Project Report: Credit Card Fraud Detection Using Machine Learning

1. Introduction

1.1 Project Overview:

This project focuses on detecting fraudulent credit card transactions using machine learning techniques. The dataset consists of transactions made by European cardholders, containing anonymized features along with transaction time, amount, and a binary class label indicating whether the transaction is fraudulent (1) or legitimate (0). The dataset is highly imbalanced, with only 492 fraudulent transactions out of 284,807 total transactions.

1.2 Objectives:

- Data Collection and Preparation: Load and explore the dataset to understand its structure and characteristics.
- Exploratory Data Analysis (EDA): Analyze the distribution of transactions, identify patterns, and visualize key features.
- Model Development: Implement machine learning models to classify transactions as fraudulent or legitimate.
- Evaluation: Assess model performance using appropriate metrics, considering the class imbalance.

2. Data Collection and Preparation

2.1 Dataset Description:

- Time: Seconds elapsed between each transaction and the first transaction.
- Amount: Transaction amount.
- Class: Binary label (0 = legitimate, 1 = fraudulent).
- V1-V28: Anonymized features resulting from PCA transformations for confidentiality.

2.2 Data Loading and Initial Exploration:

- Loaded from a PostgreSQL database into a pandas DataFrame.
- No missing values.
- Class imbalance: 99.83% legitimate vs 0.17% fraudulent.

2.3 Data Cleaning and Validation:

- Verified absence of null values.
- Confirmed binary Class variable.
- Validated row counts against the SQL database.

3. Exploratory Data Analysis (EDA)

3.1 Summary Statistics:

- Mean transaction amount: $88.35, standard deviation: $250.12.
- Maximum amount: $25,691.16.

3.2 Distribution of Transactions:

- Class imbalance visualized using count plots.
- Most transactions are low value, few high-value outliers.

3.3 Key Observations:

- Fraudulent transactions are rare but crucial to detect.
- Specialized imbalance techniques are necessary.

4. Data Preprocessing

4.1 Feature Scaling:

- StandardScaler applied to Time and Amount.

4.2 Handling Class Imbalance:

- SMOTE used to generate synthetic fraud samples.

4.3 Train-Test Split:

- 80% training and 20% testing split.

5. Model Development

5.1 Model Selection:

- Random Forest Classifier: Robust and suitable for imbalance.
- XGBoost Classifier: Effective gradient boosting method.

5.2 Hyperparameter Tuning:

- Optuna used to optimize tree count, depth, and learning rate.

6. Model Evaluation

6.1 Performance Metrics:

- Precision: Correct fraud predictions.
- Recall: Detecting all fraud cases.
- F1-Score: Balance of precision and recall.
- ROC-AUC: Class separation power.

6.2 Results:

- Random Forest: High precision/recall, ROC-AUC = 0.98.

- XGBoost: Slight improvement in recall.

6.3 Confusion Matrix:

- Showed true/false positives and negatives.

7. Conclusion and Recommendations

7.1 Key Findings:

- High performance even with imbalanced data.

- SMOTE and scaling improved model performance.

7.2 Recommendations:

- Real-time deployment for fraud detection.

- Regular updates and threshold tuning.

7.3 Future Work:

- Explore deep learning.

- Add contextual features like merchant or location.

Appendix: Code and Libraries Used

Libraries:

- pandas, numpy: Data manipulation

- matplotlib, seaborn: Visualization

- scikit-learn: Modeling and metrics

- imbalanced-learn: SMOTE

- Optuna: Hyperparameter tuning

- XGBoost: Model building

Key Snippets:

- Data Loading:

df = pd.read_sql_query("SELECT * FROM cc_data", engine)

- SMOTE Application:

smote = SMOTE(random_state=42)

Project Report: Credit Card Fraud Detection Using Machine Learning

X_resampled, y_resampled = smote.fit_resample(X_train, y_train)

- Model Training (XGBoost):

model = XGBClassifier(use_label_encoder=False, eval_metric='logloss')

model.fit(X_resampled, y_resampled)

This project shows how machine learning can reduce fraud and financial loss effectively.