Credit Card Fraud Detection Project Report

1. Introduction

Context and Importance:

Credit cards are among the most widely used financial products for online purchases and payments. While they offer convenience, they also pose significant risks, particularly related to credit card fraud. Credit card fraud involves the unauthorized use of someone else's credit card or credit card information to make purchases or withdraw cash. It is crucial for credit card companies to recognize fraudulent transactions promptly to prevent unauthorized charges to customers' accounts.

Goal:

The primary objective of this project is to build a machine-learning model to detect fraudulent credit card transactions. This will help credit card companies mitigate risks and protect their customers.

Dataset:

The dataset contains transactions made by European cardholders in September 2013. It covers transactions that occurred over two days, including 492 frauds out of 284,807 transactions. Given this, the dataset is highly unbalanced, with the positive class (frauds) accounting for only 0.172% of all transactions.

The dataset has the following structure:

- **Rows**: Each row represents a transaction.
- **Columns**: There are 31 columns:
    - **Time**: The time elapsed between this transaction and the first transaction in the dataset (in seconds).
    - **V1 to V28**: The result of a PCA (Principal Component Analysis) transformation applied to the original features (for confidentiality reasons).
    - **Amount**: The transaction amount.
    - **Class**: The class label, where 1 indicates a fraudulent transaction and 0 indicates a legitimate transaction.

2. Data Exploration and Preprocessing

Data Loading and Initial Exploration:

- The dataset is loaded using pandas, and initial exploration involves checking the first few rows, summary statistics, and data types of the columns.

Summary Statistics:

- Summary statistics provide insights into the distribution and scale of the data.

Checking for Missing Values:

- The dataset is checked for missing values, and it was found that there are no missing values in any of the columns. This ensures that the dataset is complete and ready for analysis without the need for imputation.

Distribution of Classes:

- The distribution of the classes (fraudulent vs. non-fraudulent transactions) is examined to understand the class imbalance, which is crucial for model training and evaluation.

Handling Data Imbalance:

As we see that the data is heavily imbalanced, several approaches are employed to handle this imbalance:

- **Under sampling**:
    - For balancing the class distribution, the number of non-fraudulent transactions is reduced to match the count of fraudulent transactions (492).
- **Oversampling**:
    - The number of fraudulent transactions is increased to match the count of non-fraudulent transactions.
- **SMOTE (Synthetic Minority Over-sampling Technique)**:
    - This oversampling technique uses the nearest neighbor algorithm to create synthetic data points, thereby increasing the number of fraudulent transactions.
- **ADASYN (Adaptive Synthetic Sampling)**:
    - Similar to SMOTE, ADASYN generates synthetic data, but it focuses on creating data points in regions with low density of minority class samples.

---

3. Model Building and Evaluation

Model Selection:

Various machine learning models are considered for the task:

- Logistic Regression
- Decision Trees
- XGBoost

Training and Validation:

- The dataset is split into training and testing sets to evaluate model performance on unseen data.
- Cross-validation techniques are employed to ensure the model's robustness and to avoid overfitting.

Performance Metrics:

- Key performance metrics include accuracy, precision, recall, F1-score, and the Area Under the Receiver Operating Characteristic Curve (ROC-AUC).
- A confusion matrix is used to visualise the performance of the classification models.

Hyperparameter Tuning:

- Hyperparameters of the models are tuned using techniques like GridSearchCV to find the optimal parameters that yield the best performance.

Choosing the Best Model:

- Various models were built using the different balanced datasets (Undersampling, Oversampling, SMOTE, and ADASYN).
- While many models performed well, it's important to consider the best-performing model based on key factors:
    - Under sampling techniques, despite good performance, result in loss of information and are less preferred.
    - Models built with SMOTE and ADASYN generally performed well.
    - The Logistic Regression model using the SMOTE balanced dataset showed excellent performance with an ROC score of 0.99 on the train set and 0.97 on the test set.
    - The Logistic Regression model is chosen for its simplicity, ease of interpretation, and lower computational resource requirements compared to more complex models like XGBoost.

---

4. Results and Insights

Summary of Findings:

- Key insights from the data exploration phase are summarized, including any significant patterns or anomalies detected.

Feature Importance:

- Important features impacting the model's decisions are identified, providing insights into which factors are most indicative of fraud.

Model Performance:

- The final model's performance is reported on the test set, comparing it against baseline models and discussing any improvements made through tuning.
- The chosen Logistic Regression model with SMOTE balancing technique demonstrated high recall and a strong ROC-AUC score, making it a suitable choice for deployment.

---

5. Model Deployment

Saving the Best Model:

- The best-performing model is serialized using the pickle module for deployment.
- This allows the model to be saved to disk and loaded later without needing to retrain it.

Python

```
# Logistic regression 'best_model' is the trained model that is performed

with open('best_model.pkl', 'wb') as file:

pickle.dump(best_model, file)
```

Loading and Using the Model:

- The saved model can be loaded and used for prediction on new data as follows:

Python

```
with open('best_model.pkl', 'rb') as file:

loaded_model = pickle.load(file)

# Example prediction

predictions = loaded_model.predict(new_data)
```

Deployment Using Streamlit:

- Streamlit is used to create an interactive web application for credit card fraud detection.
- The app allows users to input transaction details and get real-time predictions about the likelihood of fraud.

---

6. Cost-Benefit Analysis

Model Selection Criteria:

- While selecting the best model, it's important to consider the required infrastructure, resources, and computational power.
- Complex models like Random Forest, SVM, and XGBoost require heavy computational resources, increasing the cost of deployment.
- Simpler models like Logistic Regression require fewer computational resources, reducing the cost of building and deploying the model.

Cost vs. Performance:

- If a small increase in ROC score translates to significant monetary loss or gain, investing in a more complex model might be justified despite higher costs.
- For smaller average transaction values, high precision is crucial to minimize false positives and the burden of manual verification.
- For larger transaction values, high recall is essential to detect actual fraudulent transactions and prevent significant losses.

Conclusion:

- The Logistic Regression model with SMOTE balancing is chosen for its simplicity, high recall, and cost-effectiveness.
- This model is easier to interpret and explain to the business, making it a practical choice for deployment.

7. Summary for Business

Key Points:

- For banks with smaller average transaction values, high precision is desired to minimize false positives and manual verification efforts.
- For banks with larger transaction values, high recall is crucial to detect actual fraudulent transactions and prevent significant losses.
- The chosen Logistic Regression model with the SMOTE balancing technique offers a good balance of high recall and ease of implementation.

Final Recommendation:

- Deploy the Logistic Regression model with SMOTE balancing to detect fraudulent transactions effectively.
- This model provides a cost-effective solution with high recall, ensuring the detection of actual fraudulent transactions while being resource-efficient.