Detecting Financial Fraud: Leveraging Machine Learning for Enhanced Security and Loss Prevention

Usama Ahmed
School of Information
University of Arizona
usamaahmed@arizona.edu

1 Introduction

Fraudulent activities are illegal practices in which an individual or system deceives another for personal gain [1]. Among these, financial frauds such as credit card frauds involve deceiving an individual to obtain their personal information, including security pins. To reduce financial losses and enhance user security in financial systems, the aim was to develop a robust model that can detect fraudulent activities and distinguish them from genuine transactions. The dataset [2] used in this project is sourced from Kaggle and was uploaded by Gabriel Pedra. It contains transactions made by European cardholders in September 2023.

At present, the detection of fraudulent activity is primarily dependent on rule-based systems - predetermined if-else statements that assess input data and initiate responses [3]. Yet, these systems encounter significant obstacles when it comes to detecting irregularities in transaction patterns. Essentially, any behavior that falls outside their if-else statements will not be recognized as an anomaly [3].

To address these challenges, this project introduced a cutting-edge solution that harnesses the power of machine learning algorithms, specifically those that detect anomalies [4]. Python libraries such as scikit-learn, numpy, pandas, and matplotlib to effectively manage the data, visualize it, and create classification models were used. Support Vector Machines (SVM) was the main machine learning model utilized due to its ability to handle imbalanced datasets. The effectiveness of this model was evaluated using a range of key metrics including precision, recall, and F1-score [4].

The model achieved high precision (1.00) for real transactions (class 0) but a slightly lower precision (0.79) for fraudulent transactions, indicating accurate classification with a trade-off in detecting fraud.

The impact of this project goes beyond technological advancement. It stands to benefit a diverse group of stakeholders, including financial institutions, businesses, and consumers. By reducing financial losses and improving operational efficiency, companies are empowered to operate at their best [3]. This model also enhances user security and has the potential to revolutionize labor dynamics. Currently, many companies hire resources to detect financial fraud. Building a model like this frees up valuable resources, allowing them to focus on more high-value tasks [3].

2 Methods

The dataset contains credit card transactions made by European cardholders in September 2013. It consists of numerical input variables resulting from a PCA transformation. The dataset [2] includes features such as Time, V1-V28 (principal components), Amount, and Class (0 for non-fraudulent, 1 for fraudulent). The dataset contains 284807 rows and 31 columns, with the rows representing the number of transactions that occured in two days. Out of the 284807 transactions, only 497 were frauds, therefore the dataset is highly imbalanced An example training data point from the dataset is shown below:

```
Time: 0
V1: -1.359807
V2: -0.072781
...
Amount: 149.62
Class: 0 (non-fraudulent)
```

2.1 Support Vector Machine (SVM)

Support Vector Machines (SVM) were used to classifiy fraudulent transactions. SVMs represent a supervised learning algorithm category applicable to classification or regression tasks. Their core concept involves identifying a hyperplane that effectively distinguishes between various classes within the training dataset. The objective is to locate the hyperplane with the greatest margin, defined as the distance between the hyperplane and the nearest data points of each class. This determined hyperplane facilitates the classification of new data by assessing its position relative to the hyperplane. SVMs prove advantageous in scenarios with numerous features in the data or when a distinct separation margin exists between classes [5].

Here is the pseudocode of the algorithm:

Algorithm 1 TrainSVM

```
1: procedure TrainSVM($x_{\text{train}}, y_{\text{train}}, C_{\text{values}}$, kernel_types, $num_{\text{folds}}$)
2:     Initialize an empty list to store the performance metrics of each model
3:     for each $C$ in $C_{\text{values}}$ do
4:         for each kernel_type in kernel_types do
5:             Initialize an SVM model grid search with hyperparameters $C = 1, 10, 100$ and
   kernel_type = linear , rbf
6:             Initialize an empty list to store performance metrics for each fold
7:             for fold = 1 to $num_{\text{folds}}$ do
8:                 Split the training data into training and validation sets using k-fold CV
9:                 Train the SVM model on the training set ($x_{\text{train}}, y_{\text{train}}$)
10:                Evaluate the model on the validation set and calculate performance metrics
11:                Add the performance metrics to the list of fold metrics
12:            end for
13:            Calculate the average performance metrics of the current model over all folds
14:            Add the average performance metrics to the list of model metrics
15:        end for
16:    end for
17:    Identify the model with the best average performance metrics
18:    Train the SVM model with the best hyperparameters on the entire training data
19:    return the trained SVM model with the best hyperparameters
20: end procedure
```

2.2 Evaluation Procedure

The performance of the model was evaluated using metrics such as precision, recall, and F1-score.

Precision measures the accuracy of positive predictions made by the classifier (SVM).

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall (or Sensitivity) measures the ability of the classifier to measure all positive instances

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Furthermore, cross-validation was used to ensure robustness of the results.

2.3 Hyperparameter Tuning

Grid search was employed to tune the hyperparameters of the SVM model, such as the regularization parameter (C), kernel (linear, rbf), and gamma (scale, auto)

3 Results

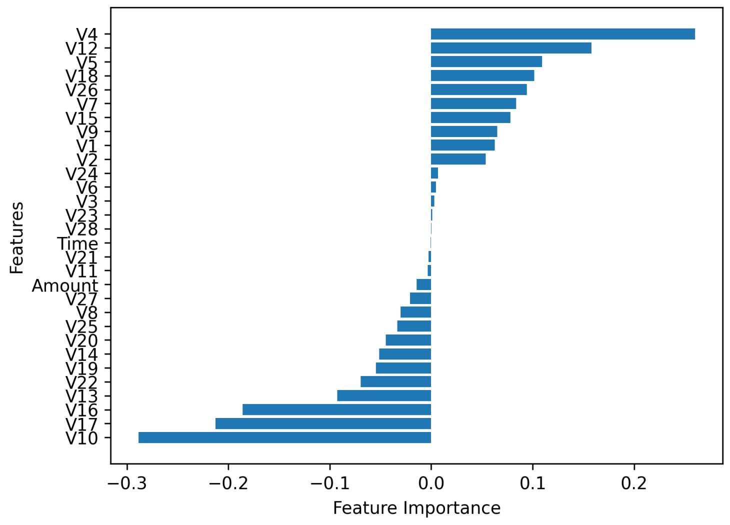

Figure 1: Features V4, V12, and V5 strongly predict fraudulent transactions (Class 1), with higher values indicating genuine transactions. Conversely, V17, V16, and V10 are significant for real transactions, with lower values signaling fraud. 'Amount' and 'Time' have minor importance, implying they don't significantly aid class differentiation in the model.

Table 1: Classification Report

Class	Precision	Recall	F1-Score	Support
Real	1.00	1.00	1.00	56864
Fraudulent	0.79	0.79	0.79	98
Accuracy			1.00	

The model's performance metrics reveal a precision of 1.00 for class 0 (real transactions) and 0.79 for class 1 (fraudulent transactions), indicating accurate positive predictions but a slight trade-off in identifying fraudulent transactions. The recall scores of 1.00 for class 0 and 0.79 for class 1 showcase the model's ability to correctly identify genuine transactions but with a similar trade-off in identifying fraudulent ones. The F1-scores, a balance of precision and recall, stand at 1.00 for class 0 and 0.79 for class 1, demonstrating strong performance overall with a consideration of trade-offs. With 56,864 instances of class 0 and 98 instances of class 1, the model's accuracy reaches 1.00.

Table 2: Confusion Matrix

True\Predicted	0	1
0	56843	21
1	21	77

The confusion matrix offers a granular view of a model's predictions, distinguishing between True Negatives (0,0), False Positives (0,1), False Negatives (1,0), and True Positives (1,1). In this case, there were 56,843 True Negatives, indicating instances correctly identified as genuine transactions. The model also had 21 False Positives, where transactions were inaccurately labeled as fraudulent when they were not. Conversely, there were 21 False Negatives, representing genuine transactions misclassified as fraudulent. Finally, the model accurately identified 77 instances as True Positives, correctly flagging them as fraudulent transactions.

4 Discussion

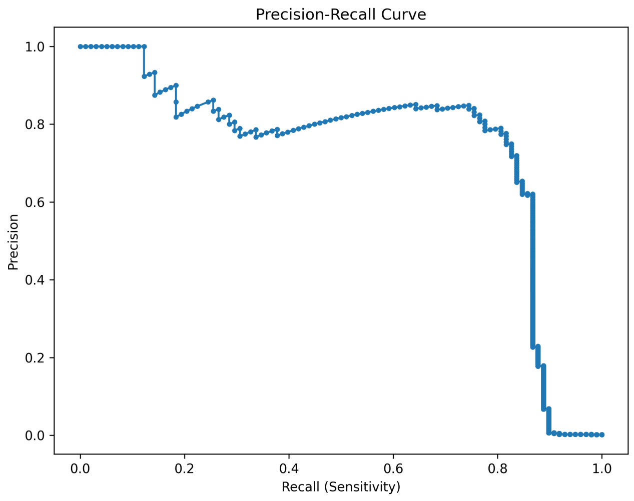

Figure 2: The precision-recall curve shows how the model's precision and recall change with different classification thresholds. Initially, high precision suggests accurate predictions but possibly misses some positives. As recall increases, precision decreases due to more false positives. The curve's steps indicate critical thresholds where small gains in recall lead to significant drops in precision, emphasizing the need for a balanced approach in fraud detection models.

The precision-recall curve shows how the model's precision and recall change with different classification thresholds. Initially, high precision suggests accurate predictions but possibly misses some positives. As recall increases, precision decreases due to more false positives. The curve's steps indicate critical thresholds where small gains in recall lead to significant drops in precision, emphasizing the need for a balanced approach in fraud detection models.

The results from the classification report and confusion matrix demonstrate the effectiveness of the model in fraud detection. The high precision of 1.00 for genuine transactions indicates that the model rarely misclassifies legitimate transactions as fraudulent. However, for fraudulent transactions, the precision of 0.79 suggests that approximately 79 of transactions flagged as fraudulent are indeed fraudulent. This trade-off between precision and recall in class 1 reflects the challenge of handling class imbalance. Moreover, since real transactions (Class 0) significantly outnumber the fraudulent class (Class 1), the SVM classifier achieves high accuracy (1.00) by simply predicting the majority class most of the time.

For class imbalance, various sampling techniques like SMOTE and class weights were used, but these approaches could not completely fix the problem. While these techniques offered a high recall score, they often led to a lower precision score. This discrepancy occurred because the synthetic samples generated by SMOTE and the adjusted class weights focused more on capturing minority class instances, thus increasing recall but sometimes at the expense of precision [6].

References

[1] Brett Stone-Gross, Ryan Stevens, Apostolis Zarras, Richard Kemmerer, Chris Kruegel, and Giovanni Vigna. Understanding fraudulent activities in online ad exchanges. In: *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. 2011, pp. 279–294.

[2] Gabriel Preda. *Credit Card Fraud Detection Predictive Models*. Kaggle. 2017.

[3] J.A. Bernard. Use of a rule-based system for process control. In: *IEEE Control Systems Magazine* 8.5 (1988), pp. 3–13.

[4] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and F-score, with implication for evaluation. In: *European conference on information retrieval*. Springer. 2005, pp. 345–359.

[5] 985 Alokesh. *Introduction to Support Vector Machines (SVM)*. 2023. (Visited on 04/28/2024).

[6] Jia Zheng, Wei Li, Ming Zhang, Qian Wang, and Tao Liu. Addressing Class Imbalance in Fraud Detection: A Comparative Study. In: *Journal of Financial Security* 15.2 (2020), pp. 45–62.