Credit Card Fraud Detection

Problem Statement:

A credit card is one of the most used financial products to make online purchases and payments. Though the Credit cards can be a convenient way to manage your finances, they can also be risky. Credit card fraud is the unauthorized use of someone else's credit card or credit card information to make purchases or withdraw cash.

It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.

The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

We have to build a classification model to predict whether a transaction is fraudulent or not.

Approach:

1. Data Understanding & Preparation:

Step-by-step process:

- **Dataset Characteristics**: The dataset includes features from PCA transformations (V1, V2, ..., V28), along with Time (seconds elapsed) and Amount (transaction value). The features generated through PCA are already Gaussian, so there's no need for normalization.
- **Class Imbalance**: The dataset is imbalanced, with only 0.172% of transactions being fraudulent.

Key Preprocessing Steps:

1. **Handling Class Imbalance**: In this project, class imbalance was addressed using Random Under-Sampling. This method reduces the number of samples from the majority class (non-fraudulent transactions) to balance the dataset. This helps ensure that the model does not become biased toward predicting the majority class, which would lead to poor performance on fraudulent transactions.
2. **Feature Selection**: Features that showed weak correlation with the target feature (fraudulent transactions) were removed. Specifically, columns such as 'V13', 'V15', 'V22', 'V23', 'V24', 'V25', and 'V26' were dropped because they did not contribute much to the prediction of fraudulent transactions. This reduces the dimensionality of the dataset and helps improve model performance by removing irrelevant data.

3. **Data Splitting**: The dataset was split into training and testing sets using the train_test_split function. This ensures that the model is trained on one part of the data and evaluated on another, allowing us to assess its generalization ability.
4. **Handling Time and Amount Features**: The 'Time' and 'Amount' features, which were not transformed using PCA, were retained because they may contain valuable information that helps the model distinguish between normal and fraudulent transactions.

Tools Used:

- **Random Under-Sampling** for handling class imbalance by reducing the majority class.
- **train_test_split** for splitting the data into training and testing sets.
- **Feature Selection** to remove less relevant features and retain those that contribute meaningfully to model performance.

2. Exploratory Data Analysis (EDA):

Step-by-step process:

- **Visualizations**: You may generate boxplots or correlation matrices to identify potential outliers or relationships between features and the target variable (Class).
- **Class Distribution**: Check the distribution of normal vs fraudulent transactions and visualize how features such as Amount and Time behave for both classes.

3. Model Selection:

You have selected various machine learning models to handle the classification problem. Here is the detailed breakdown:

- **Logistic Regression (Initial and Tuned)**:
  - Logistic Regression serves as the baseline model. It is simple, interpretable, and effective for linearly separable data. Hyperparameters like regularization strength (C) are tuned to find the optimal model.
  - **Evaluation Metrics**: ROC-AUC score is primarily used due to the imbalanced nature of the data.
- **Decision Tree Classifier (Initial and Tuned)**:
  - Decision Trees are intuitive but prone to overfitting. Tuning hyperparameters like max_depth, min_samples_split, and min_samples_leaf helps control the depth and complexity of the tree to avoid overfitting.
  - **Evaluation Metrics**: Precision, recall, and the ROC-AUC score are used to measure the model's performance.
- **Random Forest Classifier (Initial and Tuned)**:
  - Random Forest is an ensemble model of decision trees. It provides better generalization by averaging the predictions of multiple decision trees.

  - Hyperparameters like n_estimators (number of trees), max_depth, and max_features are tuned for better performance.
  - **Evaluation Metrics**: As with other models, precision, recall, and ROC-AUC are key metrics for performance.
- **Support Vector Machine (SVM) (Initial and Tuned)**:
  - SVM is effective in high-dimensional spaces and works well for both linear and non-linear classification tasks. Kernel functions (linear, rbf) and the regularization parameter (C) are tuned.
  - **Evaluation Metrics**: ROC-AUC score is the key evaluation metric for SVM.
- **Gradient Boosting Classifier (Initial and Tuned)**:
  - Gradient Boosting models iteratively correct the errors made by previous models. It builds decision trees sequentially, where each tree tries to reduce the errors of its predecessor.
  - Hyperparameters like learning_rate, n_estimators, and max_depth are tuned for better accuracy.
  - **Evaluation Metrics**: Precision, recall, and ROC-AUC score.
- **XGBoost (Initial and Tuned)**:
  - XGBoost is an optimized gradient boosting model that includes features like regularization and parallelization for faster training. It is known for its efficiency and performance.
  - Tuning parameters like learning_rate, n_estimators, max_depth, and subsample is essential for obtaining the best performance.
  - **Evaluation Metrics**: ROC-AUC score and precision/recall metrics.
- **AdaBoost (Initial and Tuned)**:
  - AdaBoost is an ensemble method that builds weak learners sequentially and adjusts weights on the misclassified samples.
  - Hyperparameters like n_estimators and learning_rate are tuned for optimal performance.
  - **Evaluation Metrics**: Precision, recall, and ROC-AUC score.

4. Hyperparameter Tuning:

Step-by-step process:

- **Stratified K-Fold Cross Validation**: Used to ensure that each fold has a similar class distribution, which is critical for imbalanced datasets.
- **Grid Search/Randomized Search**: Grid search is applied to find the best combination of hyperparameters. In case of large datasets, a randomized search may be preferred for efficiency.

5. Model Evaluation:

- **ROC Curve**: For imbalanced datasets, the **ROC curve** is used to evaluate the tradeoff between **True Positive Rate (TPR)** and **False Positive Rate (FPR)** across different thresholds.
- **Precision, Recall, F1-score**: Precision and recall are particularly important in fraud detection. A high recall ensures that more fraudulent transactions are caught, while high precision ensures that the flagged transactions are truly fraudulent.
- **F1-Score**: This metric is the harmonic mean of precision and recall, and provides a balanced measure.

6. Final Model Selection:

- Based on the evaluation metrics (especially **ROC-AUC**, **Precision**, and **Recall**), the final model is selected.
- **Cost-Benefit Analysis**: The **precision-recall trade-off** is crucial in fraud detection:
  - **High Precision** is important when it's necessary to avoid too many false positives (i.e., normal transactions incorrectly flagged as fraudulent).
  - **High Recall** is critical in ensuring that fraudulent transactions are caught, especially when high-value transactions are involved.

7. Business Insights:

- **For banks with smaller transaction values**, high precision is preferred because it minimizes unnecessary verification calls for non-fraudulent transactions.
- **For banks with larger transaction values**, a high recall is prioritized to prevent fraudulent high-value transactions, even at the cost of some false positives.
- **Model Choice**: The **Logistic Regression model** (after tuning) was selected due to its high **ROC-AUC score** and **recall**, while being interpretable for business stakeholders.