CREDIT CARD FRAUD DETECTION

Name:	**SHALVIKA SRIVASTAV**
Registration No./Roll No.:	21247
Institute/University Name:	IISER Bhopal
Program/Stream:	DSE
Problem Release date:	August 17,2023
Date of Submission:	November 19,2023

1 Introduction

Credit card fraud prediction is the process of using machine learning algorithms and data analysis techniques to identify and anticipate fraudulent transactions made using credit or debit cards.These machine learning models can then be used in real- time to assess incoming transactions and flag those that are likely to be fraudulent,enabling businesses to take immediate action to prevent potential fraud.

The training data set has 57,116 rows and 30 columns, including time and amount.The testing data set has 14,280 rows and 30 columns,including time and amount.Further,we have been provided with the class labels for the training data set.

The training data set has no null values but the testing data set has four missing values which has been imputed with the mean in this project.

There are 28 transformed features and 2 non-transformed features time and amount which have been scaled using Standard Scalar and Robust Scalar in the project.The data set has two classes, non-fraudulent represented as 0 and fraudulent represented as 1.The training data has 56974 non-fraudulent transactions (99.25 percent) and 142 fraudulent transactions (0.25 percent) and hence, is highly unbalanced , refer Figure 1. , so we have used SMOTE along with some models to balance the data set.

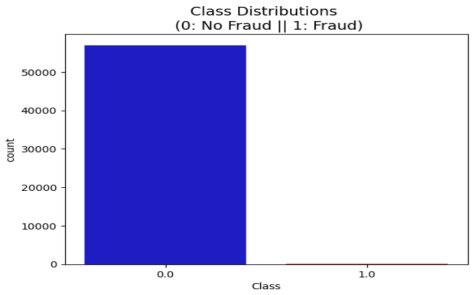

Figure 1: CLASS DISTRIBUTION OF THE DATA SET

2 Methods

A correlation matrix was plotted to understand the relationships between features in the dataset.It helps to determine which variables tend to move together (positively correlated) and which move in opposite directions (negatively correlated).It also helps in data preprocessing.

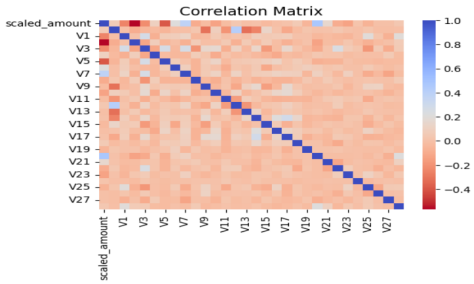

Figure 2: CORRELATION MATRIX

Time and Amount columns of training and testing datasets were scaled using Robust scalar method.Various Machine Learning Models were used like:

- Gaussian Naive Bayes
- K- Nearest Neighbour
- Decision Tree
- Logistic Regression
- Support Vector Classifier
- Random Forest

Since, the dataset was highly imbalanced, SMOTE was used with the models.Results improved for Gaussian Naive Bayes and KNN but there was very insignificant difference for decision tree and Random forest models. SMOTE generates synthetic samples of minority class to balance class distribution in imbalanced datasets.

Hyperparameter tuning was implemented through grid search cv and randomized search cv for finding optimal settings for model parameters to improve model performance.

GitHub Link for the Project

3 Experimental Setup

The libraries used in this project are numpy ,pandas,matplotlib,imblearn and scikit -learn.

The models required proper hyperparameter tuning which was done through Grid Search and Randomized Search methods.

- For Gaussian Naive Bayes, the best parameters found were priors :[0.5,0.5] (Prior probabilities of the classes.) and var smoothing : 1.0 (Portion of the largest variance of all features that is added to variances for calculation stability.)
- For Gaussian Naive Bayes with SMOTE, the best parameters found were priors :[0.5,0.5] and var smoothing : 0.599484250318941
- For K- Nearest Neighbour, the best parameters found were n neighbours:3 (Number of neighbors to use), metric: euclidean (Metric to use for distance computation.)
- For K- Nearest Neighbour with SMOTE, the best parameters found were n neighbours:3 , metric: manhattan.
- For Logistic Regression, the best parameters found were 'C': 0.1, 'penalty': 'l2'
- For Support Vector Classifier, the best parameters found were 'C': 0.7, 'kernel': 'linear'.

- For Decision tree, the best parameters found were min samples split:10 (The minimum number of samples required to split an internal node) , min samples leaf: 4 (The minimum number of samples required to be at a leaf node. A split point at any depth will only be considered if it leaves at least min samples leaf training samples in each of the left and right branches.) , max depth: 20 (The maximum depth of the tree.)
- For Random Forest, the best parameters found were n estimators : 100 (The number of trees in the forest.), min samples split:2 , min samples leaf: 1 , max depth : 20

The metrics which have been used for evaluating the results of the model are macro averaged precision , recall and F score.Macro averaged evaluation scores computes the average for each class and then takes the mean, treating all classes equally. It provides a balanced assessment of a model's performance across all classes.

- Precision = TP/(TP+FP)
- Recall = TP/(TP+FN)
- F Score = (2 X Precision X Recall)/(Precision + Recall)

where TP is true positive, the number of actual fraud transactions detected as fraud , FP is false positive, the number of actual non- fraud transactions detected as fraud, FN is false negative,the number of actual fraud transactions detected as non-fraud and TN is true negative,the number of actual non-fraud transactions detected as fraud.

4 Results and Discussion

The results obtained for various models implemented during the project is summarized in Table 1 1.

The results are obtained using classification report function from scikit learn library. The table summarizes the macro averaged precision,recall and F score obtained by applying the above mentioned hyperparameters after running the models on the data sets.

- Gaussian Naive Bayes reports macro averaged scores of Precision of 0.71 means that when the model flags a transaction as potentially fraudulent, it is correct about 71 percent of the time. Recall of 0.85 indicates that the model is successful in capturing 85 percent of the actual fraudulent transactions. F1-score of 0.77 provides a balanced measure of the model's overall performance
- Gaussian Naive Bayes with SMOTE reports a macro averaged precision score of 0.83,recall of 0.81 and F score of 0.82.These scores have clearly improved on balancing the data set .
- KNN reports macro averaged scores with precision of 0.98 accurately identifies potential fraud, minimizing false positives. It also captures 94 percent of actual fraud cases, achieving a high F1-score of 0.96, indicating a robust balance between precision and recall.
- KNN with SMOTE achieves precision of 0.93 which reflects accurate identification of potential fraud, while perfect recall (1.00) ensures comprehensive detection of all actual fraud cases, resulting in a high F1-score of 0.96.
- The results for SVC and Logistic Regression are summarized similarly in the table.
- Decision tree achieves precision of 0.98,recall of 0.93 and F score of 0.95. It indicates that the model is 93 percent capable of detecting actual fraud cases.
- Random Forest achieves a perfect score of 1.00 for precision ,recall and F score. It indicates that the model identifies all actual fraud cases as fraud.

The confusion matrix is reported in Table2 2, which summarizes the actual and predicted number of transactions belonging to each class by different models.

Table 1: Performance Of Different Classifiers Using All Features

Classifier	Macro avg Precision	Macro avg Recall	Macro avg F-measure
Gaussian Naive Bayes	0.71	0.85	0.77
Gaussian Naive Bayes with SMOTE	0.83	0.81	0.82
K-Nearest Neighbor	0.98	0.94	0.96
K-Nearest Neighbor with SMOTE	0.93	1.00	0.96
Decision Tree	0.98	0.93	0.95
Logistic Regression	0.97	0.96	0.96
Support Vector Classifier	0.95	0.93	0.94
Random Forest	1.00	1.00	1.00

Table 2: Confusion Matrices of Different Classifiers

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56840	134
Non-Fraud	41	101

Gaussian Naive Bayes

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56931	43
Non-Fraud	55	87

Gaussian Naive Bayes with SMOTE

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56969	5
Non-Fraud	21	121

Decision Tree

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56968	6
Non-Fraud	18	124

K-Nearest Neighbour

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56950	24
Non-Fraud	0	142

K-Nearest Neighbour with SMOTE

Actual Class	Predicted Class	
	Fraud	Non-Fraud
Fraud	56974	0
Non-Fraud	0	142

Random Forest

5 Conclusion

The main objective of this project was to find the most suited model in credit card fraud detection in terms of the machine learning techniques chosen for the project.After running all the models,we conclude that the best model for detecting frauds in credit card is found to be Random Forest which correctly identifies all the actual fraudulent transactions and achieves perfect scores of precision, recall and F measure considering the hyperparameters are same as mentioned.

6 References

- Credit Card Fraud Detection using ML by Meera AIEmad
- Credit Card Fraud Detection with Python and ML
- Credit Card Fraud Detection using Machine Learning by Mr. Thirunavukkarasu.M