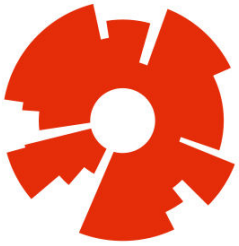

HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY

Computer Science department

Machine Learning in Cybersecurity

T-710-MLCS

# Report 3 - NIDS experiments

Saldana Giorgio
Email: giorgio24@ru.is

27th September 2024

# Contents

# Introduction

In the current digital landscape, the rapid expansion of network systems has led to an increasing vulnerability to cyber-attacks. Among these threats, network intrusions pose a significant risk to data security, necessitating the development of efficient detection mechanisms. One promising approach to identifying these threats is through the use of machine learning algorithms. This report focuses on employing decision trees and random forest classifiers to detect network intrusions using the CIC-IDS2017 dataset, a widely recognized dataset in intrusion detection research.

The primary objective of this study is to preprocess the CIC-IDS2017 dataset, apply classification techniques, and evaluate the performance of decision trees and random forests in identifying various types of network intrusions. These classifiers will be tested on two different dataset splits: a random 60%-40% split and a split based on specific days of the week. By comparing the performance of both classifiers, this report aims to highlight the strengths and weaknesses of decision trees and random forests in terms of accuracy and feature importance. Furthermore, it seeks to identify key features that contribute to the detection of different types of network attacks, offering insights into the decision-making process of these classifiers.

To reproduce the code and results presented in this report, the Python file **pipeline.py** can be used. This script is designed to handle all necessary steps, including data preprocessing, model training, evaluation, and feature importance analysis, using decision trees and random forest classifiers on the CIC-IDS2017 dataset.

# 1 Data preprocessing and splitting

The Python code for Task 1 is designed to handle the preprocessing and splitting of a network intrusion detection dataset. Instead of relying on standalone functions, the code leverages a class-based structure, which promotes better organization and reusability according to best practices in Python programming. It reads multiple CSV files, aggregates the labels into four categories (Benign, DoS, PortScan, and Exploit), and ensures the data is numeric. The data is then split into training and testing sets, either based on specific days of the week or using a randomized split, depending on the mode selected. Label encoding is applied to convert categorical labels into numerical form, and the code also provides functions for inspecting and plotting data distribution

# 2 Train and test a decision tree classifier

The code for task 2 relies on a basic implementation of a Decision Tree from the scikit-learn package. The code follows a class-based structure to ensure reusability and will also be used to generalize the implementation of a Random Forest through the same class (model_trainer.py). This class is responsible for instantiating the classifier and using the attributes of the instantiated classifier to train it. The class also provides functions to print a classification report, evaluate and compare metrics across different types of split modes and classifiers, and plot the top 10 most important features that differentiate node labels in both trees.

# 3 Decision Tree Results

In this section, is presented the performance results of the decision tree classifier using two different dataset splits: a 60%-40% random split and a split based on the days of the week (Monday-Wednesday for training and Thursday-Friday for testing). Will be included confusion matrices for a more detailed understanding of the classification outcomes.

## Decision Tree Classifier Performance on float-based split

Here, particularly will be displayed the results from the dataset splitting into 60% for training and 40% for testing. The confusion matrix helps visualize the classifier's performance by showing the exact counts of true and predicted labels for each class.

The confusion matrix provides a detailed breakdown of the decision tree classifier's predictions compared to the true labels. Each cell in the matrix represents the number of instances predicted for a particular class versus the actual class.

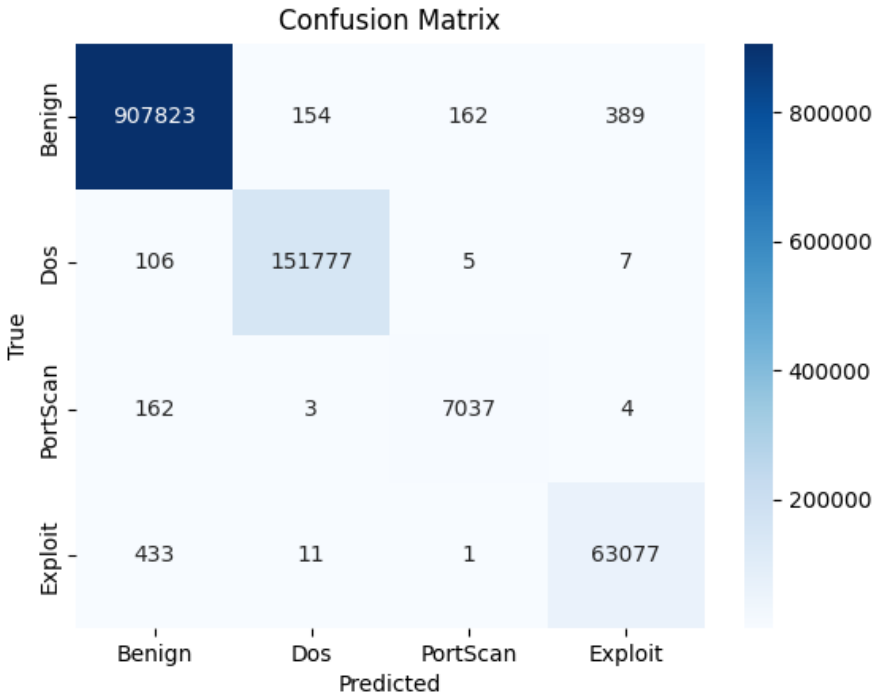


Figure 1: DT Confusion Matrix by 60/40 split

By analizing this matrix we can retrieve some useful information about our model:

- **Benign**: Of the 908,528 true benign instances, 907,823 were correctly classified. Only 154 benign instances were misclassified as DoS, 162 as PortScan, and 389 as Exploit. This very low misclassification rate is consistent with the high precision and recall values for benign traffic seen in the classification report.
- **Dos**: Out of 151,895 true DoS instances, 151,777 were correctly predicted, and only 106 were mistakenly classified as Benign. A very small number of DoS instances were misclassified as PortScan (5) or Exploit (7). This high level of accuracy confirms the decision tree's effectiveness in identifying DoS attacks, as seen in the high recall and precision scores.
- **PortScan**: For the PortScan class, there were 7,206 true instances, of which 7,037 were correctly classified. However, 162 were misclassified as Benign, and 3 were misclassified as DoS. The lower recall and precision scores for this class reflect this slight increase in misclassifications, indicating that the decision tree had more difficulty accurately detecting all PortScan attacks compared to the other classes.
- **Exploit**: The classifier correctly identified 63,077 out of 63,522 exploit instances. A small number of exploit instances were misclassified as Benign (433), DoS (11), and PortScan (1). These misclassifications are minimal, further reinforcing the strong performance for this class, as indicated by the high F1-score in the classification report.

## Decision Tree Classifier Performance on Day-Based Split

The confusion matrix provides further insight into the decision tree classifier's performance when the dataset is split by days:

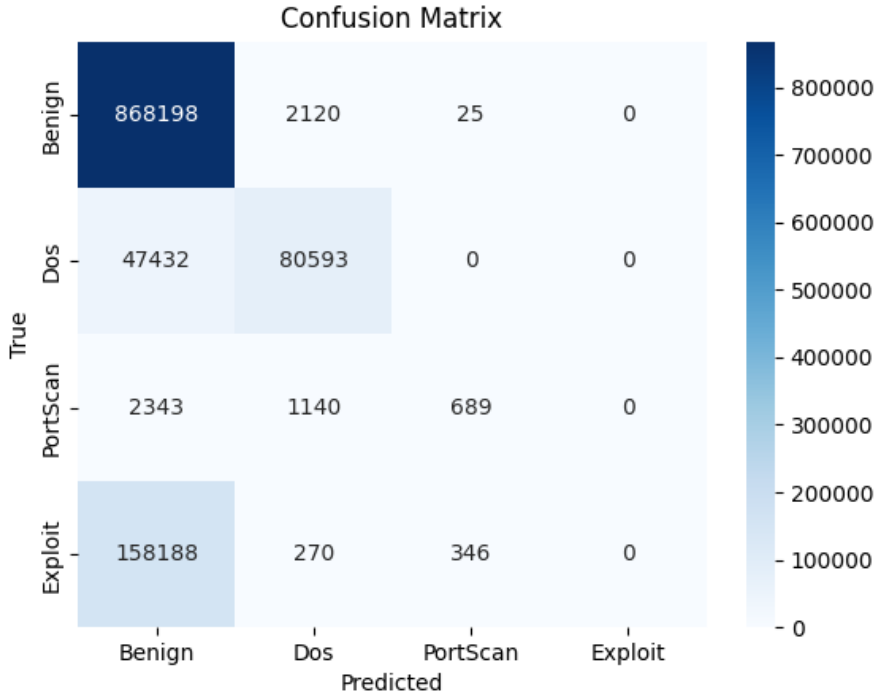


Figure 2: DT Confusion Matrix by Days split

- **Benign Traffic:** Of the 870,343 true benign instances, 868,198 were correctly classified, with 2,120 misclassified as DoS, 25 as PortScan, and none as Exploit. These numbers indicate the classifier's strong ability to correctly identify benign traffic, although some instances were confused with DoS and PortScan traffic.
- **DoS Attacks:** Of the 128,025 true DoS instances, the classifier correctly identified 80,593 as DoS. However, 47,432 instances were incorrectly labeled as benign, reflecting a substantial misclassification rate for this attack type. No DoS instances were misclassified as PortScan or Exploit.
- **PortScan:** The classifier successfully identified only 689 out of 4,172 true PortScan instances. A significant number of PortScan instances were misclassified, with 2,343 labeled as benign and 1,140 as DoS, showing the classifier's difficulty in correctly detecting PortScan attacks.
- **Exploit Attacks:** The classifier completely failed to detect any of the 158,804 Exploit instances, with 158,188 misclassified as Benign, highlighting the model's inability to detect Exploit traffic in this scenario.

The day-based split posed greater challenges for the decision tree classifier, particularly in identifying attack traffic. While the classifier maintained high accuracy for

benign traffic, it struggled significantly with DoS, PortScan, and especially Exploit attacks.

In conclusion, while the decision tree performs reasonably well on benign traffic, its ability to detect attacks, particularly exploits, is significantly limited in this day-based split. These results highlight the potential limitations of using a simple decision tree model for intrusion detection in temporally split datasets, where more sophisticated models may perform better.

# 4 Random Forest classifier and Its Results

The Random Forest classifier is implemented using the same ModelTrainer class, ensuring reusability and flexibility. By passing 'random_forest' as a parameter, the class instantiates the RandomForestClassifier from scikit-learn. This class provides functions to train the model, generate classification reports, and plot confusion matrices. Additionally, it ranks and visualizes the top 10 most important features that influence the classifier's decisions. This implementation allows for efficient handling of large datasets while providing insights into feature importance and model performance.

## Random Forest Classifier Performance on float-based split

This section discusses the results of the random forest classifier when the dataset was split randomly into 60% for training and 40% for testing. The confusion matrix provides a detailed visualization of the classifier's predictions compared to the true labels across the four aggregated traffic classes.

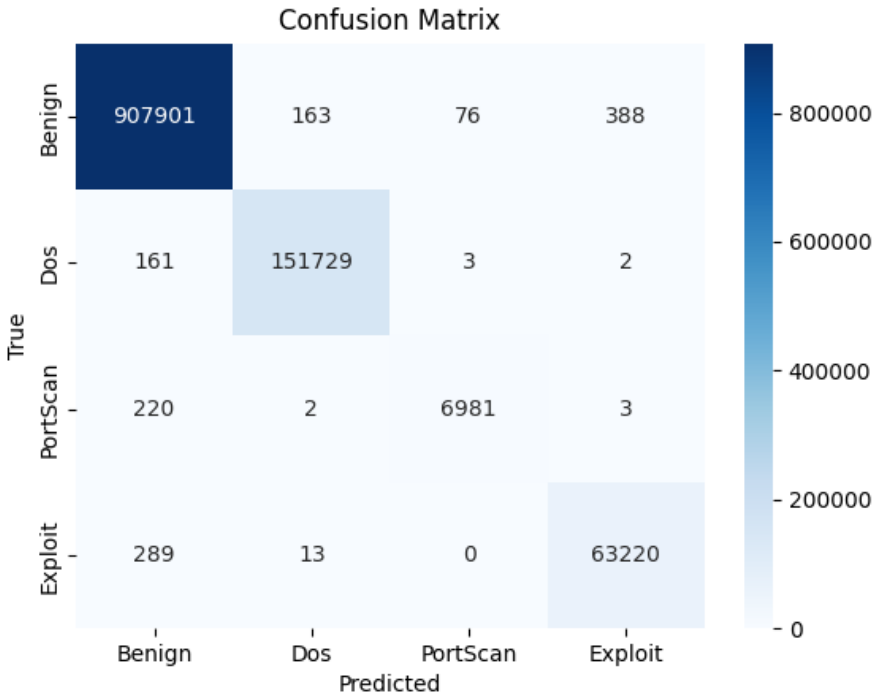


Figure 3: RF Confusion Matrix by 60/40-based split

The confusion matrix shows how the random forest classifier performed in classifying the network traffic for the 60%-40% split:

- **Benign Traffic:** Out of 908,528 true benign instances, 907,901 were correctly classified. Only 163 were misclassified as DoS, 76 as PortScan, and 388 as Exploit. The small number of misclassifications demonstrates the random forest's strong ability to detect benign traffic, significantly minimizing false positives for this class.
- **DoS Attacks:** Of the 151,895 true DoS instances, 151,729 were correctly classified, with only 161 instances misclassified as benign, and very few as PortScan (3) or Exploit (2). This shows a strong capacity to correctly classify DoS attacks with minimal errors.
- **PortScan:** For the PortScan class, 6,981 out of 7,206 instances were correctly identified, while 220 were misclassified as benign and 2 as DoS. Despite some misclassifications, the random forest classifier demonstrated high accuracy in detecting PortScan traffic.
- **Exploit Attacks:** Of the 63,522 true exploit instances, 63,220 were correctly identified, with 289 misclassified as benign and 13 as DoS. None were misclassified as PortScan. These results highlight the model's strong performance in detecting exploit traffic with minimal confusion between classes.

## Random Forest Performance on Day-based split

Instead here is provided the results of the random forest classifier when the dataset was split based on days, where data from Monday to Wednesday was used for training and data from Thursday to Friday was used for testing. The confusion matrix provides a detailed visualization of the classifier's predictions compared to the true labels across the four aggregated traffic classes.

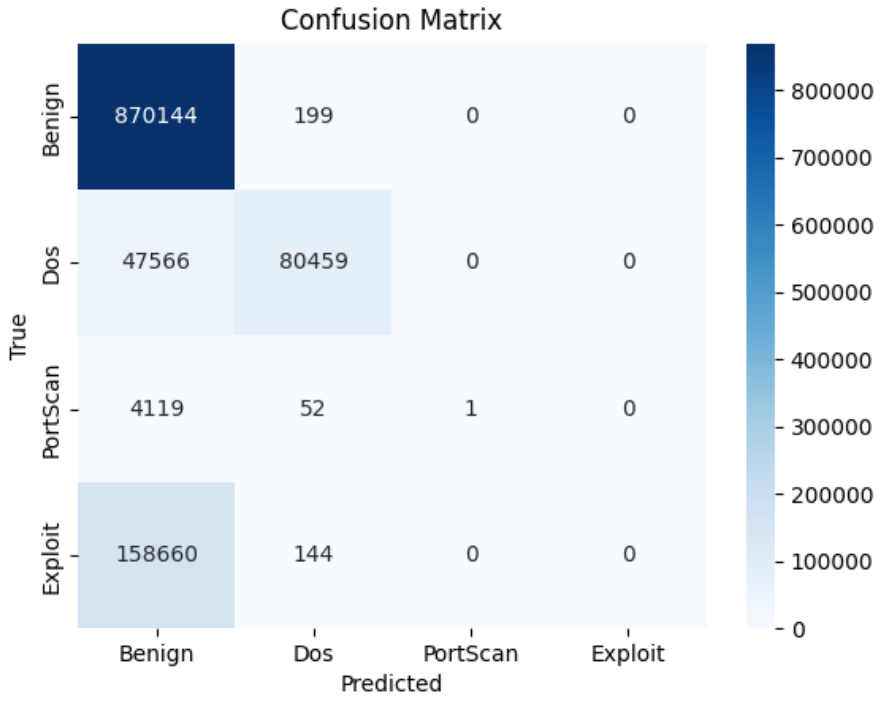


Figure 4: RF Confusion Matrix by Day-based split

The confusion matrix shows how the random forest classifier performed in classifying the network traffic for the day-based split:

- **Benign Traffic:** Out of 870,343 true benign instances, 870,144 were correctly classified. Only 199 were misclassified as DoS, and none were misclassified as PortScan or Exploit. The minimal misclassifications demonstrate the random forest's high accuracy in detecting benign traffic, with almost no confusion between attack classes.
- **DoS Attacks:** Of the 128,025 true DoS instances, 80,459 were correctly classified, while 47,566 were misclassified as benign. No DoS instances were misclassified as PortScan or Exploit. The large number of DoS instances misclassified as benign traffic indicates that the model struggled to separate these two classes in the day-based split.
- **PortScan:** For the PortScan class, out of 4,172 true instances, only 1 was correctly classified. A large number (4,119) were misclassified as benign traffic, and 52 were

misclassified as DoS. This shows that the model had significant difficulty detecting PortScan traffic under this split, misclassifying the majority of the PortScan instances.

- **Exploit Attacks:** Of the 158,804 true exploit instances, none were correctly classified. A total of 158,660 were misclassified as benign, and 144 as DoS. This reflects a complete failure of the model to detect exploit traffic, as all instances were confused with benign or DoS traffic.

In summary, the float-based split provided a more stable and consistent performance, likely because it allowed for a more representative distribution of traffic types between the training and testing sets. Conversely, the day-based split highlighted the model's difficulty in adapting to temporal shifts in network traffic, leading to significant misclassifications, particularly for attack types. This suggests that while random forests are robust with random data splits, they may struggle to generalize across time-based shifts in network traffic, a common characteristic in real-world network intrusion scenarios.

# 5 Feature Importance Analysis

In this section, will be discussed the main features identified by both the decision tree and random forest classifiers as discriminating factors. These features were extracted from both models under two different split configurations: a random 60%-40% split and a day-based split (Monday-Wednesday for training and Thursday-Friday for testing). The figures below show the importance values of the top features for each model and split.

## Decision Tree Feature Importance

The decision tree model relies on key features to classify network traffic accurately. Below, we examine the most important features identified in both the 60%-40% split and the day-based split:

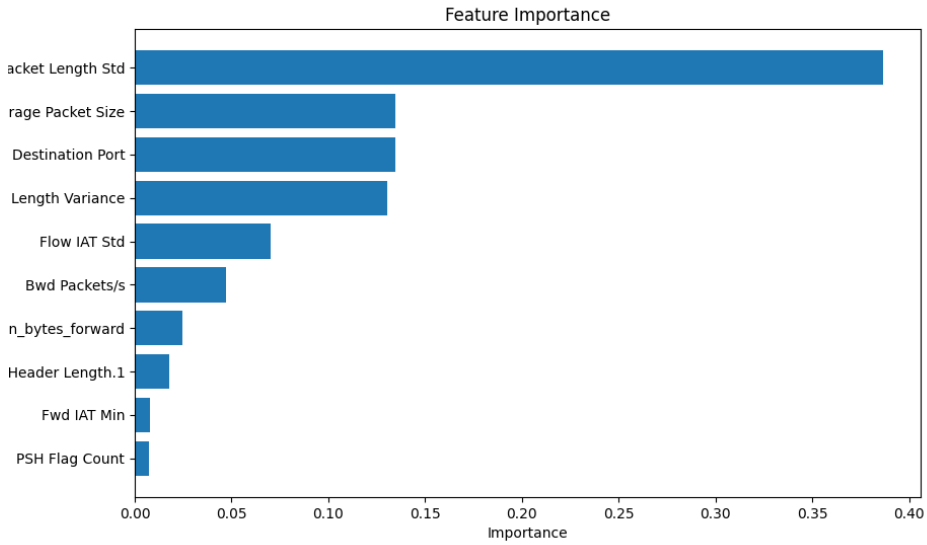


Figure 5: Decision Tree Feature Importance by 60/40 split

- **Packet Length Std:** The standard deviation of packet lengths is the most important feature in both splits, reflecting variability in traffic patterns which is crucial for distinguishing between benign and attack traffic.
- **Average Packet Size:** This feature is important as it helps the model distinguish between different traffic patterns. Larger packet sizes might be indicative of data exfiltration or file-sharing activities, while smaller sizes might be more common in benign traffic.
- **Destination Port:** This feature is important because specific ports are commonly associated with certain types of network traffic and attacks, such as DoS or PortScan attacks.

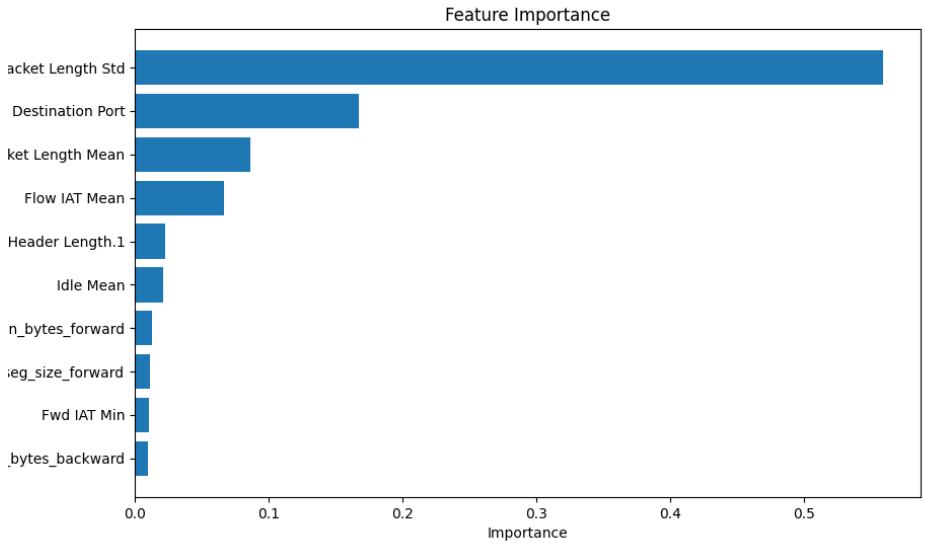


Figure 6: Decision Tree Feature Importance by Day-based split

- **Packet Length Std:** The standard deviation of packet lengths is by far the most important feature, indicating that the variability in packet sizes plays a critical role in differentiating between normal and attack traffic. This metric captures unusual patterns often present in network intrusions.
- **Destination Port:** Certain network ports are frequently targeted in specific types of attacks, such as DoS and PortScan. As a result, the destination port is a highly relevant feature for identifying malicious activity in the network.
- **Packet Length Mean:** The average packet length provides additional insights into traffic behavior, where deviations from typical values may suggest anomalous or malicious traffic, contributing significantly to attack detection.

## Random Forest Feature Importance

Random forest classifiers leverage multiple features more evenly compared to decision trees. The following figures depict the importance of features for random forest models under both split configurations.

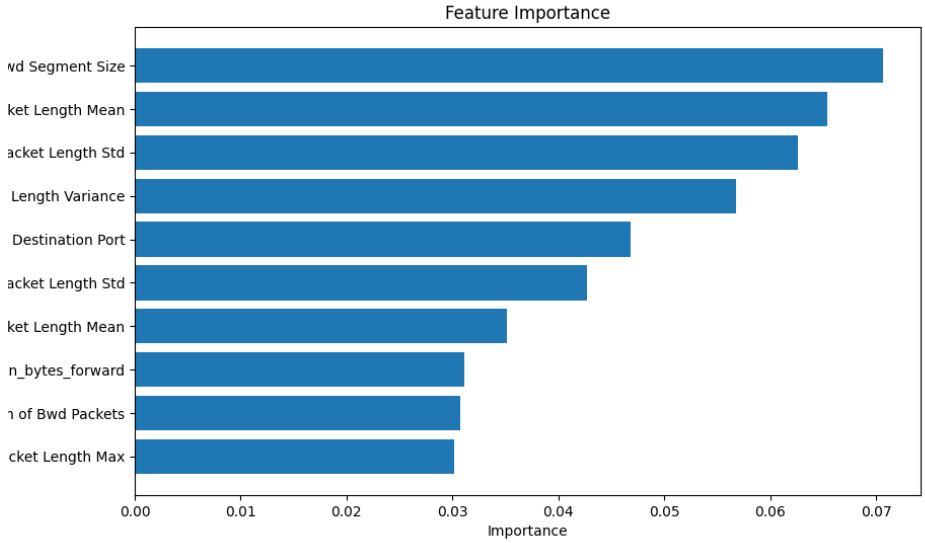


Figure 7: Random Forest Feature Importance by 60/40 split

- **Fwd Segment Size:** The size of forward segments is the most important feature in the Random Forest model, reflecting the significance of traffic size and directionality in identifying patterns that differentiate benign from malicious traffic.
- **Packet Length Mean:** The average packet length is a crucial factor in determining traffic behavior. Variations in packet length averages often indicate whether traffic is normal or involves activities such as data exfiltration or scanning attacks.
- **Packet Length Std:** The standard deviation of packet lengths remains a critical feature, just as with the decision tree model, highlighting the importance of variability in packet sizes for distinguishing between various types of network activities, including intrusions.

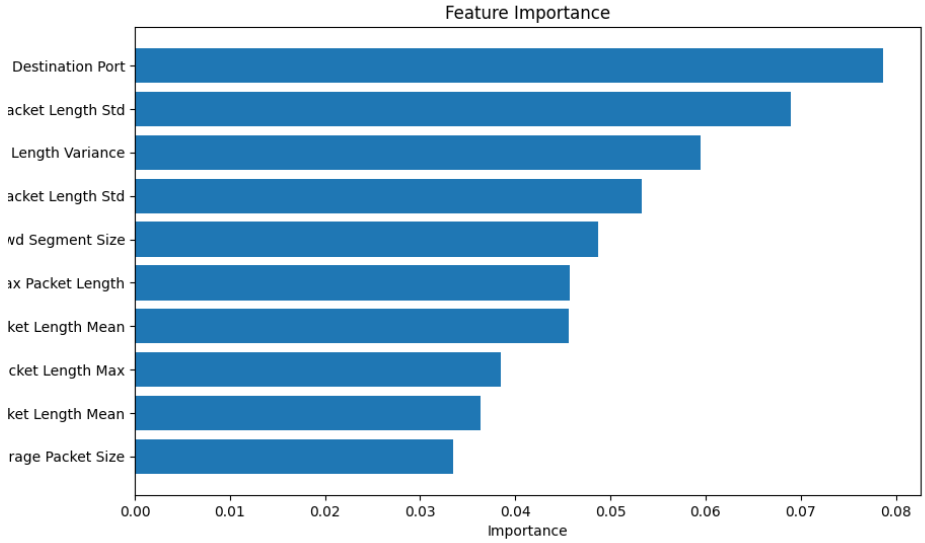


Figure 8: Random Forest Feature Importance by Day-based split

- **Destination Port:** The destination port is the most important feature in the day-based split. This makes sense as certain types of attacks target specific ports, and identifying the ports used by network traffic helps to distinguish between benign and malicious behavior.
- **Packet Length Std:** The standard deviation of packet lengths plays a significant role in detecting irregularities in network traffic. A high variability in packet lengths often correlates with attack traffic, making this feature crucial in the day-based split.
- **Length Variance:** The variance in packet lengths is another critical feature, helping to detect patterns indicative of network intrusions, particularly those involving a large range of packet sizes, such as distributed denial of service (DDoS) attacks.

In summary, the features identified as the most important in both the decision trees and random forests, such as Packet Length Std adn Destination Port play a critical role in distinguishing between benign traffic and various types of network attacks. Features related to packet size and variability, such as Packet Length Std and Length Variance, are particularly useful in detecting traffic anomalies like DoS and DDoS attacks, which often flood the network with irregular packet sizes. Similarly, the Destination Port feature helps in identifying PortScan and Exploit attacks, as these often target specific network ports. These features are essential in the decision-making process of both classifiers and highlight the ability of machine learning models to detect attack patterns based on traffic behavior, even when data is split randomly (60%/40%) or by time (day-based).