**Credit Card Fraud Detection - Project Report**

**Introduction**

This project focused on building a fraud detection system to classify credit card transactions as fraudulent or legitimate. The dataset used was the anonymized Credit Card Fraud Detection dataset available on Kaggle, which contains transactions made by European cardholders in September 2013.

**Objective**

* Detect fraudulent transactions with high accuracy and minimal false positives
* Handle imbalanced class distribution
* Implement models and evaluate them using relevant metrics
* Analyze features and their impact on fraud prediction

**Dataset Overview**

* Total transactions: 284,807
* Fraudulent transactions: 492 (0.172%)
* Features: 30 (V1–V28 PCA components, Time, Amount)
* Label: Class (1 = Fraud, 0 = Legitimate)

**Data Preprocessing**

* Normalized 'Amount' and 'Time' features using standard scaling
* Handled severe class imbalance using techniques like:
  + Undersampling the majority class
  + Oversampling the minority class using SMOTE
* Split the dataset using stratified sampling to preserve fraud ratio in train/test sets

**Model Implementation**

* Models used:
  + Logistic Regression
  + Decision Tree
  + Random Forest
  + XGBoost
* Evaluation metrics:
  + Precision, Recall, F1-score, AUC-ROC (due to class imbalance)

**Personal Findings**

* Accuracy was misleading due to the class imbalance
* AUC-ROC and precision-recall curves were more informative
* XGBoost provided the best balance between recall and precision
* Feature scaling significantly improved convergence for logistic regression

**Problems Faced**

1. **Class Imbalance**:
   * Challenge: Majority class dominated predictions, leading to poor fraud detection
   * Solution: Applied resampling (SMOTE and undersampling) and focused on precision-recall metrics
2. **Feature Interpretability**:
   * Challenge: Most features were anonymized due to PCA transformation
   * Solution: Relied on feature importance rankings from tree-based models
3. **Overfitting in Decision Trees**:
   * Challenge: Tree-based models easily overfit on minority class
   * Solution: Tuned hyperparameters like max depth and used cross-validation
4. **Runtime and Memory**:
   * Challenge: Dataset was large and required optimized processing
   * Solution: Used mini-batch processing and efficient libraries like XGBoost

**Counter Actions and Solutions**

* Chose evaluation metrics that accounted for imbalance (recall, F1)
* Performed extensive grid search for model tuning
* Used ensemble methods and cross-validation for robustness

**Performance Summary (Post-Tuning)**

| **Model** | **Precision** | **Recall** | **F1 Score** | **AUC-ROC** |
| --- | --- | --- | --- | --- |
| Logistic Reg. | ~0.83 | ~0.62 | ~0.71 | ~0.94 |
| Decision Tree | ~0.76 | ~0.68 | ~0.72 | ~0.90 |
| Random Forest | ~0.89 | ~0.76 | ~0.82 | ~0.97 |
| XGBoost | ~0.91 | ~0.80 | ~0.85 | ~0.98 |

**Conclusion**

This project emphasized the complexity of dealing with imbalanced datasets in fraud detection. Techniques like SMOTE, tree-based models, and proper evaluation metrics were key to improving performance. The project laid the groundwork for deploying a real-time fraud monitoring system in the future using advanced models and streaming data.