductoOrden: clases auxiliares que almacenan información específica para cada orden, garantizando integridad y trazabilidad histórica.

Las relaciones principales entre las entidades son:

- Un usuario puede realizar múltiples pedidos.
- Cada pedido contiene uno o varios productoOrden.
- Un producto puede tener varias imágenes.
- El usuario posee colecciones de productoCarrito, favoritos y tiempos.
- Un producto puede estar marcado como favorito por varios usuarios.

Esta estructura modular y relacional facilita la gestión eficiente y segura de los datos en Firebase, soportando las funcionalidades clave de la aplicación y garantizando escalabilidad y mantenimiento.

A continuación muestro el diagrama de clases de esta estructura explicada previamente:

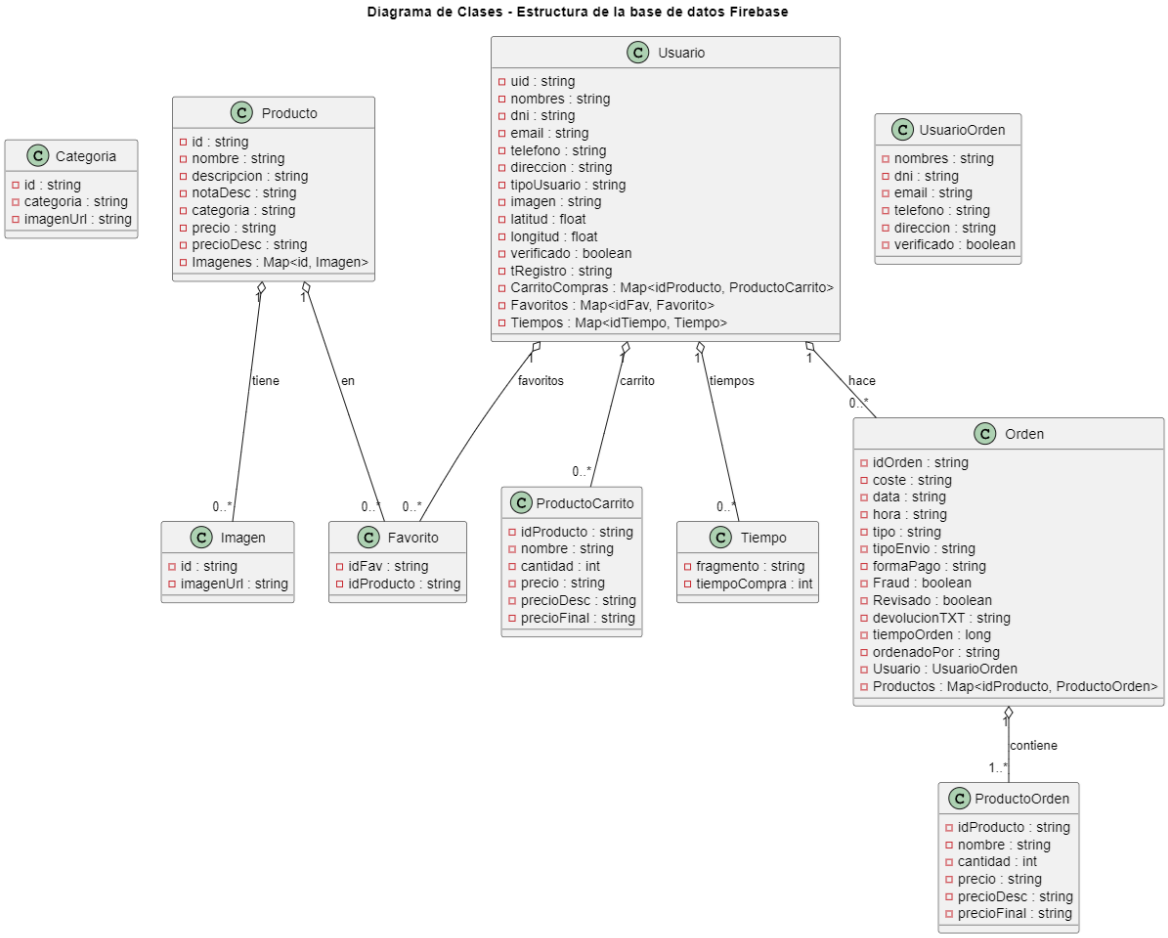

Figura 8: Diagrama de clases que representa la estructura de la base de datos online.

8. Evaluación

La evaluación del sistema desarrollado constituye una etapa esencial para verificar su eficacia, robustez y viabilidad en un contexto real de detección de fraude. En esta sección se analizan en detalle los resultados obtenidos tras el entrenamiento y validación del modelo de aprendizaje automático, así como los costes asociados a su implementación, tanto en términos computacionales como temporales.

Dado que el problema abordado implica un marcado desbalance entre clases, donde las transacciones fraudulentas representan una minoría significativa, la simple tasa de aciertos globales no basta para determinar la calidad del sistema. Por ello, se han empleado métricas especializadas que permiten valorar su rendimiento desde una perspectiva más crítica y ajustada a la naturaleza del problema.

8.1. Resultados

Para evaluar la calidad del modelo desarrollado en el presente trabajo, se han empleado métricas ampliamente utilizadas en el ámbito del aprendizaje automático, especialmente relevantes en escenarios con clases desbalanceadas, como ocurre en la detección de fraude.

Entre estas métricas, el F1-Score ocupa un lugar destacado, ya que combina de forma equilibrada dos aspectos fundamentales: la precisión y la exhaustividad también conocida como Recall. A través de la media armónica entre ambos, el F1-Score permite evaluar el rendimiento del modelo teniendo en cuenta tanto su capacidad para identificar correctamente los fraudes como su habilidad para evitar falsas alarmas. Esta métrica resulta especialmente útil en contextos donde los casos positivos —en este caso, los fraudes— son considerablemente menos frecuentes que las transacciones legítimas. El F1-Score toma valores entre 0 y 1, donde 1 representa un modelo ideal que alcanza el equilibrio perfecto entre precisión y Recall, y valores más bajos indican un rendimiento deficiente. En este caso, el modelo alcanzó un F1-Score de 0.8620.

Junto con el F1-Score, también se ha considerado la precisión global, o Accuracy, que mide la proporción total de predicciones correctas realizadas por el modelo respecto al total de casos evaluados. Aunque es una métrica fácil de interpretar, su utilidad en problemas de clases desbalanceadas es limitada, ya que un modelo puede obtener una alta precisión simplemente prediciendo la clase mayoritaria, sin aportar valor real en la detección de los casos de fraude. Por tanto, si bien se incluye como una métrica de referencia general, no constituye por sí sola un indicador fiable del rendimiento en este contexto específico. El modelo obtuvo una Accuracy de 0.8673.

Además, se ha hecho uso de la matriz de confusión, una herramienta que permite analizar en detalle los aciertos y errores del modelo al comparar las clases predichas con las clases reales. Esta matriz refleja cuántos casos fueron correctamente identificados como fraude (verdaderos positivos), cuántos fraudes no fueron detectados (falsos negativos), cuántas transacciones legítimas se clasificaron erróneamente como fraude (falsos positivos) y cuántas se identificaron correctamente como no fraude (verdaderos negativos). En este caso, los resultados fueron los siguientes:

Cuadro 3: Matriz de confusión del modelo de detección de fraude

	Predicho: Fraude	**Predicho: No Fraude**
Real: Fraude	52 923 (TP)	23 826 (FN)
Real: No Fraude	2 707 (FP)	120 544 (TN)

A partir de esta información, se derivan otras métricas relevantes:

- Precisión: 0.9513.
- Recall: 0.6896.
- Especificidad: 0.9780.

En el ámbito de la detección de fraude, se otorga especial importancia a la minimización de los falsos negativos, ya que cada uno de ellos representa un fraude no detectado, lo que puede traducirse en pérdidas económicas o daños a la reputación. Por otro lado, los falsos positivos, aunque menos graves, también deben ser controlados, ya que pueden generar desconfianza en los usuarios o costos operativos derivados de la investigación de transacciones legítimas clasificadas como sospechosas.

En conjunto, la combinación de estas métricas ofrece una visión integral del rendimiento del modelo, permitiendo valorar no solo su capacidad para detectar fraudes de forma efectiva, sino también su precisión al distinguir entre transacciones fraudulentas y legítimas. Esta evaluación rigurosa resulta fundamental para asegurar que el sistema de detección desarrollado sea eficaz y viable en escenarios reales.

8.2. Costes

La funcionalidad del proyecto es eminentemente práctica, centrada en la detección de fraudes y su representación gráfica. Para su ejecución, se ha considerado un coste total correspondiente a un equipo de programadores backend y frontend. La idea planteada es diseñar desde cero la infraestructura necesaria tanto para una aplicación móvil como para una plataforma web, permitiendo así la recepción, tratamiento y análisis de los datos utilizados en la detección de fraudes mediante técnicas de ML.

En cuanto a los recursos gráficos, se han utilizado diseños generados por Inteligencia Artificial, los cuales no están sujetos a derechos de autor. Por tanto, su coste ha sido nulo. No obstante, se debe señalar que en un entorno empresarial real, la creación y edición de imágenes de productos conllevaría un coste adicional asociado a tareas de diseño y producción gráfica.

Respecto a las licencias de software, se ha trabajado exclusivamente con programas de uso gratuito. El único coste potencial identificado es el correspondiente a los servicios de almacenamiento en la nube. Para este proyecto, se ha utilizado una versión gratuita con limitaciones de capacidad, aunque para una implementación a gran escala se recomienda la contratación de la versión de pago (en este caso, el plan *Blaze*), más adecuada para manejar volúmenes elevados de datos.

En cuanto al hardware, el desarrollo del proyecto se ha realizado en un ordenador personal equipado con una tarjeta gráfica de rendimiento competente dentro del mercado actual. El software que presenta mayores requisitos técnicos es Android Studio. Además, para la construcción del modelo de RF, la potencia de la GPU influye notablemente en el tiempo de entrenamiento: a mayor capacidad del equipo, menor será el tiempo de procesamiento. Sin embargo, para la ejecución periódica del sistema de detección, no se requiere un hardware de altas prestaciones.

Costes temporales

Durante el desarrollo del sistema, se registraron diversos costes temporales asociados a las fases más relevantes del proceso. La generación de la base de datos de entrenamiento, compuesta por un millón de registros sintéticos, requirió aproximadamente 21 minutos, lo que refleja una eficiencia considerable dada la magnitud del volumen de datos generado.

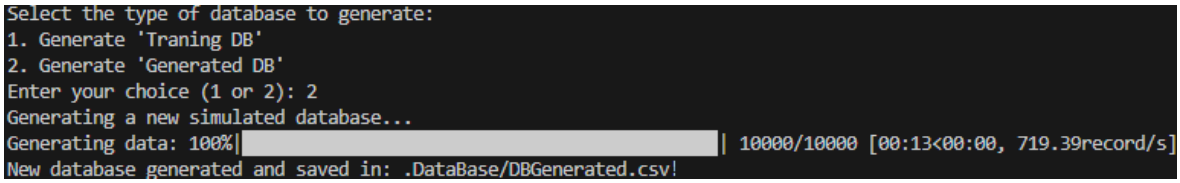

Figura 9: Tiempo de ejecución para la generación de la base de datos sintética completa.

Para facilitar pruebas más rápidas, se creó además una versión reducida de la base de datos con 10.000 registros, cuya generación tomó apenas 13 segundos.

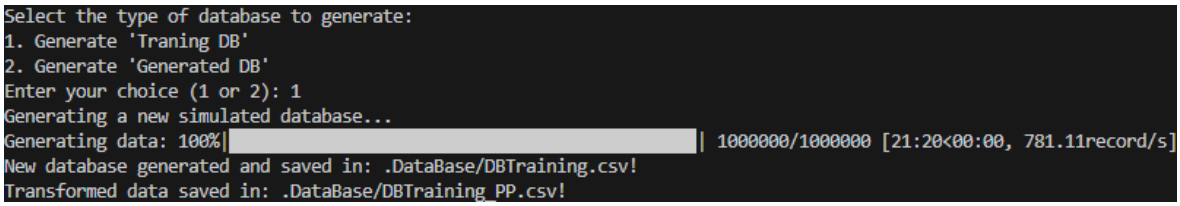

Figura 10: Tiempo de ejecución para la generación de la base de datos reducida para entrenamiento.

El análisis exploratorio de datos (EDA) se completó en alrededor de 21 segundos, beneficiándose de la buena estructura y calidad de los datos empleados.

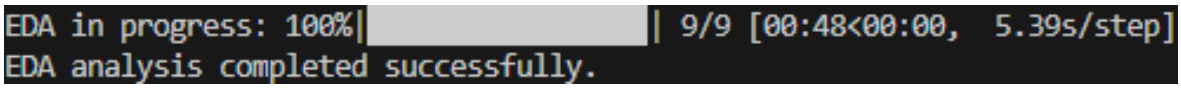

Figura 11: Resultado de la ejecución del análisis exploratorio de datos (EDA).

En cuanto al entrenamiento y evaluación del modelo de ML, específicamente un clasificador basado en RF, la duración total fue de aproximadamente 75 minutos. Este tiempo estuvo condicionado principalmente por la capacidad de procesamiento del equipo utilizado y la complejidad inherente al modelo.

Estos tiempos, obtenidos bajo condiciones controladas de desarrollo, indican un rendimiento adecuado que permite considerar viable la implementación de la solución tanto en entornos de prueba como en despliegues iniciales. No obstante, es importante destacar que dichos tiempos pueden variar en función del hardware disponible y del volumen de datos a procesar en escenarios reales.

Es destacable que una vez creado el RF, el tiempo de ejecución del algoritmo es ínfimo, ahí es donde se puede considerar que el programa funciona tal como se necesita, ya que el entrenamiento se realiza solo una vez, mientras que el uso de este es periódico.

8.3. Análisis Exploratorio de Datos

El módulo Eda.py ejecuta el análisis exploratorio de datos EDA sobre el conjunto de datos empleado para entrenar los modelos de detección de fraude. Su propósito principal es identificar patrones generales de comportamiento, detectar anomalías y facilitar decisiones en las etapas de preprocesamiento y selección de atributos.

Detección de Valores Atípicos con Boxplots

El análisis de valores atípicos es esencial para identificar comportamientos que se desvían del patrón general y que, en el contexto de la detección de fraude, pueden indicar transacciones sospechosas o entradas sintéticas no realistas. A continuación, se presentan los resultados del análisis univariante de cuatro variables numéricas clave, mediante diagramas de caja Boxplot.

COMPRA_MEDIA_USUARIO Esta variable representa el gasto medio por usuario en sus transacciones. El Boxplot revela una distribución fuertemente asimétrica, con múltiples valores atípicos por encima del límite superior. Estos casos podrían estar relacionados con usuarios fraudulentos que realizan una o pocas compras de importe elevado, generando un comportamiento anómalo respecto a la media poblacional.

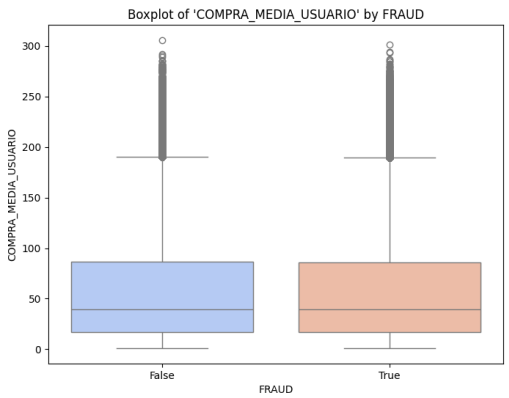

Figura 12: Boxplot de COMPRA_MEDIA_USUARIO.

DNI Aunque se espera que los DNIs sean únicos y homogéneos, el análisis muestra valores fuera del rango típico y una variabilidad no esperada. Esto sugiere la existencia de datos generados automáticamente o malformaciones intencionadas, que podrían estar asociadas a intentos de eludir controles de identidad.

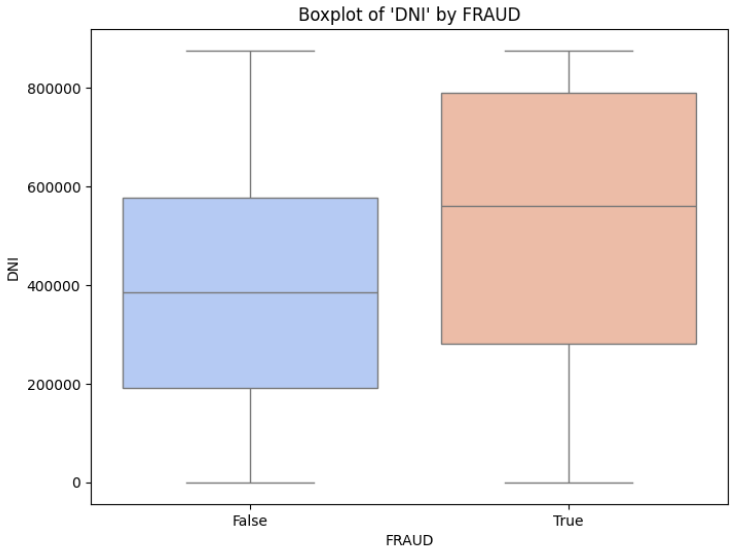

Figura 13: Boxplot sobre DNI.

IP La variable correspondiente a la dirección IP revela una concentración anómala de valores en determinadas franjas. Algunos registros presentan una repetición excesiva de IPs, lo que podría deberse a la utilización de redes privadas virtuales (VPN), proxies o simplemente a múltiples cuentas asociadas a un mismo origen, situación común en patrones de fraude automatizado.

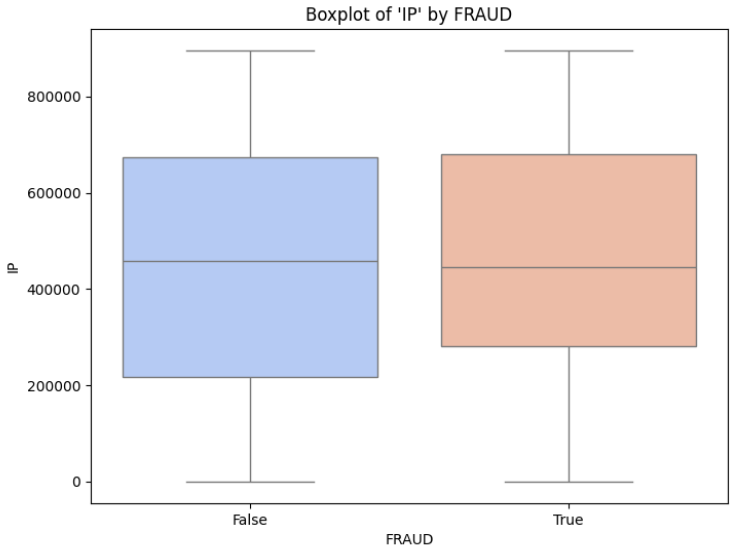

Figura 14: Boxplot de IP.

PRECIO_TOTAL El precio total de cada transacción presenta, como era de esperar, una distribución con muchos valores bajos, pero también un conjunto significativo de valores extremos muy altos. Estos outliers podrían corresponder a intentos de maximizar el beneficio en operaciones fraudulentas, especialmente si están asociados a cuentas nuevas o no verificadas.

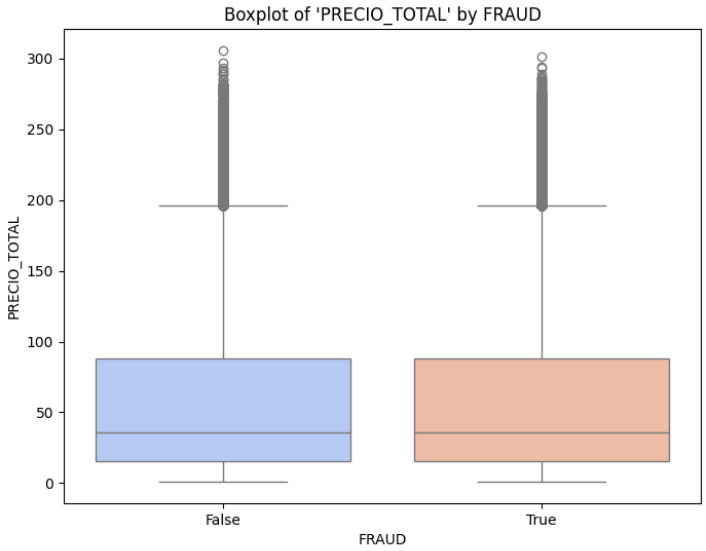

Figura 15: Boxplot de PRECIO_TOTAL.

Distribución de Variables Numéricas

Se ha realizado un análisis exploratorio de las principales variables numéricas con el objetivo de comprender su comportamiento y detectar posibles anomalías o patrones de interés:

- PRECIO_TOTAL y NUM_PRODUCTS presentan una distribución sesgada a la derecha, lo que indica que la mayoría de las compras involucran montos o cantidades bajas, con unos pocos casos de valores muy elevados que podrían corresponder a operaciones inusuales.
- TIEMPO, NUM_INTENTOS y NUM_COMPRAS muestran también una fuerte concentración en valores bajos, con colas largas que podrían señalar conductas anómalas como múltiples intentos de compra o sesiones excesivamente largas.
- CLICKS revela una interacción desigual: una gran cantidad de usuarios realiza pocos clics, mientras que otros tienen valores extremos, lo que podría apuntar a automatismos o scripts.
- DESCUENTO se agrupa en valores discretos, destacando el valor 5 como el más frecuente, posiblemente por campañas promocionales específicas.
- COMENTARIO muestra una mayoría de valores bajos o ausentes, pero con un subconjunto que introduce muchos caracteres, lo que podría estar relacionado con actividad automatizada o abusiva.

- TIPO, VERIFICADO y ENVIO_EN_CASA son variables binarias, siendo notorio el desbalance hacia una de las clases, lo cual podría tener implicaciones en modelos predictivos al introducir sesgos.
- CORREO, TELEFONO e IP presentan distribuciones relativamente uniformes, dado que son identificadores únicos o casi únicos, útiles para detectar duplicados, patrones de comportamiento repetitivo o suplantación.

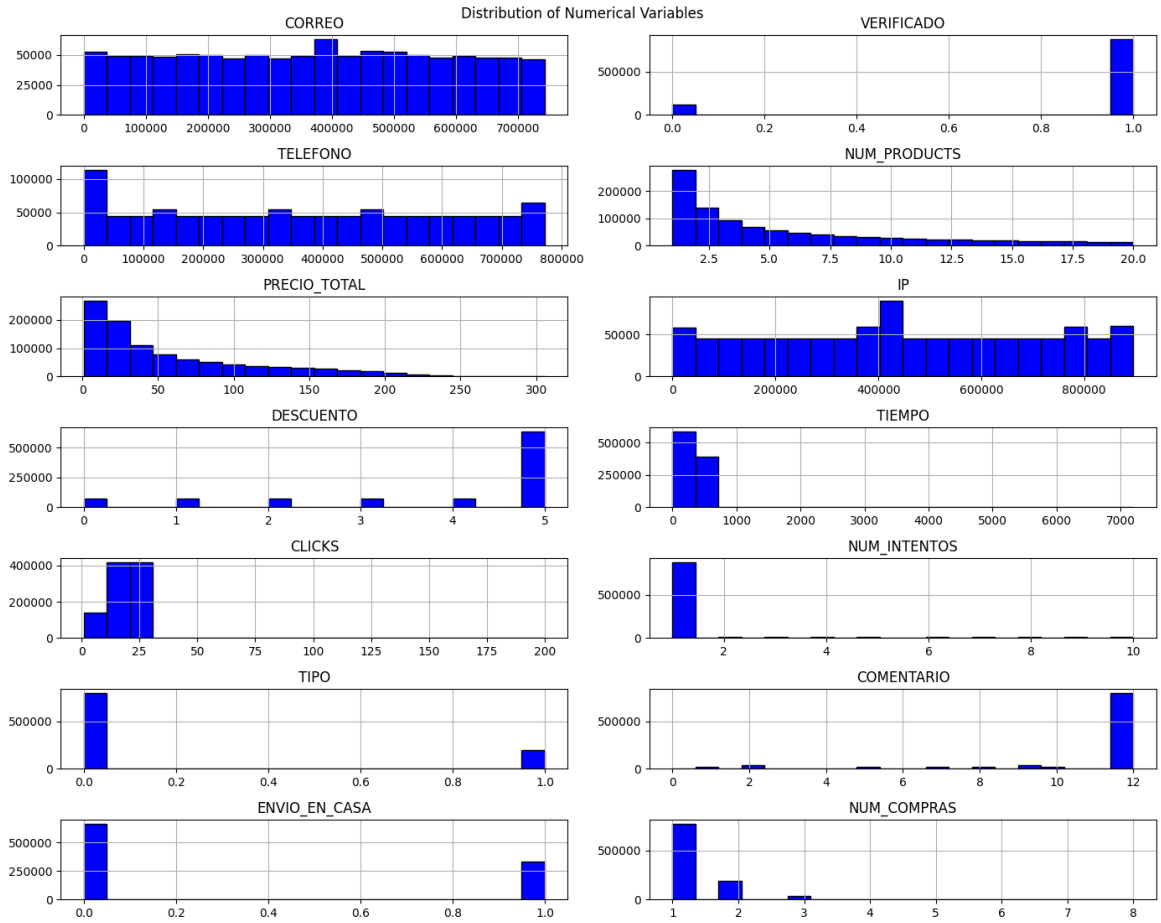

Figura 16: Distribución de las principales variables numéricas del conjunto de datos.

Número de Compras por Usuario

La mayoría de los usuarios realiza una única compra, lo que sugiere una base de usuarios esporádicos o temporales. Este patrón también puede reflejar cuentas creadas con propósitos específicos, como aprovechar descuentos únicos o realizar fraudes controlados.

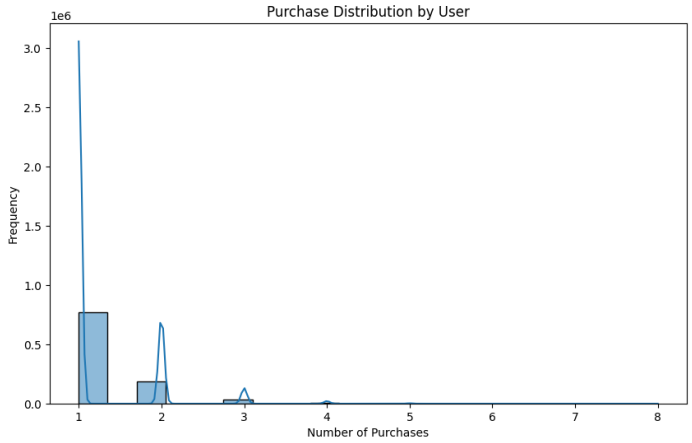

Figura 17: Diagrama de barras: Número de compras por usuario

Número de Productos por Transacción

Las transacciones con un solo producto son las más frecuentes. A medida que se incrementa el número de productos, la frecuencia disminuye, aunque los casos con muchos productos pueden estar vinculados a intentos de fraude de mayor escala.

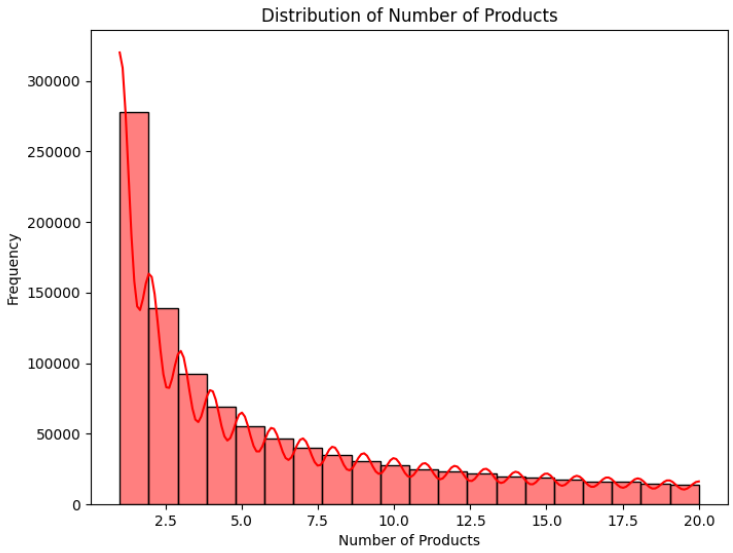

Figura 18: Diagrama de barras: Número de productos por transacción

9. Legislación y protección de datos

En el contexto de las compras online, el tratamiento de datos personales es una preocupación fundamental debido a las implicaciones que tiene en la privacidad y seguridad de los usuarios. Durante el desarrollo de este TFG, se han utilizado datos personales necesarios para simular el proceso de compras online, como el nombre, correo electrónico, dirección de envío, entre otros. Sin embargo, es fundamental asegurar que estos datos sean manejados de acuerdo con la legislación vigente en materia de protección de datos personales.

La normativa más relevante para el tratamiento de datos personales en la Unión Europea es el *Reglamento General de Protección de Datos* (Reglamento General de Protección de Datos), que establece las directrices y requisitos que deben cumplir las organizaciones al gestionar información personal. El Reglamento General de Protección de Datos exige que los datos personales se recojan de manera lícita, transparente y adecuada, con fines específicos y limitados Unión Europea, 2016. Además, impone la obligación de garantizar la seguridad de los datos mediante medidas técnicas y organizativas adecuadas, a fin de evitar su pérdida, acceso no autorizado o tratamiento ilegal Agencia Española de Protección de Datos, 2020.

En este trabajo, el uso de datos personales se ha limitado a los estrictamente necesarios para simular transacciones de compra y procesar la detección de fraudes. Para ello, se ha hecho uso de una Base de Datos Artificial creada específicamente para este propósito, lo que garantiza que los datos utilizados no correspondan a personas reales y no representen un riesgo para la privacidad. Esta base de datos ha sido diseñada para cumplir con los principios de Minimización de Datos y Anonimización, evitando la recopilación y el almacenamiento innecesarios de información sensible Agencia Española de Protección de Datos, 2021. Datos con un alto riesgo de seguridad, los asociados con la entidad bancaria, han sido obviados, considerando que estos no son relevantes al objetivo del trabajo.

Asimismo, el tratamiento de estos datos ha seguido los principios establecidos en el Reglamento General de Protección de Datos, garantizando que se realiza con el consentimiento explícito del usuario, que se le ha informado de su derecho a acceder, rectificar y eliminar sus datos, y que se han adoptado las medidas necesarias para proteger su seguridad European Data Protection Board, 2022. No obstante, cabe resaltar que el propósito de este trabajo es exclusivamente académico y de investigación, por lo que no se realiza ningún uso comercial de los datos ni se almacenan de manera prolongada.

En el entorno de almacenamiento de los datos, Firebase incorpora un sistema de protección de datos propio, utilizando reglas de acceso que limitan el acceso según el registro y rol asignado Firebase, 2024.

Es importante destacar que, en un entorno real, las empresas que gestionan datos de clientes deben cumplir con los requisitos de protección de datos, como la designación de un Delegado de Protección de Datos, la implementación de políticas de privacidad claras y la notificación de brechas de seguridad a las autoridades competentes Agencia Española de Protección de Datos, 2020. La correcta gestión de los datos personales es esencial para garantizar la confianza de los usuarios y cumplir con la legislación aplicable.

10. Implicaciones éticas

El presente proyecto, enfocado en el desarrollo de un sistema de detección de fraude mediante técnicas de aprendizaje automático y su implementación en una aplicación Android, ha sido concebido integrando principios éticos y de responsabilidad social, fundamentales para el ejercicio profesional en ingeniería. A continuación, se analizan las implicaciones del proyecto en relación con los cuatro indicadores de la competencia CT7: igualdad, medio ambiente, responsabilidad social y ética.

10.1. Igualdad y no discriminación

La igualdad y la no discriminación constituyen principios esenciales en el diseño de sistemas automatizados. Aunque los datos utilizados para el entrenamiento del modelo son generados sintéticamente, se ha puesto especial atención en evitar sesgos que puedan reproducir discriminaciones injustificadas hacia determinados perfiles de usuario. En un entorno real, resultaría imprescindible realizar auditorías éticas periódicas y aplicar métricas de equidad (Fairness) para garantizar que variables sensibles como género, edad o nacionalidad no influyan de manera directa o indirecta en la detección de fraude.

Asimismo, el diseño de la interfaz y la lógica del sistema se ha orientado a evitar favorecer o perjudicar a colectivos específicos, promoviendo así la accesibilidad y la inclusión. En futuras etapas del proyecto, se recomienda profundizar en la integración de metodologías de auditoría ética para evaluar de forma sistemática el impacto social del algoritmo.

10.2. Impacto medioambiental

El desarrollo tecnológico debe orientarse hacia la sostenibilidad ambiental. En este proyecto, se ha adoptado un enfoque que minimiza el consumo energético y los recursos computacionales empleados. La generación de datos sintéticos y el entrenamiento del modelo se han realizado en entornos locales, evitando la dependencia de infraestructuras masivas en la nube con altos consumos energéticos.

Además, la modularidad del código y la reutilización de componentes fomentan buenas prácticas de programación eficiente y sostenible, reduciendo la duplicidad de procesos. El sistema ha sido diseñado para su posible ejecución en dispositivos móviles de bajo consumo energético, como smartphones Android, lo que contribuye a reducir su huella ecológica (Foundation, 2024).

10.3. Responsabilidad social en el desarrollo tecnológico

Este proyecto reconoce la responsabilidad social inherente al desarrollo tecnológico, considerando que la ingeniería debe contribuir a construir una sociedad más justa, inclusiva y sostenible. Por ello, todas las decisiones técnicas han sido acompañadas de una reflexión ética que valora no solo la funcionalidad, sino también el impacto social y ético del sistema.

Se ha puesto especial énfasis en la protección de la privacidad y la gestión responsable de los datos personales, aspectos cruciales en sistemas de detección de fraude que manejan

información sensible. Asimismo, el proyecto se alinea con los Objetivos de Desarrollo Sostenible (ODS), en particular con el ODS 10, relativo a la reducción de desigualdades, y el ODS 12, que promueve patrones de producción y consumo responsables (Organización de las Naciones Unidas, 2023).

10.4. Ética profesional y deontología

Como futuro profesional de la ingeniería, se ha considerado imprescindible actuar conforme a los principios éticos y deontológicos propios del área de conocimiento. Esto implica reconocer las consecuencias de las decisiones técnicas y mostrar capacidad crítica y diálogo para asegurar un uso responsable de las normas que afectan al desarrollo y despliegue del sistema.

En este sentido, el proyecto contempla la importancia de la transparencia, la explicabilidad del modelo y la trazabilidad de las decisiones automatizadas para favorecer la confianza y la rendición de cuentas. Se recomienda continuar profundizando en estos aspectos en futuras iteraciones, incorporando mecanismos que permitan evaluar y mitigar riesgos éticos derivados del uso del sistema.

11. Uso responsable de la inteligencia artificial

Durante el desarrollo de este Trabajo de Fin de Grado se ha hecho un uso consciente y responsable de diversas herramientas de inteligencia artificial generativa, valorando su potencial como apoyo en tareas técnicas y creativas, pero sin delegar en ellas la responsabilidad intelectual ni autoral del proyecto.

Estas herramientas se utilizaron como complemento a los conocimientos del autor, con el objetivo de mejorar la eficiencia, la calidad de la documentación técnica y la resolución de dudas puntuales. En ningún caso se ha empleado su salida de forma directa o acrítica, sino que todas las respuestas, fragmentos de código y textos generados fueron cuidadosamente revisados, validados y adaptados por el propio estudiante, garantizando la integridad académica y la autoría del trabajo.

En concreto, se emplearon las siguientes herramientas:

- *ChatGPT* (OpenAI, 2025), utilizado como asistente conversacional para la obtención de explicaciones técnicas, propuestas de redacción, análisis de código y aclaración de conceptos teóricos en el ámbito del aprendizaje automático, la ingeniería del software y el tratamiento de datos.
- *GitHub Copilot* (GitHub, 2025), empleado en la realización de las imágenes que simulan los productos y categorías dentro de la aplicación.
- *NotebookLM* (Google, 2025), empleado como herramienta de organización de notas y materiales, permitiendo sintetizar y estructurar documentos extensos mediante resúmenes generados a partir de contenidos previamente redactados o recopilados por el autor.

El uso de estas herramientas se enmarca dentro de un enfoque ético de la inteligencia artificial, que reconoce sus beneficios sin ignorar sus limitaciones. Se ha procurado evitar sesgos en las respuestas generadas, validar cada fuente de información y garantizar que el resultado final del trabajo refleje la comprensión, el criterio y la responsabilidad personal del autor.

Este enfoque promueve una actitud crítica y reflexiva hacia el uso de la tecnología, y refuerza el papel del estudiante como agente activo en el proceso de aprendizaje y producción académica, conforme a los principios de honestidad, transparencia y responsabilidad intelectual.

12. Valoración

La realización de este Trabajo de Fin de Grado ha permitido alcanzar de forma satisfactoria los objetivos propuestos, dando lugar a una solución funcional, eficaz y técnicamente coherente para la detección de fraudes en entornos simulados. El desarrollo del sistema ha supuesto una oportunidad valiosa para aplicar, consolidar y ampliar los conocimientos adquiridos durante el grado, integrando aspectos clave de la ingeniería informática como el tratamiento de datos, el aprendizaje automático y el desarrollo de aplicaciones móviles.

El análisis y procesamiento de datos ha demostrado ser un eje fundamental en la construcción de soluciones inteligentes, y la aplicación de técnicas de ML ha evidenciado su potencial para optimizar procesos, identificar patrones anómalos y ofrecer herramientas de apoyo a la toma de decisiones en contextos empresariales. En este sentido, la implementación del modelo predictivo ha contribuido significativamente a comprender los desafíos prácticos asociados al trabajo con datos desbalanceados, la evaluación de modelos y la mejora progresiva mediante validaciones sistemáticas.

Uno de los desafíos más relevantes ha sido el diseño e implementación de la aplicación Android, un componente que inicialmente quedaba fuera de mis áreas de mayor afinidad. No obstante, su desarrollo ha representado una oportunidad de aprendizaje que ha ampliado el alcance del proyecto, dotándolo de una interfaz visual que complementa y refuerza la utilidad práctica del sistema propuesto.

Además, la integración entre el modelo de detección, el backend online y la aplicación móvil ha permitido lograr un flujo de trabajo autónomo, capaz de operar de forma periódica y ofrecer una experiencia de usuario completa. Este nivel de integración ha superado las expectativas iniciales de diseño, ofreciendo una visión más cercana a una solución real desplegable.

Pese a los resultados obtenidos, es importante destacar que el sistema es aún susceptible de mejora. Tal como se detalla en el apartado correspondiente, existen aspectos clave que podrían optimizarse en futuras iteraciones, como la diversificación de fuentes de datos, la mejora de la arquitectura backend o la ampliación de funcionalidades orientadas a un entorno productivo. Con una mayor experiencia y profundización técnica, se considera viable evolucionar esta solución hacia un sistema más robusto, escalable y adaptable a escenarios reales de mayor complejidad.

13. Mejoras Propuestas

El presente trabajo ha tenido como objetivo principal la implementación de un sistema de detección de fraudes que combine precisión, eficiencia y aplicabilidad en un entorno simulado. Si bien los resultados obtenidos han sido satisfactorios, se han identificado diversas áreas susceptibles de mejora que podrían incrementar la robustez, escalabilidad y calidad del sistema en futuras versiones.

En cuanto a la generación de la base de datos, si bien se ha empleado un sistema lógico funcional que proporciona variabilidad suficiente para realizar análisis representativos, sería deseable enriquecer el proceso de creación de datos sintéticos. La incorporación de variables más complejas podría aportar una mayor profundidad analítica y mejorar la capacidad predictiva del modelo. Estas ideas están en línea con las recomendaciones de Phua et al., 2010, donde se enfatiza la importancia de representar el comportamiento real de los usuarios para mejorar la detección de anomalías.

Respecto a la optimización del modelo RF, se ha realizado una comparación entre configuraciones básicas para seleccionar aquella que ofrecía un buen rendimiento. No obstante, el proceso podría mejorarse mediante técnicas de búsqueda sistemática de hiperparámetros como *Grid Search* o *Random Search*, que permitirían explorar un abanico más amplio de combinaciones y obtener modelos más ajustados a los datos. Complementar esta optimización con validación cruzada reforzaría la fiabilidad de las métricas obtenidas, como se sugiere en Géron, 2019, donde se describe en detalle el proceso de ajuste y validación de modelos de aprendizaje supervisado.

Otro aspecto destacable es la adquisición de datos. En esta versión del sistema, toda la información ha sido generada desde código Python y de la propia aplicación móvil, lo que simplifica el desarrollo, pero limita la escalabilidad y diversidad del sistema. Una posible mejora sería permitir la integración con fuentes externas, como APIs, plataformas web o bases de datos empresariales. Este tipo de integración es especialmente relevante en aplicaciones reales de detección de fraude, tal como exponen Kou et al., 2004 y Bolton y Hand, 2002 en sus respectivos estudios.

En relación con la aplicación Android, si bien cumple adecuadamente su propósito como entorno funcional de prueba, sería necesario mejorar la interfaz gráfica y la experiencia de usuario en caso de plantearse una evolución comercial del proyecto. Esto incluiría un diseño más atractivo y profesional, mejoras en la navegación, accesibilidad y usabilidad general, así como la incorporación de funcionalidades adicionales como visualización de estadísticas o paneles de gestión. En proyectos de base tecnológica, la experiencia de usuario es un factor determinante para la adopción, y debe considerarse al mismo nivel que el rendimiento técnico.

Finalmente, aunque Firebase ha resultado una solución adecuada como plataforma backend gracias a su facilidad de uso y sincronización en tiempo real, es recomendable evaluar otras alternativas más adaptables a diferentes escenarios productivos. La migración hacia soluciones como bases de datos relacionales, sistemas NoSQL más flexibles o servicios personalizados en la nube permitiría optimizar el sistema en función de los requisitos específicos de rendimiento, seguridad o compatibilidad con infraestructuras empresariales existentes. Esta reflexión se alinea con los principios de escalabilidad y sostenibilidad recomendados en desarrollos modernos de software orientado a datos Foundation, 2024.

Referencias Bibliográficas

Agencia Española de Protección de Datos. (2020). Guía sobre el Reglamento General de Protección de Datos (RGPD). https://www.aepd.es/sites/default/files/2020-05/rgpd-guia.pdf.

Agencia Española de Protección de Datos. (2021). Principios de minimización y anonimización de datos personales. https://www.aepd.es/es/guias-y-herramientas/guias/anonimizacion.

Bolton, R. J., & Hand, D. J. (2002). *Statistical Fraud Detection: A Review* (Vol. 17). Statistical Science.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5-32.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321-357.

European Data Protection Board. (2022). Guidelines on Consent under Regulation 2016/679. https://edpb.europa.eu/our-work-tools/our-documents/guidelines/guidelines-052020-consent-under-regulation-2016679_en.

Firebase, G. (2024). Firebase Documentation. https://firebase.google.com/docs.

Foundation, G. S. (2024). *Principles of Sustainable Software Engineering*. https://greensoftware.foundation/principles/.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2nd). O'Reilly Media, Inc.

GitHub. (2025). GitHub Copilot [Asistente de programación basado en IA]. https://github.com/features/copilot.

Google. (2025). NotebookLM [Asistente de lectura basado en IA]. https://notebooklm.google.

Kou, Y., Lu, C.-T., Sirwongwattana, S., & Huang, Y. (2004). Survey of fraud detection techniques. *IEEE International Conference on Networking, Sensing and Control, 2004*, 749-754.

OpenAI. (2025). ChatGPT (versión GPT-4) [Modelo de lenguaje grande]. https://chat.openai.com.

Organización de las Naciones Unidas. (2023). *Objetivos de Desarrollo Sostenible (ODS)*. https://www.un.org/sustainabledevelopment/es/.

Phua, C., Lee, V., Smith, K., & Gayler, R. (2010). A Comprehensive Survey of Data Mining-based Fraud Detection Research. *arXiv preprint arXiv:1009.6119*.

Powers, D. M. W. (2011). Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. *Journal of Machine Learning Technologies*, *2*(1), 37-63.

Unión Europea. (2016). Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016. https://eur-lex.europa.eu/legal-content/ES/TXT/?uri=CELEX%3A32016R0679.

Acrónimos

CSV Comma-Separated Values, formato de archivo utilizado para representar datos tabulares mediante texto plano con valores separados por comas. 26

EDA Exploratory Data Analysis, proceso de análisis preliminar de los datos que permite comprender su estructura, detectar patrones, valores atípicos y guiar decisiones de modelado. 5, 7, 25, 31, 32

ML Machine Learning, subcampo de la inteligencia artificial que se centra en desarrollar algoritmos capaces de aprender patrones a partir de datos y realizar predicciones o decisiones. 5, 6, 8, 9, 11, 12, 20, 25, 30, 31, 40, 46, 49

RF Random Forest, algoritmo de aprendizaje supervisado basado en la combinación de múltiples árboles de decisión para mejorar la precisión y reducir el sobreajuste. 5, 9, 10, 20, 25, 26, 31, 32, 41, 49

SMOTE Synthetic Minority Over-sampling Technique, técnica de sobremuestreo utilizada para equilibrar clases en datasets desbalanceados generando nuevas instancias sintéticas de la clase minoritaria. 5, 9, 20, 25, 46

SVM Support Vector Machine, algoritmo de aprendizaje supervisado utilizado para clasificación y regresión, que busca encontrar el hiperplano que mejor separa las clases en el espacio de características. 20

Términos

Accuracy Métrica de evaluación que mide la proporción total de predicciones correctas realizadas por el modelo respecto al número total de instancias evaluadas. 29

Anonimización Proceso mediante el cual los datos personales se modifican de forma que no puedan ser asociados a una persona identificable sin el uso de información adicional.. 37

Base de Datos Artificial Base de datos creada de forma sintética con datos no reales, utilizada en entornos de desarrollo y pruebas para simular escenarios sin comprometer la privacidad de individuos reales.. 37

Boxplot Gráfico estadístico que resume la distribución de un conjunto de datos mediante cinco valores clave: mínimo, primer cuartil (Q1), mediana, tercer cuartil (Q3) y máximo, e identifica posibles valores atípicos. 32

Delegado de Protección de Datos Persona encargada de velar por el cumplimiento del RGPD dentro de una organización, asegurando que se cumplan las normativas de protección de datos personales.. 37

Diagrama de Gantt Herramienta gráfica de planificación de proyectos que representa visualmente las tareas a lo largo del tiempo, facilitando el seguimiento del progreso y la organización del trabajo. 8

F1-Score Métrica de evaluación que combina la precisión y el recall en un único valor mediante su media armónica; utilizada especialmente en problemas con clases desbalanceadas para medir el rendimiento global de un modelo de clasificación. 5, 7, 10, 26, 29

Fairness Término en aprendizaje automático que se refiere a la equidad o imparcialidad en los resultados de un modelo, evitando sesgos que perjudiquen a determinados grupos sociales. 38

Fernet Algoritmo de cifrado simétrico basado en AES en modo CBC con autenticación HMAC, que garantiza la confidencialidad e integridad de los datos cifrados. 15

Firebase Plataforma de desarrollo de aplicaciones de Google que proporciona servicios backend como bases de datos en tiempo real, autenticación y alojamiento en la nube. 5, 9, 10, 14, 15, 20, 22, 26–28, 37, 41

Minimización de Datos Principio establecido por el RGPD que exige que solo se recojan los datos estrictamente necesarios para el propósito específico para el cual se recogen.. 37

Recall Métrica de evaluación que indica la proporción de instancias positivas correctamente identificadas por el modelo; también conocida como sensibilidad o tasa de verdaderos positivos. 5, 7, 10, 26, 29

Reglamento General de Protección de Datos Reglamento de la Unión Europea (2016/679) que establece directrices y requisitos para el tratamiento de los datos personales, buscando proteger la privacidad y seguridad de los individuos.. 37

14. Anexo

GenerateBD.py

El módulo `GenerateBD.py` ha sido diseñado como un generador interactivo de bases de datos simuladas, orientadas tanto al entrenamiento como a la simulación de escenarios de detección de fraude. Su funcionalidad principal consiste en ofrecer dos flujos diferenciados de procesamiento de datos, mediante una interfaz de consola sencilla para el usuario.

El script comienza cargando variables de entorno desde un archivo `.env`, lo cual permite mantener separadas las configuraciones sensibles del código fuente.

A continuación, se presenta un menú de opciones por consola para que el usuario seleccione el tipo de base de datos a generar. Solo se aceptan valores válidos (1 o 2).

```
print("Select the type of database to generate:")
print("1. Generate 'Traning DB'")
print("2. Generate 'Generated DB'")

while True:
   option = input("Enter your choice (1 or 2): ")
   if option in ["1", "2"]:
      break
   else:
      print("Invalid option. Please enter '1' or '2'.")
```

Si se selecciona la opción 1, se genera una base de datos sintética destinada al entrenamiento. El proceso incluye:

- Generacion de datos con `generate_data(DB_NUM)`
- Ordenacion alfabetica por nombre
- Guardado en `DB_TRAINING`
- Transformacion con `transform_data()`
- Guardado del resultado en `DB_TRAINING_PP`

```
datagenerated = generate_data(DB_NUM)
df_generated = pd.DataFrame(datagenerated)
df_generated = df_generated.sort_values(by='NOMBRE_COMPLETO', ascending=
    True)
df_generated.to_csv(DB_TRAINING, index=False)

data = transform_data(DB_TRAINING)
data.to_csv(DB_TRAINING_PP, index=False)
```

Con la opción 2, se genera una base de datos simulada más extensa para escenarios productivos. Incluye:

- Generacion de 10.000 registros
- Anadido de la columna `FRAUD` con el valor `"Procesando"`
- Guardado en `DB_GENERATED`

- Transformacion de los datos
- Encriptado del archivo original
- Guardado del resultado transformado en `DB_GENERATED_PP`

```
df_generated = pd.DataFrame(datagenerated)
df_generated = df_generated.sort_values(by='NOMBRE_COMPLETO', ascending=
    True)
df_generated['FRAUD'] = "Procesando"
df_generated.to_csv(DB_GENERATED, index=False)

data = transform_data(DB_GENERATED)
key = generate_key()
encrypt_file(DB_GENERATED, key, DB_GENERATED_ENCRIPTED)
data.to_csv(DB_GENERATED_PP, index=False)
```

Todas las funciones auxiliares (`generate_data`, `transform_data`, `generate_key`, `encrypt_file`, `decrypt_file`) están contenidas en el módulo `Utils`, lo que permite una separación clara de responsabilidades y mejora la mantenibilidad del sistema.

Código del Modelo de Machine Learning

A continuación, se presenta el código completo del modelo de ML desarrollado para la detección de fraude. Este script realiza la carga de datos, preprocesamiento, balanceo, entrenamiento y evaluación del modelo `RandomForestClassifier`. También guarda tanto el modelo entrenado como el escalador utilizado.

Se cargan las variables de entorno y el conjunto de datos.

Se comprueba la existencia de la columna `ID` para preservarla durante la división del conjunto de datos:

```
if "ID" in data.columns:
   ids = data["ID"]
   data = data.drop(columns=["ID"])
else:
   raise ValueError("The 'ID' column was not found in the dataset.")
```

Se separan las variables independientes (`X`) y la variable objetivo (`y`):

```
X = data.drop(columns=["FRAUD"])
y = data["FRAUD"]
```

Se aplica escalado a los datos para mejorar el rendimiento del modelo:

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
joblib.dump(scaler, "scaler.pkl")
```

Se realiza la partición del conjunto de datos y se aplica SMOTE para balancear la clase minoritaria:

```
X_train, X_test, y_train, y_test, ids_train, ids_test = train_test_split(
   X_scaled, y, ids, test_size=0.2, stratify=y, random_state=42
)

smote = SMOTE(random_state=42)
X_train_balanced, y_train_balanced = smote.fit_resample(X_train, y_train)
```

Se define y entrena el modelo `RandomForestClassifier` con la mejor configuración obtenida durante la validación:

```
best_n_estimators = 1500
best_max_depth = 50
best_min_samples_split = 2
best_min_samples_leaf = 3
best_threshold = 0.5

model = RandomForestClassifier(
   n_estimators=best_n_estimators,
   max_depth=best_max_depth,
   min_samples_split=best_min_samples_split,
   min_samples_leaf=best_min_samples_leaf,
   bootstrap=True,
   oob_score=True,
   max_features='sqrt',
   criterion='entropy',
   class_weight="balanced",
   n_jobs=-1,
   random_state=42
)
model.fit(X_train_balanced, y_train_balanced)
```

Se realiza la predicción ajustada con un umbral personalizado, y se calculan las métricas de evaluación:

```
y_pred_proba = model.predict_proba(X_test)[:, 1]
y_pred_adjusted = (y_pred_proba >= best_threshold).astype(int)

acc = accuracy_score(y_test, y_pred_adjusted)
f1 = f1_score(y_test, y_pred_adjusted, average='weighted')
precision = precision_score(y_test, y_pred_adjusted)
recall = recall_score(y_test, y_pred_adjusted)
conf_matrix = confusion_matrix(y_test, y_pred_adjusted)

TN, FP, FN, TP = conf_matrix.ravel()
specificity = TN / (TN + FP)
```

Se imprimen los resultados obtenidos y, finalmente, se guarda el modelo entrenado para su posterior uso:

```
joblib.dump(model, "modelo_random_forest.pkl")
```

Función transform_data

La función `transform_data(original)` tiene como objetivo limpiar, transformar y preparar los datos para que puedan ser utilizados por el modelo de detección de fraude. Esta transformación incluye la conversión de tipos, la codificación de variables categóricas y la generación de nuevas características. A continuación, se describe paso a paso su funcionamiento:

Se carga el archivo CSV original en un `DataFrame`.

Se reemplazan los valores `'N/A'` por `np.nan`, facilitando el manejo de valores nulos posteriormente.

```
   df.replace('N/A', np.nan, inplace=True)
```

Se definen los tipos de datos correctos para determinadas columnas, con el fin de garantizar que los valores numéricos se traten adecuadamente.

```
conversion_type = {
    'PRECIO_TOTAL': float,
    'NUM_PRODUCTS': int,
    'TIEMPO': int,
    'CLICKS': int,
    'NUM_INTENTOS': int,
    'VERIFICADO': int,
}

df = df.astype(conversion_type)
```

Se genera una nueva variable binaria llamada ENVIO_EN_CASA, que toma el valor 1 si el envío es de tipo Estandar o Express, y 0 en caso contrario.

```
df['ENVIO_EN_CASA'] = df['TIPO_ENVIO'].isin(['Estandar', 'Express']).
    astype(int)
```

Se eliminan columnas irrelevantes para el análisis, como información temporal, direcciones, detalles de envío y dispositivos.

```
df.drop(columns=['FECHA', 'HORA', 'DIRECCION', 'TIPO_ENVIO', 'METODO',
             'URL_SEGUIMIENTO', 'DISPOSITIVO', 'CODIGO_SEGUIMIENTO', '
                NUM_TRANSFERENCIA'], inplace=True)
```

Se aplica codificación ordinal a las variables categóricas o de texto. Se utiliza OrdinalEncoder de sklearn, con la opción de manejar valores desconocidos.

```
columns_to_encode = ['NOMBRE_COMPLETO', 'CORREO', 'COMENTARIO', 'DNI',
     'IP', 'DESCUENTO', 'TIPO', 'TELEFONO']

ordinal_encoder = OrdinalEncoder(handle_unknown="use_encoded_value",
    unknown_value=-1)
df[columns_to_encode] = ordinal_encoder.fit_transform(df[
    columns_to_encode].astype(str)).astype(int)
```

Se generan dos nuevas variables agregadas por usuario: NUM_COMPRAS: número total de compras realizadas por el mismo usuario. COMPRA_MEDIA_USUARIO: gasto promedio por compra del usuario, calculado a partir del PRECIO_TOTAL.

```
df['NUM_COMPRAS'] = df.groupby('NOMBRE_COMPLETO')['NOMBRE_COMPLETO'].
    transform('count')
df['COMPRA_MEDIA_USUARIO'] = df.groupby('NOMBRE_COMPLETO')['
    PRECIO_TOTAL'].transform('mean').round(2)
```

Finalmente, la función devuelve el DataFrame transformado, listo para ser usado como entrada del modelo predictivo.

Función analysis

La función `analysis(file)` se encarga de analizar un conjunto de datos almacenado en un archivo CSV, aplicar un modelo de ML previamente entrenado y guardar las predicciones sobre si una transacción es fraudulenta o no. A continuación, se detalla el funcionamiento de cada parte:

```
def analysis(file):
```

Se carga el archivo original usando `pandas.read_csv`. Se transforma el archivo mediante una función externa llamada `transform_data`, que prepara los datos para su uso en el modelo.

```
    df_original = pd.read_csv(file)
    df_trans = transform_data(file)
```

Se verifica que ambas versiones del conjunto de datos contengan una columna denominada `ID`, necesaria para identificar las filas analizadas.

```
    if "ID" not in df_trans.columns or "ID" not in df_original.columns:
        raise ValueError("Both files must contain an 'ID' column.")
```

Se filtran solo aquellas filas donde la columna `FRAUD` tenga el valor `"Procesando"`, que indica las transacciones pendientes de análisis.

```
    df_trans = df_trans.query('FRAUD == "Procesando"')
```

Se eliminan las columnas que no son relevantes para la predicción, como `ID` y `FRAUD`, y se obtiene el subconjunto de variables predictoras `X`.

```
    columns_to_remove = ["ID", "FRAUD"] if "FRAUD" in df_trans.columns
        else ["ID"]
    X = df_trans.drop(columns=columns_to_remove)
```

Si no hay filas que analizar (por ejemplo, si no existen filas con `FRAUD="Procesando"`), se imprime un mensaje y se termina la ejecución de la función.

```
    if X.empty:
        print("No hi ha files amb FRAUD='Processing'. No s'aplica el model.
            ")
```

Se carga un escalador previamente guardado y se aplica a los datos para que tengan el mismo formato que los usados durante el entrenamiento.

```
    scaler = joblib.load(SCALER_PATH)
    X_scaled = scaler.transform(X)
```

Se carga el modelo del ML entrenado (por ejemplo, un RF) y se valida que el número de características del conjunto de datos coincida con lo esperado por el modelo.

```
    model = joblib.load(MODEL_PATH)

    if X_scaled.shape[1] != model.n_features_in_:
        raise ValueError(f"The model expects {model.n_features_in_}
            features, but {X_scaled.shape[1]} were provided.")
```

Se realizan las predicciones y se convierten los resultados en cadenas de texto: `"True"` si se considera fraude, `"False"` en caso contrario.

```
y_pred = model.predict(X_scaled)
y_pred_str = ['True' if pred == 1 else 'False' for pred in y_pred]
```

Se extraen los identificadores (`ID`) de las transacciones analizadas y se actualiza la columna `FRAUD` en el archivo original con las nuevas predicciones.

```
ids = df_trans["ID"]
df_original.loc[df_original["ID"].isin(ids), "FRAUD"] = y_pred_str
```

Finalmente, se guarda el archivo CSV original actualizado, con la columna `FRAUD` modificada para reflejar los resultados de la predicción.

```
df_original.to_csv(file, index=False)
print(f"File '{file}' successfully generated with FRAUD predictions.")
```

Esta función es útil en un contexto de análisis post-fraude, ya que permite etiquetar transacciones sospechosas basándose en un modelo previamente entrenado y aplicar dichas etiquetas directamente sobre los datos originales.