Summary Report: Predicting Credit Card Fraud

1. Problem Statement

Detecting credit card fraud is essential to prevent customers from being billed for unauthorized transactions. The dataset consists of credit card transactions made by European cardholders in September 2013, totaling 284,807 transactions over two days, of which 492 are identified as fraudulent.

2. Dataset Overview

- Total Transactions: 284,807
- Fraudulent Transactions: 492
- Non-Fraudulent Transactions: 284,315
- Class Imbalance: Fraudulent transactions represent only 0.172% of the total, highlighting the severe class imbalance.

3. Exploratory Data Analysis (EDA)

- Key Findings:
- Feature Distribution: The dataset features, named V1 to V28, have been processed using PCA.

- Class Imbalance: Visualizations confirm the significant class imbalance.
- Transaction Amounts: Fraudulent transactions often involve smaller amounts.
- Class Distribution: The imbalance in fraudulent vs. non-fraudulent transactions is crucial for model training and evaluation.

4. Data Cleaning and Preprocessing

- Standardization: The Amount feature was standardized.
- Missing Values: No missing values were found in the dataset.
- Time Feature: Initially considered but later dropped due to low correlation with the target variable.

5. Dealing with Imbalanced Data

- Resampling Techniques:
- Under-sampling: Reducing the number of non-fraudulent transactions.
- Over-sampling: Increasing the number of fraudulent transactions using methods like SMOTE and ADASYN.

6. Feature Engineering

- Interaction Terms: Created to capture relationships between features.
- Feature Scaling: Min-Max scaling was applied to ensure uniformity across all features.

7. Model Building and Evaluation

- Models Used:

- Logistic Regression (LR)

- K-Nearest Neighbors (KNN)

- Decision Tree (DT)

- XGBoost (XGB)

- Performance Metrics:

- Accuracy: Correctly predicted transactions over total transactions.

- Precision: Correctly predicted fraudulent transactions over all predicted fraudulent transactions.

- Recall: Correctly predicted fraudulent transactions over all actual fraudulent transactions.

- Sensitivity and Specificity: For non-fraudulent transactions.

- F1 Score: Harmonic mean of precision and recall.

- ROC-AUC: Area under the receiver operating characteristic curve.

- Model Results:

- Detailed performance metrics for each model, including accuracy, precision, recall, F1 score, and ROC-AUC.

8. Hyperparameter Tuning

- Logistic Regression:

- Solver: Best found - liblinear

- C: Best found - 0.1

- Decision Tree:

- Criterion: Best found - gini

- Max Depth: Best found - 10

- Min Samples Split: Best found - 2

- Min Samples Leaf: Best found - 1

- XGBoost:

- Learning Rate: Best found - 0.2

- Max Depth: Best found - 5

- Subsample: Best found - 0.7

- Colsample_bytree: Best found - 1.0

- Number of Estimators: Best found - 300

Hyperparameter tuning was conducted using grid search and cross-validation to ensure optimal parameter selection.

9. Model Deployment

The best-performing model was serialized using the pickle module for deployment:

- Save the model to a file:

```python
with open('best_xgb_model.pkl', 'wb') as file:
pickle.dump(model, file)
```

- Load the model from the file:

```python
with open('best_xgb_model.pkl', 'rb') as file:
best_xgb = pickle.load(file)
```

- Use the loaded model to make predictions:

```python
preds = best_xgb.predict(X_test)
```

Conclusion:

This project effectively applied various machine learning techniques to detect fraudulent transactions in a highly imbalanced dataset. By addressing class imbalance with strategies like SMOTE and using a mix of classifiers, we achieved high performance metrics, particularly in precision and recall, which are vital in fraud detection. The high ROC-AUC score indicates the model's effectiveness in distinguishing between fraudulent and non-fraudulent transactions. However, further improvements can be made to increase the recall rate, minimizing false negatives, which are crucial in fraud detection scenarios.