Assignment 3
Network Intrusion Detection

T-710-MLCS, Machine Learning in Cyber Security

Reykjavik University - School of Computer Science

Federico Maria Cruciani

`federico24@ru.is`

29. September 2024

1 Introduction

This report covers the work done for Assignment 3 of the Machine Learning in Cyber Security course. The objective of the assignment was to experiment with the Decision Tree and the Random Forest classifiers to perform network intrusion detection. For this, the **CIC-IDS2017**[1] dataset was used.

1.1 Running the project

To run the experiments it's sufficient to execute the file **training.py** with

```
python3 training.py
```

The file allows to add a parameter to select only one model: allowed values are "**tree**" for the Decision Tree and "**forest**" for the Random Forest.

It's important that the dataset folder is placed inside the project folder before execution.

2 Data Preprocessing and Splitting

As specified in the assignment, the function `get_dataset()` in the file **preprocessing.py** splits the dataset based on the **splitmode** parameter and returns the train and test sets after doing preprocessing.

The main preprocessing steps are performed by the function `preprocess()` which does the following:

- Remaps the labels according to the groups required for the assignment
- Performs undersampling of the data points with label "Benign"
- Performs upsampling of the data points with label "Exploit"
- Removes infinite values present in the dataset, these would otherwise lead to errors being raised during execution

3 Decision Tree Classifier

The decision tree classifier is trained by the function `train_decision_tree()` in the file **training.py**.

From the confusion matrix in figure 1 we can see that the model performs well with the 60:40 random split, this is mainly thanks to the resampling done during preprocessing which greatly reduces the difference in representation between the classes. This is also reflected in the scores: precision, recall and f1-score are in fact all greatly above 90%.

[1]https://www.unb.ca/cic/datasets/ids-2017.html

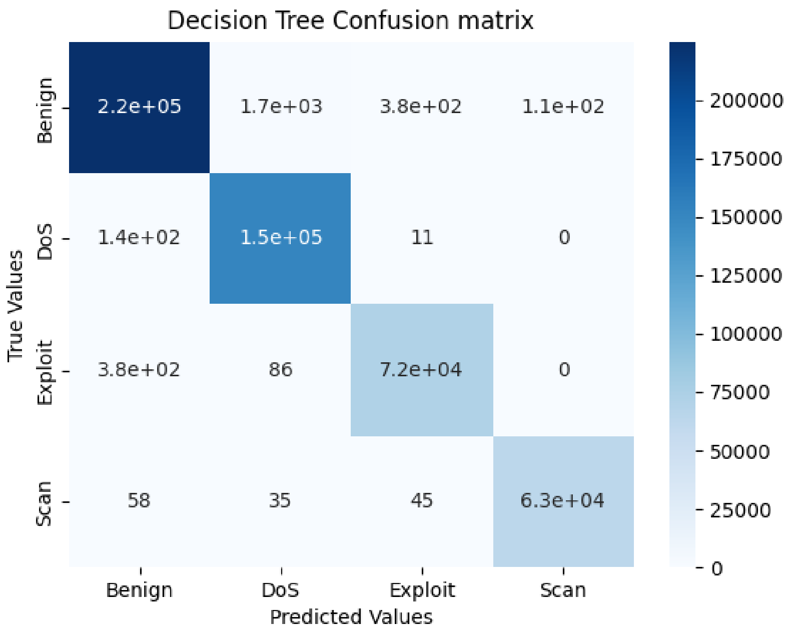


Figure 1: Decision Tree confusion matrix for 60:40 split

When using the dataset split by the days of the week, as shown in figure 2, we instead notice that the performance becomes a lot worse. The *Scan* label is never predicted and the *Exploit* data is never assigned the correct label.

This happens because of how attacks are divided in the different days, for example port scans appear only in the data for Tuesday and Friday so the training set doesn't contain any sample with the *Scan* label. Similarly, the *Exploit* label data is mainly located in those two days so it's very under-represented for the training.

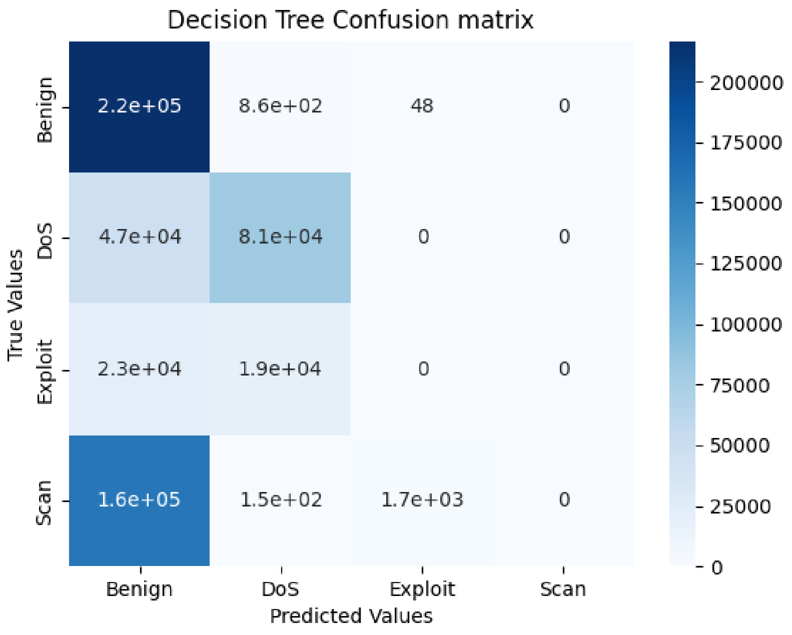


Figure 2: Decision Tree confusion matrix for day split

4 Random Forest Classifier

The Random Forest classifier is trained by the function `train_random_forest()` in the file **training.py**.

This model, as shown in figure 3, also performs well in the random-split dataset and is able to predict all the labels with similar results to the Decision Tree.

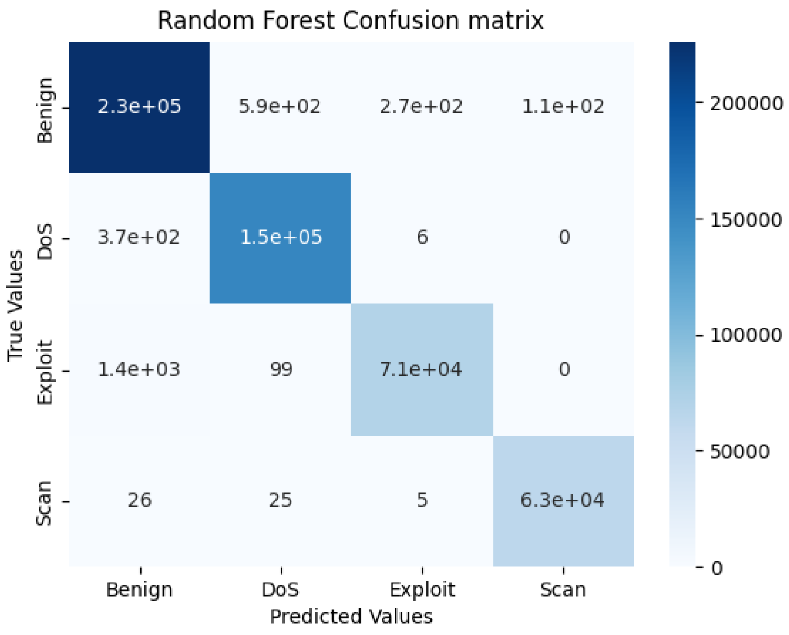


Figure 3: Random Forest confusion matrix for 60:40 split

The performance on the day-split dataset is as bad as that of the Decision Tree, confirming that this is not a good way of splitting this dataset.

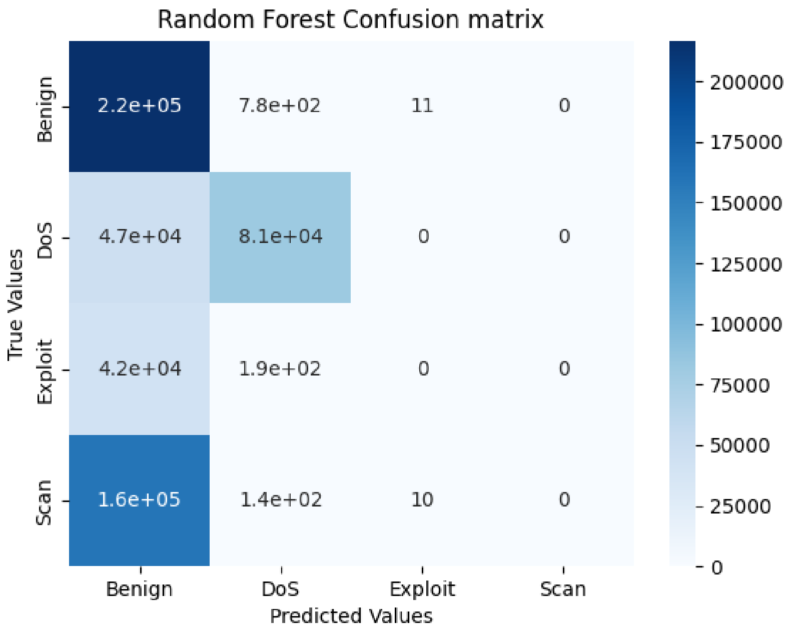
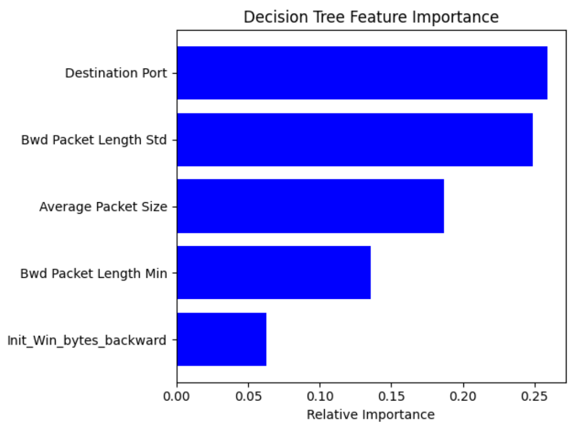
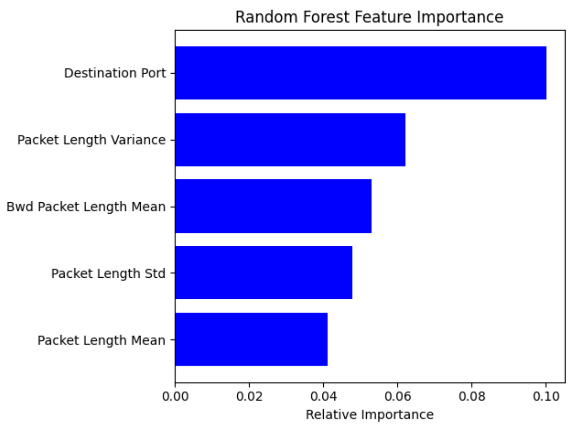


Figure 4: Random Forest confusion matrix for day split

5 Explainability



(a) Decision Tree



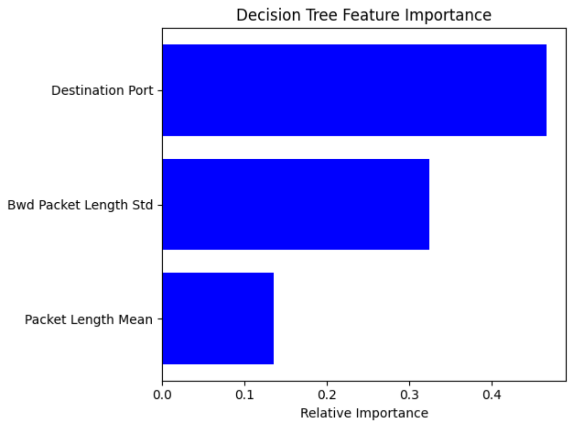
(b) Random Forest

Figure 5: Feature importance plots for 60:40 split

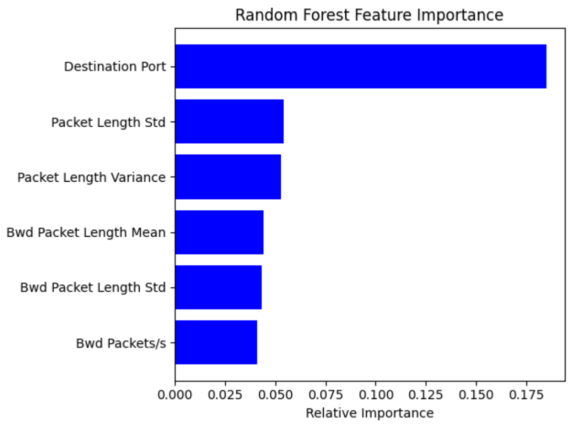
The plots shown in the figures represent the features which have an importance value greater than 0.04.

For both models we can see that the feature that was most useful for discriminating the classes was the *Destination Port*. This can in fact be used to distinguish between benign and malicious activity, as the first type normally uses a small number of the most common ports while attacks can potentially exploit vulnerabilities related to less used protocols which may use different ports.

The other group of features on which both models focused more is the packet length. This type of features can be useful for classifying DoS and DDoS attacks and also detecting port scanning activity, since both rely on sending a high amount of small packets.



(a) Decision Tree



(b) Random Forest

Figure 6: Feature importance plots for day split