Protecting Credit Card Transactions from Fraud with Machine Learning

Shreejan Kumar, B21EE088
Department of Electrical Engineering
Indian Institute of Technology Jodhpur
Rajasthan, India
kumar.301@iitj.ac.in

Vasubhya Diwan, B21AI044
Department of Artificial Intelligence
Indian Institute of Technology Jodhpur
Rajasthan, India
diwan.1@iitj.ac.in

Abstract

In this project, we present a study on credit card fraud detection using machine learning techniques. The dataset used contains credit card transactions made by European cardholders in September 2013, with a highly unbalanced distribution of only 0.172% of transactions being classified as fraudulent. To address this issue, we utilized various techniques such as undersampling, oversampling, and SMOTE to balance the dataset and improve model performance. We tested several machine learning algorithms, including KNN, LightGBM, Adaboost, Weighted SVM, and a Voting classifier, and compared their performance. The performance of the models was evaluated using metrics such as accuracy, precision, recall, and F1 score. The results showed that the combination of KNN, LightGBM, and Weighted SVM with oversampling techniques achieved the highest overall performance. This indicates that a combination of different techniques can be beneficial in improving the accuracy of credit card fraud detection models. We concluded that the best-performing model was a combination of all the successful models. Our results demonstrate the effectiveness of machine learning in credit card fraud detection, highlighting the importance of developing accurate models to prevent financial losses.

I. INTRODUCTION

Credit card fraud is a significant problem in the financial industry, leading to considerable financial losses for financial institutions and customers alike. To prevent such fraud, machine learning techniques have been widely utilized in recent years. This project presents a study on credit card fraud detection using machine learning algorithms. The aim of this project is to develop an accurate machine learning model that can predict whether a given credit card transaction is fraudulent or not.

The highly unbalanced nature of the dataset, as shown in Fig. 2, makes it a challenging task to develop an accurate fraud detection model. Therefore, the project employs various techniques such as undersampling, oversampling, and SMOTE from scratch to balance the dataset and improve model performance. The study evaluates several machine learning algorithms, including KNN, LightGBM, Adaboost, Weighted SVM, and a Voting classifier, comparing their performance in detecting fraudulent transactions. The performance of the models was evaluated using metrics such as accuracy, precision, recall, and F1 score, and the results showed that a combination of different techniques can be beneficial in improving the accuracy of credit card fraud detection models.

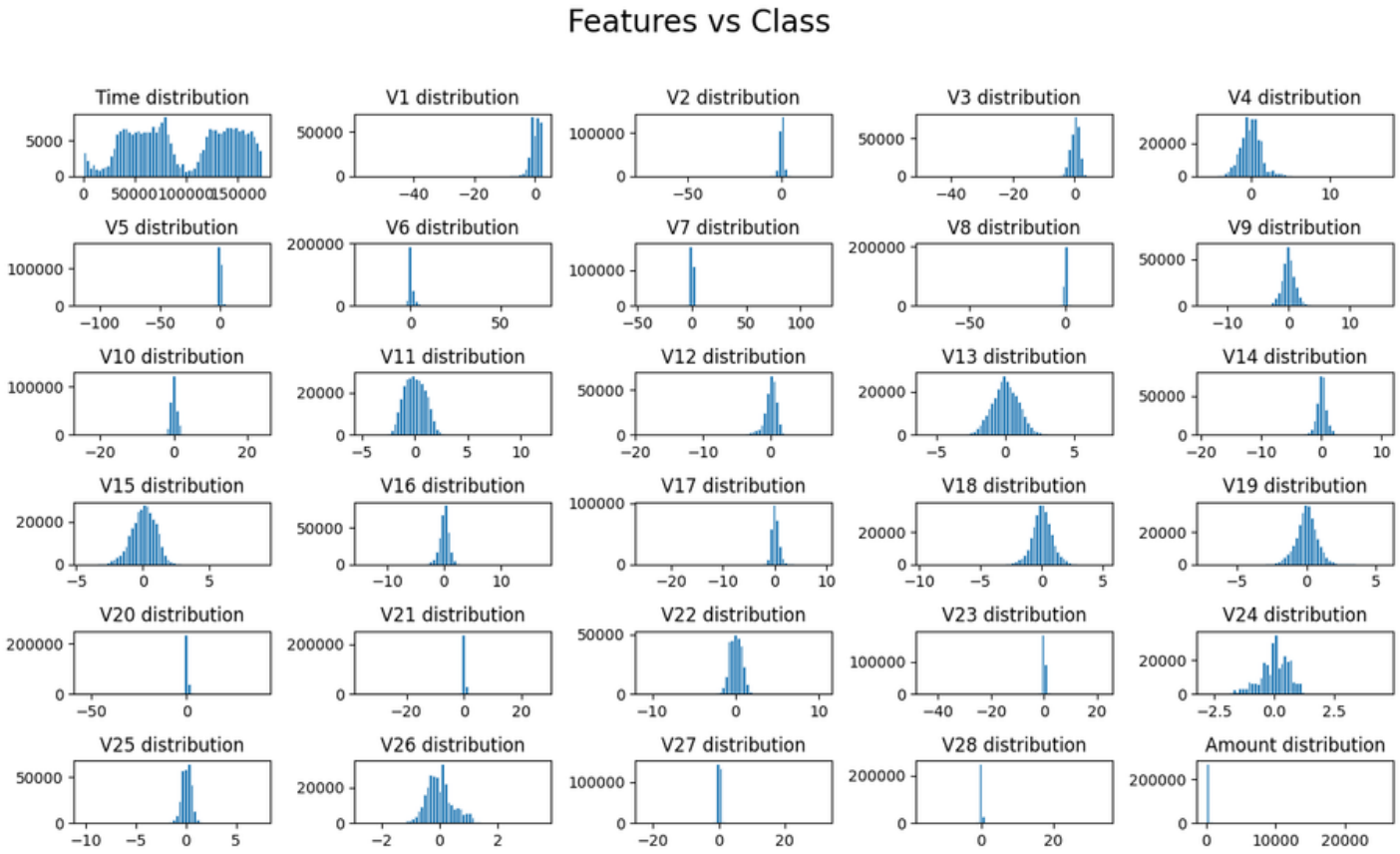


Fig. 1: Input Feature Distribution

By seeing the first graph for time in Fig. 1, we can see there are two peaks in the graph and even there are some local peaks. We can think of these as the time of the day like the peak is the day time when most of the genuine transactions happen and the dip is the night time when most people sleep and most of the fradulent transactions happen. We already know that data contains a credit card transaction for only two days, so there are two peaks for day time and one depth for one night time.

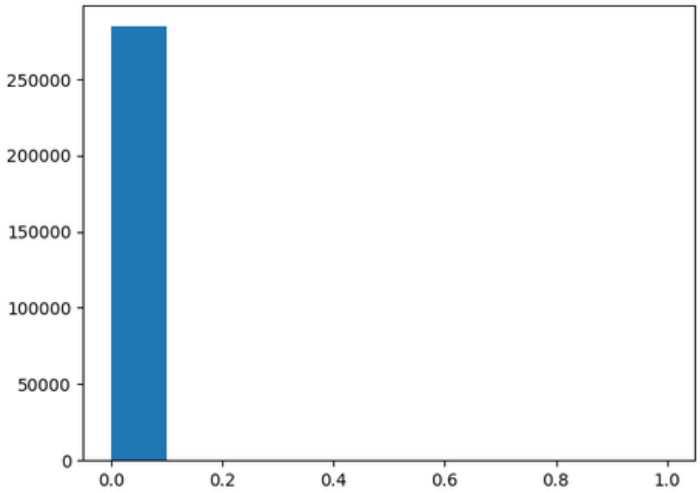


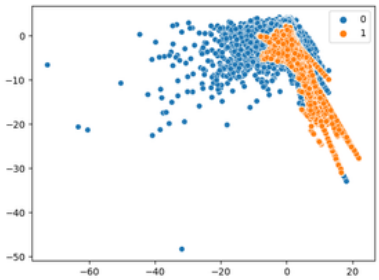
Fig. 2: Distribution of both Classes

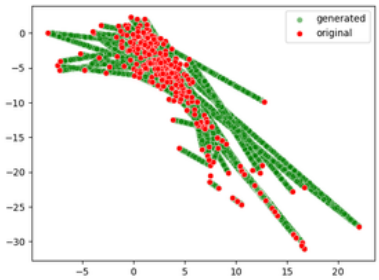
II. OUR APPROACH

Our approach was mainly based on strategies to tackle the highly imbalanced dataset we were given. We had two ways to move forward with this, One was to either make changes to the dataset through techniques like Undersampling, Oversampling, and Dimensionality reduction techniques like PCA, and the other method was to make changes in the classifiers so that they could account for the unbalanced data through techniques like Weighted data, Cost-sensitive learning, and Ensemble learning. We decided to use the LightGBM classifier to test the various methods and combinations due to its efficiency and accuracy in predicting unbalanced data. It uses a special histogram-based algorithm for binning continuous features, which is more efficient than the traditional sorting approach and finding the optimal split point. This algorithm can speed up the training process and reduce memory usage, which is especially important for large datasets. It is also known for its high accuracy in predicting the target variable. It achieves this by building a large number of decision trees sequentially and using boosting to combine the outputs of the individual trees. The metric that we used for evaluating our classifiers was F1 score instead of accuracy because since the data was highly imbalanced, we would always get a very good accuracy but the recall would be always low which is a very dangerous situation becase we cannot predict the fradulent cases through this approach. F1 score is the harmonic mean of precision and recall, so we used that to evaluate our classifiers to get a good balance between accuracy and recall.

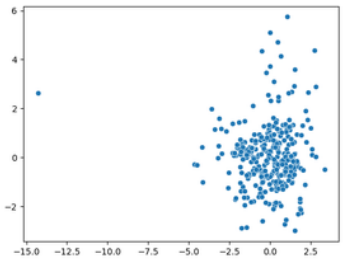
Data Manipulation

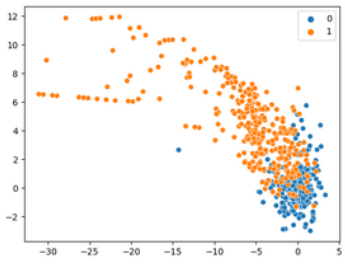
Oversampling - It is a technique for addressing class imbalance in a dataset by generating synthetic data for the minority class, in order to provide the classifier with sufficient information about the minority class to enable accurate predictions. The resulting synthetic data can then be used to balance the class distribution and improve the performance of the classifier on the minority class. We made our synthetic data by adding data points in between two closest pre-existing data points.





Undersampling - It is a technique used to address class imbalance by reducing the number of instances of the majority class. We applied the NearMiss algorithm (specifically the third variation) from scratch to undersample the majority class samples closest to the minority class samples.





Classifier Modification

Adaboost - Adaboost classifier performed well on the datset. It is able to identify and assign higher weights to the most important features, allowing it to improve its performance. Additionally, it is able to handle the imbalance in the dataset by assigning higher weights to the minority class and focusing more on correctly classifying fraudulent transactions. Furthermore, it is able to reduce the risk of overfitting by using weak learners that are simple and have high bias.

K-NN - We applied weighted K-NN with weights as distance but it did not perform well on the overall datset. The reason for this is that the credit card dataset has a large number of features, which can result in a high-dimensional space. In such a space, the distance between points can become less meaningful, and points that are close to each other in the high-dimensional space may not be close in the lower-dimensional space that represents the true underlying structure of the data.

Weighted SVM - This algorithm works by finding the hyperplane that best separates the classes while maximizing the margin between them. When dealing with imbalanced datasets, SVMs can be weighted to give more importance to the minority class, which is typically the class of interest but still it did not perform particularly well on the dataset. This may be because the dataset may have non-linear decision boundaries, which may not be captured well by linear SVMs, even when using non-linear kernels. Secondly, SVM's aims to maximize the margin between the classes, which may lead to misclassification of the minority class. Finally, SVMs are sensitive to outliers and may not perform well when the dataset has significant noise.

Tuned LGBM - We used the inbuilt Random search function to tune the previously used LGBM classifier where the parameters which were tweeked were:

- max_depth
- learning_rate
- num_leaves
- min_child_samples

And f1 score was the target metric which was maximized. Predictably we had a good performing model with much better f1 score as compared to the default model.

III. Results

Voting Classifier - Based on our analysis, we trained a voting classifier using the Lgbm classifier, Oversampled Lgbm model, and the Adaboost classifier, which were previously identified as the best classifiers. However, we observed only a minor improvement in the performance of the voting classifier compared to the individual classifiers. We hypothesize that the limited deviation in performance can be attributed to high correlation among the base classifiers, resulting in similar errors. Therefore, the voting classifier did not provide a significant improvement over the individual classifiers.

Combination Classifier - After analyzing individual performances, we decided to combine the 3 most successful methods in :

- Oversampling
- ADABoost
- Hyperparameter Tuning

This was achieved by first tuning the LGBM classifier trained on the oversampled dataset with f1 score as our target and then passing it as the base estimator for the ADABoost which will give more weight to the previously incorrectly classified points which mainly consisted of minority class samples. Thus after applying boosting on tuned LGBM classifier which was trained on the oversampled dataset, we get the best f1 score so far.

Classifier	Accuracy	F1 Score
LightGBM	100%	63%
LightGBM tuned	100%	85%
DTC with PCA	100%	56%
SMOTE with PCA	82%	82%
Near Miss with PCA	6%	6%
Undersampling	97%	53%
Oversampling	100%	91%
AdaBoost	100%	89%
KNN	100%	60%
SVM	69%	41%

Classifier	Accuracy	F1 Score
Voting Classifier	100%	82%
Combination	100%	96%