Logo

Description automatically generated

ALY 6040:

DATA MINING APPLICATIONS

Assignment 4:

Support Vector Machines (SVM) on

Online Fraud Detection Dataset

Submitted To:

Dr. Chinthaka Pathum Dinesh, PhD, Prof. Herath Gedara,

Faculty Lecturer

Submitted By:

[Abhilash Dikshit](mailto:dikshit.ab@northeastern.edu)

Academic Term: Spring 2023

Graduate Student at Northeastern University, Vancouver, BC, Canada

Master of Professional Studies in Analytics

May 11, 2023

1. **Abstract:**

This report explores the online payments fraud detection dataset obtained from [Kaggle](https://www.kaggle.com/code/haha9527/online-payments-fraud-detection), containing information related to online transactions, including details about the amount, source, and destination accounts, and whether the transaction was fraudulent. The aim of this study is to understand the characteristics of fraudulent transactions and identify patterns that can be used to prevent fraud in the future. The dataset contained over 6 million entries and required cleaning to handle missing data, duplication, and outliers. The results showed that fraudulent transactions represented a small percentage of the total, and that the amounts involved in these transactions were often much larger than in non-fraudulent transactions. The next steps would be to conduct further analysis to identify patterns and build predictive models to prevent future fraud.

1. **Introduction:**

The rise of e-commerce and online transactions has led to a significant increase in payment fraud. According to a report by Nilson, global payment card losses reached $27.85 billion in 2018, and it is predicted that the losses will continue to grow over time. Therefore, it is critical to develop effective fraud detection and prevention systems to minimize these losses.

1. **About Dataset**

The online payments fraud detection dataset obtained from Kaggle provides information related to online transactions, including details about the amount, source, and destination accounts, and whether the transaction was fraudulent. In this report, we will do the Code walk through, Interpretation and Recommendations for performing further analysis to identify patterns that can be used to prevent fraud in the future. Below are all the columns from the dataset:

|  |  |
| --- | --- |
| Step | Represents A Unit of Time Where 1 Step Equals 1 Hour |
| Type | Type Of Online Transaction |
| Amount | The Amount of The Transaction |
| Nameorig | Customer Starting the Transaction |
| Oldbalanceorg | Balance Before the Transaction |
| Newbalanceorig | Balance After the Transaction |
| Namedest | Recipient Of the Transaction |
| Oldbalancedest | Initial Balance of Recipient Before the Transaction |
| Newbalancedest | The New Balance of Recipient After The Transaction |
| Isfraud | Fraud Transaction |

1. **Code Walk Through:**

The data exploration, analysis and interpretation were performed using below libraries:

A screenshot of a computer program

Description automatically generated with medium confidence

*Fig 1. Libraries used.*

The first step was to load the dataset and examine the number of entries, variables, and data types. The dataset contained over 6 million entries and 11 variables.

Hence, we split the dataset into train and test data with 80:20 split and we will be using test dataset further analysis.

A screenshot of a computer program

Description automatically generated with medium confidence

*Fig 2. Test and Train Data Split (80:20)*

The next step was to examine the EDA on test data for null count and unique values.

A screenshot of a computer code

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated

*Fig 3. EDA on test dataset*

1. **Analysis:**

In order to verify the overall extent of fraudulent activity, we are currently extracting a subset of the dataset from the "isFraud" column, specifically isolating instances where the values are either 0 or 1.A picture containing text, font, screenshot

Description automatically generated

*Fig 4. Total fraud in the dataset*

Presently, we shall proceed with creating a visual representation pertaining to the distribution of transactions based on the "amount" and "step" columns.A screenshot of a computer

Description automatically generated with medium confidence

*Fig 5. Distribution of Transaction Amount and Transaction Step*

**Count plot to show Payment Type vs Count:**

|  |  |
| --- | --- |
| **A screenshot of a computer code  Description automatically generated with low confidence**  *Fig 6. Payment Type vs Count* | In this instance, we shall examine each payment category provided by the bank, along with their corresponding transaction volumes. This analysis will provide us with a comprehensive understanding of the frequency of payment channels utilized. |

**Count plot to show the Frequency of Transaction Types where Fraud happened:**

|  |  |
| --- | --- |
| **A screenshot of a computer code  Description automatically generated with medium confidence**  *Fig7. Frequency of Transaction Types for fraud* | In this instance, we shall examine the fraud happened in each payment category provided by the bank.  As depicted, the fraud happened in payment rails “cash\_out” and “transfer” and the frequencies has been shown in the bar plot. |

To enhance the visual representation, we categorized the columns into "numerical" and "categorical" types and generated a boxplot for each numerical variable to assess its skewness.

A screenshot of a computer screen

Description automatically generated with low confidence

*Fig 8. Boxplot for each numerical variable*

Correlation Matrix Plot on our dataset:

A close-up of a computer screen

Description automatically generated with low confidence

|  |  |
| --- | --- |
| A screenshot of a graph  Description automatically generated with low confidence  *Fig 9. Imbalanced Correlation Matrix* | As shown in the heatmap, we see there is an imbalance in our variables.  *Note: We are considering our test dataset as the number of rows is > 6 million.* |

**Interpretation and Recommendations:**

**Clustering Technique:**

We want to cluster based on the transaction amount, transaction type, and whether the transaction is fraudulent or not. We can drop the other columns from the dataset and proceed with clustering.

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated with medium confidence | Here, I added the mask parameter to hide the upper triangular part of the plot, set cbar=True to add a color bar, and set xticklabels and yticklabels to the column names of the correlation matrix for better readability. Finally, I also set linewidths=.5 to make the lines between the cells of the heatmap thinner. |
|  | This will perform clustering using KMeans with 2 clusters based on the transaction amount and type and visualize the clusters. |

*Fig 10: KMeans Clustering Technique*

**Random Forest:**

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with low confidence

*Fig 11: Random Forest*

Confusion Matrix:

A screenshot of a computer program

Description automatically generated with low confidence

*Fig 12: Confusion Matrix*

Random Forest with No under/Oversampling:

A screenshot of a computer program

Description automatically generated with low confidence

*Fig 13: Random Forest with No under/Oversampling*

**Support Vector Machines (SVM):**

1. We will select the columns of interest to create the new dataframe.

A screenshot of a computer

Description automatically generated with medium confidence

*Fig 12: New dataframe with selected columns*

1. Then we are going to preprocess the dataset by scaling the numeric features. We will use the StandardScaler() function from scikit-learn to scale the numeric variables

A picture containing text, font, screenshot

Description automatically generated

*Fig 13: Preprocess the dataset*

1. Let’s split the dataset into training and testing sets using the train\_test\_split() function from scikit-learn.

A picture containing text, font, screenshot, line

Description automatically generated

*Fig 14: Train and Test dataset*

1. Train the SVM model using the SVC () class from scikit-learn.

A screenshot of a computer program

Description automatically generated with low confidence

*Fig 15: SVM Model*

1. Evaluate the performance of the model using appropriate evaluation metrics, such as accuracy, precision, recall, and F1 score. For example, we will use the classification\_report() function from scikit-learn to calculate these metrics:

A screenshot of a computer

Description automatically generated with medium confidence

*Fig 16: Model Performance*

1. **Results:**

We conducted a fraud detection analysis using a support vector machine (SVM) on a dataset of financial transactions. The SVM was trained on a subset of the data and tested on a separate validation set.

Overall, the SVM model achieved a high accuracy of 1.00 on the validation set, indicating that it was able to correctly predict the class labels for all instances in the test set. However, when we examined the performance of the model on the positive class (fraudulent transactions), we found that its recall was relatively low at 0.32, indicating that the model was missing a significant number of fraudulent transactions. The precision for the positive class was high at 0.99, indicating that the model had a low rate of false positives.

We also examined the features that were most important in the SVM model. We found that the transaction amount, old balance of the origin account, and new balance of the origin account were the most important features for predicting fraud.

In summary, our analysis using SVM suggests that it is possible to achieve a high level of accuracy in predicting fraudulent financial transactions. However, further research is needed to address the relatively low recall rate observed in this analysis. Future work could focus on improving the performance of the model on the positive class by addressing the imbalanced nature of the dataset and experimenting with different classification algorithms. Additionally, our analysis highlights the importance of the transaction amount and origin account balances in detecting fraudulent transactions, which may have implications for developing more effective fraud detection systems in the future.

1. **Conclusion: (Based on SVM Model Performance)**

Based on the classification report output, the SVM model has an overall accuracy of 1.00 (or 100%) on the test set. This means that the model correctly predicted the class labels for all instances in the test set.

However, when we look at the precision, recall, and F1-score values for the positive class (fraudulent transactions, label "1"), we see that the precision is 0.99, recall is 0.32, and F1-score is 0.48. This indicates that the model is able to correctly identify most of the fraudulent transactions (high precision) but is missing a significant number of them (low recall). In other words, the model has a high rate of false negatives.

The low recall score may be due to the imbalanced nature of the dataset, where the number of non-fraudulent transactions far exceeds the number of fraudulent transactions. This can lead to a biased model that performs well on the majority class (non-fraudulent transactions) but poorly on the minority class (fraudulent transactions).

In conclusion, while the overall accuracy of the model is high, its performance on the positive class is not optimal due to the low recall score. Therefore, further analysis and experimentation may be needed to improve the model's ability to identify fraudulent transactions.

1. **References:**

Kaggle. (n.d.). Online Payments Fraud Detection. Retrieved from <https://www.kaggle.com/datasets/rupakroy/online-payments-fraud-detection-dataset>

Nilson Report. (2019). Card fraud losses reach $27.85 billion. Retrieved from <https://nilsonreport.com/publication_chart_and_graphs_archive.php>