Project Report - Real-time Fraud Detection API

Author: Bruno Fonkeng
Date: 9th August 2025

1. Business Problem

Credit card fraud is a serious global issue costing billions annually. Financial institutions need solutions that can **detect fraudulent transactions in real time** to prevent losses and protect customers. Traditional fraud detection methods often suffer from high false positives, poor scalability, and inability to adapt quickly to new fraud patterns.

2. Objective

Develop an **end-to-end machine learning system** that:

- Trains a fraud detection model on real-world transaction data
- Handles class imbalance with oversampling techniques
- Deploys the model as a **REST API**
- Provides a **user-friendly frontend** for prediction
- Is scalable, containerized, and cloud-deployed

3. Data Overview

Dataset: Credit Card Fraud Dataset (Kaggle)

- **Rows:** 284,807 transactions
- **Features:** 30 numerical features (PCA transformed) + Class label (0 = Non-fraud, 1 = Fraud)
- **Imbalance:** Only 0.172% of transactions are fraudulent

4. Methodology / Pipeline

Step 1 - Data Exploration

- Checked missing values, outliers, feature distributions
- Verified heavy class imbalance

Step 2 - Preprocessing

- Applied **SMOTE** (Synthetic Minority Oversampling Technique) to balance classes
- Normalized/standardized features

Step 3 - Model Training

- Trained and evaluated **Random Forest** and **Logistic Regression**
- Evaluation metric: AUC Score
- Selected Random Forest (AUC ≈ 0.9609) for deployment

Step 4 - Backend Development

- Created REST API using **FastAPI**
- /predict endpoint accepts JSON with 30 scaled features
- Returns fraud probability and classification

Step 5 - Frontend Development

- Built **Streamlit** UI for easier interaction
- Allows manual input via sliders or CSV upload

Step 6 - Deployment

- Dockerized both backend and frontend
- Deployed to **Render** (separate services for API & UI)
- Configured **CORS** for cross-origin access

---

5. Technologies Used

Category	Tools / Libraries
Programming	Python 3.10
ML & Data	scikit-learn, imbalanced-learn, NumPy, pandas
Visualization	Matplotlib, Seaborn
Backend	FastAPI, Uvicorn
Frontend	Streamlit
Deployment	Docker, Render

---

6. Results

Model	AUC Score
Logistic Regression	0.9619
Random Forest	0.9609

- Fraud prediction accuracy: **94%**
- API response time: **<200ms**

---

7. System Architecture

Components:

1. **User Input** (via Streamlit UI)
2. **REST API** (FastAPI)
3. **Deployed Model** (Random Forest)
4. **Cloud Hosting** (Render)

---

8. Live Demo Links

- **Frontend UI (Streamlit)**: https://fraud-detection-api-frontend.onrender.com
- **Backend API (FastAPI)**: https://fraud-detection-api-27ae.onrender.com
- **Swagger Docs**: https://fraud-detection-api-27ae.onrender.com/docs

---

9. Future Improvements

- Add authentication & API key security
- Integrate monitoring/logging tools
- Improve model with Gradient Boosting or Neural Networks
- Implement batch transaction processing
- Deploy on AWS/GCP with autoscaling

---

10. Conclusion

This project demonstrates how to take a machine learning model from raw data to a fully deployed, production-ready API with a working frontend. The architecture is scalable and can be extended to handle other real-time classification problems.