FINAL CAPSTONE PROJECT
BY
DEEPANSHU KUSHWAHA

Summary Report: Prediction of Credit Card Fraud

1.Problem Statement

Credit card fraud detection is critical to ensure that customers are not charged for fraudulent transactions. The dataset provided contains transactions made by credit cards in September 2013 by European cardholders. It includes 284,807 transactions over two days, with 492 transactions identified as fraudulent.

2.Dataset Overview

Total Transactions: 284,807

Fraudulent Transactions: 492

Non-Fraudulent Transactions: 284,315

Class Imbalance: The dataset is highly imbalanced with fraudulent transactions constituting only 0.172% of the total transactions.

3. Exploratory Data Analysis (EDA)

Key Findings:

Distribution of Features: The dataset has been pre-processed with PCA, so features are named V1, V2, ..., V28.

Class Imbalance: Visualizations confirmed the severe class imbalance.

Amount Distribution: The transaction amounts vary significantly, with frauds often involving smaller amounts.

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

4. Data Cleaning and Preprocessing

Standardization: The Amount feature was standardized.

Missing Values: No missing values were found in the dataset.

Time was 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 techniques like SMOTE (Synthetic Minority Over-sampling Technique) imblearn.over_sampling
ADASYN

6. Feature Engineering

Interaction Terms: Created interaction terms to capture relationships between features.

Feature Scaling: Applied Min-Max scaling to ensure all features are on the same scale

7.Model Building and Evaluation

Models Used:

We make the four Model

Logistic Regression (LR)

K-Nearest Neighbors (KNN)

Decision Tree (DT)

XGBoost (XGB)

Model Performance Metrics:

Accuracy: The ratio of correctly predicted transactions to the total transactions. Precision (Positive Predictive Value): The ratio of correctly predicted fraudulent transactions to all predicted fraudulent transactions.

Recall (Sensitivity): The ratio of correctly predicted fraudulent transactions to all actual fraudulent transactions.

Sensitivity: The ratio of correctly predicted non-fraudulent transactions to all actual nonfraudulent transactions

Specificity: The ratio of correctly predicted non-fraudulent transactions to all actual nonfraudulent transactions

F1 Score: The harmonic mean of precision and recall.

ROC-AUC: The area under the receiver operating characteristic curve, which plots TPR vs. FPR.

Model Results:

The performance of each model was evaluated using a confusion matrix and classification report, with key metrics summarized below:

Name	**Accuracy**	**Precision**	**Recall**	**F1scorer**	ROCAUC	Sensitivity	Specificity	FPR	PPV	NPV
Logistic Regression	0.906195	0.017182	0.952381	0.033755	0.982625	0.9524	0.9061	0.0939	0.0172	0.9999
K-Nearest Neighbors	0.998127	0.475472	0.857143	0.611650	0.928116	0.8571	0.9984	0.0016	0.4755	0.9998
Decision Tree	0.997519	0.386760	0.755102	0.511521	0.876519	0.7551	0.9979	0.0021	0.3868	0.9996
XGBoost	0.999122	0.714286	0.816327	0.761905	0.958659	0.8163	0.9994	0.0006	0.7143	0.9997

8.Hyperparameter Tuning:

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

Hyperparameter lr model,dt model,xgboost model

For each model, hyperparameter tuning was performed to identify the best set of parameters that maximize the model's performance. Below are the details for each model:

Logistic Regression:

. **Solver:** The algorithm to use in the optimization problem. Best found: lib linear

. **C:** Inverse of regularization strength. Best found: 0.1

Decision Tree:

- **criterion**: The function to measure the quality of a split. Best found: gini
- **Max Depth**: The maximum depth of the tree. Best found: 10
- **Min Samples Split**: The minimum number of samples required to split an internal node. Best found: 2
- **Min Samples Leaf**: The minimum number of samples required to be at a leaf node. Best found: 1

Fitting 5 folds for each of 72 candidates, totalling 360 fits

XGBoost:

Fitting 5 folds for each of 162 candidates, totalling 810 fits

- Learning Rate (0.2)

- **Max Depth**: Maximum depth of a tree. Best found: 5
- **Subsample**: The fraction of samples to be used for fitting the individual base learners. Best found: 0.7
- **Colsample_bytree**: The fraction of features to be used for fitting the individual base learners. Best found: 1.0
- **Number of Estimators**: The number of gradient boosted trees. Best found: 300

Hyperparameter tuning was performed using grid search and cross-validation to ensure robust selection of parameters.

9. 8. Model Deployment:

The best-performing model is serialize using the pickle module for deployment

We make the model deploy use to pickle file best_xgb model

First to Save the model to a file

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

pickle.dump(model, file)

#and later on, load the model from the file with

open('best_xgb_model.pkl', 'rb') as file:

best_xgb = pickle.load(file)

# Use the loaded model to make prediction

preds = best_xgb.predict(X_test) and now

print the model

Conclusion:

The project successfully demonstrated the application of various machine learning techniques to detect fraudulent transactions in a highly imbalanced dataset. By employing strategies like SMOTE to address class imbalance and using a combination of different classifiers, we were able to achieve high performance metrics, particularly in precision and recall, which are crucial in fraud detection scenarios.

The high ROC-AUC score indicates that our model is effective at distinguishing between fraudulent and non-fraudulent transactions. However, there is still room for improvement, especially in increasing the recall rate to minimize false negatives, which are critical in fraud detection.