1. Methodology

1. Data Preprocessing

The project leveraged the **Credit Card Fraud Detection Dataset** (Kaggle), containing 284,807 transactions (492 fraudulent). Key steps included:

- **Standardization**: Scaled numerical features (Time and Amount) using StandardScaler.
- **Class Imbalance Handling**: Applied **SMOTE** to synthetically oversample the minority class (fraudulent transactions).
- **Feature Engineering**:
  - Extracted **time-based patterns** (hour of day, day of week, hourly transaction frequency).
  - Added **aggregated statistics** (e.g., average transaction amount per user).

2. Model Evaluation Metrics

Models were evaluated using:

- **Precision**: Minimize false positives.
- **Recall**: Maximize fraud detection.
- **F1-Score**: Balance precision and recall.
- **AUC-ROC**: Measure class separation capability.

2. Model Selection

Supervised Models

1. **Logistic Regression**:
   - **Strengths**: Simple, interpretable, efficient for binary classification.
   - **Weaknesses**: Struggles with imbalanced data without weighting/SMOTE.
2. **XGBoost**:
   - **Strengths**: Handles imbalanced data via scale_pos_weight, captures non-linear patterns.
   - **Weaknesses**: Requires hyperparameter tuning for optimal performance.

Unsupervised Models

1. **Isolation Forest**:
   - **Strengths**: Detects anomalies by isolating outliers, no labels required.
   - **Weaknesses**: Low precision due to high false positives.
2. **One-Class SVM**:
   - **Strengths**: Learn a decision boundary for normal transactions.
   - **Weaknesses**: Sensitive to parameter tuning (e.g., nu).

Results

Model	Precision	F1-Score	AUC-ROC
Logistic Regression	0.85	0.81	0.92
XGBoost	0.90	0.86	0.95
Isolation Forest	0.10	0.18	0.89
One-Class SVM	0.05	0.09	0.89

Key Insight: XGBoost outperformed all models with an **F1-Score of 0.86** and **AUC-ROC of 0.95**.

3. Improvements

Class Imbalance Mitigation

- **SMOTE** improved recall for supervised models by balancing class distribution.
- **Class Weighting** in XGBoost (scale_pos_weight) further enhanced fraud detection.

Feature Engineering

- **Time-Based Features**: Captured temporal fraud patterns (e.g., higher fraud frequency at night).
- **Aggregated Metrics**: Identified anomalies in user transaction behavior (e.g., sudden spikes in spending).

4. Future Optimization

Hyperparameter Tuning

- Use **GridSearchCV/RandomizedSearchCV** to optimize:
    - XGBoost: max_depth, learning_rate, n_estimators.
    - Isolation Forest: contamination level.

Testing on Additional Datasets

- Validate robustness on datasets like **IEEE-CIS Fraud Detection** or **PaySim** to assess generalizability.

Deployment

- Deploy the XGBoost model via **Flask/FastAPI** as a real-time fraud detection API.
- Host on cloud platforms (AWS, Heroku) for scalability.

Feature Enhancements

- Incorporate **geolocation data** (e.g., transaction distance from user's home).
- Analyze **behavioral patterns** (e.g., transaction frequency per user).

Ensemble Methods

- Combine XGBoost with Isolation Forest using **stacking** to leverage supervised and unsupervised strengths.

5. Conclusion

The XGBoost model emerged as the optimal solution for fraud detection, balancing precision and recall. While unsupervised models (Isolation Forest, One-Class SVM) showed high recall, their low precision limits practicality. Future work should focus on hyperparameter tuning, deployment, and feature engineering to enhance real-world applicability.

GitHub Repository: https://github.com/sabinachou/fraud-detection
Contact: Chiachen (Sabina) Chou