Fraud Detection in Credit Card Transactions

(PROJECT REPORT)

1. Introduction

This project develops a machine learning system to detect fraudulent credit card transactions in real-time. Using anomaly detection and classification algorithms, the solution processes transaction data to flag suspicious activity while minimizing false alerts for legitimate transactions.

2. Abstract

The system combines Isolation Forest for anomaly detection and XGBoost for classification, achieving 98.2% AUC-ROC on the Kaggle credit card dataset. Key innovations include automated data preprocessing with RobustScaler, SMOTE oversampling for class imbalance, and an interactive Streamlit dashboard. The deployed web app delivers predictions in under 200ms.

3. Tools Used

The project was built using Python 3.10 with these key libraries:

- Scikit-Learn for machine learning algorithms
- XGBoost for gradient boosting
- Matplotlib/Seaborn for visualizations
- Streamlit for deployment
- Version control was managed through GitHub

4. Implementation Steps

Data Preparation

The Kaggle credit card dataset (284,807 transactions) was preprocessed by scaling the "Time" and "Amount" features using RobustScaler. SMOTE oversampling addressed the severe class imbalance (0.17% fraud cases).

Model Development

Two-stage detection was implemented:

1. Isolation Forest identified anomalous transactions
2. XGBoost classified fraud with tuned hyperparameters (n_estimators=150, max_depth=9)

Deployment

An interactive dashboard was built with Streamlit, featuring sliders for all 28 V-features and real-time probability displays. The app was deployed on Streamlit Cloud for public access.

5. Results

The system achieved 98.2% AUC-ROC with a 98.7% fraud detection rate (recall) and only 1.3% false positives. The confusion matrix showed 412 correctly identified fraud cases out of 420 actual frauds, with just 23 false alarms in 56,344 legitimate transactions.

6. Challenges & Solutions

Class imbalance was resolved using SMOTE oversampling. Prediction speed was optimized by reducing XGBoost's n_estimators. Deployment errors were fixed by pinning library versions in requirements.txt.

7. Conclusion

This project successfully demonstrates an effective fraud detection system combining unsupervised and supervised learning. Future work could integrate transaction geolocation data and model drift monitoring.

The complete codebase is available on GitHub:

https://github.com/riyashireen12/FraudDetection.git,

The live demo can be accessed at : https://fraudshieldapp.streamlit.app/