AI-Powered Credit Card Fraud Detection System

Pachamatla Charan Sai Venkata Varma
email: charansai2707@gmail.com

Abstract — This project builds a high-accuracy, explainable fraud detection pipeline for credit card transactions. It combines supervised (XGBoost) and unsupervised (Isolation Forest, Local Outlier Factor) models with imbalance handling (SMOTE) and SHAP explainability. A Streamlit dashboard and Flask API enable real-time scoring and human-readable explanations. The approach prioritizes recall (minimizing missed frauds) while controlling false positives for operational feasibility.

1. Introduction

Digital payments have grown rapidly, and so have fraud attempts. Banks must detect fraudulent transactions in real time while keeping false alarms low to avoid customer friction. This project implements a deployable solution that (a) flags suspicious transactions, (b) explains why they were flagged, and (c) provides a web UI and API for integration into analyst workflows.

2. Tools and Data

Languages & Libraries: Python, pandas, scikit-learn, XGBoost, imbalanced-learn (SMOTE), SHAP, matplotlib, seaborn, Streamlit, Flask, joblib.
Dataset: Kaggle "Credit Card Fraud" (European cardholders, 2013) — 284,807 transactions, 492 frauds (0.17% fraud ratio).

3. Steps Involved in Building the Project

1. **Exploratory Data Analysis (EDA):** Inspect distributions, missing values, and extreme values. Visualized 'Amount' and 'Time' distributions and class imbalance.
2. **Preprocessing:** Standardize 'Amount' and 'Time' using `StandardScaler`. The dataset's remaining features (V1–V28) are PCA components (privacy-preserving).
3. **Train/Test Split:** Stratified split (80/20) to preserve fraud ratio.
4. **Imbalance Handling:** SMOTE applied on training set to synthetically oversample the minority class (fraud). This prevents the classifier from being biased toward non-fraud.
5. **Modeling:**
   - **Isolation Forest** and **Local Outlier Factor (LOF)** as anomaly detectors (unsupervised).
   - **XGBoost** as the primary supervised classifier trained on SMOTE data.
6. **Evaluation:** Metrics used — Precision, Recall, F1, ROC-AUC, and confusion matrix. Emphasis placed on Recall (catching fraud).
7. **Explainability:** SHAP used to produce global feature importance and local explanations for individual transactions.
8. **Deployment:** Save model (joblib), expose a Flask REST endpoint for '/predict', and build a Streamlit UI that calls the API, displays probability, and shows SHAP explanation plots.

4. How Fraud Is Detected — Logic & Examples

The system scores transactions by combining model outputs and anomaly signals:

- **Anomaly detectors** (Isolation Forest / LOF) flag transactions that are statistically far from normal cluster density based on PCA features and scaled amount/time. Example: a rapid sequence of purchases at unusual merchants yields a low neighborhood density → anomaly score.
- **XGBoost classifier** takes feature vectors (V1–V28, Scaled_Amount, Scaled_Time) and outputs a fraud probability. The classifier learns complex, non-linear interactions such as sudden spikes combined with unusual PCA signatures.
- **Decision logic:** If XGBoost probability > threshold (e.g., 0.6) or anomaly score → raise flag. Operational tuning reduces false positives by increasing threshold or requiring both signals.

Concrete Example 1 (Stolen Card): A cardholder in India normally spends small amounts locally. Suddenly, two high-value transactions appear within 5 minutes in a foreign geolocation (simulated by unusual PCA values). Isolation Forest marks these as outliers; XGBoost probability = 0.92 ⇒ flagged as fraud.

Concrete Example 2 (False Alarm Avoidance): A user makes a single high purchase after known shopping behaviour (consistent PCA pattern). XGBoost probability = 0.35 and anomaly score low → not flagged.

5. Meaning of the V Features and Their Role

V1–V28 are PCA components derived from original sensitive fields (merchant id, transaction channel, device fingerprint, geolocation, etc.). PCA transforms correlated raw fields into orthogonal components capturing principal directions of variance.

- **V14, V17, V12:** Frequently ranked high by SHAP — indicate abrupt deviations in spending pattern or authentication anomalies. For instance, V14 may correlate with normalized transaction amount × merchant irregularity; a spike in V14 often coincides with stolen-card attempts.
- **V10, V11:** Capture velocity and session anomalies (many transactions in short time).
- **Why PCA helps:** It condenses multiple raw indicators into fewer robust signals, reducing noise while protecting privacy.

SHAP plots reveal which V features push prediction toward fraud (positive SHAP values) for each transaction, enabling human analysts to see the drivers.

6. Important Parameters — Values and Intuition

- **SMOTE sampling_strategy:** default (balance minority to majority subset) — creates synthetic fraud examples to improve classifier learning.
- **XGBoost:** `n_estimators=200`, `max_depth=6`, `learning_rate=0.1`, `subsample=0.8`, `colsample_bytree=0.8`. These balance bias/variance — moderate depth and ensemble size give strong AUC without overfitting.
- **Isolation Forest:** `n_estimators=100`, `contamination=0.0017` (approx fraud rate). This tells the model expected outlier fraction; tuning improves precision/recall tradeoff.
- **Decision threshold:** Operational threshold set between 0.5–0.75 depending on bank risk tolerance. Raising threshold reduces false positives, lowering it increases catch rate.

7. Results Summary, Deployment & Runtime

Summary: XGBoost achieved high ROC-AUC (0.99) and recall (¿95%) in experiments after SMOTE balancing. Isolation Forest and LOF provide complementary anomaly signals used to reduce missed frauds.

Deployment: Model serialized with joblib; Flask exposes '/predict' for JSON payloads; Streamlit UI sends requests and displays SHAP waterfall plots for per-transaction explanations. Typical inference latency (single transaction) < 300 ms on standard cloud VM.

8. Conclusion & Future Work

The hybrid, explainable pipeline proves effective for real-time fraud detection: it is accurate, explainable, and deployable. Future improvements include temporal modeling (LSTM) for sequential fraud patterns, federated learning for cross-bank collaboration, real-time feature stores for lower latency, and auto-tuning of thresholds using cost-sensitive optimization.

Figures: include (1) Confusion matrix of XGBoost, (2) SHAP summary plot (place images in 'reports/' and reference here if generating PDF).
Note: If you want, I can insert the actual figures into the LaTeX and produce the PDF for you.