📄 Credit Card Fraud Detection System

End-to-End Machine Learning Project

1. Introduction

With the rapid growth of online payments and digital transactions, credit card fraud has become a major challenge for financial institutions. Fraudulent transactions lead to significant financial losses and negatively impact customer trust. Traditional rule-based fraud detection systems struggle to detect evolving fraud patterns due to their static nature.

This project presents an end-to-end machine learning–based **Credit Card Fraud Detection System** designed to intelligently identify fraudulent transactions with a strong focus on high fraud recall, while minimizing costly false negatives.

The project covers the complete machine learning lifecycle, including:

- Data understanding
- Exploratory Data Analysis (EDA)
- Feature engineering
- Model selection and evaluation
- Model deployment using Streamlit

---

2. Problem Statement

Objective

To build a machine learning model that predicts whether a credit card transaction is **fraudulent** or **normal**, based on transaction, temporal, and geographical attributes.

Why This Problem Is Challenging

- Fraudulent transactions are extremely rare.
- The dataset is highly imbalanced.
- Accuracy alone is misleading.
- False negatives (missed fraud) are very costly in real-world systems.

Business Requirement

In fraud detection systems, **catching fraud (high recall)** is more important than achieving high overall accuracy.

---

3. Dataset Description

Dataset Source

A publicly available credit card transaction dataset widely used for fraud detection research and machine learning experimentation.

Dataset Size (Exact & Transparent)

- **The original dataset contains 1,048,576 transaction records.**
- **Each record represents a single credit card transaction with customer, merchant, temporal, and geographical attributes.**
- **The dataset includes both fraudulent and normal transactions, making it suitable for supervised learning.**

Class Distribution (Original Dataset)

Class	Count	Percentage
Normal Transactions	~1,042,570	~99.43%
Fraudulent Transactions	~6,006	~0.57%
Total	1,048,576	100%

⚠ The dataset is extremely imbalanced, with fraud cases forming less than 1% of total transactions.
This imbalance heavily influenced feature engineering, model choice, and evaluation metrics.

Data Used for Modeling

- **Due to the large dataset size and computational constraints, the data was:**
  - **Cleaned**
  - **Stratified to preserve fraud ratio**
- **The dataset was then split into:**
  - **Training set (~80%)**
  - **Testing set (~20%)**

This resulted in approximately 209,715 transactions in the test set, which was used for model evaluation.

✅ This approach preserves real-world data characteristics while ensuring efficient and reliable model training.

Key Features Used

Feature	Description
amt	Transaction amount
category	Merchant category
gender	Customer gender
state	Customer state
zip	Customer ZIP code
lat, long	Customer location
merch_lat, merch_long	Merchant location
city_pop	Population of customer city
trans_date_trans_time	Transaction timestamp
is_fraud	Target variable (0 = Normal, 1 = Fraud)

---

4. Exploratory Data Analysis (EDA)

Why EDA Was Necessary

EDA was performed to:

- Understand feature distributions.
- Quantify class imbalance.
- Identify fraud-related behavioral patterns.
- Guide feature engineering decisions.

Key Insights from EDA

1️⃣ Class Imbalance

- Fraud cases are extremely rare.
- Accuracy is not a reliable evaluation metric.
- Recall and ROC-AUC are more meaningful.

2️⃣ Transaction Amount

- Fraud transactions often occur at unusual or higher amounts.
- Normal transactions follow smoother distributions.

3️⃣ Time-Based Patterns

- Fraud occurs more frequently during odd hours.
- Late-night and early-morning transactions show higher fraud probability.

4️⃣ Location Patterns

- Large geographical distance between customer and merchant increases fraud likelihood.
- City population impacts transaction behavior.

Note: EDA directly influenced feature engineering and model selection.

---

5. Feature Engineering

Why Feature Engineering Matters: Raw transaction data is rarely optimal for machine learning. Feature engineering transforms raw features into more informative representations that improve model performance.

Feature Engineering Steps

🔹 Time-Based Features

Extracted from transaction timestamps to capture customer behavior patterns:

- Transaction hour
- Day of week
- Weekend indicator

🔹 Amount Transformation

Applied log transformation to transaction amount ($log(x+1)$) to handle skewed distributions and extreme values.

🔹 Geographical Features

Calculated customer–merchant distance using latitude and longitude. Large distances often indicate abnormal transactions.

🔹 Categorical Encoding

Categorical features (Merchant category, Gender, State) were encoded using One-Hot Encoding with handle_unknown="ignore", ensuring production safety.

🔷 Feature Selection

Only meaningful features were retained to reduce noise and prevent overfitting.

---

6. Model Selection Strategy

Why Multiple Models Were Evaluated

Different models were tested to compare fraud recall, precision, stability on imbalanced data, and generalization ability.

Models Evaluated

1️⃣ Logistic Regression

- Simple baseline model.
- Poor fraud recall.
- Unable to capture complex patterns.

2️⃣ Decision Tree

- Captures non-linear relationships.
- Prone to overfitting.
- Unstable on imbalanced datasets.

3️⃣ Random Forest Classifier

- *Why used:* Ensemble learning reduces overfitting; handles non-linear feature interactions.
- *Observation:* Showed very strong validation performance, but generalization and fraud recall were not optimal compared to XGBoost.

4️⃣ XGBoost (Final Model)

- *Why selected:* Designed for tabular data, handles class imbalance effectively, and learns complex feature interactions.
- **Result:** Best overall generalization performance.

---

7. Model Evaluation

Evaluation Metrics Used

Metric	Reason
Recall (Fraud)	**Most important** – catching fraud
Precision	Avoid excessive false alerts
F1-Score	Balance between precision & recall
ROC-AUC	Overall discriminative power

Final XGBoost Performance (Approximate)

- **Accuracy:** ~98%
- **Fraud Recall:** ~96%
- **ROC-AUC:** Highest among all tested models

Key Achievement: Despite high accuracy, the focus remained on strong **fraud recall**, aligning with real-world business requirements.

---

8. Model Serialization

The final trained XGBoost pipeline was saved using joblib:

Python

joblib.dump(xgb_model, "fraud_xgb_model.pkl")

This enables:

- Fast loading.
- Reusability during deployment.
- Clear separation of training and inference.

---

9. Deployment Using Streamlit

Why Streamlit?

- Fast deployment.
- Minimal frontend complexity.
- Interactive UI.

- Ideal for ML demonstrations.

Application Features

- **User Inputs:** Transaction amount, Time-related features, Customer/Merchant location, Category, Gender, City population.
- **Output:** Fraud probability and clear classification.
    - ✅ Normal Transaction
    - 🚨 Fraudulent Transaction

---

10. End-to-End Workflow Summary

1. Problem understanding
2. Data exploration (EDA)
3. Feature engineering
4. Model experimentation
5. Model selection (XGBoost)
6. Evaluation with correct metrics
7. Model serialization
8. Streamlit deployment

---

11. Real-World Impact

This project reflects industry-aligned practices used in fraud detection systems and demonstrates how machine learning can:

- Reduce financial loss.
- Improve fraud detection efficiency.
- Support scalable ML-based security solutions.

---

12. Limitations & Future Improvements

Current Limitations

- Offline batch training.

- No real-time transaction stream.
- Static decision threshold.

Future Enhancements

- Real-time fraud detection pipeline.
- Automated model retraining.
- Explainable AI (SHAP) integration.
- API-based deployment.

---

13. Conclusion

This project demonstrates a complete machine learning pipeline focused on solving a real-world problem using best practices. Special attention was given to imbalanced data handling, feature engineering, and correct metric selection, making the solution both practical and industry-relevant.

✨ **Key Takeaway:** This project is not about maximizing accuracy — **it is about making the right ML decisions for real-world impact.**