Botnet Detection in Network Traffic Data

Name:	**Porus Vaid**
Registration No./Roll No.:	21208
Institute/University Name:	IISER Bhopal
Program/Stream:	DSE
Problem Release date:	August 17, 2023
Date of Submission:	November 17, 2023

1 Introduction

1.1 Botnets

Botnets, consisting of remotely controlled compromised computers, are used for malicious activities such as sending spam and launching DDoS attacks, some of the most severe cybersecurity threats. These networks often exploit devices with vulnerabilities like outdated software and open ports.

Detecting botnets is essential for protecting corporate reputation, customer trust, and service availability. The focus is usually on identifying the command and control infrastructure of the botnet, though individual bot detection remains challenging.

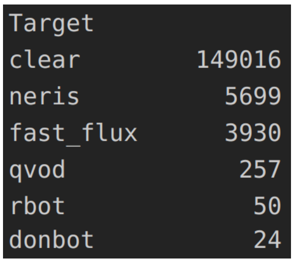
1.2 Problem Statement

The challenge addressed in this project is the identification of such botnets using network traffic data. This involves developing supervised machine learning algorithms to classify and predict different types of botnets, a crucial step for enhancing cybersecurity measures.

2 Methodology

2.1 Dataset Description

The given data are in the CSV file format (train_data.csv, test_data.csv, train_data_label.csv),
Training Data - Training data has 158976 rows & 84 columns. The data types of the 84 columns are as follows: [dtypes: bool(4), float64(61), int64(17), object(2)]. There are unique values as well as null values in certain columns.
Testing Data - Testing data has 17664 rows & 84 columns the column types are the same as training data, There are unique values as well as null values in certain columns.



```
Target
clear        149016
neris          5699
fast_flux      3930
qvod            257
rbot             50
donbot           24
```

Figure 1: Target Labels

2.2 Data Preprocessing

Cleaning:

1. There were a lot of features that had missing values. But all those features that had missing values were of numeric data type (either int or float). Data Imputation was done on those features. **SimpleImputer** was used, the strategy as **median** because if the mean were used, the model would become sensitive to outliers.
2. Added the Column Name as "Target" to the Target Labels from the train_data_label.csv

Encoding:

1. Binary encoder was used for the Dtype: Object. Binary encoding is a combination of Hash encoding and one-hot encoding. The categorical feature is first converted into numerical using an ordinal encoder in this encoding scheme. Then, the numbers are transformed in the binary number. After that, the binary value is split into different columns. Binary Encoder was selected over other encoders as it keeps the number of features restricted.
2. For the Labels - Label Encoding was used. It was favoured apart from any other encoding techniques primarily to reduce the sparseness in the confusion matrix.

Standardization:

1. I used **StandardScaler**, scaling all features to a similar range, typically with a mean of 0 and a standard deviation of 1 - mainly for the models sensitive to the data scale (KNN and SVM). Also, to increase the overall performance of the model.

Exploratory Data Analysis:

1. PCA (Principal Component Analysis) and t-SNE (t-Distributed Stochastic Neighbor Embedding) were employed to visualize and derive meaningful insights from the data.
2. The insights from both PCA and t-SNE suggest that for this dataset, a combination of linear and non-linear features may be necessary for effective botnet detection.

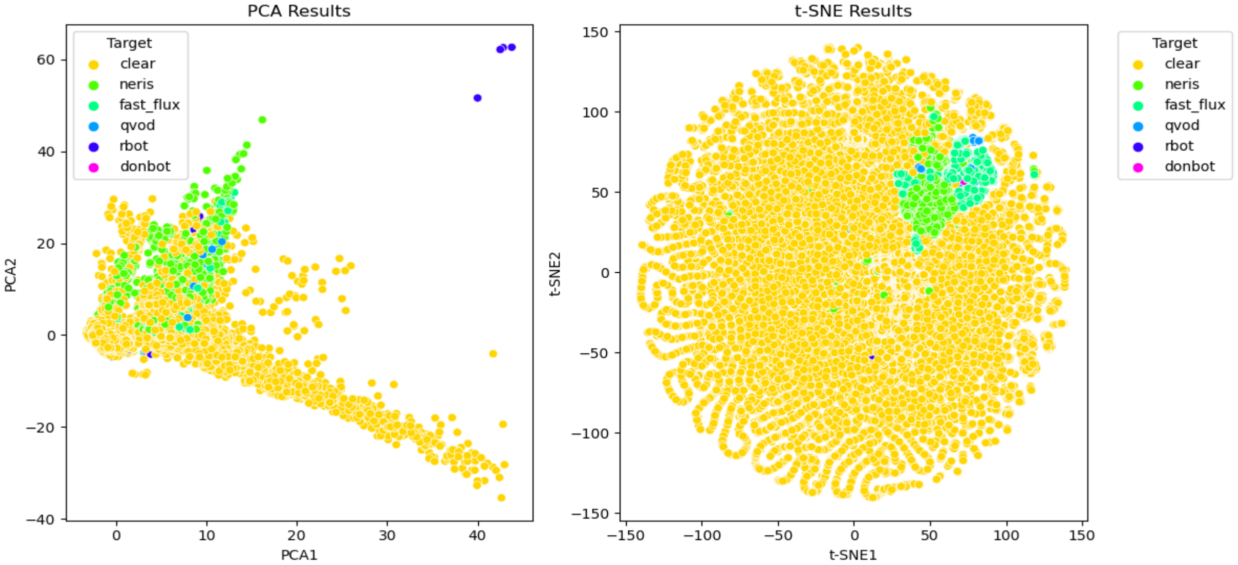
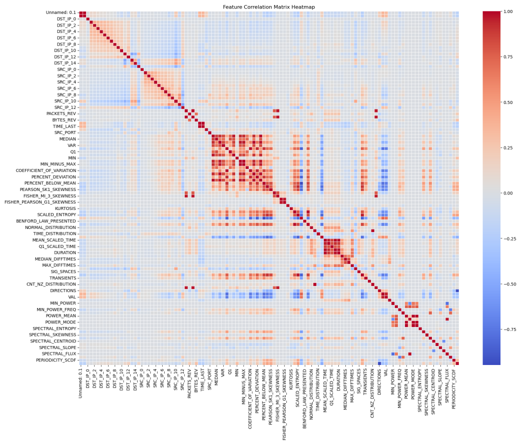


Figure 2: PCA and t-SNE Analysis

2.3 Feature Selection



Multiple Feature Selection techniques were used, but a point to be noted here is, that the final model was evaluated on all features as well as selected features, and there were no major changes in the evaluation metrics of them. The techniques used were:

1. The Correlation Matrix and its visualization were done using heatmaps. The features that had values greater than 0.3 were selected, as after that, the values decreased drastically.

2. Recursive Feature Elimination (RFE) - This method iteratively fits the model and removes the least important feature at each step, as determined by the classifier, until the specified number of features is reached.

RandomForest Classifier is used and top 10 features were selected.

3 Algorithm and Experimental Results

3.1 Handling Data Imbalance:

There was a high data imbalance in the given dataset. To handle that, an innovative approach was implemented:
Synthetic Minority Over-sampling Technique (SMOTE): This technique generates synthetic examples of the minority class, enhancing its representation in the training set and mitigating the model's bias towards the majority class.
Random Undersampling: Concurrently, the majority class is undersampled—randomly reduced to half its original size for each model in the ensemble. This further balances the class distribution, ensuring no single class dominates the learning process.

3.2 Ensemble Technique:

Ensemble Technique was used to get the most out of the data and get accurate predictions:
Multiple Models: An ensemble of five Random Forest models is trained, each on a balanced subset of the training data obtained through SMOTE and random undersampling. This introduces variety, with each model potentially capturing different aspects of the data.
Majority Voting: Predictions from all models are combined using a majority voting scheme, where the final prediction for each instance is the one that most models agree on. This approach reduced the variance of predictions, resulting in a more accurate and stable model than any single classifier in the ensemble.

3.2.1 Majority Voting

Aggregation Method: The code uses a statistical mode to aggregate predictions across the ensemble, selecting the most common prediction (the mode) for each data point. This technique is akin to a voting system where the most frequent prediction is considered the final output.

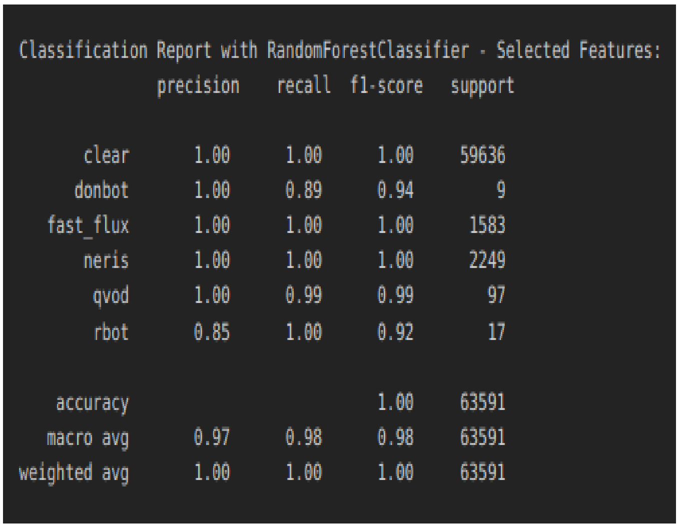
Enhanced Robustness: By combining multiple models' predictions, the system mitigates individual model errors, leading to a more reliable and consistent performance. This is particularly effective in managing the diverse behaviours captured by different models due to the variance in training data subsets, if any.

3.3 Evaluation:

For Evaluation purposes, I have used the Confusion Matrix and Performance Report (which consists of Precision, Recall, F_Score)

4 Results

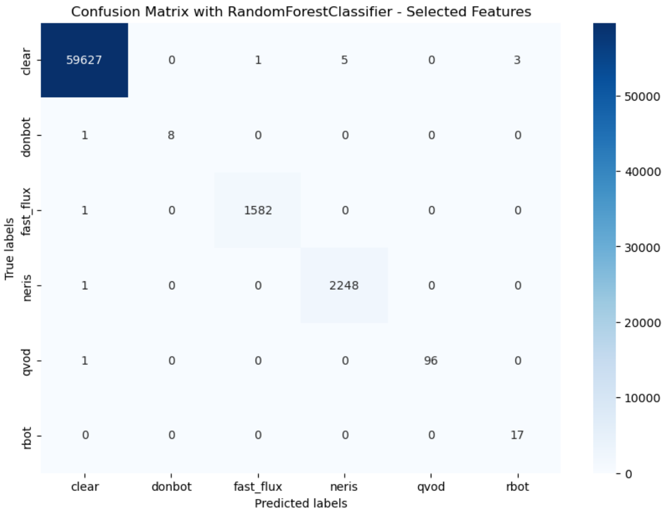
The Confusion matrix and Performance report are as follows:



```
Classification Report with RandomForestClassifier - Selected Features:
               precision    recall  f1-score   support

       clear       1.00      1.00      1.00     59636
      donbot       1.00      0.89      0.94         9
   fast_flux       1.00      1.00      1.00      1583
       neris       1.00      1.00      1.00      2249
        qvod       1.00      0.99      0.99        97
        rbot       0.85      1.00      0.92        17

    accuracy                           1.00     63591
   macro avg       0.97      0.98      0.98     63591
weighted avg       1.00      1.00      1.00     63591
```



Confusion Matrix with RandomForestClassifier - Selected Features

True labels \ Predicted labels	clear	donbot	fast_flux	neris	qvod	rbot
clear	59627	0	1	5	0	3
donbot	1	8	0	0	0	0
fast_flux	1	0	1582	0	0	0
neris	1	0	0	2248	0	0
qvod	1	0	0	0	96	0
rbot	0	0	0	0	0	17

Figure 3: Confusion Matrix and Performance Report

4.1 Key Points:

1. Apart from the Random Forest Classifier, I have used the KNearestNeighbour and the Support Vector Classifier. However, the Random Forest Classifier gave the best results.

2. Once the model was trained on all the features, its performance was as good as when it was trained on reduced features. But to optimize the program and reduce the time complexity.

3. The Confusion Matrix and Performance Report of other classifiers are here.

4. The labels for the provided test set is uploaded on the drive link and can be found here.

5 Conclusion

Botnet detection project demonstrates the potential for effective network traffic classification using machine learning techniques. The combination of data preprocessing, feature selection, and ensemble learning yielded favourable results. Key takeaways from this project include:

1. Feature Importance: Feature selection techniques, such as Recursive Feature Elimination (RFE), guided us in focusing on the most relevant predictors, enhancing model efficiency and interoperability.

2. Ensemble Robustness: The ensemble of Random Forest classifiers, employing both synthetic oversampling (SMOTE) and random undersampling, contributed to robust predictions. Aggregating predictions from multiple models reduced variance and improved overall model reliability.

6 References

1. Data set - Botnet Data
2. Pieces of information on Botnet - BOTNET
3. For Machine Learning models and techniques: Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011. 4. Other pieces of information were taken from the DSE 317/617 lecture notes.
5. Youtube, Geeksforgeek, and towardsdatascience - Medium were used for guidance in this project.