School of Computer Science and Informatics

Final Year Individual Project

Detecting DDoS attacks with Network Traffic

Francesco Zucchelli| C1841044

Module Code: CM3203

Supervisor: Amir Javed

Moderator: Jose Camacho Collados

Proforma:

Title: Detecting DDoS attack on a network

Description: Using Cyber range to simulate DoS attack on a network – Within the project student will work on already collected dataset. They will then pre-process the data and build supervised/unsupervised machine learning model to detect an ongoing attack.

Due date: 27/05/2022

Abstract

Distributed Denial of Service attacks cause great difficulty in the industry and to average users of online services. In the growing war against DDoS attacks, a new weapon has been investigated in recent years: Machine Learning. Combining Machine Learning together with Cyber Security, studies have shown success when creating models that can detect malicious network traffic with an accuracy of above “99%”. In this study, old and new datasets with over 10 years of age difference are used to teach different models’ detection of DDoS attacks. These models are then put against each other in metrics such as accuracy and feature importance. The results from this study show that both models’ performance has little to no difference when effectively competing on the same data. Both datasets achieved an accuracy of over “99%” on their own data and over “68%” on the other. Therefore, this study finds it is advantageous to use old in combination with modern data to teach current-day machine learning models.

Acknowledgements

This Final-Year Individual Project will be the final piece of work I submit for my Undergraduate Degree. With all the twists and turns my project has taken, including the changing of the project's method, I would like to thank dearly my supervisor Amir Javed for his great patience with me. Furthermore, despite not being my direct supervisor, I would like to thank Andrew Hood for giving me substantial support throughout this project, especially when the method was changed later in the timeline due to complications.

On a personal note, I would like to thank my family and friends for providing a warm comfort during my final year and particularly during this individual project.

Table of Contents

Table of figures

1. Introduction

More than ever, denial of service attacks (DoS) are being used in cybercrime [1]. DoS attacks attempt to overwhelm a targeted device or network, rendering it unable to manage legitimate traffic and function properly [2]. A DoS attack can become more troublesome by using multiple computers or devices to distribute itself. The difficulty with stopping distributed denial of service attacks (DDoS) comes from their distributed nature precisely because it can be hard to differentiate the real traffic from the malicious. Furthermore, filtering the traffic is harder: if the packets are still clogging the network connection to the internet, this will still prevent regular traffic from reaching the network, thus the filtering does not help against the DDoS attack. The most common type of distributed attack is when the hacker uses a botnet "army" assembled through illegal means by infiltrating into people's devices unsuspectingly and using a fraction of their computational power to carry out small attack (using a small amount of computational power) happens at a large scale (across all of the botnet) on each compromised device [4].

With the frequency of DDoS attacks increasing, coupled with an increase in the use of devices that access the internet - especially ones with poor security that open the door for hackers to create more botnets - it is important to understand how we can combat them and look to what tools we have to improve our arsenal against these attacks. With more for us to defend, there is more for hackers to use to attack and there are more hackers that initiate attacks.

By using machine learning algorithms, we are able to equip ourselves with tools that can recognise patterns of attacks that we cannot. Machine learning algorithms in particular are useful in determining cyber traffic and can be used to prevent attacks in the future through classification [5].

1.1 Aims and Objectives:

The aim of this project is to compare the results of using different datasets to train machine learning algorithms in the detection of DDoS attacks. For detection, the models will be classifying packets/traffic as malicious or benign. To do this, objectives were described:

Core Objectives:

Throughout my project there have been complications which have directly affected the core objectives that were originally communicated in the initial plan.

Previous Core objectives include:

1. ~~Create a dataset using virtual machine simulations of network topologies.~~
2. Create a machine learning model that is able to detect DoS attacks on network topologies on one operating system.
    a. Specifically able to detect at least one type of DoS attack.
    b. Using the traffic and processing data that was taken from various types of network topologies. Traffic and processing data is likely to contain:
        i. Date and time
        ii. IP address of the device
        iii. The protocol the packet was sent over.

Previous Desirable objectives:

1. Multiple types of DoS attacks are detected by the ML model
2. ~~Over 5 types of network topologies~~ with real-life use cases.
3. Detection can occur across various operating systems

Additional Core Objectives:

Due to some of my previous objectives being removed, I decided to add some new objectives to replace them.

1. Find differing datasets that you can use to teach the machine learning algorithm with.
2. Evaluate performance by comparing each ML Model against the different datasets.

1.2 Intended Audience:

This study focuses on combining the use of machine learning algorithms and cyber security. Therefore, this study is directed at people who are interested in either topic or particularly at people who are planning to involve themselves in research relevant to this combination.

1.3 Report Layout:

Section 1 – Introduction: This section introduces the reason for this study, the aim of this study (including the changes to the aims for the initial plan) and the people who will be interested in the study.

Section 2- Background: This section outlines and explains the necessary knowledge required to understand what is being discussed throughout the project. Furthermore, it establishes what people have already done around this topic and their results.

Section 3- Approach and Implementation: The approach focuses on the possible ways to reach the objectives and what can be implemented into the final solution. The implementation is the solution that was chosen for the project.

Section 4- Results and Evaluation: This section highlights the results of the study followed by comparing the results to prior studies mentioned in the background.

Section 5- Conclusion: The conclusion summarises the findings from the project and outlines the potential flaws and shortcomings in this study. Furthermore, this section can begin to discuss potential future works that can be built on top of this study.

Section 6- Reflection on Learning: This section presents an opportunity to reflect on the lessons learnt from carrying out the study.

2. Background

The background will consist of four main parts: A brief explanation of different types of DoS attacks with their different categories, a background on machine-learning algorithms with example use cases, how different datasets are handled by machine-learning algorithms and what research has already been done on a combination of these two topics.

2.1 Types of DDoS attacks

In most literature, DoS attacks are divided into three categories [6][7].

1. Volume Based Attacks: As the name suggests, volume-based attacks focus on flooding a network device with a large amount of network traffic. These attacks are often sent in large quantities through protocols that allow high amounts of packet frequency such as “UDP” and “ICMP”. The most common type of volume-based attack is a UDP flood attack where

the device becomes unable to receive any other packets or respond to the UDP requests. These attacks are often measured in Bps (Bits per second).

2. Protocol Specialised Attacks: Protocol attacks are types of attacks that consume server resources typically abusing a specific characteristic of protocols such as the “three-way handshake” from the TCP protocol. A TCP DoS attack attempts to open many TCP connections but not to close them. By doing this, the server is unable to open new connections with legitimate users and the hackers are able to hold the server hostage.
3. Application Layer Attacks: Similarly to the protocol attack, the application layer attack focuses on taking resources from a server by sending packets to a specific layer whilst seeming legitimate when inspected. Application layer attacks are especially effective because they require very low amounts of resources to launch but require a large number of resources to be defended against.

2.2 Types of Machine Learning Algorithms:

Machine Learning can be divided into many subcategories. The first large division that identifies the type of model is whether it is supervised or unsupervised.

Supervised machine-learning models use labelled datasets to learn to give the desired outcome. The dataset will contain a series of inputs called “features” and their correct outputs. An example of this would be input data as network traffic (IP, Protocol, Packet Size, Timestamp) and the output value labelling the record (single row in database) as malicious or benign.

The algorithm can then check its accuracy with a “loss function” letting the programmer iteratively change the algorithm until the error margin is minimised enough.

Supervised learning has two types:

Classification: Classification uses a machine-learning algorithm that must give a result within a discrete set of answers. Is something true or false? I.E is this packet part of a DDoS attack?
In classification models, there is a table that can help visualise the performance of a model called a confusion matrix.

There are four types of results you can yield from a confusion matrix:

TN: A true negative is when the True class and Predicted class both state the observation as negative.

TP: A true positive is when the True class and Predicted class both state the observation as positive.

FN: A false negative is when the true class identifies the result as positive, but the prediction class predicts a negative.

FP: A false positive is when the true class identifies the result as negative, but the prediction class predicts a positive.

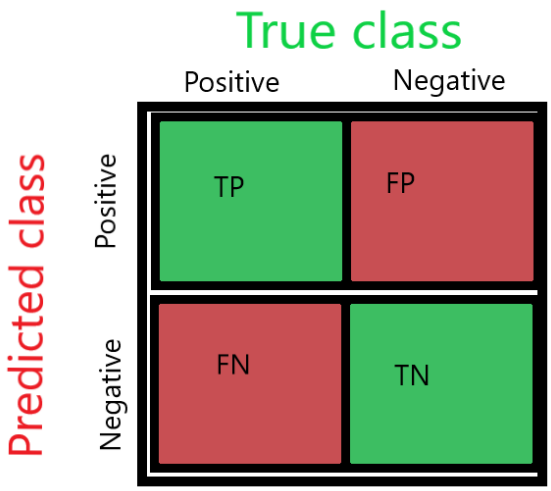


Figure 1: This is a table that shows the confusion matrix layout

Measuring Accuracy using the CM is used to predict the fraction of correct predictions over the total number of observations. If measured as a percentage, the best result is 100% and the worst is 0%. The formula for accuracy using the CM is:

$$Accuracy = \frac{\sum(TP+TN)}{\sum(TP+TN+FP+FN)} \text{ x } 100$$

Regression: Regression focuses on giving an answer that is continuous. I.E a percentage that someone may die of a disease that depends on many variables.

Unsupervised: Rather than teaching a computer through labelled datasets, the unsupervised machine learning model is given unlabelled data and time to build its own “patterns” to discover hidden similarities without human intervention. An ideal scenario of using unsupervised machine learning is for image recognition to determine whether a picture is a bee or a bus. Another common scenario is when it is used for cross-selling strategies (when a merchant attempts to sell you a memory card when you have bought a camera).

2.3 Datasets:

Datasets are built on 3 pillars:

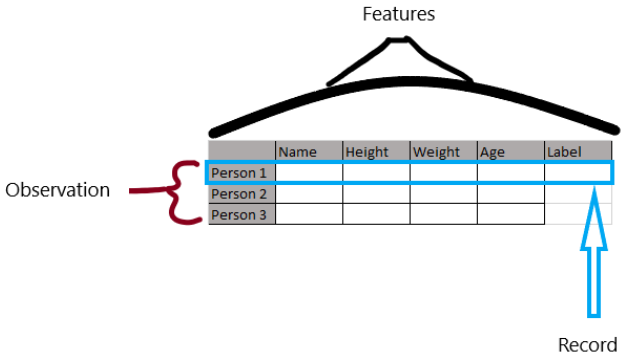


Figure 2: This visualises what a dataset would look like on an excel document. It directly outlines what features, observations labels and records are.

Features: These are the headings for the columns in the dataset. An example of a feature would be a height or weight column in a patient record. [Figure 2]

Observations: Observations are the records of a dataset. For example, a record would be an entire row of information that identifies a specific patient. [Figure 2]

Labels: Labels help navigate through a dataset by outlining the correct result the machine learning algorithm should identify the record as. For example, the label could contain the value of whether a packet (a record) is malicious or benign (the label). [Figure 2]

Low dimensional datasets: A LDD is a dataset that contains fewer features than observations. Generally, LDDs are considered to be more reliable as fewer assumptions are made when using them to teach MLMs.

High dimensional datasets: A HDD is a dataset that contains more features than observations. HDDs often appear in medical research because of the large amount of data you can take note of regarding patients[9][10]. HDDs are found to be less accurate when the models are applied to real-life use cases. The reason is MLMs can be confused by finding patterns in data where there is a correlation but not causation.

To calculate whether your dataset is a low or high dimensional dataset you must follow the formula[9]:

”if *features* p **>>** *Number of Observations* N the dataset is now **high** dimensional”

“if *features* p **<<** *Number of Observations* N the dataset is now **low** dimensional”

Typically people assume that their dataset is high dimensional due to their large number of features[9]. However, if the dataset has 20,000 features and 50,000 observations, by plugging it into the formula above:

p (features) **= 20,000**

N (observations) **= 50,000**

20,000<<50,000 therefore the dataset is a
Low dimensional dataset

2.4 Research Literature:

With the recent surge in cyber attacks in the last decade, there were many studies conducted recently to aid in the detection and prevention of DDoS attacks. Furthermore, the rise in MLMs being used in real-life use cases has given way to many studies that combine the two fields of cyber security and machine learning. Here are some examples:

The first study was a study that used a Random Forest Algorithm to perform a classification of normal and malicious attack samples[11]. The study concluded the accuracy of the MLM to be 99.76%[11]. They specifically studied the use of the "ping of death" attack, a type of protocol DDoS attack, to see if MLMs could be used as a form of detection against them. They used a pre-existing dataset called the "NSL-KDD" from the Canadian Institute for Cybersecurity[12].

Another study used the Random Forest Algorithm to specifically detect volume-based attacks by applying a two-year dataset from a recursive server (2014-2016) [13]. The study ended with the final MLM having an accuracy of 99.2% on its traffic classification.

The two studies both highlighted the reason they decided to do the study was that there has been a large increase in DDoS attacks in recent years [12][13]. The studies also highlighted that from their results, it was evident using an ML model would be a useful tool to tackle DDoS attacks [12][13].

Both the studies mentioned so far focus on only one type of attack. This theme was common across a lot of papers whilst researching this topic [12][13][14].

Instead of having a focused approach, this study capitalised on a varied dataset and created a generalised MLM that could detect various types of DDoS attacks [15]. The study found success when they reduced the number of features that were selected for analysis prior to teaching the MLM.

2.5 Research Questions:

In order to understand how different datasets change the outcome of a generalised MLM's accuracy, this study will compare and contrast the results of the MLMs' success. By analysing the different changes made to each model, this study will search for the most successful way of training MLMs. Another question it will attempt to answer is why there is a difference in performance. By evaluating the performances and understanding possible paths of increased success, this knowledge can be applied to the industry use cases of machine learning models when preventing and defending against DDoS attacks.

3. Approach

For the project to be successful, a Machine learning algorithm must be used to construct an MLM. To achieve this, the approach to this problem must follow certain criteria:

1. Find pre-existing datasets that have differing factors.
2. Find an algorithm that compliments the type of learning that the project requires.
3. Label the chosen pre-existing datasets.
4. Write a program that uses a chosen ML algorithm.
5. Teach the same algorithm with different labelled datasets to create different models.
6. Extract results of different models' performance.

To tackle the first task, a differing factor was required that could differentiate the datasets from each other. There was an opportunity to use Cardiff University's Forensics Labs to create a dataset on their cyber range. The initial approach was to design an artificial dataset using the CUFL and

compare it to a real modern-day dataset. By creating a network and simulating a DDoS attack (check supporting material VIDEO titled: “Artificial Network Part one, two and three), data could be captured and labelled appropriately for the MLM to learn from. However, there were constraints which I met early on in the construction of this dataset that limited me from creating it.

Firstly, creating a dataset that contained packets from a large network was out of the scope of this project’s allocated time. Constructing a dataset that extracted data from many machines and replicated a realistic network would require additional time to investigate and perfect.

In addition, the machines needed to perform “normal” tasks that mirrored the actions of an average user. Therefore, the artificial network would need an API that could create background noise, adding a layer of realism to the captured dataset.

Considering the factors of time constraints and artificially producing the datasets, specifically without prior experience, the choice was made to search for other factors to compare among pre-existing datasets.

To navigate through the problem of finding differing datasets, research on previous studies that performed analysis but on single datasets was conducted [12][13][14][15]. These studies pointed toward institutes that held credible datasets. The first institute was the Canadian Institute for Cybersecurity [16]. A dataset that stood out was the Intrusion Detection Evaluation Dataset (CIC-IDS2017). “CICIDS2017 dataset contains benign and the most up-to-date common attacks, which resembles the true real-world data (PCAPs).” [17]. The second institute was the Massachusetts Institute of Technology [18]. There is an old dataset that contains PCAPs of intrusion detection named “DARPA” [19].

The benefit of both these datasets is they both contain varied types of attacks rather than just one specific type of DDoS attack [17][20]. Ultimately, the most important factor that we were considering when looking for different datasets was met: The differing factor. The CICIDS2017 dataset is from 2017 and the DARPA is from 1999 meaning I could use this difference as the focus of this study. This concludes the first requirement in the criteria “Find pre-existing datasets that have differing factors.”.

To construct a machine-learning model to detect DDoS attacks, a choice must be made regarding what algorithm will be used. The initial choice was whether the algorithm should be supervised or unsupervised [21]. The main benefit of using an unsupervised model is that it does not require the data to be labelled. This is especially good when the data is hard or impossible to label by traditional methods. However, since this study attempts to find pre-existing datasets on traffic data, a supervised machine learning model was appropriate.

The datasets will be divided into benign and malicious data. The statistical binary nature of the dataset means that for the MLM it is appropriate to choose a classification algorithm [22].

There are different types of supervised ML algorithms:

Support Vector Machines: SVM are a set of supervised learning methods for classification or regression [23]. SVM is memory efficient and good to use with high dimensional datasets. However, the training times for large datasets can be long and choosing a kernel function is not easy. The outcome of the SVC heavily depends on the kernel function chosen. Finally, it is difficult to interpret the final model. For example, if you want to understand the weighting of each feature, it will require you to pick a kernel that supports that extraction.

Naïve Bayes: NB methods are supervised algorithms that apply Bayes' theorem with the assumption of conditional independence between every pair of features given the value of the label (the assumption makes it "naïve" [24]. The benefits of using NB are that it is easy to implement, and the size of the training data does not need to be large. Although these benefits are convincing to use as the overall algorithm, NB falls short when values are found in the test data when they were not present in the training data. The NB model will assign the probability value of zero and will be unable to make adequate predictions.

Decision Trees: "DT are a non-parametric supervised learning method used for classification" [25]. The model predicts values for the target variable by learning simplified decision rules observed from the features and corresponding records. DTs are very useful for visualising results, and they can be easily repeated. By repeating the test on the DTs many times, the model's reliability can be investigated. However, DTs tend to overfit the test data because the model struggles to create complex trees. The result is a model that is not generalised and requires external pruning and setting a minimum number of samples per leaf node. Overall, the decision tree model is useful for comparing different datasets but struggles with overfitting, especially when creating generalised models, meaning it is not fit to be the algorithm for this project.

Random Forest: RF is an ensemble classification algorithm that uses many decision trees simultaneously to learn a dataset [26]. Although each decision tree tends to overfit when learning from the same dataset, once you apply the use of averaging to teach many decision trees at once from different subsamples, you control the risk of overfitting. RF can handle large datasets efficiently and provide higher levels of accuracy in predicting outcomes, because of its ability to generalise models and not overfit them. On the other hand, once a model is trained, it struggles to perform real-time predictions with large amounts of data (like packet captures in Wireshark).

As mentioned previously, my investigation in the project is on the different MLMs' performances according to the dataset that was taught to them, therefore, I do not require the model to perform real-time predictions and the Random Forest algorithm is suitable for this study.

Random forest uses two methods to achieve its algorithm. The first thing that RF does is bootstrap data. Bootstrapping data is random sampling with replacement from the original dataset [FIGURE.]. By doing this you can get more than one of the same records in the dataset which is what causes overfitting on an individual DT. However, over each decision tree overfitting will occur, overall causing no overfitting to any particular record. After iterating the bootstrapped data to each DT, you aggregate the results to create one model based on the aggregation of the DTs. This process is called "bagging"[27].

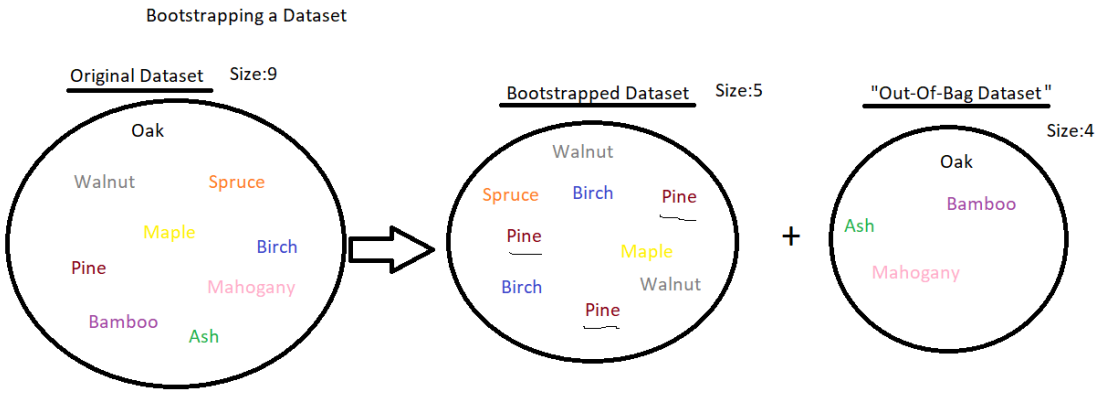


Figure 3: Demonstrates how a dataset can be bootstrapped. It also highlights how some values from the original dataset can be left out of the bootstrapped dataset (sample).

For the labelling, the datasets had web pages that outlined the "ground truth" documents. This information could be used to label the datasets using information directly found on these pages [28] [29]. Data that was used to identify the attacks against benign data would be features such as Date-Time, Source IP, Destination IP and the duration of the attack [28] [29]. The files are kept as PCAP format and because of this, I will have to extract the required data to identify the attack as well as the data I plan to train the MLM with. I will then label the data, remove the unnecessary columns and use those corrected CSV files to train the model.

The next requirement to fulfil the approaches criteria was "Write a program that uses a chosen ML algorithm". Having little experience in this topic, the project will use the help of various websites and videos to program the algorithm [26][30][31][32].

Teaching the algorithm with the different datasets can have many permutations of training and testing. To keep the variables consistent over the experiments conducted, the following order will be

Please assume DARPA is "A" and the CIC-IDS2017 is "B"

In the first model (Model A), the training will use training set "A" and have the initial test done on "A-test data". After completing the first level of training and testing, Model A will be tested against "B-test data" to see how it performs against data it has not seen before.

In the second model (Model B), the training will use training set "B" and have the initial test done on "B-test data". After completing the first level of training and testing, Model B will be tested against "A-test data" to see how it performs against data it has not seen before.

The third model will aim to combine the two datasets using a data frame structure to create another MLM (Model AB) [33]. Model "AB" will then be tested on its own dataset "AB-Test" and without training again, testing specifically on "A-test" and "B-Test".

Finally, to extract the results of all the tests I will use the basic outputs included in "sklearn-Metrics" library[34]. Metrics that would be interesting to analyse are weighting and accuracy. Accuracy was discussed previously, but weighting in RF is calculated in a particular way:

The feature importance is equivalent to what weighting is in most models. In RF using Sklearn, it is possible to display the “feature importance” for each feature in the dataset. The way it is calculated is:

1. Get the overall accuracy on all trained models.
2. Permute features iteratively on the models to see if they change the accuracy significantly of the same model.
3. Calculating the average over all the training sets will demonstrate what features have a larger influence on the accuracy which can then translate to “feature importance”.

Weighting as a maximum score of 1 and a minimum score of 0. Although, it is important to remember that weighting of all features is calculated as a sum. Therefore, all the weightings of the features should add up to 1.

4. Implementation

In the current implementation of the project, the MLM is a supervised binary classifying model that uses a random forest algorithm developed on a Python script.

4.1 Obtaining datasets

The dataset was downloaded from the websites mentioned earlier [16][17][18][19]. These datasets were in PCAP format and were transferred onto WireShark[36]. Using WireShark before I know how to open the file onto the software but there was documentation on their website that I could follow to manipulate and extract data. It was important to check whether the data was similar in format because it was vital to remove as many different factors as possible (apart from the date of the packet capture).

Both the files contained similar headings that could be used to analyse the datasets. The features that were chosen to represent the observations are:

1. Delta (time difference between the current and previous packet):
   The delta of previously captured packets is a useful feature to use because it shows if the network is being flooded with packets.
2. Protocol:
   The protocol is an important feature to have in the datasets because it allows the algorithm to analyse protocol attacks that are one of the types of DDoS attacks (See Introduction and Background for more details) [7].
3. Packet Length:
   The packet length is a measurement that has been shown to be useful when detecting specific types of DDoS attacks [37].
4. Source Port:
   Specific types of DDoS attacks happen through certain ports as shown by a UDP Flood attack that floods one port with large volumes of data [38]. Therefore, a source port is a useful feature.

5. Destination Port:
   Specific types of DDoS attacks happen through certain ports as shown by a UDP Flood attack that floods one port with large volumes of data [38]. Therefore, a destination port is an important feature.

Since the datasets had to be similar in size to remove differences other than the date recorded, the DARPA dataset was limited to only outside data. Fortunately, the majority of the DDoS attacks performed and recorded were found in the outside data (See reference to see how data is divided into "Inside" and "Outside" PCAPs) [39].

Types of attacks labelled:
To see each type of attack recorded for the DARPA dataset, access the supporting material document called "list of attacks". It will outline each type of attack, a brief description taken from the MIT website and the dates that the attacks occurred [20].

4.2 Choosing an Algorithm:

This study chose to use the Random Forest Algorithm to compare the performance of the two datasets. The benefits of using Random Forest are:

- Reduced risk of overfitting the model
- Increased performance when handling larger datasets
- Efficient with memory management

The only drawback of using RF was that it would struggle to create live predictions if the number of packets in a specific measurement of time was too large. In this study there is no need to predict live data, meaning this drawback does not affect the study.

4.3 Labelling the dataset:

Labelling the data for CIC followed a methodical approach:

1. Access the website and find the description of each attack [17]. This would include:
   a. The time of each attack.
   b. The source IP of the attack.
   c. The duration of the attack.
2. Extract the features and data described in "Obtaining Datasets" into a .CSV file.
3. Take note of the attack's information and look for it within the WireShark UI [17].
4. Find the Packet ID Number on Wireshark and search for it on the .CSV file ID column.
5. Filter by the correct Source IP
6. Label each malicious packet in a "label" column.
7. If the packet is not labelled as malicious, label it as "Benign".

Following this guide, the CIC-IDS2017 dataset was labelled quickly.

Labelling the DARPA dataset was significantly harder. When first looking to label the data, information found on the website contradicted the information found in the PCAP file read into Wireshark.

As seen from the figure below, the beginning of the PCAP file imported into Wireshark states the first packet arrived at 14:00 pm (If you inspect the "time" column) (Figure.).

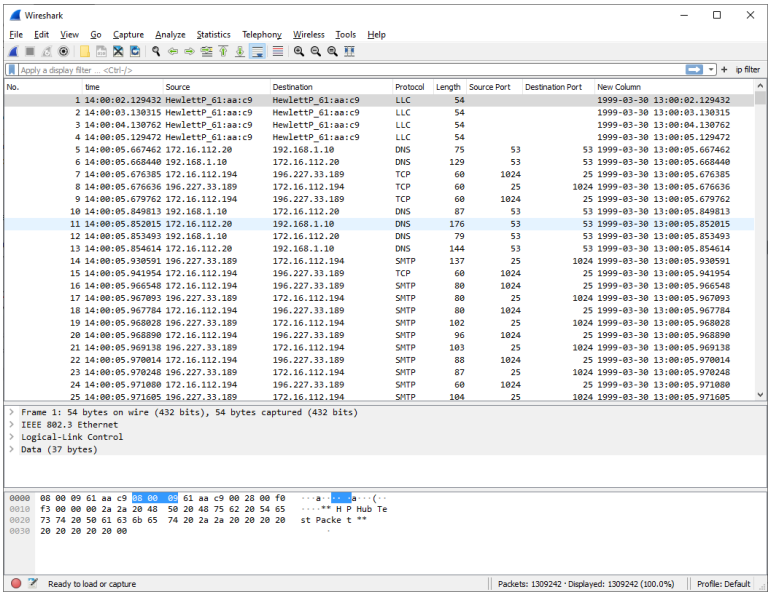


Figure 4: This screenshot shows the beginning of a PCAP file of the date 30/03/1999 (DD/MM/YYYY)

After checking on the website in the figure below, it was clear there was a mismatch of the start time from the PCAP files and the date-time information shown (Figure.).

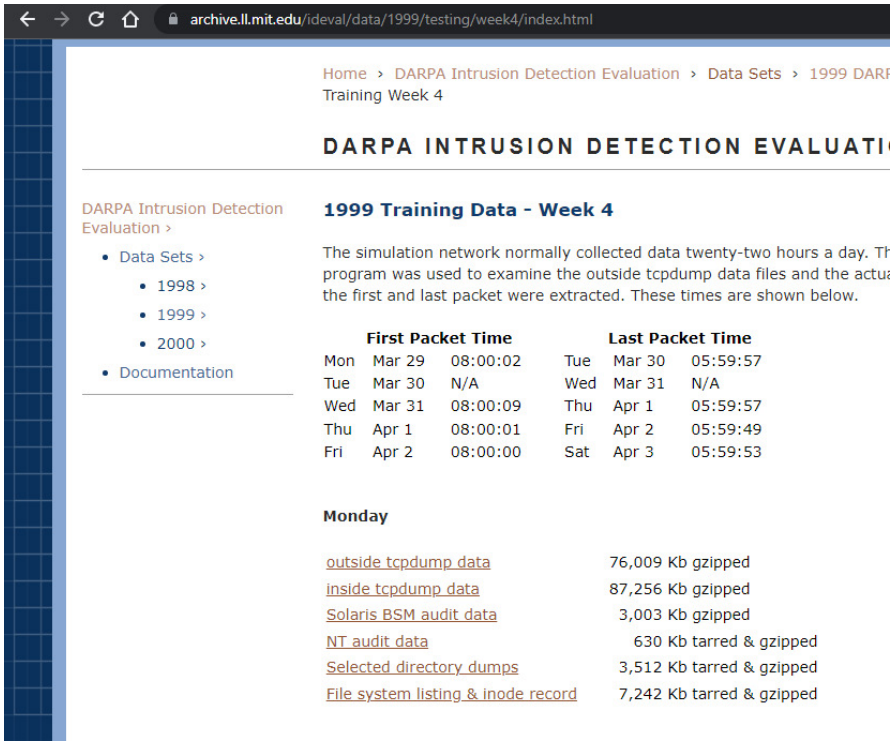


Figure 5: This shows the training data – week 4 "First packet time".

This began to cause issues because the time of the PCAPs was critical to labelling the dataset. By checking if the “ground truth” document had similar issues, I could tell if the problem’s source was on my end or if the “ground truth” was labelled badly (Figure.).

```
Summarized attack: 42.210410
 IDnum    Date       StartTime Duration Destination    Attackname insider? manual? console?success? aDump? oDump iDumpBSM? SysLogs FSlisting StealthyNew? Category OS
 42.21041003/30/1999 21:04:10  00:15:07 172.016.112.100crashiis   out      auto     rem      succ      aDmp   oDmp       not  SysLg   not       Clr      Old  llDOS

Summarized attack: 43.080401
```

Figure 6: An extract from the ground truth document that shows the time for a “Crashiis” attack

However, after inspecting the PCAP file on Wireshark to see if the time matched the document, I found the malicious packet was not there. This is also where another column was inserted at the end of the file called “New Column”. After assigning another time feature to it, it was evident time was relative on the packet’s information.

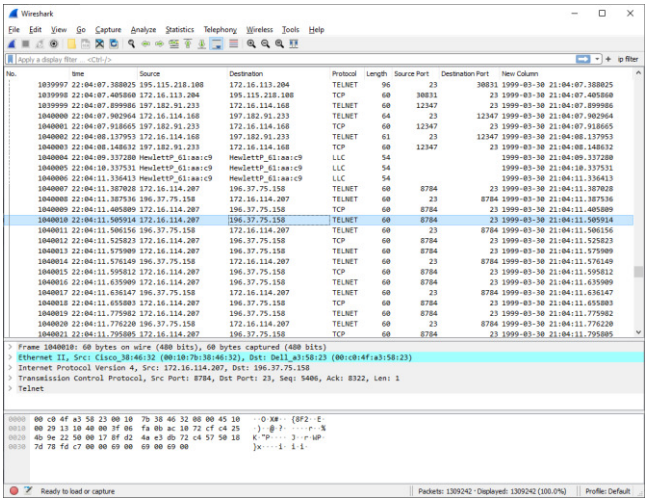


Figure 7: A screenshot of Wireshark demonstrating the packet for the attack is missing.

A quick google search showed that the difference in time could be to do with time zone differences between the MIT packet data and the current location of my device reading the file. Observing the difference in the two figures below clearly demonstrates there is a leap of five hours (Figure.) (Figure.).

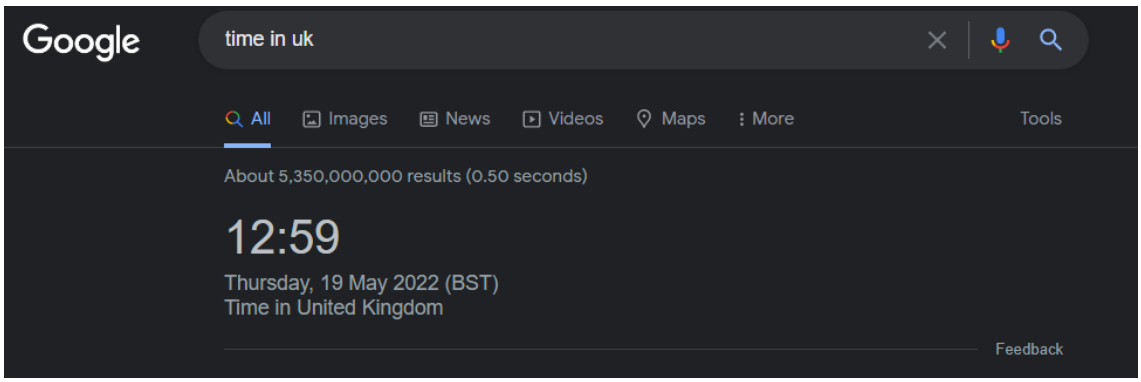


Figure 8: A Google Search of the “time in the UK”

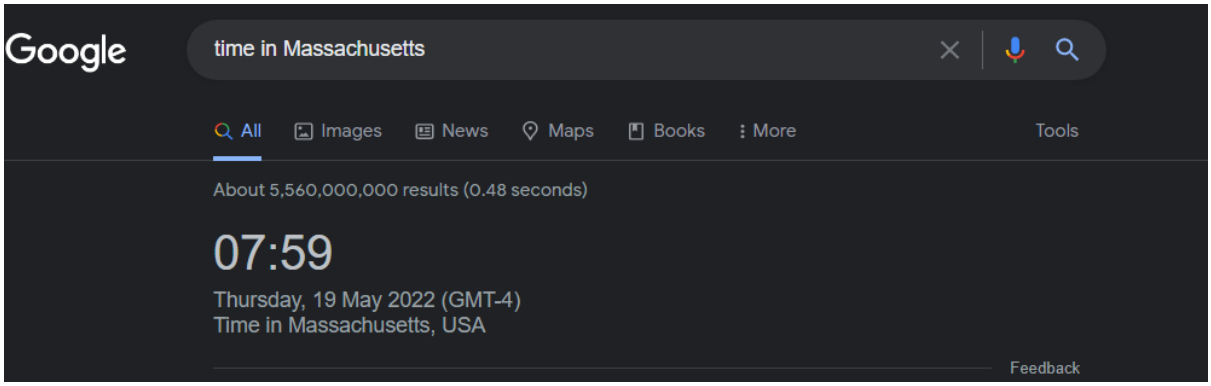


Figure 9: A Google Search of the "time in Massachusetts"

After searching for the same packet found on the "ground truth" document with the adjustment for time, the malicious data was discovered and could be labelled appropriately in the CSV file (Figure.).

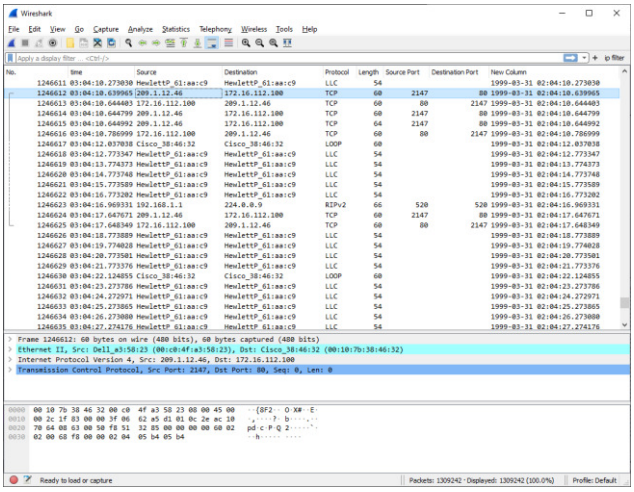


Figure 10: A screenshot from Wireshark showing the presence of the real attack at a different time is stated from the "ground truth" document.

Therefore, the methodical approach followed changed when labelling the DARPA dataset:

1. Access the document and find the description of each attack [20]. This would include:
    a. The time of each attack.
    b. The destination IP of the attack.
    c. The duration of the attack.
2. Extract the features and data described in "Obtaining Datasets" into a .CSV file.
3. Take note of the attack's information and look for it within the WireShark UI by adding five hours to the time [20].
4. Find the Packet ID Number on Wireshark and search for it on the .CSV file ID column.
5. Filter by the correct Destination IP
6. Label each malicious packet in a "label" column.
7. If the packet is not labelled as malicious, label it as "Benign".

It is important to take note that when editing CSV files in Microsoft Excel that the data cannot exceed an estimate of 1,000,000 packets. This is because the file will remove the remaining values.

Due to this, when extracting data from Wireshark to label, you must only extract one million packets at a time and have multiple CSV files to combine in the code of the ML algorithm.

4.4 Code Analysis

This section will outline important pieces of code with both text and images of the Python script.

The code below outlines the libraries used throughout the program(Figure). Pandas is used to manipulate and analyse data in Python.

```
import pandas as pd
from matplotlib import pyplot as plt
import numpy as np
```

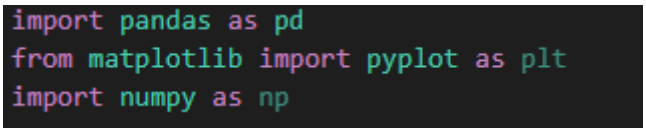


Figure 11: This shows the libraries imported.

"pd.concat" was a useful addition to the code because it makes it possible to combine CSVs into one dataframe (Figure.). This code was necessary because excel did not allow for editing of CSVs with more records than around one million [40].

```
df=pd.concat(map(pd.read_csv,[r"C:\Users\User\Documents\ThirdYear\FinalYearPro
\FridayWorkingHours1.csv",r" ……
```

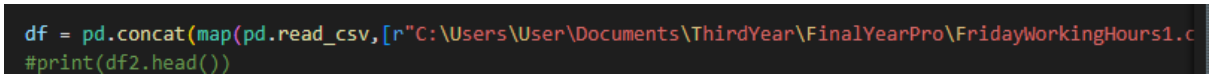


Figure 12: This shows the concatenation of CSV files into a data frame.

Dropping the "source" and "No." columns clean the data of unnecessary columns that were used to identify malicious packets when labelling. Although useful when labelling, they will not be required for teaching the MLMs.

```
df.drop(['Source'], axis=1, inplace=True)
df.drop(['No.'], axis=1, inplace=True)
```

```
#dropping irrelavent data
df.drop(['Source'], axis=1, inplace=True)

df.drop(['No.'], axis=1, inplace=True)
#print(df.head())
```

Figure 13: This line of code drops data that is not used to train the MLM.

Some machine learning algorithms can learn from qualitative data. However, Sklearn's Random Forest Classifier is unable to do so and requires workarounds. To avoid complicating the script, each unique value of a feature was assigned an ascending number as seen in the figure below (Figure.). A challenge faced was the number of protocols that were in the datasets. Using excel, I was able to extract unique lists of all the protocols and create an excel document to write the python code (Access supporting materials to see the spreadsheet titled "Protocol EXCEL").

```
#convert non numeric -> numeric
#changing labels to 0 and 1
df.label[df.label == 'BENIGN'] = 0
df.label[df.label == 'DDoS'] = 1
df.label[df.label == 'DDOS'] = 1
#changing protocols to numbers
df.Protocol[df.Protocol == 'ARP'] = 1
df.Protocol[df.Protocol == 'ASAP'] = 2
df.Protocol[df.Protocol == 'BitTorrent'] = 3
```

```
17  #convert non numeric ->  numeric
18  #changing labels to 0 and 1
19  df.label[df.label == 'BENIGN'] = 0
20  df.label[df.label == 'DDoS'] = 1
21  df.label[df.label == 'DDOS'] = 1
22  #changing protocols to numbers
23  df.Protocol[df.Protocol == 'ARP'] = 1
24  df.Protocol[df.Protocol == 'ASAP'] = 2
25  df.Protocol[df.Protocol == 'BitTorrent'] = 3
26  df.Protocol[df.Protocol == 'BJNP'] = 4
27  df.Protocol[df.Protocol =='ASAP'] =5
28  df.Protocol[df.Protocol =='Auto-RP'] =6
29  df.Protocol[df.Protocol =='BitTorrent'] =7
30  df.Protocol[df.Protocol =='BJNP'] =8
31  df.Protocol[df.Protocol =='BOOTP'] =9
```

Figure 14: This shows a snippet of code that converts values in the features from non-numerical to numerical data

In this study, the delta (time difference between packets captured" was rounded to four decimal places as seen in the figure below (Figure .). When inspecting the dataset, it was noticeable that the number of packets was so high that if rounded to fewer decimal places, the feature importance of delta is likely to go down.

```
#rounding delta to 4dp
df.Delta = round(df.Delta,4)
```

```
#rounding delta to 4dp
df.Delta = round(df.Delta,4)
```

Figure 15: This coded rounds the delta feature to four decimal places (4dp.)

This section of the code removes all values that are still null. Most algorithms struggle to handle data that is empty[41]. To mitigate this, assigning integers that are not in the subset is a good measure to take. Inspecting previous code will reveal that this was done, however, if a record was missed or something did not work, this line of code helps avoid errors(Figure .).

```
#removing Null values
df = df.dropna()
```


```
#removing Null values
df = df.dropna()
```


Figure 16: This code drops any row that contains a null value.

This is where the code splits the dataset into two sections, the training set, and the testing set (Firgure .). I chose to split it in half because this is commonly seen as a viable divide [26]. The "random_state = 5" means that it always creates the same random subsamples (Figure .). If the random state is not specified, the algorithm will create different subsamples which could potentially change the outcome of the test.

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=.5,
random_state = 5)
print("size of training")
print("X_train")
print(X_train.shape[0])
print("Y_train")
print(Y_train.shape[0])
```


```
#split data into training and testing data
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=.5, random_state = 5)
print("size of training")
print("X_train")
print(X_train.shape[0])
print("Y_train")
print(Y_train.shape[0])
```


Figure 17: This screenshot shows the splitting of the dataset into two groups: Training and Testing.

Finally the figure below shows the actual training of the model (Figure .). The “n_estimators” is a parameter that instructs the algorithm on how many decision trees to make [42]. As mentioned before, the random state is kept consistent for test conditions.

```
from sklearn.ensemble import RandomForestClassifier
# creating a model
model = RandomForestClassifier(n_estimators= 10 , random_state=5)
# fitting the model to training data
model.fit(X_train, Y_train)
#running a test to see the prediction
prediction_test = model.predict(X_test)

print("prediction_test")
print(prediction_test)
```

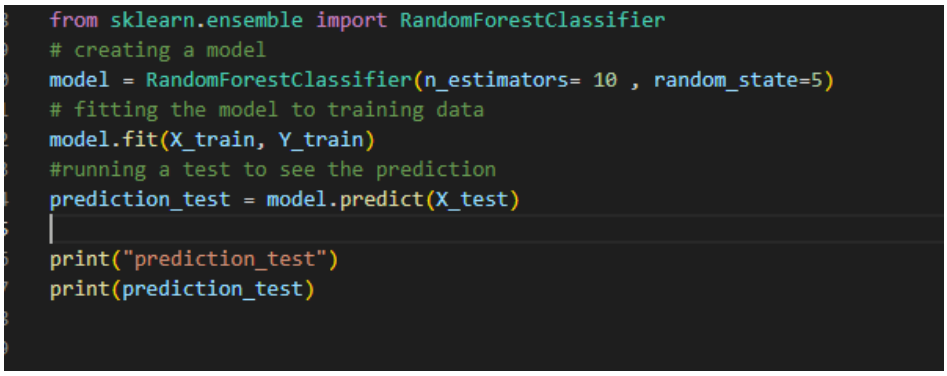


Figure 18: This part shows how the model is created using a specific random state and the model being taught with the X and Y training data.

4.5 Number of Models

For the implementation of this project, there were three models created in total:

- The first model was created using the DARPA dataset.
- The second model was created using the CIC-IDS2017 dataset.
- The third model was created using a combination of the DARPA and CIC-IDS2017 datasets.

4.6 Extracting results

The sklearn metrics library was imported to the Python script to allow accuracy, feature importance and total data sizes to be extracted (Figure .). The figure below shows accuracy being based on the test data rather than the performance on training and test data (Figure.).

```
from sklearn import metrics
print("Accuracy =" , metrics.accuracy_score(Y_test, prediction_test))
```

```
print("key features")
feature_list = list (X.columns)
feature_imp = pd.Series(model.feature_importances_, index=
feature_list).sort_values(ascending=False)
print(feature_imp)
```

```
#checking for accuracy of tested model
from sklearn import metrics
print("Accuracy =" , metrics.accuracy_score(Y_test, prediction_test))

print("key features")
feature_list = list (X.columns)
feature_imp = pd.Series(model.feature_importances_, index= feature_list).sort_values(ascending=False)
print(feature_imp)
```

Figure 19: This section of the code contains metric outputs.

5. Results and Evaluation

5.1 Results

This section will discuss the results acquired from the three different models created in this project. The models will be represented in the following way:

- The model taught by the DARPA dataset will be identified as “Model A”. The training and testing data used to teach “Model A” will be called “Train A” and “Test A”.
- The model taught by the CIC-IDS2017 dataset will be identified as “Model B”. The training and testing data used to teach “Model B” will be called “Train B” and “Test B”
- The model taught by the combination of DARPA and CIC-IDS2017 datasets will be identified as “Model AB”. The training and testing data used to teach “Model AB” will be called “Train AB” and “Test AB”

5.1.1 Model AB (CIC-IDS2017 AND DARPA)

Model AB tested on AB

The accuracy of Model AB is “99.220%” when tested against Test AB (Figure .).
The feature importance of this model in ranking and rating is distributed in the following order:

1. Source Port: Importance of rating of “0.416050”.
2. Destination Port: Importance rating of “0.304603”.
3. Delta: Importance rating of “0.149049”.
4. Packet Length: Importance rating of “0.089780”.
5. Protocol: Importance rating of “0.040518”

```
total data size
30885692
size of training
X_train
15442846
Y_train
15442846
prediction_test
[0 0 0 ... 0 0 0]
Accuracy = 0.9922060350792853
key features
SourcePort          0.416050
DestinationPort     0.304603
Delta               0.149049
Length              0.089780
Protocol            0.040518
dtype: float64
```

Figure 20: This image shows the accuracy, size and feature importance of the Model AB tested against Test AB. The model results were taught with delta rounded to four decimal places.

This Second Model AB was tested independently of other tests to measure what would change if the Delta rounding of decimal places was two instead of four.

The accuracy of Model AB is “98.950%” when tested against Test AB (Figure .).
The feature importance of this model in ranking and rating is distributed in the following order:

1. Source Port: Importance of rating of “0.511101”.
2. Destination Port: Importance rating of “0.323497”.
3. Packet Length: Importance rating of “0.094221”.
4. Protocol: Importance rating of “0.049496”
5. Delta: Importance rating of “0.021685”.

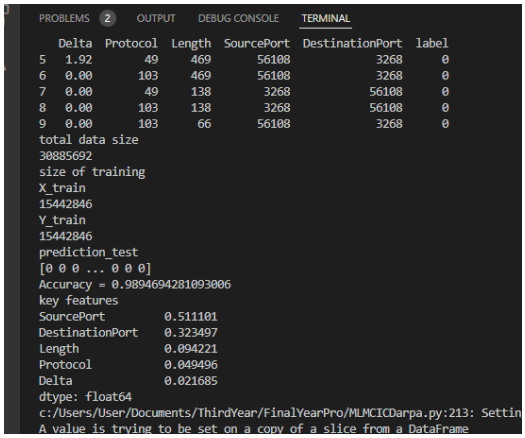


Figure 21: This image shows the accuracy, size and feature importance of the Model AB tested against Test AB. The model results were taught with delta rounded to two decimal places.

Model AB tested on DARPA Data

The accuracy of Model AB is “99.567%” when tested against Test A (Figure .).

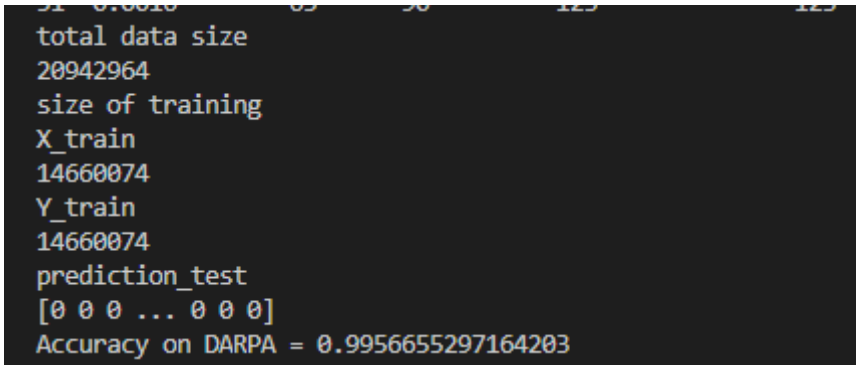


Figure 22: This image shows the accuracy, size and feature importance of the Model AB tested against Test A. The model results were taught with delta rounded to four decimal places.

Model AB tested on CIC-IDS2017 Data

The accuracy of Model AB is “99.006%” when tested against Test B (Figure .).

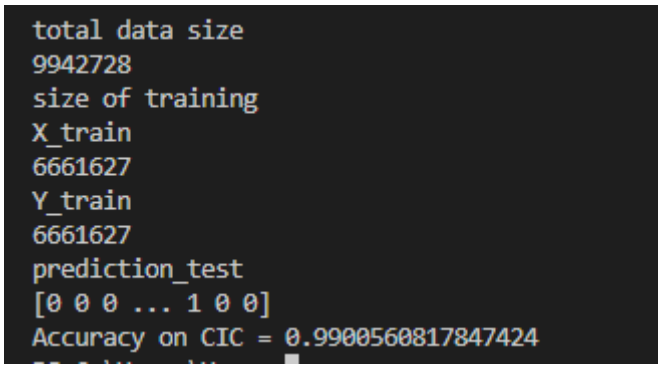


Figure 23: this image shows the accuracy, size and feature importance of the Model AB tested against Test B. The model results were taught with delta rounded to four decimal places.

5.1.2 MODEL A -DARPA

Darpa tested on Darpa

The accuracy of Model A is “99.490%” when tested against Test A (Figure .).
The feature importance of this model in ranking and rating is distributed in the following order:

1. Source Port: Importance of rating of “0.373638”.
2. Destination Port: Importance rating of “0.363212”.
3. Delta: Importance rating of “0.168702”.
4. Packet Length: Importance rating of “0.053107”.
5. Protocol: Importance rating of “0.041342”

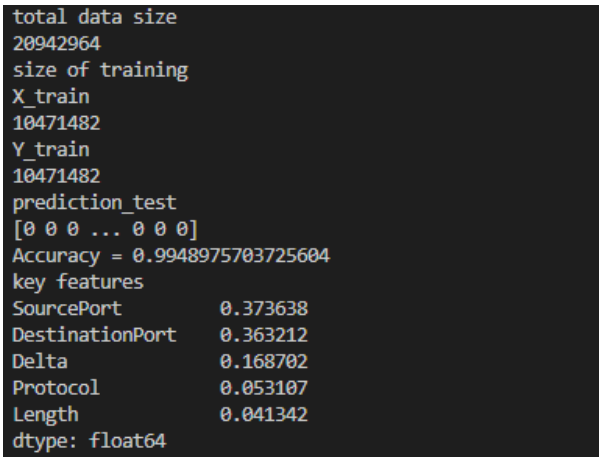


Figure 24: This image shows the accuracy, size and feature importance of the Model A tested against Test A. The model results were taught with delta rounded to four decimal places.

Darpa tested on CIC-IDS2017

The accuracy of Model A is “68.605%” when tested against Test B (Figure .).

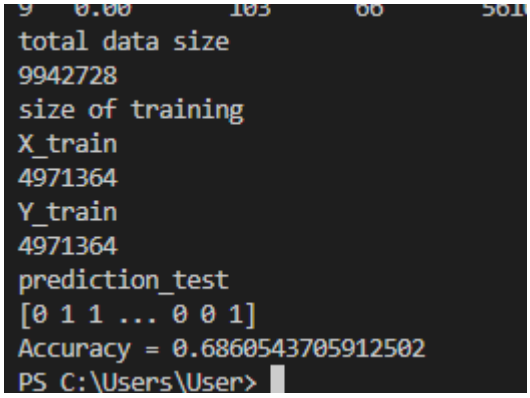


Figure 25: This image shows the accuracy, size and feature importance of the Model A tested against Test B. The model results were taught with delta rounded to four decimal places.

5.1.3 Model B - CIC-IDS2017

CIC-IDS2017 trained Model tested on CIC-IDS2017

The accuracy of Model B is “99.011%” when tested against Test B (Figure .).
The feature importance of this model in ranking and rating is distributed in the following order:

1. Source Port: Importance of rating of “0.452862”.
2. Destination Port: Importance rating of “0.268189”.
3. Delta: Importance rating of “0.125777”.
4. Packet Length: Importance rating of “0.108400”.
5. Protocol: Importance rating of “0.044771”

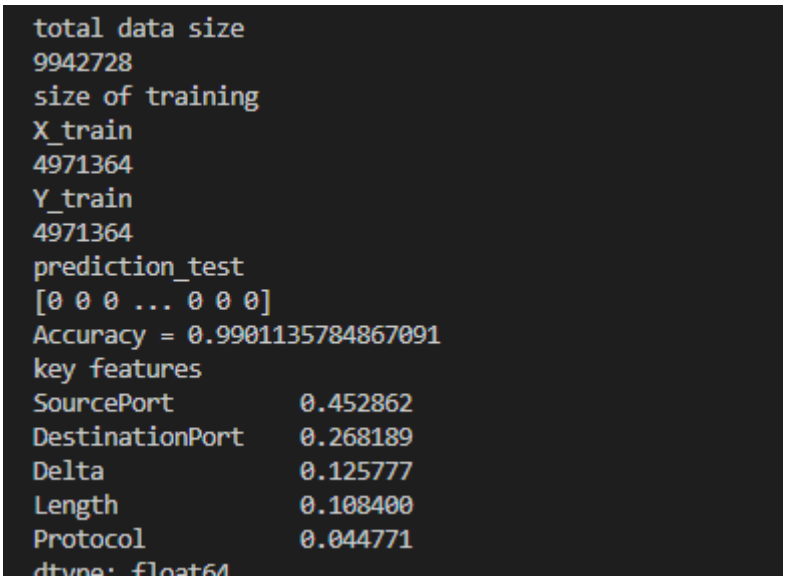


Figure 26: This image shows the accuracy, size and feature importance of the Model B tested against Test B. The model results were taught with delta rounded to four decimal places.

CIC-IDS2017 tested on DARPA

The accuracy of Model A is “68.832%” when tested against Test B (Figure .).

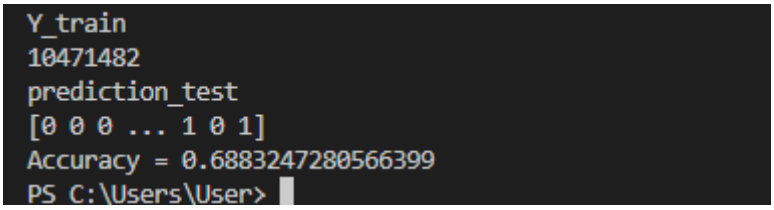


Figure 27: This image shows the accuracy, size and feature importance of the Model B tested against Test A. The model results were taught with delta rounded to four decimal places.

5.1.4 Summary of results

As seen from the results above, the trend is that there is little difference between the performances of Model A and Model B, taught by the DARPA and CIC-IDS2017 datasets respectively. They both performed over “99%” accuracy when it comes to their own data. When attempting to predict the labels of the opposing dataset, they both achieved “68%”. In both occasions the DARPA dataset Model (Model A) performed slightly better. The combined Model AB performed very well when testing on the Test AB data with “99.220%”. This study noted that Model AB performed better in the individual data tests compared to Models A and B. For example, Model A achieved “99.490%” when predicting Test A data. However, Model AB achieved “99.567%” when tested with Test A data.

5.2 Evaluation

5.2.1 Result interpretation:

The results draw 3 main points:

The first point is clearly old data is still viable for us to use to teach modern-day models. Evidence for this comes from the results that Model A was able to achieve when tested against Test B data. The accuracy for Model A was “68%” which was the same as Model B on Test A data.

Observing the combined Model AB’s results, it is evident that the performance increase is worth the trouble of combining the data. I think this means that the policy when teaching models for detection of DDoS attacks should be variety. Therefore, the more varied the model, the more likely it is to correctly predict the packet as “benign” or “malicious”.

When training Model AB, the Delta decimal place rounding was changed from four decimal places to two decimal places on one test. The model that had four decimal places performed slightly better and also saw Delta have a significant increase in feature importance. My conclusion from this is if Delta’s value is restricted to not having enough decimal places to show its true value, it will naturally have very low feature importance (as seen in Model AB’s 2 decimal place test that gave Delta a feature importance of 2%). This is due to the many packets that will come into a network of a significant size (similar to network sizes of the recorded datasets).

Study constraints:

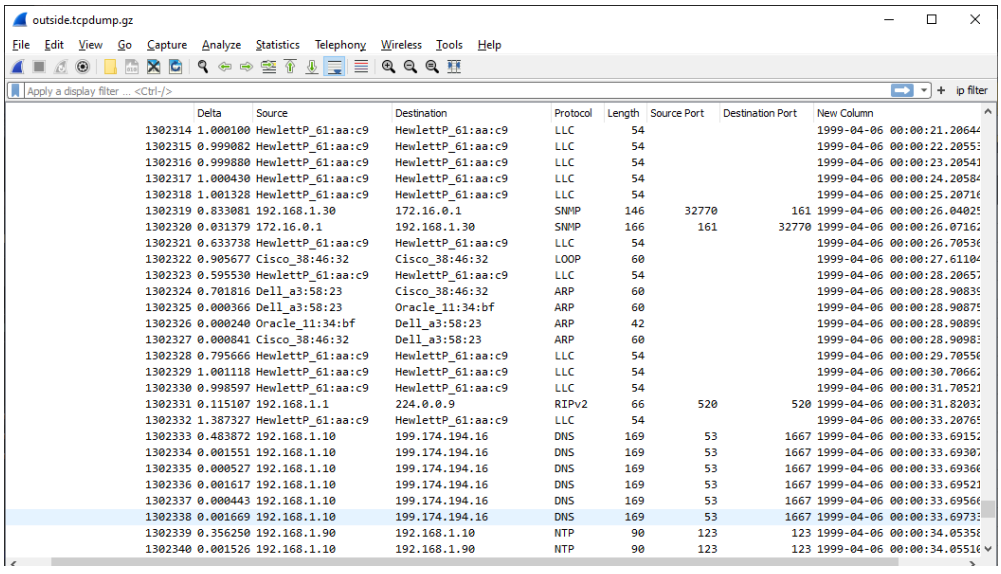
This section will cover the possible constraints that the study faced and should be factors to consider when reading the results.

To begin it is important to address that Model AB's performance on Test A and B data could be heavily affected by the training it went through. The large dataset coupled with the fact it was allowed to see data from both sources meant it was the best-equipped model. To confirm whether this is true, there should be a third dataset which is implemented. The benefit of this would not only be the ability to test the results of this study so far. But also, that the third data set would allow for many more permutations of model combinations. If we assume DARPA is "A", CIC-IDS2017 is "B" and the third dataset is "C", the list of combinations become:

- Models A, B and C
- Models AB, BC and AC
- Model ABC

Secondly, many studies conducted showing how well machine learning and cybersecurity complement each other, use artificially simulated datasets [6][11][12][15]. Therefore, even if we all tested the same dataset and concluded the same results, there should still be doubt about how valid it is without real-world data. It is important to note that there are studies that have seen success with the same methods using real-world data [5][43].

Finally, I labelled the dataset despite having no experience labelling datasets in the past. There could have been errors that I made causing the models to learn from incorrect data. In addition, there was an issue where the DARPA data was mislabelled in the ground truth document, meaning the dataset lost out on possibly useful attacks. Below I have inserted multiple figures showing how the "ground truth" document did not address the correct packet information. As seen below, the attack was not present in the "outside" PCAP file despite being written on the "ground truth" document (Figures). As seen from the figure at the end of this section, you can clearly see the attack was present in the "Inside" PCAP file which differs from what was written in the "ground truth" document (Figure .).



	Delta	Source	Destination	Protocol	Length	Source Port	Destination Port	New Column
1302314	1.000100	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:21.20644
1302315	0.999082	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:22.20553
1302316	0.999880	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:23.20541
1302317	1.000430	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:24.20584
1302318	1.001328	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:25.20716
1302319	0.833081	192.168.1.30	172.16.0.1	SNMP	146	32770	161	1999-04-06 00:00:26.04025
1302320	0.031379	172.16.0.1	192.168.1.30	SNMP	166	161	32770	1999-04-06 00:00:26.07162
1302321	0.633738	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:26.70536
1302322	0.905677	Cisco_38:46:32	Cisco_38:46:32	LOOP	60			1999-04-06 00:00:27.61104
1302323	0.595530	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:28.20657
1302324	0.701816	Dell_a3:58:23	Cisco_38:46:32	ARP	60			1999-04-06 00:00:28.90835
1302325	0.000366	Dell_a3:58:23	Oracle_11:34:bf	ARP	60			1999-04-06 00:00:28.90875
1302326	0.000240	Oracle_11:34:bf	Dell_a3:58:23	ARP	42			1999-04-06 00:00:28.90895
1302327	0.000841	Cisco_38:46:32	Dell_a3:58:23	ARP	60			1999-04-06 00:00:28.90983
1302328	0.795666	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:29.70556
1302329	1.001118	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:30.70662
1302330	0.998597	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:31.70521
1302331	0.115107	192.168.1.1	224.0.0.9	RIPv2	66	520	520	1999-04-06 00:00:31.82032
1302332	1.387327	HewlettP_61:aa:c9	HewlettP_61:aa:c9	LLC	54			1999-04-06 00:00:33.20765
1302333	0.483872	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69152
1302334	0.001551	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69307
1302335	0.000527	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69360
1302336	0.001617	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69521
1302337	0.000443	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69566
1302338	0.001669	192.168.1.10	199.174.194.16	DNS	169	53	1667	1999-04-06 00:00:33.69731
1302339	0.356250	192.168.1.90	192.168.1.10	NTP	90	123	123	1999-04-06 00:00:34.05358
1302340	0.001526	192.168.1.10	192.168.1.90	NTP	90	123	123	1999-04-06 00:00:34.05516

Figure 28: This image demonstrates the attack was not present in the "Outside" PCAP file.

```
Summarized attack: 51.200037
 IDnum    Date       StartTime Duration Destination  Attackname insider? manual? console?success? aDump? oDump iDumpBSM? SysLogs FSListing StealthyNew? Category OS
 51.20003704/05/1999 20:00:27  00:15:00 172.016.113.*udpstorm   out      auto    rem     succ     not    oDmp  iDmp not  not     not                Old  llDOS   ◆llALL
 51.20003704/05/1999 20:00:27  00:15:00 172.016.112.*udpstorm   out      auto    rem     succ     not    oDmp  iDmp not  not     not                Old  llDOS   ◆llALL

Summarized attack: 51.201715
```

Figure 29: An image of the "ground truth" document labelling the attack as in the "Outside" PCAP file.

	First Packet Time			Last Packet Time	
Mon	Apr 5	08:00:02	Tue	Apr 6	05:59:56
Tue	Apr 6	08:00:00	Wed	Apr 7	05:59:58
Wed	Apr 7	08:00:00	Thu	Apr 8	05:59:52
Thu	Apr 8	08:00:00	Fri	Apr 9	05:59:53
Fri	Apr 9	08:00:04	Sat	Apr 10	05:59:58

Monday

outside tcpdump data	122,874 Kb gzipped
inside tcpdump data	146,149 Kb gzipped
Solaris BSM audit data	6,932 Kb gzipped
NT audit data	913 Kb tarred & gzipped
NT audit data	12,259 Kb tarred & gzipped
Selected directory dumps	3,610 Kb tarred & gzipped
File system listing & inode record	8,960 Kb tarred & gzipped

Tuesday

Figure 30: A snippet of the website's timetable showing both Inside and Outside PCAP data files present.

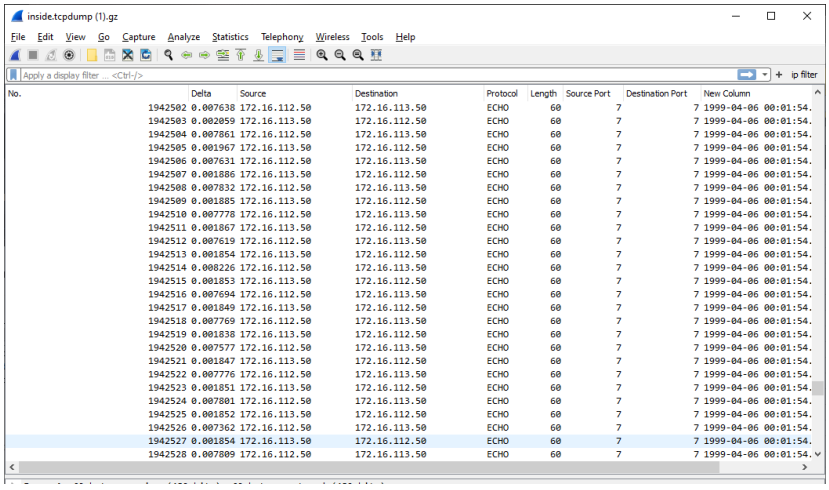


Figure 31: A screenshot from Wireshark showing that the DDoS attack is present in the "Inside" PCAP file instead of the "Outside".

6. Future Work

For future work there are many possible paths in which you could build on top of this project. There are four interesting paths which could add to this study:

First, another study could reproduce the project by following the same methodology that was conducted. This could validate the results found in this study and explore different errors that may have been skipped in this study.

Another addition which could expand the scope of this study would be achieving the original goal of creating a virtual dataset and observing how that affects the dynamic of the models' performance. As mentioned before, this would also expand the possible combinations of models that could be achieved (See evaluation section for a list of possible combinations). Note that this would require a much larger amount of time to complete as there would be a need for a "background noise" application that could make fake "realistic traffic" on the network.

Thirdly, the use of the different algorithms on the same datasets would be interesting to explore. The Random Forest algorithm was able to perform well but has some shortcomings in specific places (see results above to have more details on where it performed worse). It would be valuable to see how different models affect the results directly when using the same datasets that was used in this study. Since the data used has already been labelled, it would be the easiest addition to execute.

Finally, another path for future studies would be to replicate the study's format but find two new datasets that have different Date-Times. For example, the DARPA dataset was recorded in 1999 and the CIC-IDS2017 was recorded in 2017. If the additional study were to find two new datasets that have a similar difference in age, it would be beneficial to see how the Models interact differently or similarly to the ones in this study.

7. Conclusion

Denial of Service attacks are difficult to defend, especially if they adopt a distributed nature. DDoS attacks can be detected by machine learning algorithms and have been proven to be effective in circumnavigating them. To better equip these algorithms, it is critical that they are given the best datasets to represent real-world network traffic. In this study, using old and new datasets to teach MLMs demonstrated no difference in overall performance when put against each other. Both datasets achieved an accuracy of "99%" on their own data and "68%" on the other. Therefore, this study concludes it is advantageous to use old data to teach current-day MLMs. It is especially effective to combine both new and old datasets to provide a variety of DDoS attacks. If done similarly to this study, the models should show increased performance overall when compared to a single time period.

8. Reflection on Learning

This section will cover lessons learnt through the study. It will also discuss the transferable skills and personal developments that grew in this project.

Initially, whilst trying to make the dataset, I allocated one month on the Gantt chart I had created for the initial plan. By three weeks into the plan, it was evident that time was going to be a constraint in this project. I greatly underestimated the time it would take for me to learn the necessary skills to construct the dataset on a virtual network. The main issue was my lack of experience when it came to using the software provided at the Cardiff University Forensics Lab. A large portion of time was spent getting familiar with the software and understanding how the data would be created and captured. Despite my failure to complete this part of the project, I was able to learn how to create a dataset and what kind of data is used to teach algorithms about network activity. Furthermore, if I were to participate in a similar study, I would grasp the time required to complete objectives better.

My second failure in this project was relying on the labs to be able to provide access to the necessary application to make the virtual networks simulate attacks. Once the project was delayed further by me only being able to access and use the labs at specific times, I was left with less time to complete the project. Once I was made aware some of the software on the machines was not functioning properly, I had to submit an extenuating circumstances request. Otherwise, I would have been unable to complete the project in time. Overall, working toward finishing the project and studying in a shorter space of time taught me better time management and how to work under more pressure. These two developments will help in my future work whether it is in an office or in academia.

Throughout this project, I was able to grasp to a beginner's level how to create datasets, label and apply them to a machine learning algorithm. This may greatly benefit my work because I may participate in future studies or use this skill in a future workplace.

The last point that I learned during this study was that we must be cautious when looking at machine learning studies that use simulated or sanitised data. The issue with a lot of studies that are currently conducted that utilises this data, is the validity of their results. Although many researchers could replicate the same study, if we all conclude the same things from "realistic" but not real data, how can we know that these methods are effective at tackling problems such as DDoS attacks in real-world scenarios? It will be important for future studies to try and keep data as "fresh" as possible from their original source. This will add credibility to their results and encourage others in research to do the same.

Even if the battle against DDoS attacks is becoming easier thanks to machine learning algorithms at a rapid pace, it is vital to look to the future and find other methods that can be added to our arsenal. I strongly believe that cyber security and machine learning complement each other to the point

where the effectiveness of the combination is greater than the sum. However, I doubt this will end the battle against DDoS attacks.

9. Table of Abbreviations and Acronyms:

AI Artificial Intelligence

API Application Programming Interface

CIC-IDS2017 Canadian Institute for Cybersecurity-Intrusion Detection Evaluation Dataset 2017

CM Confusion Matrix

CSV Comma-separated values

CUFL Cardiff University's Forensics Labs

DARPA Defense Advanced Research Projects Agency

DoS Denial of Service

DDoS Distributed Denial of Service

DT Decision Tree

FN False Negative

FP False Positive

GDPR General Data Protection Regulation

GUI Graphics User Interface

HDD High Dimensional Dataset

ICMP Internet Control Message Protocol

IoT Internet of things

LDD Low Dimensional Dataset

ML Machine Learning

MLA Machine Learning Algorithm

NB Naïve Bayes

PCAP Packet Capture

RF Random Forest

SVM Support Vector Machine

SVC Support Vector Machine

TCP Transmission Control Protocol

TN True Negative

TP True Positive

UDP User Datagram Protocol

UI User Interface

10. References

[1] https://www.comparitech.com/blog/information-security/ddos-statistics-facts/#:~:text=DDoS%20attacks%20have%20been%20steadily,to%20the%20previous%20three%20months.

Cook, S., 2022. DDoS attack statistics, Facts and Trends for 2018-2022. *Comparitech*. Available at: https://www.comparitech.com/blog/information-security/ddos-statistics-facts/#:~:text=DDoS%20attacks%20have%20been%20steadily,to%20the%20previous%20three%20months. [Accessed May 27, 2022].

[2] https://www.bloomsburycollections.com/book/the-coming-swarm-ddos-actions-hacktivism-and-civil-disobedience-on-the-internet/chintro-introduction-searching-for-the-digital-street

[3] https://www-sciencedirect-com.abc.cardiff.ac.uk/science/article/pii/S0045790622000404

[4] https://ieeexplore.ieee.org/abstract/document/7842850

Sauter, M. (2014). Introduction: Searching for the digital street. *The Coming Swarm*, 1–17. https://doi.org/10.5040/9781628926705.0006

[5] https://www-sciencedirect-com.abc.cardiff.ac.uk/science/article/pii/S0045790622000404#bib0004

Cook, S. (2022). *DDoS attack statistics, Facts and Trends for 2018-2022*. Comparitech. Retrieved May 27, 2022, from https://www.comparitech.com/blog/information-security/ddos-statistics-facts/#:~:text=DDoS%20attacks%20have%20been%20steadily,to%20the%20previous%20three%20months.

[6] https://link-springer-com.abc.cardiff.ac.uk/article/10.1007/s41635-021-00119-z

Abbasi, H., Ezzati-Jivan, N., Bellaiche, M., Talhi, C., & Dagenais, M. R. (2021). The use of anomaly detection for the detection of different types of ddos attacks in cloud environment. *Journal of Hardware and Systems Security*, *5*(3-4), 208–222. https://doi.org/10.1007/s41635-021-00119-z

[7] https://www.imperva.com/learn/ddos/ddos-attacks/

DDoS attack types & mitigation methods: Imperva. Learning Center. (2021). Retrieved May 27, 2022, from https://www.imperva.com/learn/ddos/ddos-attacks/

[8] https://www.cloudflare.com/en-gb/learning/ddos/application-layer-ddos-attack/

Application layer ddos attack - cloudflare. (n.d.). Retrieved May 27, 2022, from https://www.cloudflare.com/learning/ddos/application-layer-ddos-attack/

[9] https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

The Iris dataset. scikit. (n.d.). Retrieved May 27, 2022, from https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html

[10] https://www.statology.org/high-dimensional-data/

Zach. (2021, February 10). *What is high dimensional data? (definition & examples)*. Statology. Retrieved May 27, 2022, from https://www.statology.org/high-dimensional-data/

[11] https://link-springer-com.abc.cardiff.ac.uk/chapter/10.1007/978-981-15-8469-5_5

[12] https://www.hindawi.com/journals/scn/2022/6577274/

In-text: (Liu, 2022)

Your Bibliography: Liu, W., 2022. Cognitive Computing Model Based on Machine Learning Algorithm in Artificial Intelligence Environment. *Security and Communication Networks*, 2022, pp.1-7.

[13] https://www.sciencedirect.com/science/article/pii/S1877050918311426

In-text: (Chen et al., 2018)

Your Bibliography: Chen, L., Zhang, Y., Zhao, Q., Geng, G. and Yan, Z., 2018. Detection of DNS DDoS Attacks with Random Forest Algorithm on Spark. *Procedia Computer Science*, 134, pp.310-315.

[14] https://www.sciencedirect.com/science/article/pii/S1319157820304857

In-text: (Reddy and Shyam, 2020)

Your Bibliography: Reddy, S. and Shyam, G., 2020. A machine learning based attack detection and mitigation using a secure SaaS framework. *Journal of King Saud University - Computer and Information Sciences*,.

[15] https://onlinelibrary-wiley-com.abc.cardiff.ac.uk/doi/full/10.1002/nem.2152

In-text: (Marvi, Arfeen and Uddin, 2020)

Your Bibliography: Marvi, M., Arfeen, A. and Uddin, R., 2020. A generalized machine learning-based model for the detection of DDoS attacks. *International Journal of Network Management*, 31(6).

[16] https://www.unb.ca/cic/

In-text: (Canadian Institute for Cybersecurity | UNB, 2022)

Your Bibliography: Unb.ca. 2022. *Canadian Institute for Cybersecurity | UNB*. [online] Available at: <https://www.unb.ca/cic/> [Accessed 27 May 2022].

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

In-text: (IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB, 2022)

Your Bibliography: Unb.ca. 2022. *IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB*. [online] Available at: <https://www.unb.ca/cic/datasets/ids-2017.html> [Accessed 27 May 2022].

[18] https://www.mit.edu/

In-text: (The Massachusetts Institute of Technology (MIT), 2022)

Your Bibliography: Massachusetts Institute of Technology. 2022. *The Massachusetts Institute of Technology (MIT)*. [online] Available at: <https://www.mit.edu/> [Accessed 27 May 2022].

[19] https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset

In-text: (1999 DARPA Intrusion Detection Evaluation Dataset | MIT Lincoln Laboratory, 2022)

Your Bibliography: Ll.mit.edu. 2022. *1999 DARPA Intrusion Detection Evaluation Dataset | MIT Lincoln Laboratory*. [online] Available at: <https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset> [Accessed 27 May 2022].

[20] https://archive.ll.mit.edu/ideval/docs/attackDB.html

In-text: (MIT Lincoln Laboratory: DARPA Intrusion Detection Evaluation, 2022)

Your Bibliography: Archive.ll.mit.edu. 2022. *MIT Lincoln Laboratory: DARPA Intrusion Detection Evaluation*. [online] Available at: <https://archive.ll.mit.edu/ideval/docs/attackDB.html> [Accessed 27 May 2022].

[21]https://librarysearch.cardiff.ac.uk/view/action/uresolver.do?operation=resolveService&package_service_id=11764648880002420&institutionId=2420&customerId=2415&VE=true

In-text: (Albalate and Minker, 2013)

Your Bibliography: Albalate, A. and Minker, W., 2013. *Semi-Supervised and Unervised Machine Learning*. Hoboken: Wiley-ISTE [Imprint].

[22]https://librarysearch.cardiff.ac.uk/discovery/fulldisplay?docid=alma995510243402420&context=L&vid=44WHELF_CAR:44WHELF_CAR_VU1&lang=en&search_scope=CU_Search_ALL&adaptor=Local%20Search%20Engine&tab=CSCOP_EVERYTHING&query=any,contains,An%20Introduction%20to%20Support%20Vector%20Machines%20and%20other%20kernel-based%20learning%20methods.&offset=0

In-text: (Alpaydın, 2014)

Your Bibliography: Alpaydın, E., 2014. *Introduction to machine learning*. Cambridge, Massachusetts: The MIT Press.

[23] SVM https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

In-text: (sklearn.svm.SVC, 2022)

Your Bibliography: scikit-learn. 2022. *sklearn.svm.SVC*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html> [Accessed 27 May 2022].

[24] Naïve bayes https://scikit-learn.org/stable/modules/naive_bayes.html

In-text: (1.9. Naive Bayes, 2022)

Your Bibliography: scikit-learn. 2022. *1.9. Naive Bayes*. [online] Available at: <https://scikit-learn.org/stable/modules/naive_bayes.html> [Accessed 27 May 2022].

[25] Decision Tree https://scikit-learn.org/stable/modules/tree.html

In-text: (1.10. Decision Trees, 2022)

Your Bibliography: scikit-learn. 2022. *1.10. Decision Trees*. [online] Available at: <https://scikit-learn.org/stable/modules/tree.html> [Accessed 27 May 2022].

[26] Random Forest https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

In-text: (sklearn.ensemble.RandomForestClassifier, 2022)

Your Bibliography: scikit-learn. 2022. *sklearn.ensemble.RandomForestClassifier*. [online] Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Accessed 27 May 2022].

[27] https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm Bagging

[28] https://www.unb.ca/cic/datasets/ids-2017.html CIC label

Not found

[29] https://archive.ll.mit.edu/ideval/files/master-listfile-condensed.txt

In-text: (2022)

Your Bibliography: Archive.ll.mit.edu. 2022. [online] Available at: <https://archive.ll.mit.edu/ideval/files/master-listfile-condensed.txt> [Accessed 27 May 2022].

[30] https://www.youtube.com/watch?v=YYjvkSJoui4 RF

In-text: (60 - How to use Random Forest in Python?, 2019)

Your Bibliography: 2019. *60 - How to use Random Forest in Python?*. [video] Available at: <https://www.youtube.com/watch?v=YYjvkSJoui4> [Accessed 27 May 2022].

[31] https://machinelearningmastery.com/random-forest-ensemble-in-python/

In-text: (Brownlee, 2022)

Your Bibliography: Brownlee, J., 2022. *How to Develop a Random Forest Ensemble in Python*. [online] Machine Learning Mastery. Available at: <https://machinelearningma

[32] https://www.youtube.com/watch?v=ok2s1vV9XW0

In-text: (Machine Learning Tutorial Python - 11 Random Forest, 2018)

Your Bibliography: 2018. *Machine Learning Tutorial Python - 11 Random Forest*. [image] Available at: <https://www.youtube.com/watch?v=ok2s1vV9XW0> [Accessed 27 May 2022].

[33] https://stackoverflow.com/questions/20906474/import-multiple-csv-files-into-pandas-and-concatenate-into-one-dataframe

In-text: (DataFrame and Singh, 2022)

Your Bibliography: DataFrame, I. and Singh, G., 2022. *Import multiple csv files into pandas and concatenate into one DataFrame*. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/20906474/import-multiple-csv-files-into-pandas-and-concatenate-into-one-dataframe> [Accessed 27 May 2022].

[34] https://scikit-learn.org/stable/modules/model_evaluation.html

In-text: (3.3. Metrics and scoring: quantifying the quality of predictions, 2022)

Your Bibliography: scikit-learn. 2022. *3.3. Metrics and scoring: quantifying the quality of predictions*. [online] Available at: <https://scikit-learn.org/stable/modules/model_evaluation.html> [Accessed 27 May 2022].

[35]https://librarysearch.cardiff.ac.uk/discovery/fulldisplay?docid=alma9911455176102420&context=L&vid=44WHELF_CAR:44WHELF_CAR_VU1&lang=en&search_scope=CU_Search_ALL&adaptor=Local%20Search%20Engine&tab=CSCOP_EVERYTHING&query=any,contains,Introduction%20to%20Machine%20Learning&offset=0

In-text: (Alpaydın, 2014)

Your Bibliography: Alpaydın, E., 2014. *Introduction to machine learning*. Cambridge, Massachusetts: The MIT Press.

[36]https://www.wireshark.org/

In-text: (Wireshark · Go Deep., 2022)

Your Bibliography: Wireshark.org. 2022. *Wireshark · Go Deep.*. [online] Available at: <https://www.wireshark.org/> [Accessed 27 May 2022].

[37] https://www.hindawi.com/journals/scn/2017/3691629/

In-text: (Zhou, Liao, Yuan and Zhang, 2017)

Your Bibliography: Zhou, L., Liao, M., Yuan, C. and Zhang, H., 2017. Low-Rate DDoS Attack Detection Using Expectation of Packet Size. *Security and Communication Networks*, 2017, pp.1-14.

[38] https://www.cloudflare.com/en-gb/learning/ddos/udp-flood-ddos-attack/

In-text: (UDP flood attack, 2022)

Your Bibliography: 2022. *UDP flood attack*. [online] Available at: <https://www.cloudflare.com/en-gb/learning/ddos/udp-flood-ddos-attack/> [Accessed 27 May 2022].

[39] https://archive.ll.mit.edu/ideval/data/1999/testing/week4/index.html

In-text: (MIT Lincoln Laboratory: Communication Systems and Cyber Security: Cyber Systems and Technology: DARPA Intrusion Detection Evaluation, 2022)

Your Bibliography: Archive.ll.mit.edu. 2022. *MIT Lincoln Laboratory: Communication Systems and Cyber Security: Cyber Systems and Technology: DARPA Intrusion Detection Evaluation*. [online] Available at: <https://archive.ll.mit.edu/ideval/data/1999/testing/week4/index.html> [Accessed 27 May 2022].

[40] https://support.microsoft.com/en-us/office/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3

In-text: (Excel specifications and limits, 2022)

Your Bibliography: Support.microsoft.com. 2022. *Excel specifications and limits*. [online] Available at: <https://support.microsoft.com/en-us/office/excel-specifications-and-limits-1672b34d-7043-467e-8e27-269d656771c3> [Accessed 27 May 2022].

[41] https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e

In-text: (7 Ways to Handle Missing Values in Machine Learning, 2022)

Your Bibliography: Medium. 2022. *7 Ways to Handle Missing Values in Machine Learning*. [online] Available at: <https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e> [Accessed 27 May 2022].

[42]https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/#:~:text=n_estimators%20%3A,but%20makes%20your%20code%20slower.

In-text: (Random Forest Parameter Tuning | Tuning Random Forest, 2022)

Your Bibliography: Analytics Vidhya. 2022. *Random Forest Parameter Tuning | Tuning Random Forest*. [online] Available at: <https://www.analyticsvidhya.com/blog/2015/06/tuning-random-forest-model/#:~:text=n_estimators%20%3A,but%20makes%20your%20code%20slower.> [Accessed 27 May 2022].

[43]https://www.mdpi.com/2079-9292/11/4/602

In-text: (Saghezchi et al., 2022)

Your Bibliography: Saghezchi, F., Mantas, G., Violas, M., de Oliveira Duarte, A. and Rodriguez, J., 2022. Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs. *Electronics*, 11(4), p.602.