Malware Detection in Network Traffic

Yurim Park

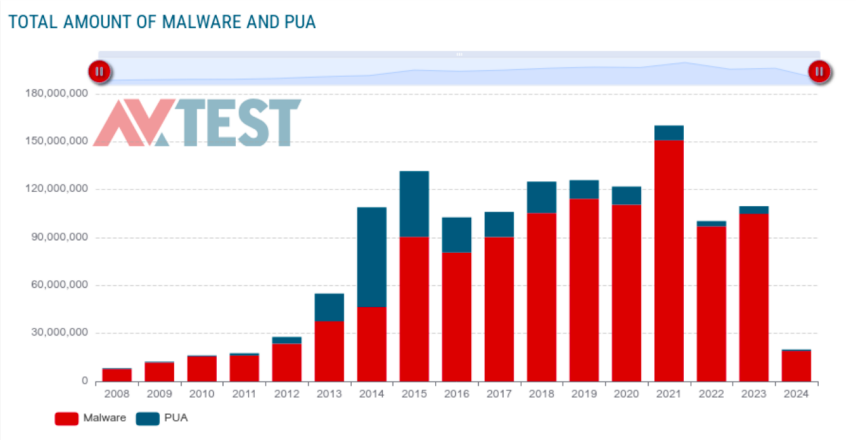
May 3, 2024

1 Abstract

This report presents an analysis of malware detection in network traffic, emphasizing the use of machine learning techniques to identify malicious activity. The motivation for the study stems from the increasing frequency and sophistication of cyberattacks, highlighting the need for robust detection mechanisms. By leveraging labeled datasets, including malicious and benign network traffic, two machine learning classifiers—Support Vector Machines (SVM) and Naive Bayes—were trained and evaluated. The results indicate that while the SVM model demonstrated respectable accuracy, the Naive Bayes classifier outperformed it, achieving higher accuracy and better generalization. Through comprehensive data preprocessing, feature encoding, and cross-validation, this study demonstrates the effectiveness of machine learning in enhancing cybersecurity, with a particular focus on proactive malware detection.

2 Introduction

2.1 Background



In the realm of cybersecurity, malware stands as a persistent and pervasive threat, capable of wreaking havoc on networks and systems worldwide. Malicious software, or malware, encompasses a broad range of malicious programs designed to infiltrate, disrupt, or damage computer systems, often with nefarious intent. From viruses and worms to ransomware and spyware, malware manifests in various forms, each posing unique challenges to cybersecurity professionals.

One of the primary vectors through which malware propagates and operates is network traffic. Malware often relies on network communications to spread across systems, communicate with

command-and-control servers, or exfiltrate sensitive data. As such, analyzing network traffic data has become a crucial component of cybersecurity defense strategies.

2.2 Motivation and Application

The escalating frequency and sophistication of malware incidents underscore the critical importance of detecting malicious activity within network traffic. In today's interconnected digital landscape, where systems seamlessly communicate and exchange data, malware finds fertile ground for propagation and exploitation. The interconnected nature of modern networks not only facilitates the rapid spread of malware but also amplifies its potential impact, as compromised systems can serve as launching pads for further attacks or as conduits for exfiltrating sensitive information.

The motivation for detecting malware in network traffic extends beyond mere safeguarding against immediate threats; it embodies a proactive approach to cybersecurity that emphasizes preemptive defense and threat mitigation. By monitoring network traffic in real-time and analyzing it for signs of malicious behavior, organizations can effectively identify and isolate potential threats before they escalate into full-blown security breaches. This proactive stance not only minimizes the risk of data breaches and system compromise but also bolsters overall resilience against evolving cyber threats.

In essence, the motivation for detecting malware in network traffic lies in the imperative to proactively safeguard interconnected systems and data assets against evolving cyber threats. By leveraging advanced detection technologies and proactive defense strategies, organizations can mitigate the risk of cyber attacks, preserve trust and confidence in digital ecosystems, and uphold the resilience of critical infrastructure in an increasingly interconnected world.

3 Methodology

3.1 Dataset and Collection

Field	Description	Type
ts	The timestamp of the connection event.	time
uid	A unique identifier for the connection.	string
id.orig_h	The source IP address.	addr
id.orig_p	The source port.	port
id.resp_h	The destination IP address.	addr
id.resp_p	The destination port.	port
proto	The network protocol used (e.g., 'tcp').	enum
service	The service associated with the connection.	string

Field	Description	Type
duration	The duration of the connection.	interval
orig_bytes	The number of bytes sent from the source to the destination.	count
resp_bytes	The number of bytes sent from the destination to the source.	count
conn_state	The state of the connection.	string
local_orig	Indicates whether the connection is considered local or not.	bool
local_resp	Indicates whether the connection is considered local or not.	bool
missed_bytes	The number of missed bytes in the connection.	count
label	A label associated with the connection (e.g., 'Malicious' or 'Benign').	string

In response to this ever-evolving threat landscape, researchers at Stratosphere labs have undertaken the task of meticulously labeling network traffic data to identify connections associated with malicious activity. This meticulous labeling process involves categorizing network flows based on various indicators of compromise (IOCs) such as known malicious IP addresses, suspicious communication patterns, or anomalous behavior.

The resulting dataset serves as a valuable resource for training machine learning models to automatically detect malware in network traffic. By leveraging the vast amount of labeled data, researchers can develop and refine machine learning algorithms capable of identifying subtle patterns and anomalies indicative of malware presence.

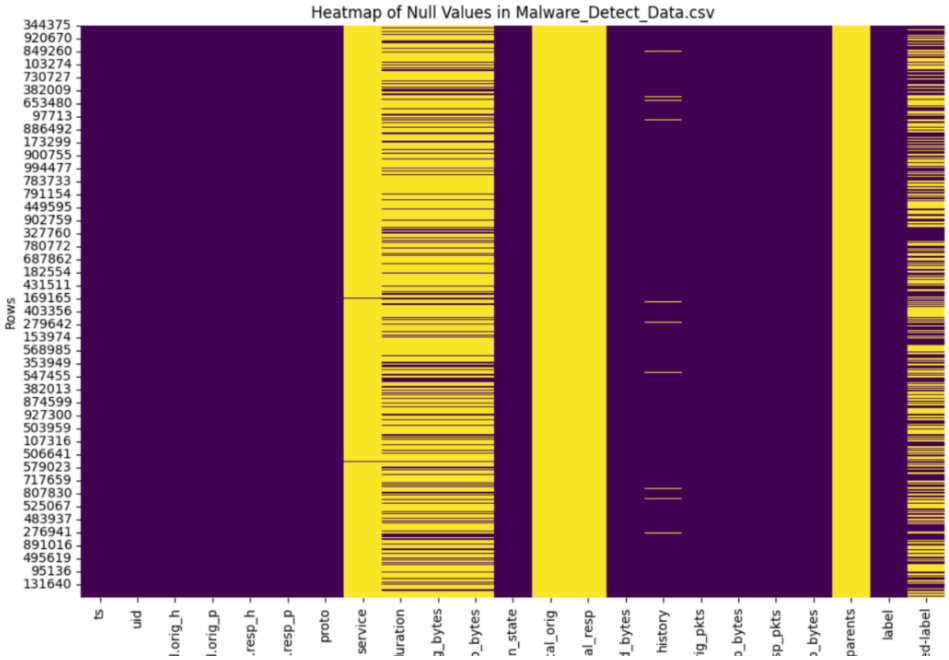
This dataset contains labels that categorize the connections between network flows based on their association with potentially harmful or malicious activity. These labels provide insights into

the nature and intent of the connections, aiding in the identification and classification of cybersecurity threats.

The "Attack" label indicates instances where an attack originates from an infected device and is directed towards another host within the network. Conversely, the "Benign" label is assigned to connections where no suspicious or malicious activities have been detected. The "C&C (Command and Control)" label denotes connections established by infected devices with Command-and-Control servers. Command and Control servers serve as centralized hubs for coordinating and controlling malicious activities across a network of compromised devices. "Distributed Denial of Service (DDoS)" is assigned to connections involving infected devices actively participating in a Distributed Denial of Service attack. These attacks involve flooding a target server or network with an overwhelming volume of traffic, rendering it inaccessible to legitimate users. The presence of this label indicates the involvement of the infected device in coordinated efforts to disrupt or disable network services. Lastly, the "FileDownload" label signifies connections where a file is being downloaded to the infected device.

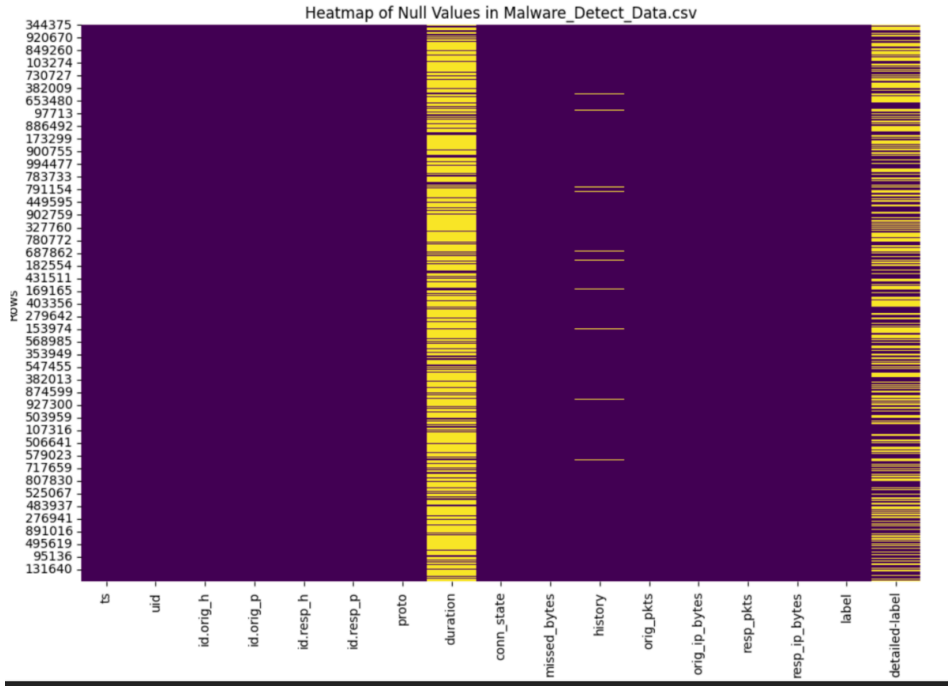
3.2 Data Preprocessing – Removing null values

For data preprocessing, several steps were undertaken to prepare the dataset for machine learning analysis. These steps included removing columns containing null values, label encoding categorical variables, replacing specific values, and converting data types as necessary.

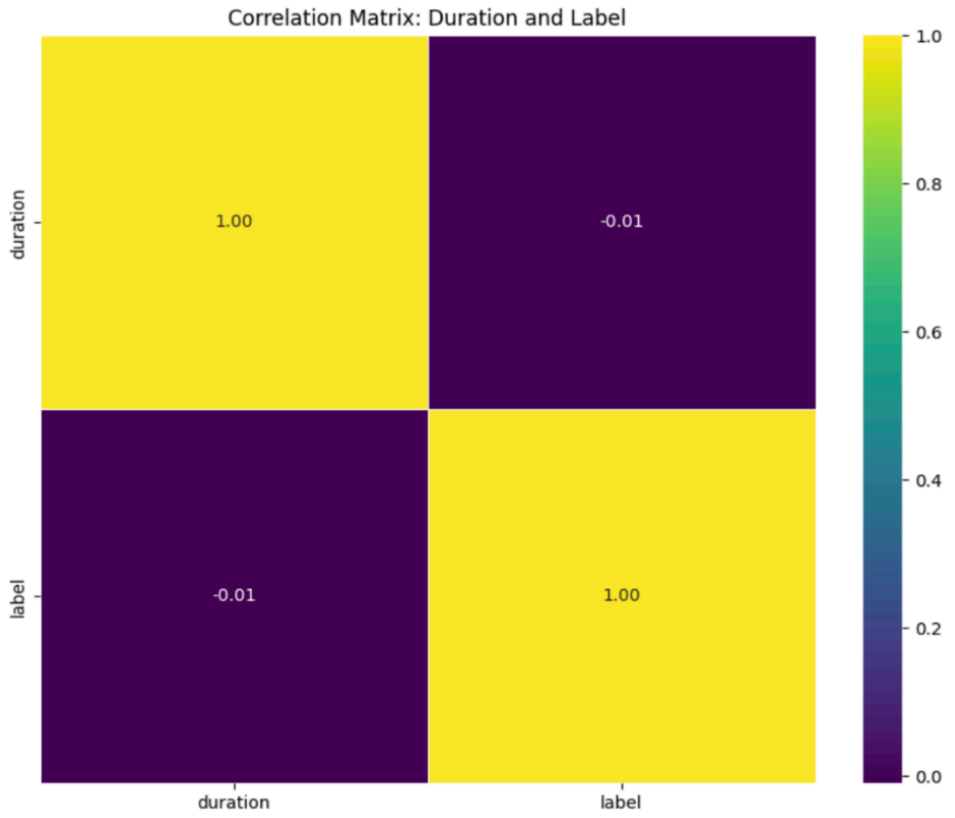


The preprocessing process began with the detection of null values within the dataset. To visualize the distribution of null values across the data, a heatmap was generated. This heatmap provided a clear overview of which columns contained missing values and the extent of their presence.

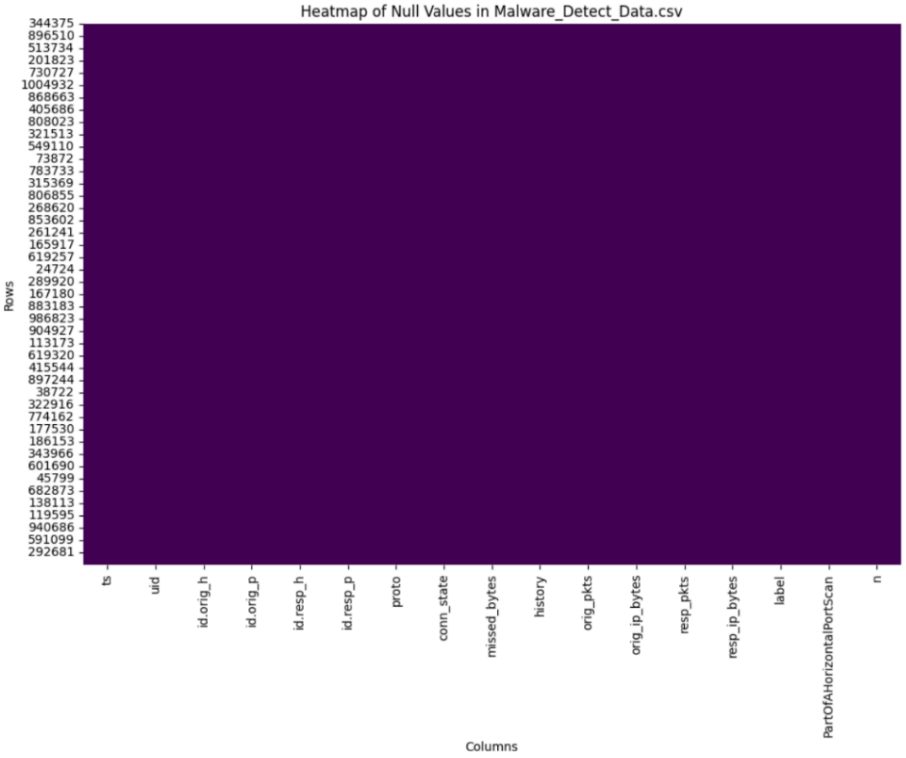
Following the identification of null values, columns containing these missing values were dropped from the dataset. This step ensured that the subsequent analysis would not be affected by incomplete or unreliable data. Additionally, the duration column was identified as problematic due to its lack of correlation with the label column. Despite exploring potential correlations, none were found, leading to the decision to drop the duration column from the dataset as well.



The duration value was the problem. It had null values, but some didn't. So, I drew a correlation matrix to check the relation between duration and label.



It turned out to have no relations between them. So, the duration value was removed also.



So the null values were completely removed.

3.3 Data Preprocessing – One hot encoding

In the data preprocessing step, one-hot encoding was employed to transform categorical variables into a numerical format. This technique is particularly useful when dealing with categorical features that cannot be directly interpreted by machine learning algorithms.

One-hot encoding works by creating new binary columns for each unique category present in the categorical variable. For each observation in the dataset, only one of these binary columns will have a value of "1," indicating the presence of the corresponding category, while all other binary columns associated with other categories will have a value of "0," indicating their absence.

In the context of this dataset, one-hot encoding was applied to both the label data and the protocol (proto) data. For the label data, which includes categories such as "Attack," "Benign," "C&C," "DDoS," and "FileDownload," one-hot encoding generated separate binary columns for each of these categories. This transformation allows machine learning models to interpret and learn from the categorical labels more effectively, enabling accurate classification of network connections based on their associated threats or benign activities.

Similarly, the protocol (proto) data, which may include categories such as "TCP," "UDP," "ICMP," and others, underwent one-hot encoding to represent each protocol type as a binary feature. By encoding protocol data in this manner, machine learning algorithms can leverage the numerical representations of protocols to discern patterns and relationships between different types of network traffic, aiding in the detection of anomalous or malicious behavior.

3.4 Model Training

In the process of model training, two classifiers were utilized: Support Vector Machines (SVM) and Naive Bayes. Each of these classifiers offers unique strengths and approaches to classification tasks, contributing to the robustness and diversity of the overall modeling strategy.

Support Vector Machines (SVM) stand as a powerful tool in supervised learning, excelling in both classification and regression tasks. The fundamental principle behind SVM involves finding a hyperplane that optimally separates different classes of data within a high-dimensional feature space. This hyperplane serves as the decision boundary, maximizing the margin between data points of different classes and thus enhancing the classifier's ability to generalize to unseen data. SVMs are particularly effective in scenarios where the data is not linearly separable, as they can leverage kernel functions to map the data into higher-dimensional spaces where separation is feasible. By employing rigorous optimization techniques, SVMs strive to identify the hyperplane that best discriminates between classes, thereby facilitating accurate and reliable classification.

On the other hand, Naive Bayes classifiers belong to the family of probabilistic classifiers grounded in Bayes' Theorem. These classifiers operate under the assumption of feature

conditional independence, given the class label. Despite its simplistic assumption, Naive Bayes classifiers often exhibit remarkable performance, especially in situations where the training data is limited, or the feature space is vast. By leveraging probability distributions and Bayes' Theorem, Naive Bayes classifiers calculate the posterior probability of each class given the input features and select the class with the highest probability as the predicted label. This probabilistic approach enables Naive Bayes classifiers to handle categorical and numerical data alike, making them versatile and easily adaptable to various classification tasks.

3.5 Model Evaluation

1. Accuracy and Classification Report

To assess the performance of the models, we employed the accuracy_score and classification_report metrics.

Accuracy Score: The accuracy_score function measures the proportion of correctly predicted instances out of the total instances in the dataset.

Classification Report: The classification_report provides a comprehensive summary of various metrics such as precision, recall, F1-score, and support for each class in the dataset. It offers insights into the model's performance across different classes.

2. Confusion Matrix

We utilized the confusion_matrix to visualize the model's predictions compared to the actual values. This matrix allows us to assess the model's performance in terms of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN).

3. Visualizing the Confusion Matrix

Visualizing the confusion matrix facilitates a deeper understanding of the model's predictive performance across different classes.

In summary, the combination of accuracy metrics, classification reports, confusion matrices, and k-fold cross-validation provides a comprehensive evaluation framework for assessing the performance of the SVM and Naive Bayes classifiers. These techniques offer valuable insights into the models' predictive capabilities, aiding in informed decision-making and model refinement efforts.

4 Results

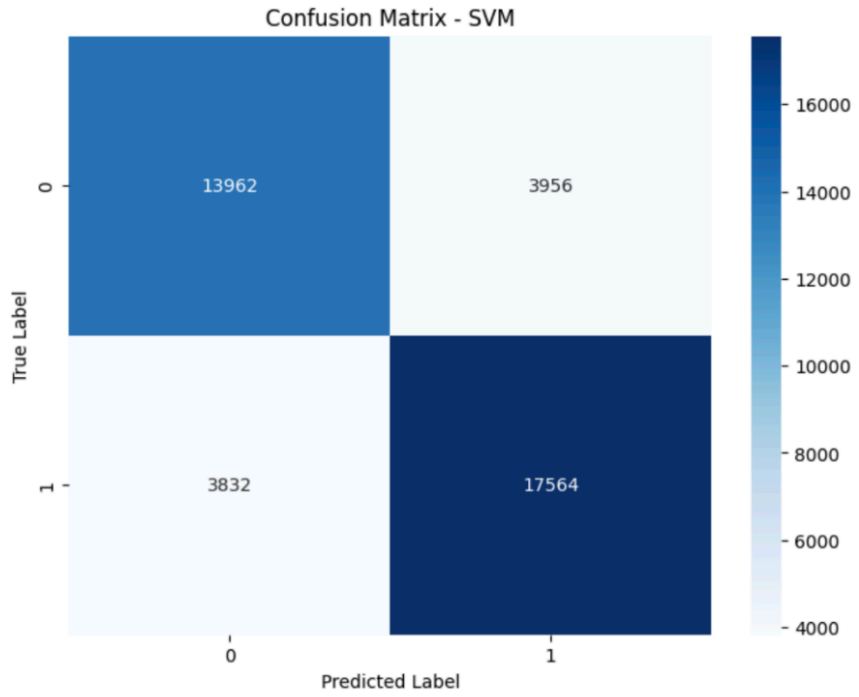
4.1 Results with SVM classifier

```
Accuracy: 0.80190263010632 35
Classification Report:
              precision    recall  f1-score   support

           0       0.78      0.78      0.78     17918
           1       0.82      0.82      0.82     21396

    accuracy                           0.80     39314
   macro avg       0.80      0.80      0.80     39314
weighted avg       0.80      0.80      0.80     39314
```

```
Cross-validation results: [0.54508457 0.54540252 0.53958665 0.545182
83 0.54429253 0.54041335
 0.5463275  0.54759936 0.53933227 0.54689984]
Mean accuracy: 0.5440121431663502
```



The SVM classifier achieved an accuracy of approximately 80.19%. This indicates that the model correctly classified about 80.19% of the instances in the dataset.

Classification Report:

The precision, recall, and F1-score metrics provide insights into the SVM classifier's performance for each class.

- Precision: The precision measures the proportion of true positive predictions out of all positive predictions. For class 0, the precision is 78%, and for class 1, it is 82%.

- Recall: Recall, also known as sensitivity, calculates the proportion of true positive predictions out of all actual positives. Both class 0 and class 1 have a recall of approximately 78% and 82%, respectively.

- F1-Score: The F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics. Class 0 has an F1-score of 78%, while class 1 has an F1-score of 82%.

Cross-validation Results:

Cross-validation was performed using a 10-fold configuration, with accuracy as the scoring metric. The results show the accuracy scores for each fold:

- The accuracy scores range from approximately 53.93% to 54.68% across different folds.

- The mean accuracy of the SVM classifier, calculated from the cross-validation results, is approximately 54.40%.

Overall, the SVM classifier demonstrates respectable performance with an accuracy of around 80.19%. The classification report highlights balanced precision, recall, and F1-score metrics for both classes, indicating reliable classification capabilities. However, the cross-validation results suggest a slight variance in accuracy across different folds, with a mean accuracy of approximately 54.40%. Further analysis and refinement may be necessary to enhance the model's generalization ability and address potential sources of variance.

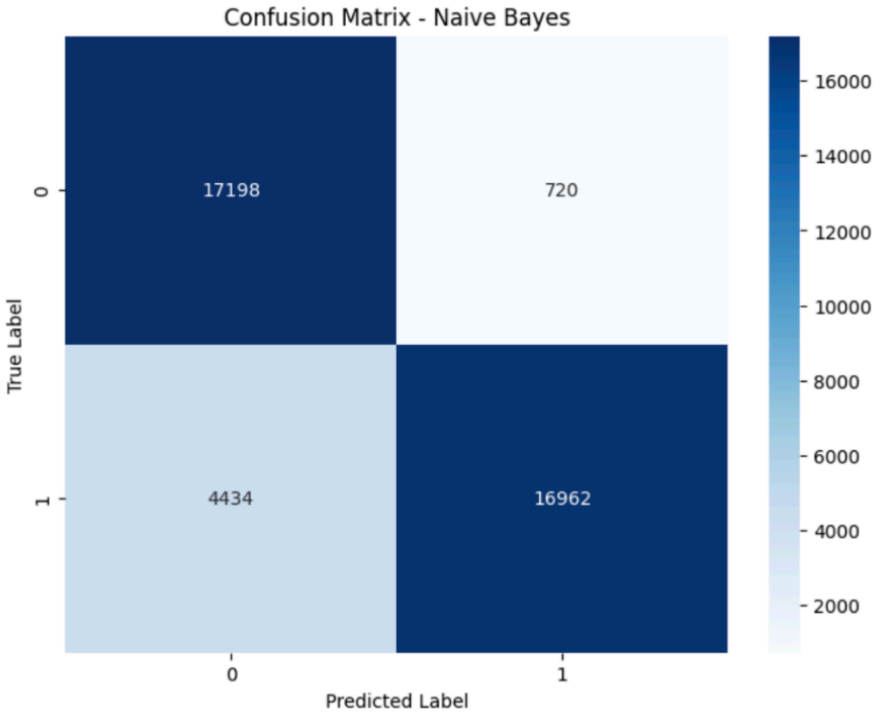
4.2 Results with Naïve Bayes classifier

```
Accuracy (Naive Bayes): 0.8689016635295315
Classification Report (Naive Bayes):
              precision    recall  f1-score   support

           0       0.80      0.96      0.87     17918
           1       0.96      0.79      0.87     21396

    accuracy                           0.87     39314
   macro avg       0.88      0.88      0.87     39314
weighted avg       0.88      0.87      0.87     39314
```

```
Cross-validation results: [0.93691975 0.92490144 0.93647059 0.924006
36 0.93869634 0.89825119
 0.930938   0.92559618 0.93265501 0.92400636]
Mean accuracy: 0.9272441219638294
```



Accuracy:

The Naive Bayes classifier achieved an accuracy of approximately 86.89%. This indicates that the model correctly classified about 86.89% of the instances in the dataset.

Classification Report:

The precision, recall, and F1-score metrics provide insights into the Naive Bayes classifier's performance for each class.

- Precision: The precision measures the proportion of true positive predictions out of all positive predictions. For class 0, the precision is 80%, and for class 1, it is 96%.

- Recall: Recall, also known as sensitivity, calculates the proportion of true positive predictions out of all actual positives. Class 0 has a recall of approximately 96%, while class 1 has a recall of about 79%.

- F1-Score: The F1-score is the harmonic mean of precision and recall, providing a balance between the two metrics. Both class 0 and class 1 have an F1-score of approximately 87%.

Cross-validation Results:

Cross-validation was performed using a 10-fold configuration, with accuracy as the scoring metric. The results show the accuracy scores for each fold:

- The accuracy scores range from approximately 89.83% to 93.87% across different folds.

- The mean accuracy of the Naive Bayes classifier, calculated from the cross-validation results, is approximately 92.72%.

The Naive Bayes classifier demonstrates excellent performance with an accuracy of around 86.89%. The classification report highlights balanced precision, recall, and F1-score metrics for both classes, indicating reliable classification capabilities. Additionally, the cross-validation results suggest consistent high accuracy across different folds, with a mean accuracy of approximately 92.72%. Overall, the Naive Bayes classifier exhibits strong predictive power and generalization ability, making it a suitable choice for classification tasks in cybersecurity applications.

5 Discussion

5.1 Evaluation of Results

The Naive Bayes classifier surpasses the SVM in terms of accuracy, precision, recall, and F1-score metrics. It demonstrates superior performance in correctly classifying instances, especially for class 0 (benign connections).

SVMs are effective in capturing complex decision boundaries and handling high-dimensional data. However, in this case, the SVM classifier may struggle with generalization, as evidenced by the lower cross-validation mean accuracy.

Naive Bayes: Naive Bayes classifiers rely on the assumption of feature conditional independence, making them efficient and robust for classification tasks. The Naive Bayes classifier's strong performance and high cross-validation mean accuracy highlight its ability to generalize well to unseen data.

While both classifiers demonstrate respectable performance, the Naive Bayes classifier exhibits superior accuracy and generalization ability compared to the SVM. Its robustness and efficiency make it a suitable choice for malware detection in network traffic, offering reliable predictions and consistent performance across different datasets.

In summary, the Naive Bayes classifier emerges as the preferred choice for malware detection in this scenario, owing to its superior accuracy, generalization ability, and efficient classification approach.

6 References

https://www.stratosphereips.org/datasets-iot23

https://www.kaggle.com/datasets/agungpambudi/network-malware-detection-connection-analysis