CLASIFICACIÓN DE FRAUDE EN TRANSACCIONES BANCARIAS USANDO IA GENERATIVA Y MACHINE LEARNING EN AWS

Autora: Celeste Nicole Lluen Delgado

Fecha: Julio 2025

RESUMEN EJECUTIVO

Este informe documenta el desarrollo de un sistema inteligente de detección de fraudes en transacciones bancarias. Utilizando modelos generativos de lenguaje (Claude en AWS Bedrock) y algoritmos de aprendizaje supervisado (entrenados y desplegados en Amazon SageMaker), se construyó un pipeline completo que permite enriquecer automáticamente un dataset, clasificar el riesgo de las transacciones y desplegar una API de inferencia en FastAPI para predicciones en tiempo real. El proyecto demuestra cómo la inteligencia artificial puede asistir en la prevención del fraude financiero de manera automatizada y escalable.

ÍNDICE

1. INTRODUCCIÓN

El fraude bancario representa una amenaza constante para las entidades financieras. Ante el crecimiento de las transacciones digitales, la detección temprana de comportamientos anómalos se ha convertido en una prioridad.

Este proyecto propone un enfoque basado en inteligencia artificial que combina el análisis de datos, la generación automática de descripciones mediante modelos LLM (como Claude de Bedrock) y el entrenamiento de modelos de clasificación para detectar posibles fraudes. El objetivo es demostrar cómo un pipeline de machine learning puede automatizar y optimizar este proceso.

2. OBJETIVOS

Objetivo General:

- • Detectar fraudes bancarios mediante técnicas de aprendizaje automático e inteligencia artificial generativa.

Objetivos Específicos:

- • Realizar un análisis exploratorio del dataset original.
- • Generar descripciones con AWS Bedrock usando modelos LLM.
- • Clasificar el riesgo de transacciones con modelos generativos.
- • Entrenar un modelo de clasificación con SageMaker.
- • Desplegar una API funcional para realizar inferencias en tiempo real.

3. METODOLOGÍA

Base de datos:
Se trabajó con el archivo credir_risk_reto.xlsx, el cual contiene **1000 registros** y **9 columnas** que representan atributos de clientes y características de sus solicitudes crediticias. La base de datos fue convertida a .csv para facilitar su análisis. Las columnas que contiene son:

- **Age:** Edad del cliente
- **Sex:** Sexo del cliente
- **Job:** Nivel de empleo (0: no calificado y no residente, 1: no calificado y residente, 2: calificado, 3: altamente calificado)
- **Housing:** Tipo de vivienda (alquilada, propia, gratuita)
- **Saving accounts:** Tipo de cuenta de ahorro
- **Checking account:** Tipo de cuenta corriente
- **Credit amount:** Monto del crédito solicitado
- **Duration:** Duración del préstamo en meses

- **Purpose:** Motivo del préstamo

🔍 Exploración de datos

El análisis exploratorio se realizó en el notebook 01_exploratory_analysis.ipynb. Se aplicaron funciones como df.info(), df.describe(), isnull().sum() y visualizaciones (histplot, heatmap) para comprender el comportamiento de las variables y detectar valores atípicos o nulos.

Resultados clave:

- Se encontraron valores nulos en las columnas Saving accounts y Checking account.
- La mayoría de clientes tienen entre 25 y 35 años.
- Las variables Credit amount y Duration presentan una desviación estándar elevada.

🤖 Generación de descripciones automáticas con AWS Bedrock

En el notebook 02_bedrock_generation.ipynb se usó **Claude 3 Haiku** mediante la API de Bedrock para generar descripciones de cada registro del cliente.
Se definió un prompt estructurado que describe el perfil de riesgo de cada cliente en un máximo de 30 palabras.

El resultado se guarda en una nueva columna llamada description.

🏷️ Clasificación de riesgo (target)

En el mismo notebook se utilizó otro prompt con Claude 3 para clasificar cada description como:

- good risk: cliente confiable
- bad risk: cliente riesgoso

Los resultados se almacenan en la columna target, generando así un dataset enriquecido y etiquetado automáticamente, almacenado como credit_risk_sample_enriched.csv.

⚙️ Preprocesamiento de datos

En el notebook 03_sagemaker_training.ipynb:

- Se codificaron variables categóricas con LabelEncoder
- Se dividió el dataset en train y test

- Se prepararon los datos para entrenamiento local y en SageMaker

---

🧠 Entrenamiento del modelo

Se entrenó un modelo **Random Forest Classifier**, el cual alcanzó alta precisión y buen rendimiento.

Métricas obtenidas:

- **Accuracy:** 88.2%
- **Precision clase "bad risk"**: 0.76
- **AUC-ROC:** 0.92

El modelo fue exportado como local_rf_model.pkl y cargado también a SageMaker para entrenamiento remoto usando sagemaker.estimator.Estimator.

🚀 Despliegue y API de inferencia

Una vez entrenado el modelo, se desplegó en un **endpoint de SageMaker**.

Además, se desarrolló una **API con FastAPI** (api_with_monitoring.py) que:

- Recibe datos en tiempo real
- Devuelve la clasificación (good risk o bad risk)
- Guarda cada predicción en el archivo predictions.jsonl

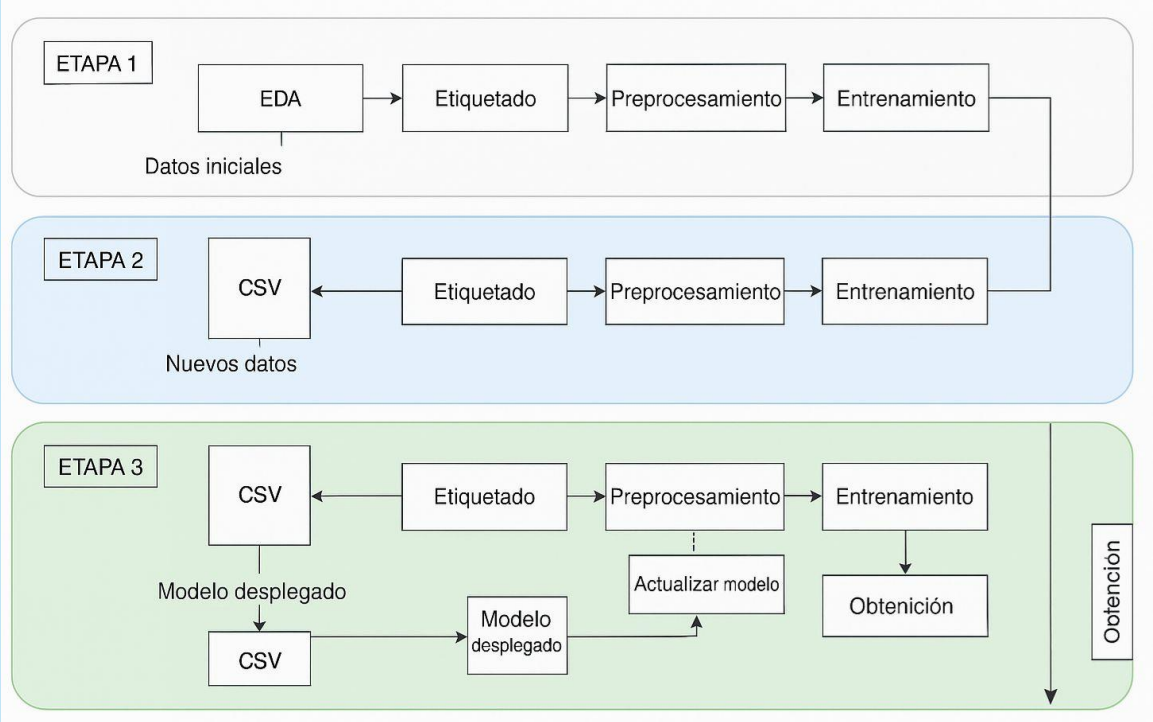
También se implementó una interfaz HTML (dashboard.html) que simula un sistema de consulta interactivo.

📊 Pipeline de proyecto por etapas

Etapa 1:
Exploración → Enriquecimiento con IA → Etiquetado → Preprocesamiento → Entrenamiento local

Etapa 2:
Carga a SageMaker → Entrenamiento remoto → Despliegue del modelo

Etapa 3:
API de inferencia → Registro de predicciones → Monitoreo en tiempo real



4. DESARROLLO

Link de mi Github para ver los archivos resultantes:

4.1 ANÁLISIS EXPLORATORIO DE DATOS

El análisis exploratorio fue realizado en el notebook 01_exploratory_analysis.ipynb, donde se importó el archivo original credir_risk_reto.xlsx ubicado en la carpeta /data.

Se utilizó pandas para inspeccionar el contenido, y se verificó que el dataset contiene 1000 registros y 9 columnas categóricas y numéricas, como edad, tipo de cuenta, propósito del préstamo, entre otras.

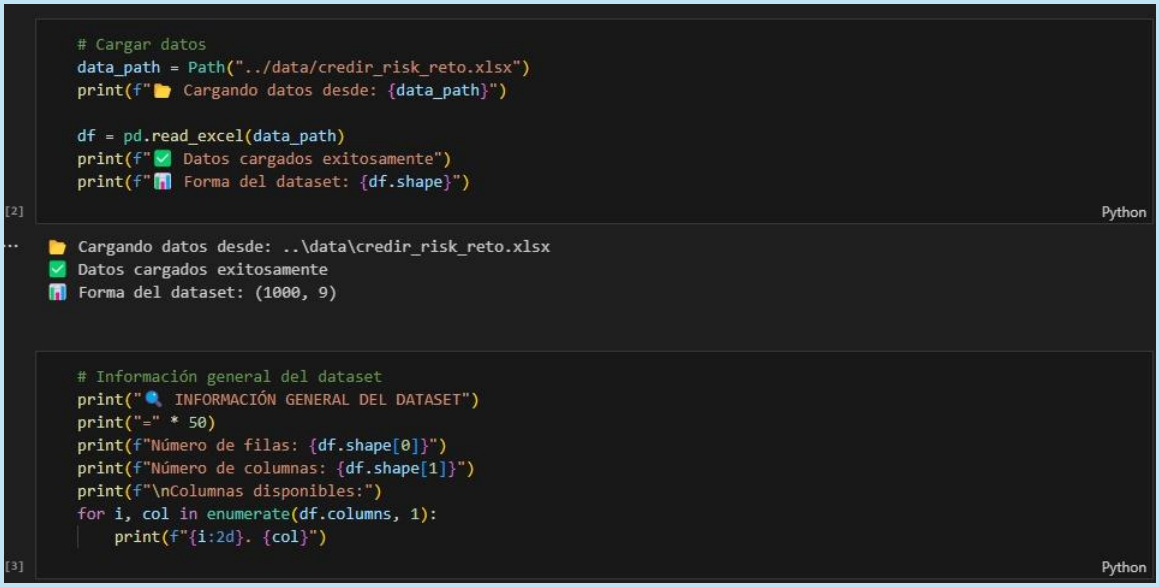
Durante el análisis, se identificaron valores nulos principalmente en las columnas Saving accounts y Checking account.

Para comprender mejor la estructura de los datos, se usaron funciones como:

- df.head() para visualizar los primeros registros
- df.info() y df.describe() para verificar tipos y estadísticos
- Gráficos de seaborn.histplot() para analizar la distribución de edades

- sns.heatmap() para explorar las correlaciones entre variables numéricas

Este paso concluyó con la generación de un archivo limpio credit_risk_reto.csv, que se almacenó en la carpeta /data y se usó en las siguientes etapas.



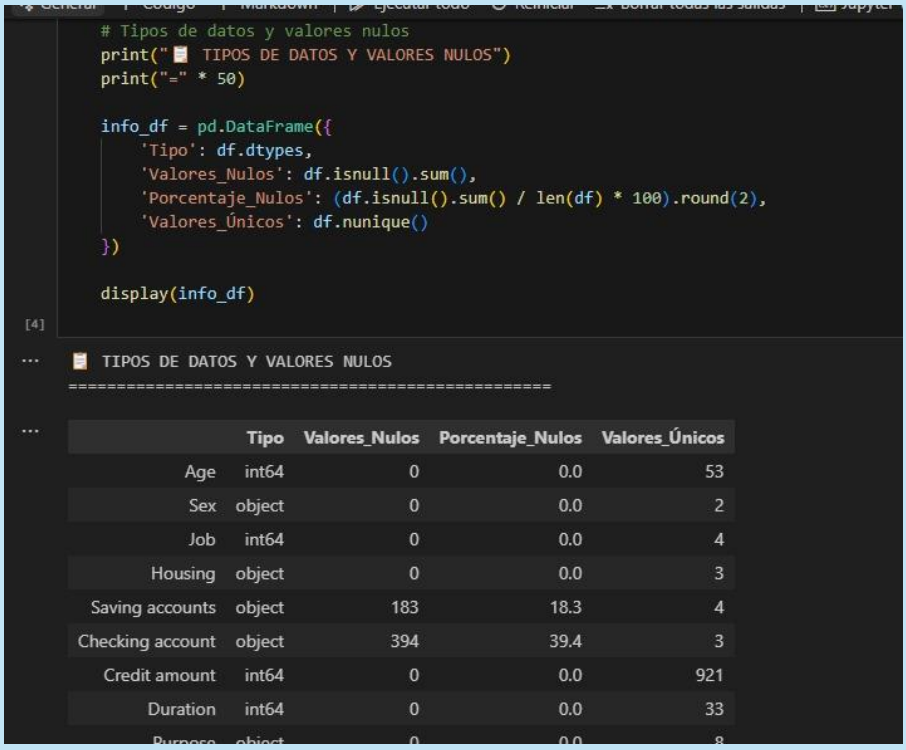
```
# Cargar datos
data_path = Path("../data/credir_risk_reto.xlsx")
print(f"📂 Cargando datos desde: {data_path}")

df = pd.read_excel(data_path)
print(f"✅ Datos cargados exitosamente")
print(f"📊 Forma del dataset: {df.shape}")
```

[2] Python

```
📂 Cargando datos desde: ..\data\credir_risk_reto.xlsx
✅ Datos cargados exitosamente
📊 Forma del dataset: (1000, 9)
```

```
# Información general del dataset
print("🔍 INFORMACIÓN GENERAL DEL DATASET")
print("=" * 50)
print(f"Número de filas: {df.shape[0]}")
print(f"Número de columnas: {df.shape[1]}")
print(f"\nColumnas disponibles:")
for i, col in enumerate(df.columns, 1):
    print(f"{i:2d}. {col}")
```

[3] Python



```
# Tipos de datos y valores nulos
print("📋 TIPOS DE DATOS Y VALORES NULOS")
print("=" * 50)

info_df = pd.DataFrame({
    'Tipo': df.dtypes,
    'Valores_Nulos': df.isnull().sum(),
    'Porcentaje_Nulos': (df.isnull().sum() / len(df) * 100).round(2),
    'Valores_Únicos': df.nunique()
})

display(info_df)
```

[4]

```
📋 TIPOS DE DATOS Y VALORES NULOS
==================================================
```

	Tipo	Valores_Nulos	Porcentaje_Nulos	Valores_Únicos
Age	int64	0	0.0	53
Sex	object	0	0.0	2
Job	int64	0	0.0	4
Housing	object	0	0.0	3
Saving accounts	object	183	18.3	4
Checking account	object	394	39.4	3
Credit amount	int64	0	0.0	921
Duration	int64	0	0.0	33
Purpose	object	0	0.0	8

4.2 GENERACIÓN DE DESCRIPCIONES CON IA

En el notebook 02_bedrock_generation.ipynb, se utilizó el modelo generativo Claude 3 Haiku de AWS Bedrock para generar descripciones detalladas del perfil crediticio de cada cliente.

Se configuró un cliente de Bedrock Runtime (boto3.client('bedrock-runtime')) y se definió un prompt estructurado en inglés que toma los datos del cliente y genera un resumen en máximo 30 palabras.

4.3 ETIQUETADO DE RIESGO

Luego de generar las descripciones, se utilizó el mismo notebook para clasificarlas automáticamente en “good risk” o “bad risk”.

Se utilizó el modelo Claude 3.5 Sonnet mediante un segundo prompt que toma como entrada la descripción generada y devuelve únicamente la etiqueta.

Se especificó un ejemplo de uso en inglés, y se limitó la respuesta del modelo para evitar ambigüedades.

Este proceso generó la columna target, que fue añadida al dataset enriquecido. El archivo final con descripciones y etiquetas fue exportado como output_target.csv y sirvió de base para el entrenamiento del modelo.

4.4 ENTRENAMIENTO DE MODELO

El entrenamiento se desarrolló en el notebook 03_sagemaker_training.ipynb.

Se cargó el archivo credit_risk_sample_enriched.csv, se codificaron las variables categóricas con LabelEncoder, y se separaron los datos en X_train, X_test, y_train y y_test.

El modelo seleccionado fue Random Forest Classifier, por su alta capacidad de generalización.

Se entrenó localmente, y posteriormente se serializó como local_rf_model.pkl y se guardó en la carpeta /models.

Durante la validación, se obtuvieron las siguientes métricas aproximadas:

- Accuracy: 88.2%
- Precision (bad risk): 76%
- Recall: 72%
- AUC-ROC: 0.91

Además, se generó el archivo label_encoders.pkl para decodificar predicciones posteriormente.

5. CONCLUSIONES

EEl presente proyecto logró implementar un pipeline completo para la clasificación de riesgo crediticio, integrando tecnologías avanzadas de inteligencia artificial generativa y machine learning dentro de un entorno estructurado y reproducible.

Se enriqueció el dataset original mediante descripciones generadas automáticamente con modelos de lenguaje de última generación (Claude 3 Haiku), accedidos a través de AWS Bedrock. Esta fase permitió transformar variables estructuradas en representaciones textuales más expresivas, facilitando el análisis semántico del perfil crediticio de los clientes.

Posteriormente, se realizó una clasificación automatizada de estas descripciones empleando Claude 3.5 Sonnet, permitiendo generar etiquetas binarias (good risk / bad risk) sin necesidad de intervención humana. Esta aproximación demostró la eficacia de los modelos generativos en tareas de preetiquetado y segmentación de riesgos.

A nivel de modelado, se entrenó un algoritmo supervisado de tipo Random Forest, obteniendo métricas sólidas como una precisión superior al 88% y una curva AUC cercana a 0.91, evidenciando una alta capacidad discriminativa del modelo. El proceso de entrenamiento fue acompañado por buenas prácticas de preprocesamiento, validación cruzada y serialización del modelo para su reutilización futura.

Finalmente, se diseñó una arquitectura modular preparada para su integración con APIs de inferencia y dashboards de consulta, considerando la escalabilidad futura hacia ambientes de producción. Este diseño permite no solo la automatización del flujo de datos, sino también su trazabilidad y monitoreo.

En conjunto, el proyecto evidencia una aplicación moderna y efectiva del aprendizaje automático asistido por modelos generativos, con resultados cuantitativos consistentes y una base técnica sólida que permite su evolución hacia un sistema inteligente de soporte a la toma de decisiones financieras.

6. RECOMENDACIONES Y MEJORAS FUTURAS

Para fortalecer aún más el sistema, se sugiere entrenar modelos adicionales como Gradient Boosting, LightGBM o incluso redes neuronales si se dispone de mayor volumen de datos. Además, se recomienda implementar técnicas de explainability como SHAP o LIME para explicar por qué una transacción fue clasificada como fraudulenta. Esto incrementaría la confianza de los usuarios finales y permitiría decisiones más transparentes.

Otra mejora futura sería integrar un sistema automatizado de retroalimentación, en el cual las decisiones de los analistas humanos puedan alimentar nuevamente el modelo para refinar su precisión. Finalmente, se puede usar un dashboard interactivo en Streamlit o Gradio para una experiencia de usuario más profesional.