Machine Learning Based Botnet Detection

Zainab Fatima (2020-CE-35)
University of Engineering and Technology, Lahore

ABSTRACT

The project investigates machine learning methods for flow data analysis in relation to network security. With an emphasis on the CTU-Malware-Capture-Botnet-42 dataset, the study attempts to create a useful system for classifying flow data. An Artificial Neural Network (ANN) and conventional classifiers are among the varied group of machine learning models that are trained and assessed. The project tackles model integration and data loading efficiency by utilising Google Colab and Pickle. The final objective is to improve network security by using solid flow data analysis.

INTRODUCTION

Effective flow data analysis in the digital age requires novel methods to network security. In order to strengthen network security protocols, this study explores the relationship between the fields of machine learning and flow data analysis. Accurate flow data classification is becoming increasingly important as data traffic continues to increase. The investigation of diverse machine learning models and techniques serves as the cornerstone for an all-encompassing framework designed to prevent future cyber risks.

LITERATURE REVIEW

Machine learning applications are being used to support botnet detection analysis for network security, according to recent literature. In order to detect intrusions in network traffic, studies examine classic classifiers including SVMs, Decision Trees, and Logistic Regression. More sophisticated flow data classification has become more popular as a result of the development of deep learning, especially ANN.

Research emphasises how important feature engineering is for deriving valuable insights from flow data and improving model performance. Recent research has focused on how computationally efficient models are, particularly in real-time situations. This assessment of the literature highlights the importance of machine learning in strengthening network security through flow data analysis, which informs the project's methodology.

METHODOLOGY

Dataset:

The dataset used for this project is Scenario 1 from the CTU-13 dataset, focusing on the Neris botnet. This dataset was captured at the CTU University in the Czech Republic in 2011, with the primary objective of providing a large-scale dataset containing real botnet traffic intermingled with normal and background traffic. The dataset comprises thirteen scenarios, each executed with specific malware, utilizing various protocols and actions. [1]

Scenario1 Overview:

- **Name:** CTU-13 Scenario 1 (Neris Botnet)
- **Date Captured:** August 10, 2011
- **Duration:** 6.15 hours
- **Pcap Sizes:**
  - Complete: 52GB
  - Botnet: 56MB
  - NetFlow: 1GB

Dataset Loading:

Loading the dataset is a crucial initial step in the flow data analysis project. The dataset used for this study is stored in a Google Drive pickle file named 'flowdata.pickle'. The loading process is implemented using the Google-Colab environment and the pickle module.

Pickle File:

A **pickle file** is a serialized binary file format used in Python to store and exchange data between programs efficiently. It allows complex data structures, such as lists, dictionaries, and even custom objects, to be serialized and saved to a file. The **pickle** module in Python provides functions for serializing and deserializing objects, making it easy to store and retrieve complex data structures.

Model Implementation:

Classifiers used:

This project involves the following machine learning classifiers:

- ➢ SVM
- ➢ Logistic Regression
- ➢ Decision Tree
- ➢ Gaussian Naive Bayes
- ➢ K Nearest Neighbors.
- ➢ Artificial Neural Network

Following are the steps involved in model’s implementation:

Data Preprocessing
Description:

Prior to model training, comprehensive data preprocessing steps were employed to ensure the input features were appropriately prepared for machine learning. [2]

The primary preprocessing steps include:

1. **Normalization:**

- The z-score normalization technique was applied to both the training (X) and test (XT) datasets.

- Normalization ensures that the features have a consistent scale, preventing certain features from dominating others during model training.

Model Training

Description: Each underwent a structured training process:

1. **Initialization:**

    - A model class (e.g., SVMModel or logmodel) was created for each classifier.

    - The constructor received training and test datasets (X, Y, XT, YT), and an optional accuracy label (accLabel).

2. **Data Copying:**

    - Duplicate arrays were created for training (X, Y) and test (XT, YT) datasets to ensure the original data integrity.

3. **Normalization:**

    - Z-score normalization was applied separately to training (X) and test (XT) datasets, ensuring that the mean and standard deviation of each feature were consistent.

4. **Model-Specific Training:**

    - The specific machine learning model was instantiated and trained using the normalized training data (X, Y).

DISCUSSION

Model Performance

In this section, we assess the performance of various machine learning models on the flow data from the CTU-13 dataset. We utilize key metrics such as Accuracy, Precision, Recall, F1 Score, and the Confusion Matrix to comprehensively evaluate each model's classification capabilities.

Definitions and Formulas:

- **Accuracy:**

    The proportion of correctly classified instances among the total instances.

$$\boldsymbol{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision:**

    The ratio of correctly predicted positive observations to the total predicted positives.

$$\boldsymbol{Precision} = \frac{TP}{TP + FP}$$

- **Recall (Sensitivity):**

    The ratio of correctly predicted positive observations to the all observations in the actual class.

$$\boldsymbol{Recall} = \frac{TP}{TP + FN}$$

- **F1 Score:**

    The harmonic means of Precision and Recall, providing a balance between the two metrics.

$$\boldsymbol{F1\ Score} = \frac{2 * Precision * Recall}{Precision + Recall}$$

- **Confusion Matrix:**

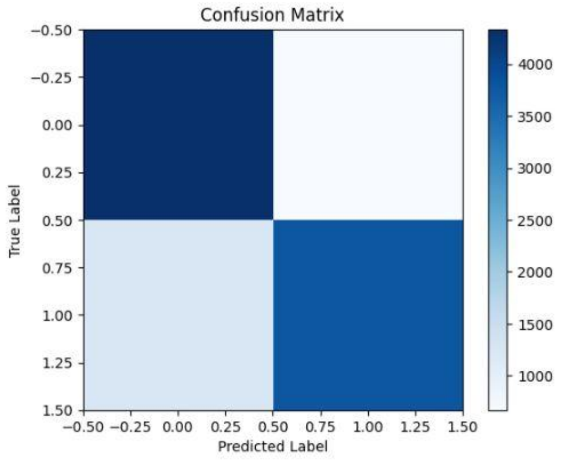
    A matrix representing the performance of a classification algorithm, illustrating True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN).

$$\textbf{Confusion Matrix} = \begin{vmatrix} TN & FP \\ FN & TP \end{vmatrix}$$

RESULTS:

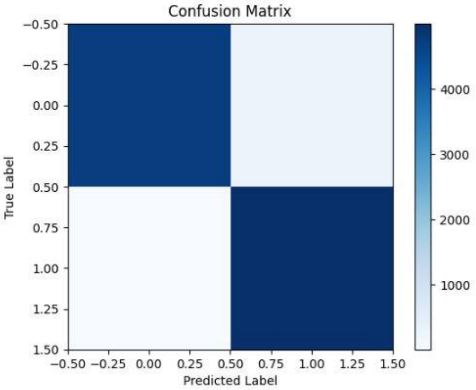
SVM Model:

- Accuracy: 0.81
- Precision: 0.85
- Recall: 0.76
- F1 Score: 0.80
- Confusion Matrix:



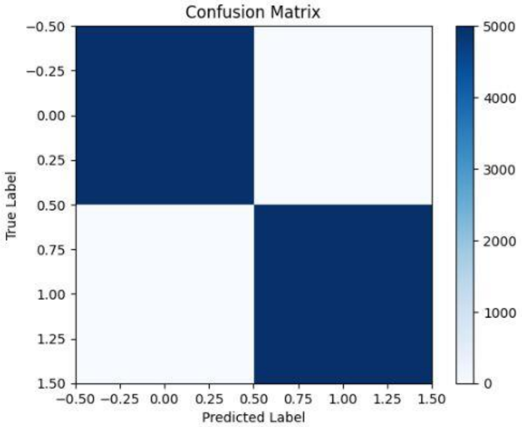
Logistic Regression Model:

- Accuracy: 0.97
- Precision: 0.94
- Recall: 1.00
- F1 Score: 0.97
- Confusion Matrix:



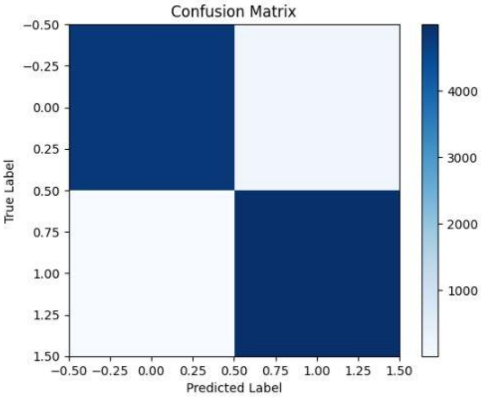
Decision Tree Model:

- Accuracy: 1.00%
- Precision: 1.00
- Recall: 1.00
- F1 Score: 1.00
- Confusion Matrix:



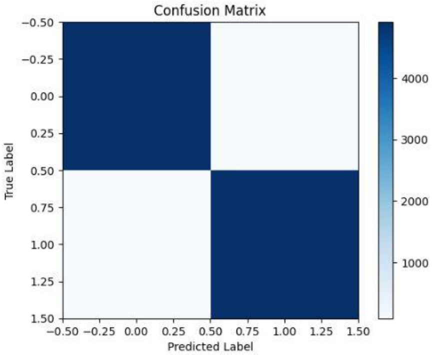
Naive Bayes Model:

- Accuracy: 0.98
- Precision: 0.97
- Recall: 1.00
- F1 Score: 0.98
- Confusion Matrix:



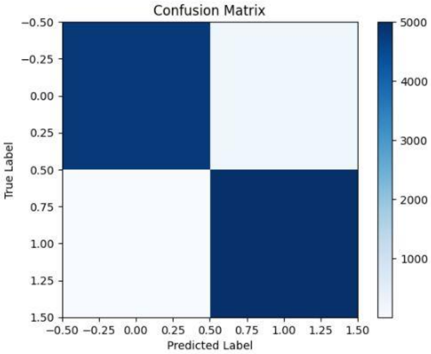
Knn Model:

- Accuracy: 0.98%
- Precision: 0.98
- Recall: 0.98
- F1 Score: 0.98
- Confusion Matrix:



ANN Model:

- Accuracy: 0.98%
- Precision: 0.96
- Recall: 1.00
- F1 Score: 0.98
- Confusion Matrix:



Performance Summary:

The models exhibit varying degrees of accuracy and efficacy in classification tasks, as evidenced by the evaluation metrics.

- Decision Tree Model outperforms other models in Accuracy, Precision, Recall, and F1 Score, making it an overall top performer.
- Logistic Regression Model also demonstrates high Accuracy and Precision.
- Naive Bayes Model excels in Recall, showcasing its effectiveness in capturing true positives.
- ANN Model achieves a balanced performance across all metrics.

CONCLUSION

The study demonstrates the effectiveness of various machine learning models in flow data analysis. Traditional classifiers like Decision Tree and SVM exhibit high accuracy, while the ANN model presents an alternative with potential for improved performance. Understanding the trade-offs between models is essential for selecting the most suitable approach based on specific network security requirements.

FUTURE WORK

Future work should focus on:

Feature Engineering:

Exploring additional features for improved model performance, considering the dynamic nature of network traffic.

Model Optimization:

Fine-tuning hyperparameters to enhance accuracy and responsiveness to emerging threats.

Real-time Analysis:

Adapting models for real-time flow data analysis, ensuring timely detection and response to potential security incidents.

REFERENCES

[1] *The CTU-13 dataset. A labeled dataset with botnet, normal and background traffic.* (no date) *Stratosphere IPS*. Available at: https://www.stratosphereips.org/datasets-ctu13 (Accessed: 05 January 2024).

[2] *GitHub*. Available at: https://github.com/NagabhushanS/Machine-Learning-Based-Botnet-Detection/blob/master/src/dataset_load.py (Accessed: 05 January 2024).