Predicción de fraude en transacciones de cuentas bancarias

12/08/2024 - 18/08/2024

Juan Hernández Sánchez-Agesta
Universidad de Granada
jhernandezsanchezagesta@gmail.com
Granada (Granada)

Índice:

Visión general

El vertiginoso crecimiento de las transacciones bancarias online, impulsado por la digitalización y la pandemia, ha convertido la seguridad en una prioridad absoluta. Paralelamente, el aumento exponencial en la actividad de hackers y la sofisticación de sus técnicas han dado lugar a un incremento significativo de las brechas de seguridad.

Desafíos

1. **Vulnerabilidad de los sistemas:** La complejidad creciente de los sistemas bancarios los hace más susceptibles a ataques.
2. **Evolución constante de las amenazas:** Los hackers desarrollan continuamente nuevas tácticas, dificultando la detección de fraudes.
3. **Comportamiento del usuario:** La creciente población de usuarios mayores, menos familiarizada con las transacciones online, puede ser un blanco fácil para los estafadores.
4. **Amenazas físicas:** La existencia de hardware capaz de robar datos bancarios (skimmers, etc.) sigue siendo una preocupación en el mundo físico.

Objetivo

Desarrollar un modelo de detección de fraudes capaz de identificar transacciones fraudulentas en tiempo real, protegiendo así los fondos de los clientes y la reputación de la entidad bancaria.

Enfoque del proyecto

1. **Recopilación y análisis de datos:** Se recopilarán grandes volúmenes de datos históricos de transacciones, incluyendo información sobre el cliente, la transacción y el dispositivo utilizado.
2. **Ingeniería de características:** Se crearán nuevas características a partir de los datos existentes para mejorar la capacidad del modelo de detectar patrones de fraude.
3. **Selección del modelo:** Se evaluarán diferentes algoritmos de aprendizaje automático (redes neuronales, random forest, etc.) para seleccionar el modelo más adecuado.

4. **Validación y ajuste:** El modelo se validará utilizando técnicas como cross-validation y se ajustarán sus hiperparámetros para optimizar su rendimiento.
5. **Implementación:** El modelo se integrará en un sistema de detección de fraudes en tiempo real, permitiendo la identificación y bloqueo de transacciones sospechosas.

Beneficios esperados

I. Reducción de pérdidas económicas:

Al detectar y bloquear las transacciones fraudulentas a tiempo, se minimizan las pérdidas para la entidad bancaria y sus clientes.

II. Mejora de la reputación:

Un sistema de detección de fraudes eficaz refuerza la confianza de los clientes en la seguridad de sus transacciones.

III. Optimización de recursos:

Al automatizar la detección de fraudes, se reducen los costos operativos y se liberan recursos humanos para otras tareas.

Consideraciones adicionales

1. **Ética:** Es importante garantizar que el modelo no discrimine a ningún grupo de usuarios y que se respeten los derechos de privacidad de los clientes.
2. **Interpretabilidad:** En algunos casos, puede ser útil desarrollar modelos interpretables para entender mejor las razones detrás de las predicciones.
3. **Actualización continua:** El modelo debe ser actualizado periódicamente para adaptarse a la evolución de las amenazas y a los cambios en el comportamiento de los usuarios.

Presentación del problema :

Los pagos digitales están en constante evolución, pero también lo están los ciberdelincuentes. Según el Índice de Brechas de Datos, se roban más de 5 millones de registros diariamente, lo que demuestra que el fraude sigue siendo muy común tanto en transacciones presenciales como no presenciales. En un mundo digital donde se realizan

billones de transacciones con tarjeta diariamente, detectar el fraude se convierte en un desafío significativo.

Para la extracción de este conjunto de datos he utilizado como fuente Kaggle(adjunto enlace):

https://www.kaggle.com/datasets/dhanushnarayananr/credit-card-fraud

Descripción de Parámetros

- **distancia_desde_hogar:** Distancia desde el domicilio del titular de la tarjeta hasta el lugar donde se realizó la transacción.
- **distancia_ultima_transaccion:** Distancia entre la ubicación de la transacción actual y la ubicación de la transacción anterior.
- **ratio_precio_compra_mediana:** Relación entre el precio de la compra realizada y el precio medio de las compras del usuario.
- **transaccion_repetida:** Indica si la transacción se realizó en el mismo comercio.
- **uso_chip:** Indica si se utilizó el chip de la tarjeta para realizar la transacción.
- **uso_pin:** Indica si se utilizó el número PIN para realizar la transacción.
- **compra_online:** Indica si la transacción fue realizada de forma online.
- **fraude:** Indica si la transacción fue fraudulenta.

Procedimiento seguido:

La detección de fraudes en transacciones bancarias es un desafío complejo pero crucial en el entorno actual. Este proyecto busca desarrollar una solución innovadora y efectiva para proteger los fondos de los clientes y garantizar la seguridad de las transacciones.

I. Extracción de los datos (Extract)

Para la extracción de los datos existen múltiples fuentes , ya sea de bases de datos públicas, privadas, APIs de terceros, o fuentes de datos publicas tales como Kaggle o Mockaroo. En este caso he seleccionado Kaggle ya que es la que me ofrecía dataset mas ajustados a lo que yo estaba buscando.De manera que una vez extraído el csv con los datos de las transacciones , he procedido a limpiarlos.

II. Transformación de los datos (Transform)

En esta fase he llevado a cabo el limpiado , normalización y estandarización de los datos, haciendo comprobaciones sobre el dataset gracias a librerías como pandas .

III. Carga de los datos (Load)

He comenzado a explorar los datos y a partir de análisis estadísticos he establecido cuales son las variables más importantes y que más afectan a la existencia o no de fraude en la transacción bancaria.He extraído también algunos indicadores estadísticos de interés como los siguientes:

	distance_from_home	distance_from_last_transaction	ratio_to_median_purchase_price	repeat_retailer	used_chip	used_pin_number	online_order	fraud
count	1000000.000000	1000000.000000	1000000.000000	1000000.000000	1000000.000000	1000000.000000	1000000.000000	1000000.000000
mean	26.628792	5.036519	1.824182	0.881536	0.350399	0.100608	0.650552	0.087403
std	65.390784	25.843093	2.799589	0.323157	0.477095	0.300809	0.476796	0.282425
min	0.004874	0.000118	0.004399	0.000000	0.000000	0.000000	0.000000	0.000000
25%	3.878008	0.296671	0.475673	1.000000	0.000000	0.000000	0.000000	0.000000
50%	9.967760	0.998650	0.997717	1.000000	0.000000	0.000000	1.000000	0.000000
75%	25.743985	3.355748	2.096370	1.000000	1.000000	0.000000	1.000000	0.000000
max	10632.723672	11851.104565	267.802942	1.000000	1.000000	1.000000	1.000000	1.000000

Una vez hecho esto y teniendo una visión rápida de los datos de manera agregada, he procedido a calcular las correlaciones entre las variables para intentar establecer una dependencia o por el contrario independencia.

```
                     distance_from_home     fraud
distance_from_home             1.000000  0.187571
fraud                          0.187571  1.000000
                                 distance_from_last_transaction     fraud
distance_from_last_transaction                         1.000000  0.091917
fraud                                                  0.091917  1.000000
                                 ratio_to_median_purchase_price     fraud
ratio_to_median_purchase_price                         1.000000  0.462305
fraud                                                  0.462305  1.000000
```

De aqui se pueden extraer algunas conclusiones sobre variables cuantitativas como:

- La distancia a casa y el fraude están positivamente correlacionados en un 18,75%
- La distancia a la última transacción está correlacionado positivamente en un 9,19%
- El ratio sobre la compra media está correlacionado con el fraude positivamente en un 46,23%.

Para obtener información también sobre las variables de tipo 0/1 (SÍ / NO) , hemos calculado el estadístico chi-cuadrado sobre estas variables y el fraude:

```python
from scipy.stats import chi2_contingency

# Crear una tabla de contingencia
contingency_table = pd.crosstab(df['fraud'], df['online_order'])

# Realizar el test chi-cuadrado
chi2, p, dof, expected = chi2_contingency(contingency_table)
print(f"Chi-squared test statistic: {chi2}")
print(f"p-value: {p}")
```

```
Chi-squared test statistic: 36852.02374794533
p-value: 0.0
```

```python
# Crear una tabla de contingencia
contingency_table = pd.crosstab(df['fraud'], df['repeat_retailer'])

# Realizar el test chi-cuadrado
chi2, p, dof, expected = chi2_contingency(contingency_table)
print(f"Chi-squared test statistic: {chi2}")
print(f"p-value: {p}")
```

```
Chi-squared test statistic: 1.827827480587841
p-value: 0.17638436830458504
```

```python
# Crear una tabla de contingencia
contingency_table = pd.crosstab(df['fraud'], df['used_pin_number'])

# Realizar el test chi-cuadrado
chi2, p, dof, expected = chi2_contingency(contingency_table)
print(f"Chi-squared test statistic: {chi2}")
print(f"p-value: {p}")
```

```
Chi-squared test statistic: 10057.412546099067
p-value: 0.0
```

Las interpretaciones sobre estos resultados obtenidos son las siguientes:

Con respecto a la variable ***online_order*** :

- Significa que hay una **relación muy fuerte** entre la variable **fraude** y la variable **online_order**. En otras palabras, las transacciones realizadas en línea tienen una probabilidad significativamente mayor de ser fraudulentas en comparación con las transacciones que no se realizaron en línea.

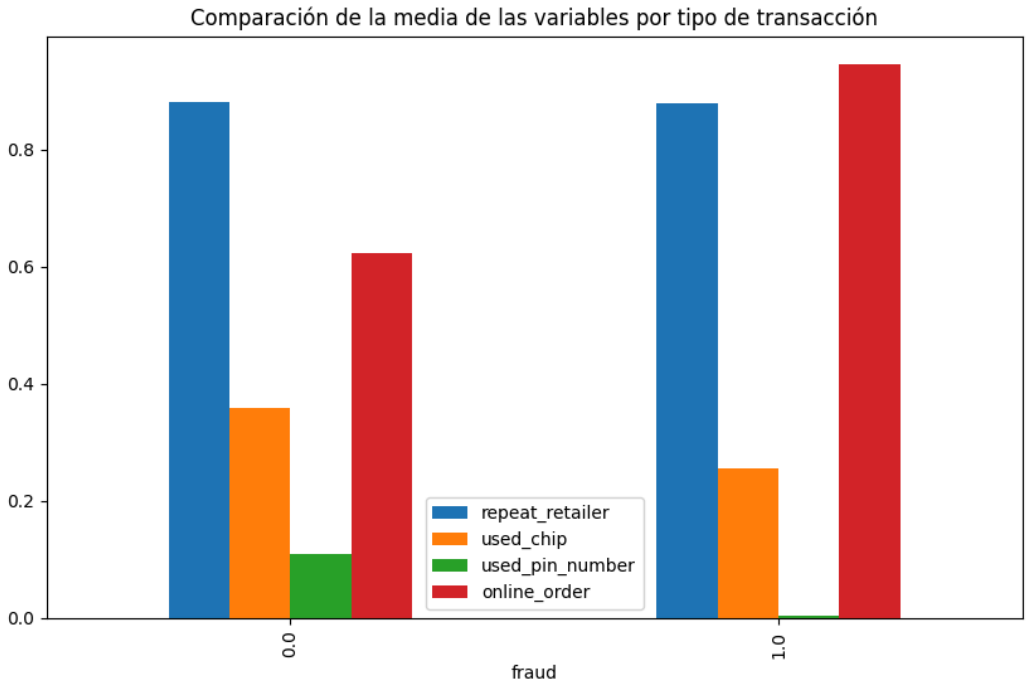
Con respecto a la variable ***repeat_retailer***:

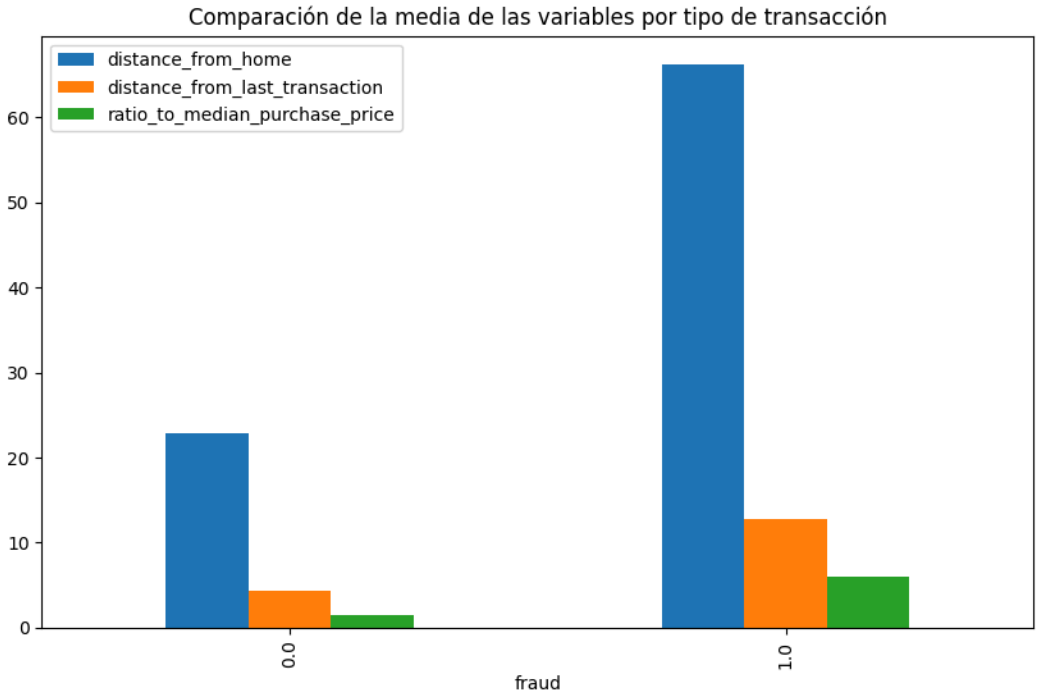
- En este caso, los resultados del test chi-cuadrado sugieren que **no hay una asociación significativa** entre las dos variables que estás analizando.
    - **Bajo estadístico de prueba chi-cuadrado:** El valor de 1.827827480587841 es relativamente bajo, lo que indica una pequeña discrepancia entre los valores observados y los esperados bajo la hipótesis nula.
    - **Valor p mayor que 0.05:** Un valor p de 0.17638436830458504 es mayor que el nivel de significancia típico de 0.05, por lo que no podemos rechazar la hipótesis nula.

Con respecto a la variable ***used_pin_number***:

- **El estadístico chi-cuadrado es extremadamente alto (10057.412546099067).** Esto indica una fuerte discrepancia entre las frecuencias observadas y las esperadas en la tabla de contingencia entre la variable **used_pin_number** y la variable **fraude**.
- **El valor p es 0.0,** lo que significa que hay evidencia estadísticamente significativa para rechazar la hipótesis nula.
- Dado que rechazamos la hipótesis nula, podemos concluir que **existe una fuerte asociación entre el uso del número PIN y la ocurrencia de fraude**. En otras palabras, el uso del número PIN parece estar relacionado con la probabilidad de que una transacción sea fraudulenta.

El siguiente paso fue relacionar las variables de las cuales tenemos información con la variable que indica la fraudulencia la transacción:





De este análisis se pueden extraer cierta información de interés, algunas conclusiones son :

- En los casos en los que se usa el número de pin es casi imposible que se cometa fraude

- En prácticamente la totalidad de las transacciones la modalidad ha sido online
- En los casos de fraude, la distancia de la transacción al hogar es mucho mayor
- En los casos de fraude , la distancia desde la transacción actual a la anterior es mayor
- En los casos de fraude , la relación entre el precio de la compra realizada y el precio medio de las compras del usuario es mucho mayor a 1 , lo cual significa que si : Precio compra/Precio medio de compra >1 , el precio de la compra en esta transacción es mayor al de la media de transacciones de esa tarjeta , si Precio compra/Precio medio de compra > 2 , significa que el precio de la compra de esa transacción es más del doble de la media , .. y así sucesivamente.

IV. Entrenamiento del modelo

Para el entrenamiento del modelo , he empleado un modelo de regresión logística. Para ello he escalado las variables (con StandardScaler) para que tengan sentido en su conjunto.

Para elaborar el modelo he separado las variables entre variables dependientes y variables independientes. En el conjunto de variables independientes se encuentran las variables 'distance_from_home', 'distance_from_last_transaction', 'ratio_to_median_purchase_price', 'repeat_retailer', 'used_chip', 'used_pin_number', 'online_order' , variables que intentaremos que en mayor o menor medida explique la variable independiente, que será 'fraud'.

El siguiente pasó fue dividir nuestro conjunto de datos en conjunto de entrenamiento y conjunto de prueba , de manera que tengamos un conjunto de datos que se centre en el entrenamiento del modelo y otro que sirva para evaluar el comportamiento del modelo.

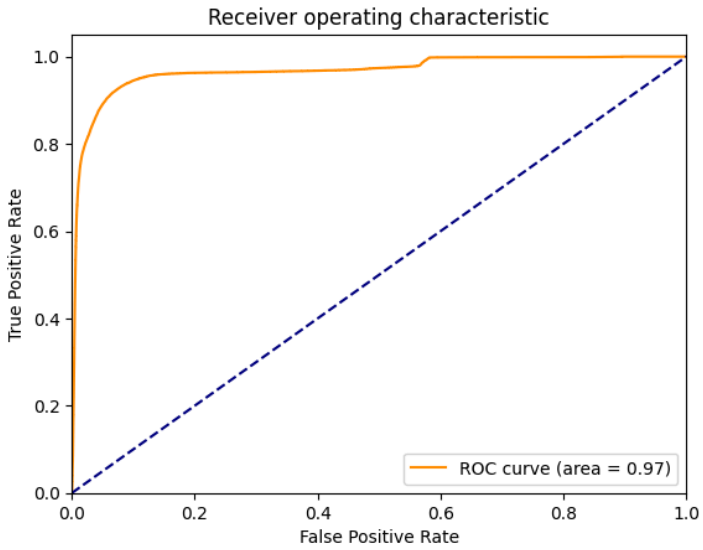
V. Evaluación del modelo

Para la evaluación del modelo he empleado 2 técnicas, la curva ROC (Receiving Operating Characteristic) y la matriz de confusión

Curva ROC: Muestra el rendimiento del modelo a diferentes umbrales de clasificación. Es útil para evaluar el equilibrio entre la tasa de verdaderos positivos y la tasa de falsos positivos.

Matriz de confusión: Muestra una tabla que resume el número de predicciones correctas e incorrectas. Es útil para evaluar la precisión, el recall y la F1-score del modelo.

```
Matriz de confusión:
[[181283   1274]
 [  6976  10467]]
```



¿Qué significa un AUC de 0.97?

- **Alto poder discriminativo:**
  Un valor cercano a 1 significa que el modelo tiene una gran capacidad para distinguir entre transacciones fraudulentas y legítimas. En otras palabras, es muy bueno en clasificar correctamente ambas clases.
- **Baja tasa de falsos positivos y falsos negativos:** Esto implica que el modelo tiene pocas probabilidades de clasificar una transacción legítima como fraudulenta (falso positivo) y también pocas probabilidades de clasificar una transacción fraudulenta como legítima (falso negativo).

¿Qué nos dice esta matriz?

Una matriz de confusión es una herramienta esencial para evaluar el rendimiento de un modelo de clasificación. Cada celda representa una combinación de la clase real y la clase predicha. En tu caso, parece que estás trabajando con un problema de clasificación binaria (fraude o no fraude).

Desglosando la matriz:

- **[181283, 1274]:** Esta fila representa las transacciones que realmente no eran fraudulentas (clase negativa).

    - **181283:** El modelo predijo correctamente que estas transacciones no eran fraudulentas (verdaderos negativos).
    - **1274:** El modelo clasificó erróneamente estas transacciones como fraudulentas (falsos positivos).
- **[6976, 10467]:** Esta fila representa las transacciones que realmente eran fraudulentas (clase positiva).
    - **6976:** El modelo clasificó erróneamente estas transacciones como no fraudulentas (falsos negativos).
    - **10467:** El modelo predijo correctamente que estas transacciones eran fraudulentas (verdaderos positivos).

Interpretando los resultados:

- **Falsos positivos:** 1274 transacciones legítimas fueron marcadas como fraudulentas. Esto podría llevar a inconvenientes para los clientes legítimos, como bloqueos de tarjetas o investigaciones innecesarias.
- **Falsos negativos:** 6976 transacciones fraudulentas no fueron detectadas. Esto implica una pérdida financiera para la empresa y un riesgo para los clientes.

A partir de la matriz de confusión, podemos calcular diversas métricas para evaluar el rendimiento del modelo de manera más precisa:

Descripción de las Métricas

- **Accuracy:** Representa la proporción de predicciones correctas totales, tanto para casos positivos como negativos. En este caso, el modelo predice correctamente el 95.875% de las veces.
- **Precision:** Indica la proporción de predicciones positivas que son realmente positivas. En este caso, cuando el modelo predice una transacción como fraudulenta, está en lo correcto el 89.15% de las veces.
- **Recall:** Mide la capacidad del modelo para identificar correctamente los casos positivos. En este caso, el modelo identifica correctamente el 60.01% de las transacciones fraudulentas.
- **Specificity:** Mide la capacidad del modelo para identificar correctamente los casos negativos. En este caso, el modelo identifica correctamente el 99.30% de las transacciones no fraudulentas.
- **F1-score:** Es la media armónica de precisión y recall, proporcionando un equilibrio entre ambos. En este caso, el F1-score es de 0.7173, indicando un buen equilibrio entre precisión y recall.

```
Accuracy: 0.95875
Precision: 0.8914913550804872
Recall: 0.6000687955053603
Specificity: 0.9930213577129335
F1-score: 0.7173108552631579
```

Análisis de los Resultados

- **Alto accuracy:** El modelo tiene una alta precisión general, lo que indica que predice correctamente la mayoría de las transacciones.
- **Buena precisión:** El modelo tiene una buena precisión, lo que significa que cuando predice una transacción como fraudulenta, suele ser correcto.
- **Recall moderado:** El modelo tiene un recall moderado, lo que indica que no identifica todas las transacciones fraudulentas. Esto podría ser un problema si es importante detectar la mayoría de los fraudes.
- **Alta especificidad:** El modelo es muy bueno para identificar transacciones legítimas, lo que significa que tiene una baja tasa de falsos positivos.

Desarrollo de la aplicación web:

Para el desarrollo de la aplicación he utilizado el modelo entrenado a partir del análisis de datos y gracias a Streamlit me ha sido fácil conseguir una aplicación que utilizando el modelo y con una interfaz amigable e intuitiva en la que introduciendo los parámetros de una transacción el modelo realiza una predicción. Explicaré como ha sido el despliegue de la misma

Despliegue

1. **Preparación para el Despliegue:**
    - **Código en GitHub:** Se subió el código fuente de la aplicación a un repositorio en GitHub. Esto incluye el archivo principal de la aplicación (`app.py`), el archivo de dependencias (`requirements.txt`), y los archivos exportados del modelo (`modelo_prediccion_fraude_credito.joblib` y `scaler_fraude_credito.joblib`).
2. **Despliegue en Streamlit Cloud:**
    - **Streamlit Cloud:** Se eligió Streamlit Cloud para desplegar la aplicación debido a su integración directa con aplicaciones desarrolladas en Streamlit.
    - **Pasos para Desplegar:**
        1. **Subir el Repositorio:** Iniciar sesión en Streamlit Cloud y conectar la cuenta de GitHub.

2. **Seleccionar el Repositorio:** Elegir el repositorio que contiene el código de la aplicación.
3. **Desplegar la Aplicación:** Streamlit Cloud automáticamente instalará las dependencias listadas en `requirements.txt`, cargará el código, y pondrá en marcha la aplicación. El enlace de la aplicación se proporciona al final del proceso de despliegue.

3. **Configuración y Verificación:**
   - **Configuración del Entorno:** Verificar que todas las configuraciones y dependencias estén correctamente establecidas en el entorno de despliegue.
   - **Verificación:** Probar la aplicación en el entorno de producción para asegurar que funciona correctamente, incluyendo la correcta carga del modelo y el procesamiento de datos ingresados por el usuario.
4. **Mantenimiento:**
   - **Actualizaciones:** Para realizar actualizaciones en la aplicación, se puede modificar el código en el repositorio de GitHub y luego redesplegar desde Streamlit Cloud.
   - **Monitoreo:** Supervisar el funcionamiento de la aplicación y gestionar cualquier problema que surja durante su operación en producción.