Anomaly Detection in Networks Using Machine Learning

Umar Zaib

A report submitted for the Course of Machine Learning

Supervisor: Dr. Arshad
School of Computing Sciences
University of PAF-IAST

January 2024

Abstract

Everyday millions of people and hundreds of thousands of institutions communicate with each other over the Internet. In the past two decades, while the number of people using the Internet has increased very fast. Parallel to these developments, the number of attacks made on the Internet is increasing day by day. Although signature-based methods are used to prevent these attacks, they are abortive against zero-day attacks. On the other hand, the Anomaly-based approach is an alternative solution to the network attacks and can detect zero-day attacks as well. In this study, it is aimed to detect network anomaly using machine learning methods. In this context, the CICIDS2017 has been used as dataset because of its up-to datedness, and wide attack diversity. On this dataset, feature selection was made by using the Random Forest Regressor algorithm. Seven different machine learning algorithms have been used in the application step and achieved high performance. Machine learning algorithms and success rates are as follows: Naive Bayes 86%, QDA 86%, Random Forest 94%, ID3 95%, AdaBoost 94%, MLP 83%, and K Nearest Neighbors 97%

1 Introduction

1.1 Motivation

Everyday millions of people and hundreds of thousands of institutions communicate with each other over the Internet. In the past two decades, while the number of people using the Internet has increased very fast, today this number has exceeded 4 billion and this increase is continuing rapidly [1, 2].

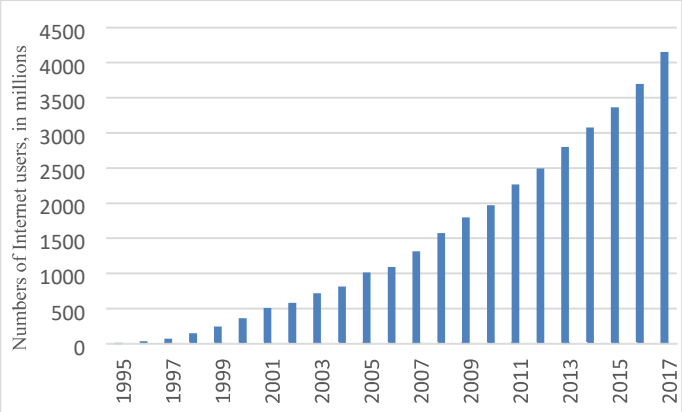


Figure 1. Increase in The Number of the Internet Users According to Years (1995-2017) [1, 2].

Parallel to these developments, the number of attacks made on the Internet is increasing day by day. Against these attacks, there are two basic methods used to detect the attacks to ensure information security; identification based on signature, and detection based on anomaly.

In this project, it was aimed to develop a system that detects network anomaly quickly and effectively by means of machine learning methods.

1.2 Goals and Objectives

1.2.1 **Goals** The goals that are aimed to achieve at the end of this project are as follows:

- Examination of machine learning algorithms that can be used to detect network anomalies.
- To detect network attacks in a fast and effective way by studying network anomaly with machine learning methods.
- To determine the success level of the study by comparing the results obtained in it with the studies previously conducted in this area.
- Contributing to the literature by obtaining close results from previous studies in the detection of network anomalies.

1.2.2 Objectives

The objectives that are aimed to achieve at the end of this study are as follows:

- To examine the previous work done in the field by doing extensive field research.
- Selecting the appropriate dataset by performing comprehensive research on the alternatives to the dataset.
- Choosing suitable algorithms by conducting extensive research on machine learning algorithms.

- Deciding on right algorithms by performing exhaustive research on machine learning methods.
- Selecting the appropriate software platform.
- Choosing the suitable hardware/equipment platform.
- Deciding on the right evaluation criteria.
- Choosing the benchmark studies to be compared during the evaluation phase.

2 Background and Related Work

2.1 Datasets

In the detection of network anomaly by machine learning methods, there is a need for a large amount of harmful and harmless network traffic for training and testing steps. However, it is not possible for real network traffic to be used publicly because of privacy issues. To meet this

CICIDS 2017

Finally, dataset to be addressed is the CICIDS 2017 (Intrusion Detection Evaluation Dataset) created by the Canadian Institute for Cybersecurity at the University of New Brunswick. This dataset consists of a 5-day (3rd July- 7th July 2017) data stream on a network created by computers using up-to-date operating systems such as Windows Vista / 7 / 8.1 / 10, Mac, Ubuntu 12/16 and Kali. Details of the dataset can be seen from Table 1[4, 7].

Flow Recording Day (Working Hours)	**pcap File size**	**Duration**	**CSV File Size**	**Attack Name**	**Flow Count**
Monday	10 GB	All Day	257 MB	No Attack	529918
Tuesday	10 GB	All Day	166 MB	FTP-Patator, SSH-Patator	445909
Wednesday	12 GB	All Day	272 MB	DoS Hulk, DoS GoldenEye, DoS slowloris, DoS Slowhttptest, Heartbleed	692703
Thursday	7.7GB	Morning	87.7 MB	Web Attacks (Brute Force, XSS, Sql Injection)	170366
		Afternoon	103 MB	Infiltration	288602
Friday	8.2GB	Morning	71.8 MB	Bot	192033
		Afternoon	92.7 MB	DDoS	225745
		Afternoon	97.1 MB	PortScan	286467

Table 1. Details of CICIDS2017 Dataset

2.3 Attacks

In this section, the types of attacks that the data set contains are examined in detail.

All data in the dataset is tagged with 15 labels. One (Benign) of these tags represents normal network movements while the other 14 represent attacks. The benign record formed using Mail services, SSH, FTP, HTTP, and HTTPS protocols represent a non-harmful / normal data stream on the network, created by simulating real user data.

2.3.1 **DoS HULK**

HULK (HTTP (Hypertext Transfer Protocol) Unbearable Load King) can be used to perform both DoS and DDoS attacks. Because this attack influences the server load first-hand, it can disable the server in a very short time, like a minute or so. HULK has the capability to conceal the real user agent and use a different template for each attack request. During the attack, it creates TCP-SYN (Transmission Control Protocol - Synchronization) floods and multiple HTTP-GET flood requests. To better understand this attack, TCP-SYN flood and HTTP-GET flood methods can be examined.

TCP-SYN Flood

The TCP-SYN flood attack is intended to exploit the 3-way handshake method to consume server resources and render the server out of service. The 3-way handshake method involves the following steps:

- The client that wants to connect to the server sends a SYN packet to the server to start the communication.
- The server receiving the SYN packet sends the corresponding SYN-ACK packet.
- Finally, the client initiates communication by sending an ACK packet to the server.

In the TCP-SYN flood attack, the first two steps occur; the attacker (client) sends the SYN packet to the victim (server), and the server responds with the SYN-ACK packet in response. However, the attacker does not send the ACK packet that should be sent, and the connection waits as incomplete. The server reserves a resource by keeping these requests in a log queue. When the number of requests increases, the server becomes inaccessible.

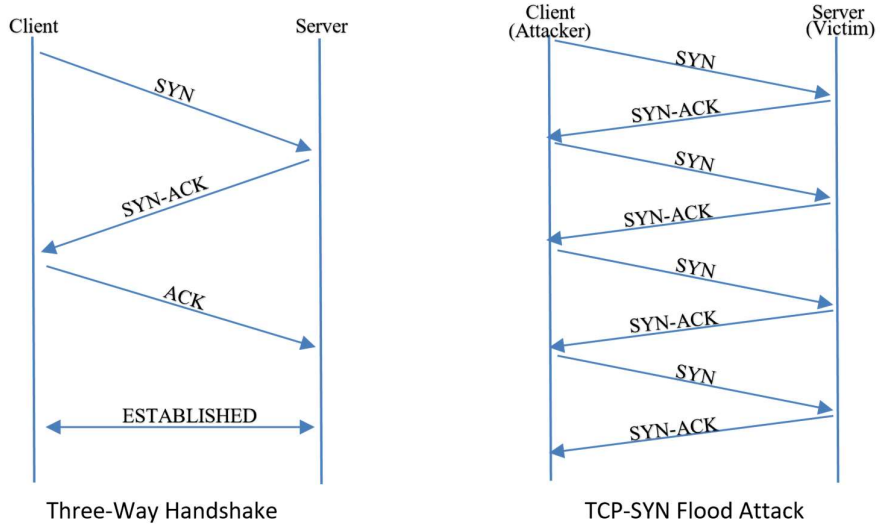


Figure 4. A comparison of TCP-SYN Flood Attack with a successful Three-Way Handshake.

HTTP-GET Flood

HTTP GET is the command used by the client to request a file from a web server. The purpose of an HTTP GET flood attack is to reduce the number of HTTP connections that the web server can keep open by consuming server resources, rendering the server unresponsive to legitimate HTTP GET requests. In this type of attack, the attacker sends too many HTTP GET requests to

the target server. The attacker can send an HTTP GET request on each TCP connection or can send multiple HTTP GET requests for each TCP connection using the HTTP feature. Moreover, these requests do not require high network traffic.

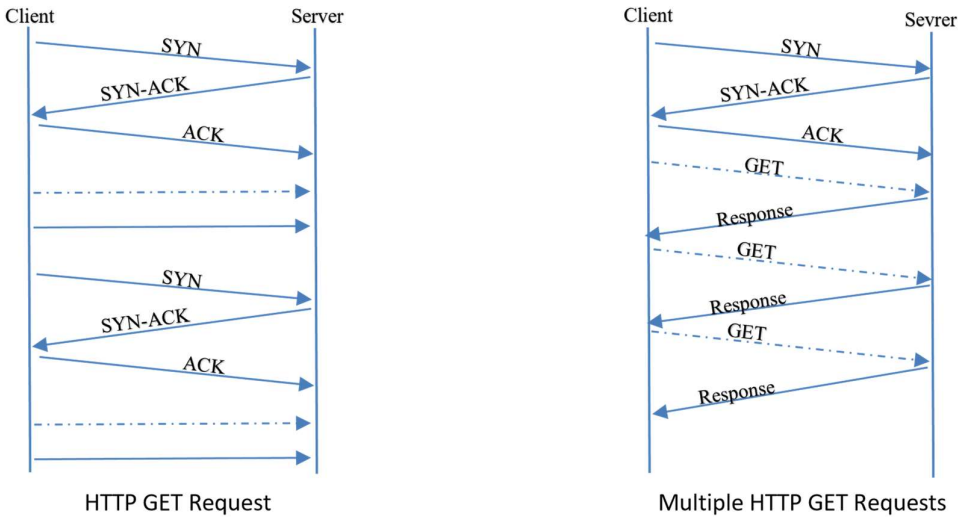


Figure 5. The comparison of singular and multiple HTTP GET requests in HTTP GET Flood attack [17]

2.3.2 **PortScan**

The PortScan attack in the CICIDS2017 dataset is made by the program named Nmap and is located on Friday afternoon section. Ports are abstract connection points named with numbers between 0 and 65535 and are used to determine what type of service you want to receive from the reached computer. Port Scan is a popular attack type used to gather information about how attackers can get into the system. With this attack, the attacker can learn much critical information about the connected devices, such as operating system, running services, and port status. In a PortScan attack, the attacker sends a message to each port. The type of response received from the victim gives information about whether the port is used or not. By using this information, the attacker learns about the used system and detects its weaknesses. Port Scanning types can be listed as follows:

DDoS

The DDoS attack, that sent HTTP, TCP and UDP requests. is located in Friday-afternoon section. A DDoS (Distributed Denial-of-Service) attack is another type of attack designed to prevent a legitimate user from accessing a website or web service, such as a DoS attack. Unlike the DoS attack, the DDoS attack is not made from a central computer but is made from many computers to a single source.

Hundreds of proxy computers can be used to attack a computer. These computers used were infected by malicious software so that they could be used remotely during the DDoS attack. These devices are called "zombies", and the network created by these devices is called "botnet".

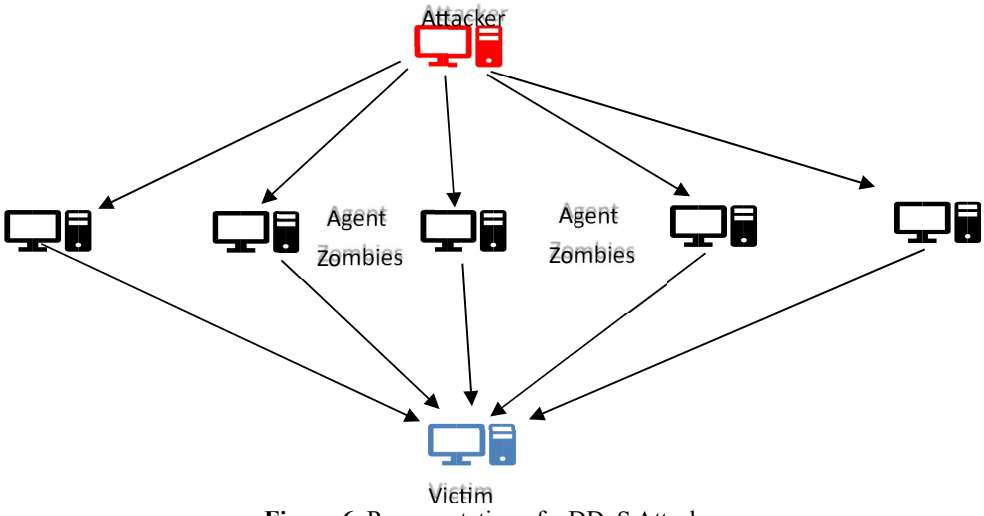


Figure 6. Representation of a DDoS Attack

FTP-Patator

FTP-Patator attack is made using the Patator, a multithreaded tool written in the Python program. This attack takes place on Tuesday-morning-records within the CICIDS2017 dataset. FTP (File Transfer Protocol) is a network protocol that provides file transfer between client and server in a network. To transfer a file using FTP, a valid username and password are required on the network to be sent. The FTP-Patator attack is a brute-force attack aimed at seizing this username and password.

Brute-force is a cryptographic attack designed to obtain passwords. In this attack, the attacker tries to log on to the system as a legitimate user by trying possible usernames and passwords on the system. Software that automatically generates passwords is often used for this process.

A large number of unsuccessful entry attempts over a short period of time is a characteristic feature for brute force attacks. In this context, it is possible to see dense packet flow during brute-force attack. In addition, failed entries do not contain high sizes files, so bandwidth consumption and bytes count are low.

SSH-Patator

SSH-Patator attack is made using the Patator, a multithreaded tool written in the Python program. This attack takes place on Tuesday-afternoon-records within the CICIDS2017 dataset.

SSH (Secure Shell) is a cryptographic protocol that allows secure operation of various network services over a network in an unsecured environment (e.g. The Internet). The most common use of this protocol is to log on to a remote system.

In this attack, the intent of the attacker is to gain remote access to a system and gain full control over it. This attack consists of three steps:

Scanning Step: This is the first phase of the SSH-Patator attack. The goal in this phase is to learn about the system to be attacked (See 2.2.2 PortScan). The attacker tries to find the host

running SSH by performing a scan on a specific port number for IP blocks on a network or subnets.
Brute-Force Step: At this stage, the attacker tries to log in to the system he has discovered during the scanning phase with a large number of username and password combinations.

Die-off Step: The attacker gains the legitimate user authority by successfully logging in.

While during the first step of this attack, a large number of incomplete TCP packets with SYN flags are observed, in the second step (brute-force), small sized completed TCP packets are seen. The number of packets in the stream is high, but the packet sizes are low.

Botnet
Botnet attack is made using Ares, an attack tool written in the Python program. This attack takes place on Friday-morning-records within the CICIDS2017 dataset.

Botnet is a network created by malware-infected computers. These computers that make up botnets are called bots or zombies. Because of the malicious software, these computers perform various harmful activities without the knowledge of their owners. These computers can be remotely controlled by bot-masters and used to make SPAM (unsolicited emails) or DDoS attacks.

The detection of the bots' network behavior is only possible in the active state (during the attack). When the botnet is passive, there is no network activity and cannot be detected. In the active state, the number of bytes per packet and the number of packets with PSH flag make it possible to detect this attack.

Web Attack
Web Attack takes place on Thursday-afternoon -records in the CICIDS2017 dataset. In this group of attacks, three different web-based attacks were launched against a vulnerable PHP (Hypertext Pre-processor - a script programming language)/ MySQL (an open-source database administration framework) web application identified as the victim. These attacks are:

XSS: XSS (Cross-Site Scripting), a popular web-based attack type, is implemented by injecting code into a web page. When the victim wants to view this web page, this malicious code fragment can lead to undesirable results such as data theft, session seizing, special code execution, and fraud.

SQL Injection: SQL (Structured Query Language – a database management system) Injection attack is one of the most popular and most dangerous attacks on the web. A typical dynamic web page keeps the user's various information in the database for later use and occasionally makes various queries in the database to reach this data.

These attacks are usually done by exploiting a security vulnerability in the database layer of a web application. The attacker can access the database using the exploits of the connection between the web application and the database. Then he could be able to steal, delete, or modify the data by sending malicious commands to the database server.

Heartbleed

This attack is made using the Heartleech, a Heartbleed exploit tool written in C programming language. This attack takes place on Wednesday - afternoon -records within the CICIDS2017 dataset. The normal operation of heartbeat is as follows: The client transmits to the server a request consisting of random payload. the server replies to this request with a packet containing the same payload.

During the attack, a specially created heartbeat request is sent to the server. This demand is empty but points out that it carries very high amounts of data. At this stage, due to a vulnerability in OpenSSL, the server sends a piece of memory in the specified length in response to this message. These parts contain various confidential information such as personal information, usernames and passwords that should not be sent. The size of these packages can be up to 16 KB and the attacker can repeat this operation without any limit.

During this attack, some anomalies on the network flow are observed. The length of the incoming messages is less than the heartbeat minimum message size of 20 bytes. Outgoing message lengths are in the kilobyte level, and for a heartbeat response, this size is too big. The difference in size between outgoing and incoming messages is another way to understand the attack. In normal heartbeat messages, the lengths of the outgoing and incoming messages are the same. In case of attack, incoming message lengths are too small, outgoing message lengths are too large.

3.2 **Implementation**

In this section, various pre-processing and actual applications are performed to detect anomaly by machine learning techniques. For this purpose, the data cleansing process is performed in the first step and the dataset is cleaned from mistakes and defects. Then, the data set is divided into two parts, training, and test. After these operations, the properties to be used by the algorithms are decided at the step of feature selection. Finally, the section ends with the implementation of machine learning algorithms. Figure 12 illustrates the implementation process in detail.

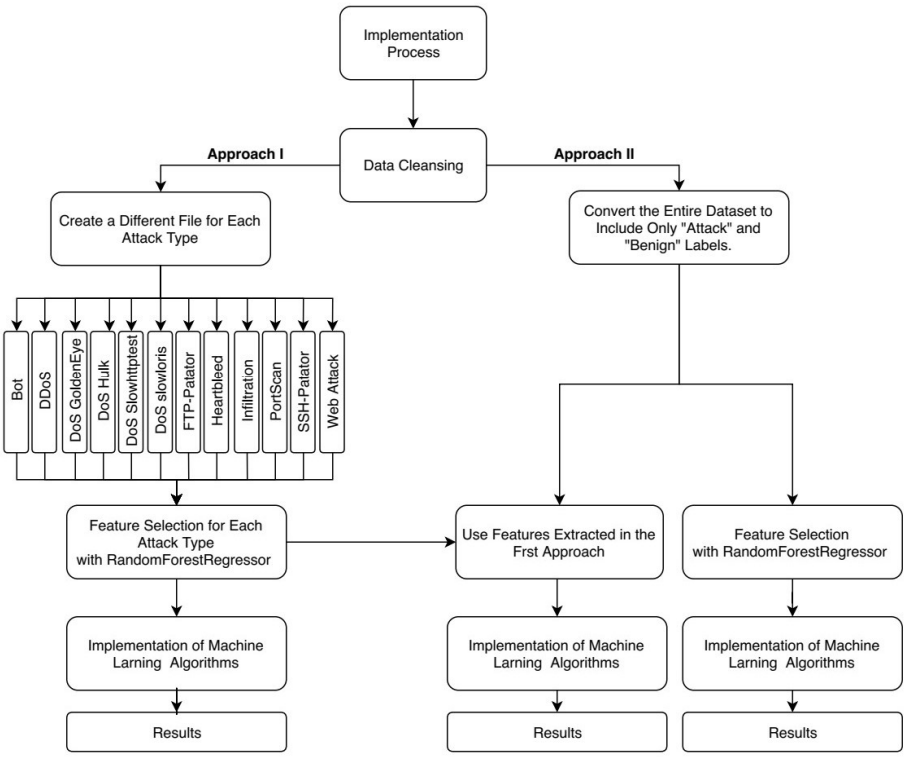


Figure 12. The Implementation Process

Data Cleansing

It may be necessary to make some changes to the dataset before using it in practice, making it more efficient. For this purpose, in this section, some defects of the CICIDS2017 dataset are corrected, and some data are edited.

The dataset file contains 3119345 stream records. The distribution of these stream records can be seen from Table 2. When these records are examined, the 288602 record is incorrect / incomplete. The first step in the pre-processing process will be to delete these unnecessary records.

Label Name	Number
Benign	2359289
Faulty	288602
DoS Hulk	231073
PortScan	158930
DDoS	41835
DoS GoldenEye	10293
FTP-Patator	7938
SSH-Patator	5897
DoS slowloris	5796
DoS Slowhttptest	5499
Bot	1966
Web Attack – Brute Force	1507
Web Attack – XSS	652
Infiltration	36
Web Attack – SQL Injection	21
Heartbleed	11

Table 2. Distribution of stream records in the CICIDS2017 dataset.

Another error about the dataset is in the columns that make up the features. The dataset file consists of 86 columns that define the flow properties such as Flow ID, Source IP, Source Port etc. However, the `Fwd Header Length` feature (which defines the forward direction data flow for total bytes used) was written two times (41st and 62nd columns). This error is corrected by deleting the repeating column (column 62).

However, although the "Label" tag is a categorical feature, no changes have been made on it. The reason is that during the processing, the original categories are needed in order to classify the attack types in different forms and to try different approaches.

3.2.2 **Creation of Training and Test Data**

During the machine learning process, data is needed so that learning can take place. The data sets used are the result of this need. In addition to the data required for training, test data is needed to evaluate the performance of the algorithm and to see how well it works. The algorithm acquires a skill on the training data and applies it to the test data. The result of the test data is the performance of the machine learning algorithm [40].

However, the CICIDS2017 dataset used does not contain dedicated training and test data, but it contains a single unbundled dataset. Therefore, the data should be divided into training and test data parts. In the application phase, a Sklearn command, train_test_split[59] is used. This command divides the data into 2 parts at the sizes specified by the user. Generally preferred partitioning is 20% test, 80% training data [40] and this ratio is also preferred in this application. The train_test_split command makes the selection random when creating data groups. This process is known as cross-validation. In order to ensure that the results obtained

during the application are solid, the creation of the training and the test data have been performed 10 times in succession. The results obtained are the arithmetic mean of the repeated operations.

3.2.3 **Feature Selection**

In this section, the features in the dataset are evaluated to determine which features are important to define which attack.

3.2.3.1 Feature Selection According to Attack Types

To do this calculation, a special file is created for every kind of attack by isolating the attack from other attacks. This file contains the entire stream identified as the attack and the data stream identified as randomly selected "Benign" (30% Attack, 70% Benign).

The Random Forest Regressor class of Sklearn is used when importance weights of features are calculated. This algorithm creates a decision-forest. In this decision forest, each feature is given a weight of importance as to how useful they are in the construction of the decision-tree. When the process is finished, these importance weights of features are compared and sorted. The sum of the importance weights of all the properties gives the total importance weight of the decision tree. The comparison of the score of any feature to the score of the whole tree gives information about the importance of that feature in the decision tree.

The distribution of features with the most significance value can be seen below:

Bot attack importance list:

	Features	importance
1	Bwd Packet Length Mean	0.338129
2	Flow IAT Max	0.024407
3	Flow Duration	0.008495
4	Flow IAT Min	0.007067
5	Flow IAT Mean	0.005609
6	Flow Bytes/s	0.003518
7	Flow IAT Std	0.002213
8	Flow Packets/s	0.000667
9	Fwd Packet Length Mean	0.000418
10	Total Length of Bwd Packets	0.000333
11	Total Backward Packets	0.000223
12	Bwd Packet Length Max	0.000212
13	Fwd IAT Total	0.000141
14	Bwd Packet Length Std	0.000083
15	Total Fwd Packets	0.000077
16	Fwd Packet Length Std	0.000075
17	Fwd Packet Length Min	0.000069
18	Total Length of Fwd Packets	0.000048
19	Fwd Packet Length Max	0.000036
20	Bwd Packet Length Min	0.000005

DoS Slowhttptest attack importance list:

	Features	importance
1	Flow IAT Mean	0.653023
2	Fwd Packet Length Min	0.101739
3	Bwd Packet Length Mean	0.029976
4	Total Length of Bwd Packets	0.007153
5	Fwd Packet Length Std	0.007107
6	Fwd Packet Length Mean	0.006074
7	Bwd Packet Length Max	0.003857
8	Bwd Packet Length Std	0.002934
9	Flow IAT Min	0.002184
10	Fwd Packet Length Max	0.001245
11	Total Fwd Packets	0.000808
12	Flow Duration	0.000802
13	Total Length of Fwd Packets	0.000604
14	Bwd Packet Length Min	0.000545
15	Flow Bytes/s	0.000421
16	Flow IAT Max	0.000327
17	Fwd IAT Total	0.000284
18	Flow IAT Std	0.000252
19	Flow Packets/s	0.000159
20	Total Backward Packets	0.000130

Infiltration attack importance list:

	Features	importance
1	Fwd Packet Length Max	0.204751
2	Fwd Packet Length Mean	0.163696
3	Flow Duration	0.052250
4	Total Length of Fwd Packets	0.026823
5	Bwd Packet Length Mean	0.018539
6	Fwd Packet Length Std	0.017846
7	Flow IAT Max	0.017182
8	Flow Bytes/s	0.010812
9	Flow IAT Mean	0.008816
10	Flow IAT Min	0.008310
11	Fwd IAT Total	0.007726
12	Flow IAT Std	0.002862
13	Bwd Packet Length Max	0.002737
14	Bwd Packet Length Std	0.002030
15	Fwd Packet Length Min	0.001937
16	Total Fwd Packets	0.001862
17	Total Backward Packets	0.001680
18	Bwd Packet Length Min	0.001610
19	Flow Packets/s	0.001406
20	Total Length of Bwd Packets	0.000000

DDoS attack importance list:

	Features	importance
1	Bwd Packet Length Std	0.471368
2	Total Backward Packets	0.093576
3	Fwd IAT Total	0.010827
4	Flow Duration	0.006320
5	Flow IAT Min	0.006110
6	Total Length of Fwd Packets	0.006037
7	Flow IAT Std	0.005439
8	Flow IAT Mean	0.005238
9	Flow Bytes/s	0.004741
10	Flow IAT Max	0.004703
11	Fwd Packet Length Max	0.001647
12	Fwd Packet Length Std	0.001406
13	Flow Packets/s	0.001019
14	Fwd Packet Length Mean	0.000619
15	Bwd Packet Length Max	0.000577
16	Bwd Packet Length Min	0.000405
17	Bwd Packet Length Mean	0.000197
18	Total Length of Bwd Packets	0.000085
19	Total Fwd Packets	0.000042
20	Fwd Packet Length Min	0.000017

DoS slowloris attack importance list:

	Features	importance
1	Flow IAT Mean	0.473856
2	Total Length of Bwd Packets	0.057384
3	Bwd Packet Length Mean	0.053022
4	Total Fwd Packets	0.021895
5	Fwd Packet Length Std	0.002831
6	Flow Bytes/s	0.000965
7	Total Backward Packets	0.000769
8	Fwd IAT Total	0.000755
9	Fwd Packet Length Max	0.000728
10	Flow IAT Std	0.000664
11	Bwd Packet Length Max	0.000608
12	Flow IAT Max	0.000603
13	Total Length of Fwd Packets	0.000579
14	Fwd Packet Length Mean	0.000537
15	Bwd Packet Length Std	0.000528
16	Flow Packets/s	0.000429
17	Flow Duration	0.000375
18	Flow IAT Min	0.000348
19	Fwd Packet Length Min	0.000152
20	Bwd Packet Length Min	0.000021

PortScan attack importance list:

	Features	importance
1	Flow Bytes/s	0.313933
2	Total Length of Fwd Packets	0.304613
3	Fwd IAT Total	0.000427
4	Flow Duration	0.000398
5	Fwd Packet Length Max	0.000150
6	Flow IAT Max	0.000059
7	Flow IAT Mean	0.000054
8	Flow Packets/s	0.000031
9	Flow IAT Min	0.000031
10	Total Length of Bwd Packets	0.000022
11	Total Fwd Packets	0.000021
12	Fwd Packet Length Mean	0.000019
13	Bwd Packet Length Std	0.000018
14	Flow IAT Std	0.000017
15	Total Backward Packets	0.000014
16	Fwd Packet Length Std	0.000009
17	Bwd Packet Length Max	0.000004
18	Bwd Packet Length Mean	0.000002
19	Bwd Packet Length Min	0.000002
20	Fwd Packet Length Min	0.000001

DoS GoldenEye attack importance list:

	Features	importance
1	Flow IAT Max	0.413073
2	Bwd Packet Length Std	0.111039
3	Flow IAT Min	0.054021
4	Total Backward Packets	0.044895
5	Fwd Packet Length Min	0.009531
6	Flow IAT Mean	0.004040
7	Flow IAT Std	0.003863
8	Fwd Packet Length Max	0.002899
9	Bwd Packet Length Mean	0.001228
10	Flow Bytes/s	0.000928
11	Flow Duration	0.000927
12	Flow Packets/s	0.000892
13	Fwd IAT Total	0.000719
14	Bwd Packet Length Max	0.000445
15	Total Length of Fwd Packets	0.000406
16	Fwd Packet Length Mean	0.000393
17	Total Length of Bwd Packets	0.000217
18	Total Fwd Packets	0.000065
19	Bwd Packet Length Min	0.000033
20	Fwd Packet Length Std	0.000027

FTP-Patator attack importance list:

	Features	importance
1	Fwd Packet Length Max	1.098307e-01
2	Fwd Packet Length Std	2.437956e-02
3	Fwd Packet Length Mean	2.200624e-03
4	Bwd Packet Length Std	8.997715e-04
5	Bwd Packet Length Mean	7.081365e-04
6	Bwd Packet Length Max	6.054917e-04
7	Total Length of Bwd Packets	4.015506e-04
8	Flow IAT Min	2.611238e-04
9	Total Length of Fwd Packets	1.766284e-04
10	Total Fwd Packets	1.634693e-04
11	Flow Duration	1.330083e-04
12	Fwd IAT Total	9.598814e-05
13	Flow IAT Mean	9.187611e-05
14	Total Backward Packets	8.884482e-05
15	Flow IAT Max	8.359459e-05
16	Flow Packets/s	7.030109e-05
17	Fwd Packet Length Min	4.619749e-05
18	Flow IAT Std	2.987659e-05
19	Flow Bytes/s	2.244798e-05
20	Bwd Packet Length Min	4.764849e-07

SSH-Patator attack importance list:

	Features	importance
1	Fwd Packet Length Max	0.000881
2	Flow Duration	0.000748
3	Flow IAT Max	0.000497
4	Total Length of Fwd Packets	0.000448
5	Flow IAT Mean	0.000425
6	Flow Packets/s	0.000423
7	Flow Bytes/s	0.000375
8	Fwd IAT Total	0.000329
9	Flow IAT Std	0.000177
10	Fwd Packet Length Mean	0.000158
11	Flow IAT Min	0.000111
12	Total Backward Packets	0.000100
13	Bwd Packet Length Min	0.000099
14	Fwd Packet Length Std	0.000070
15	Total Fwd Packets	0.000070
16	Fwd Packet Length Min	0.000040
17	Bwd Packet Length Max	0.000032
18	Total Length of Bwd Packets	0.000027
19	Bwd Packet Length Mean	0.000014
20	Bwd Packet Length Std	0.000008

DoS Hulk attack importance list:

	Features	importance
1	Bwd Packet Length Std	5.148222e-01
2	Fwd Packet Length Std	7.321079e-02
3	Fwd Packet Length Max	4.515548e-03
4	Flow IAT Min	1.675778e-03
5	Flow Duration	1.218072e-03
6	Total Backward Packets	3.813481e-04
7	Flow IAT Std	2.572354e-04
8	Flow IAT Max	2.517998e-04
9	Total Length of Bwd Packets	1.778769e-04
10	Fwd IAT Total	1.739909e-04
11	Flow IAT Mean	9.875828e-05
12	Flow Packets/s	8.114421e-05
13	Bwd Packet Length Mean	5.449508e-05
14	Flow Bytes/s	2.752602e-05
15	Total Fwd Packets	1.227050e-05
16	Bwd Packet Length Max	1.004453e-05
17	Bwd Packet Length Min	9.303096e-06
18	Fwd Packet Length Mean	8.013636e-06
19	Total Length of Fwd Packets	4.604820e-06
20	Fwd Packet Length Min	1.810830e-08

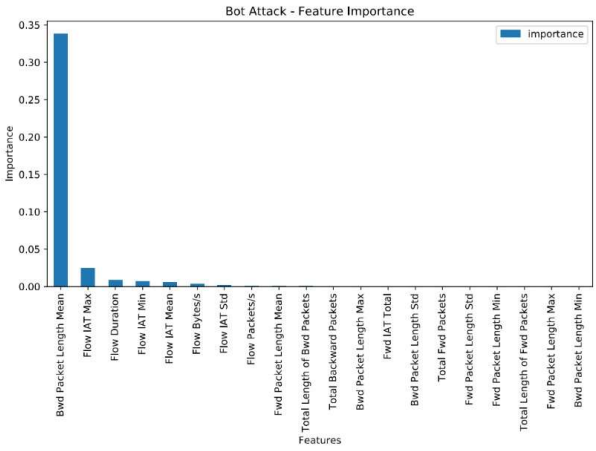
Heartbleed attack importance list:

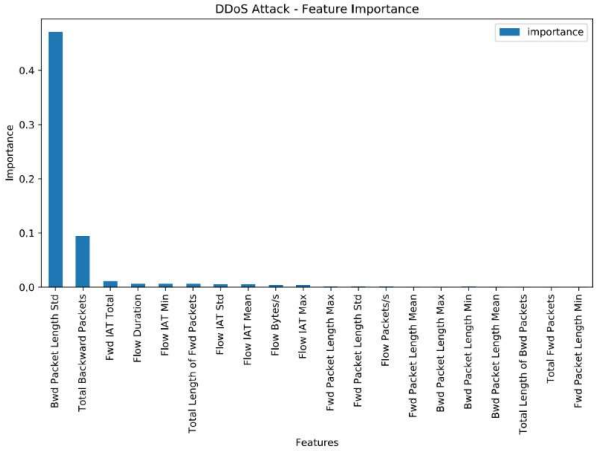
	Features	importance
1	Total Backward Packets	0.052
2	Fwd Packet Length Max	0.048
3	Flow IAT Min	0.048
4	Bwd Packet Length Max	0.048
5	Total Length of Fwd Packets	0.044
6	Total Length of Bwd Packets	0.044
7	Bwd Packet Length Mean	0.044
8	Bwd Packet Length Std	0.036
9	Total Fwd Packets	0.028
10	Flow Duration	0.016
11	Fwd IAT Total	0.004
12	Fwd Packet Length Std	0.004
13	Fwd Packet Length Mean	0.000
14	Flow Bytes/s	0.000
15	Flow Packets/s	0.000
16	Flow IAT Mean	0.000
17	Flow IAT Std	0.000
18	Flow IAT Max	0.000
19	Fwd Packet Length Min	0.000
20	Bwd Packet Length Min	0.000

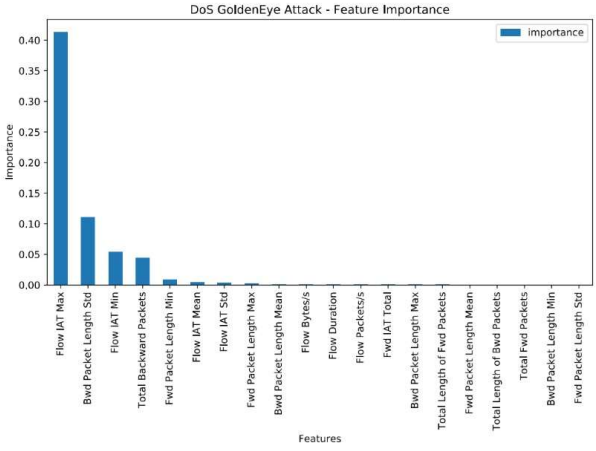
Web Attack attack importance list:

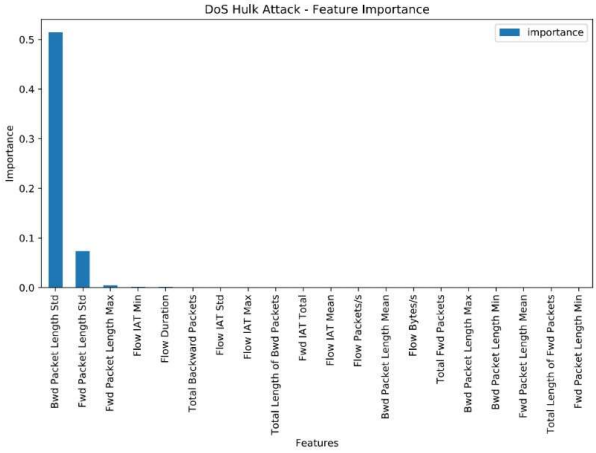
	Features	importance
1	Bwd Packet Length Std	0.007255
2	Total Length of Fwd Packets	0.006046
3	Flow Bytes/s	0.003366
4	Flow IAT Max	0.002102
5	Bwd Packet Length Max	0.001728
6	Flow IAT Mean	0.000760
7	Fwd Packet Length Max	0.000638
8	Fwd Packet Length Std	0.000616
9	Flow Duration	0.000541
10	Flow Packets/s	0.000526
11	Total Fwd Packets	0.000506
12	Fwd IAT Total	0.000505
13	Flow IAT Min	0.000499
14	Fwd Packet Length Mean	0.000490
15	Total Length of Bwd Packets	0.000454
16	Total Backward Packets	0.000177
17	Flow IAT Std	0.000140
18	Bwd Packet Length Mean	0.000102
19	Bwd Packet Length Min	0.000016
20	Fwd Packet Length Min	0.000008

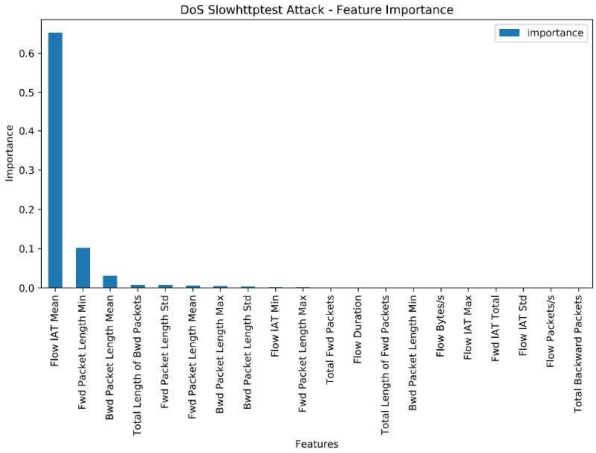
The distribution of features with the most significance can be seen below:

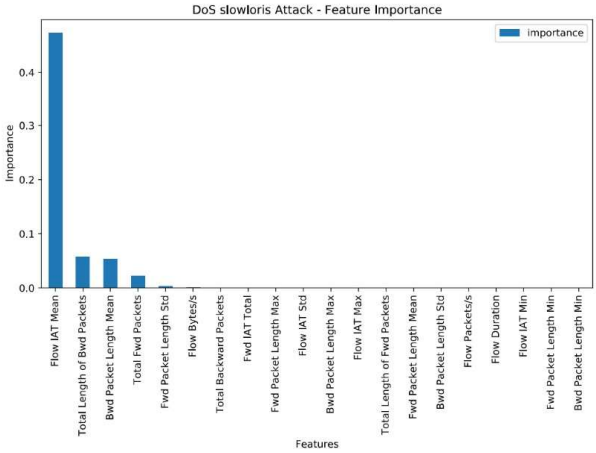

Bot Attack - Feature Importance
importance
Importance
Features
Bwd Packet Length Mean
Flow IAT Max
Flow Duration
Flow IAT Min
Flow IAT Mean
Flow Bytes/s
Flow IAT Std
Flow Packets/s
Fwd Packet Length Mean
Total Length of Bwd Packets
Total Backward Packets
Bwd Packet Length Max
Fwd IAT Total
Bwd Packet Length Std
Total Fwd Packets
Fwd Packet Length Std
Fwd Packet Length Min
Total Length of Fwd Packets
Fwd Packet Length Max
Bwd Packet Length Min

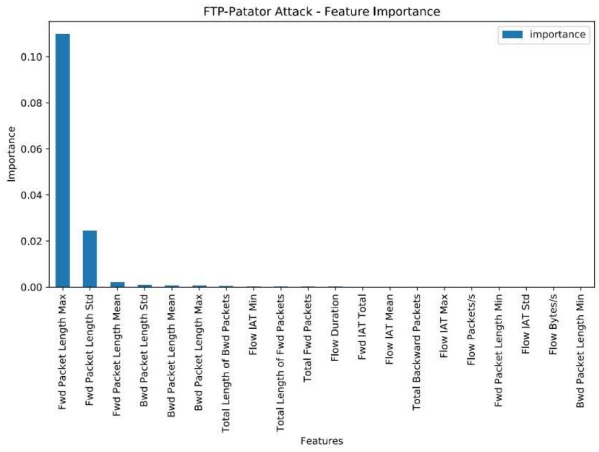

DDoS Attack - Feature Importance
importance
Importance
Features
Bwd Packet Length Std
Total Backward Packets
Fwd IAT Total
Flow Duration
Flow IAT Min
Total Length of Fwd Packets
Flow IAT Std
Flow IAT Mean
Flow Bytes/s
Flow IAT Max
Fwd Packet Length Max
Fwd Packet Length Std
Flow Packets/s
Fwd Packet Length Mean
Bwd Packet Length Max
Bwd Packet Length Min
Bwd Packet Length Mean
Total Length of Bwd Packets
Total Fwd Packets
Fwd Packet Length Min

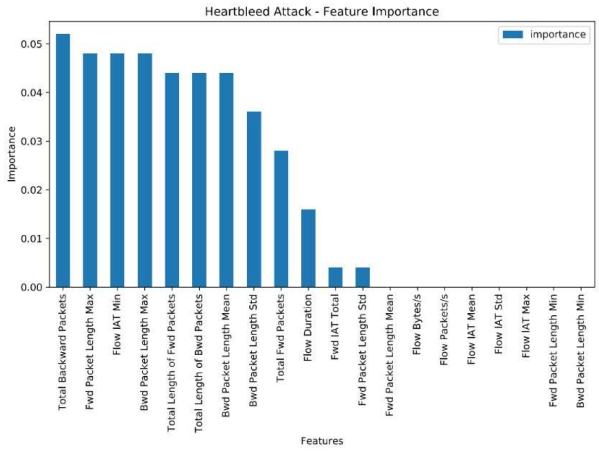

DoS GoldenEye Attack - Feature Importance
importance
Importance
Features
Flow IAT Max
Bwd Packet Length Std
Flow IAT Min
Total Backward Packets
Fwd Packet Length Min
Flow IAT Mean
Flow IAT Std
Fwd Packet Length Max
Bwd Packet Length Mean
Flow Bytes/s
Flow Duration
Flow Packets/s
Fwd IAT Total
Bwd Packet Length Max
Total Length of Fwd Packets
Fwd Packet Length Mean
Total Length of Bwd Packets
Total Fwd Packets
Bwd Packet Length Min
Fwd Packet Length Std

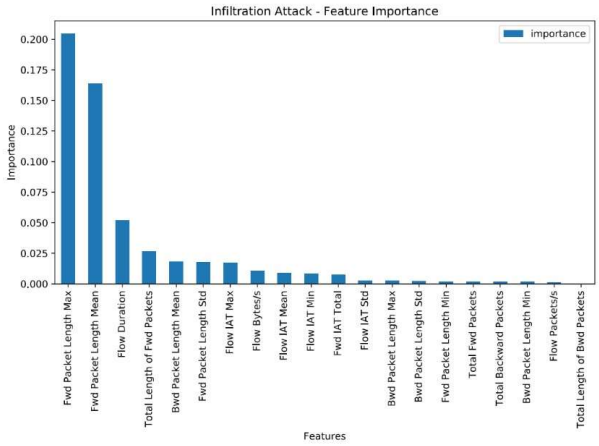

DoS Hulk Attack - Feature Importance
importance
Importance
Features
Bwd Packet Length Std
Fwd Packet Length Std
Fwd Packet Length Max
Flow IAT Min
Flow Duration
Total Backward Packets
Flow IAT Std
Flow IAT Max
Total Length of Bwd Packets
Fwd IAT Total
Flow IAT Mean
Flow Packets/s
Bwd Packet Length Mean
Flow Bytes/s
Total Fwd Packets
Bwd Packet Length Max
Bwd Packet Length Min
Fwd Packet Length Mean
Total Length of Fwd Packets
Fwd Packet Length Min

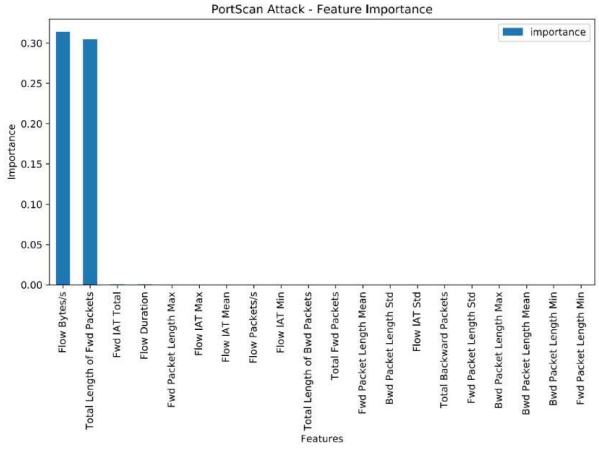

DoS Slowhttptest Attack - Feature Importance
importance
Importance
Features
Flow IAT Mean
Fwd Packet Length Min
Bwd Packet Length Mean
Total Length of Bwd Packets
Fwd Packet Length Std
Fwd Packet Length Mean
Bwd Packet Length Max
Bwd Packet Length Std
Flow IAT Min
Fwd Packet Length Max
Total Fwd Packets
Flow Duration
Total Length of Fwd Packets
Bwd Packet Length Min
Flow Bytes/s
Flow IAT Max
Fwd IAT Total
Flow IAT Std
Flow Packets/s
Total Backward Packets

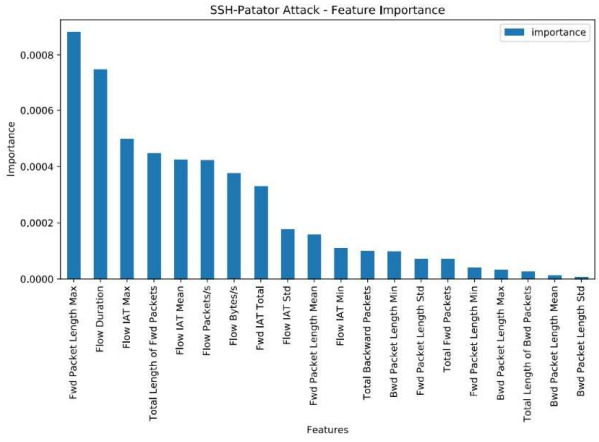

DoS slowloris Attack - Feature Importance
importance
Importance
Features
Flow IAT Mean
Total Length of Bwd Packets
Bwd Packet Length Mean
Total Fwd Packets
Fwd Packet Length Std
Flow Bytes/s
Total Backward Packets
Fwd IAT Total
Fwd Packet Length Max
Flow IAT Std
Bwd Packet Length Max
Flow IAT Max
Total Length of Fwd Packets
Fwd Packet Length Mean
Bwd Packet Length Std
Flow Packets/s
Flow Duration
Flow IAT Min
Fwd Packet Length Min
Bwd Packet Length Min

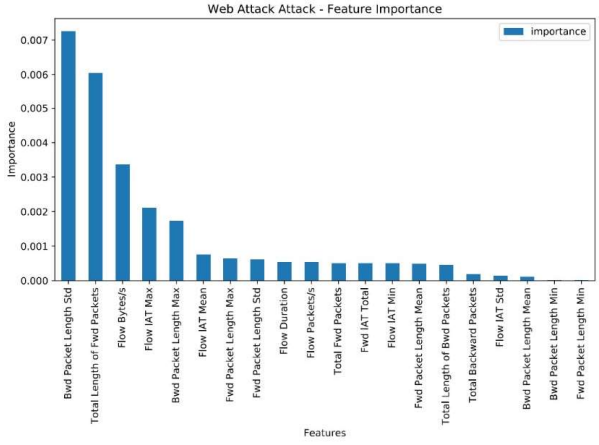












Feature Selection According to Attack or Benign

Another approach in feature selection is to apply the Random Forest Regressor operation to the whole dataset by collecting all attack types under a single label; “attack”. So, the data in this file contains only the attack and benign tags. As a result of this operation, the feature list obtained is shown in Table 4 and graphics of feature in Figure 16.

Feature Name	**Importance Weight**		**Feature Name**	**Importance Weight**
Bwd Packet Length Std	0.246627		Flow IAT Mean	0.003266

Flow Bytes/s	0.178777		Total Length of Bwd Packets	0.001305
Total Length of Fwd Packets	0.102417		Fwd Packet Length Min	0.000670
Fwd Packet Length Std	0.063889		Bwd Packet Length Mean	0.000582
Flow IAT Std	0.009898		Flow Packets/s	0.000541
Flow IAT Min	0.006946		Fwd Packet Length Mean	0.000526
Fwd IAT Total	0.005121		Total Backward Packets	0.000169
Flow Duration	0.004150		Total Fwd Packets	0.000138
Bwd Packet Length Max	0.004007		Fwd Packet Length Max	0.000125
Flow IAT Max	0.003579		Bwd Packet Length Min	0.000084

Table 4. According Attack and Benign Labels Feature Importance Weight List

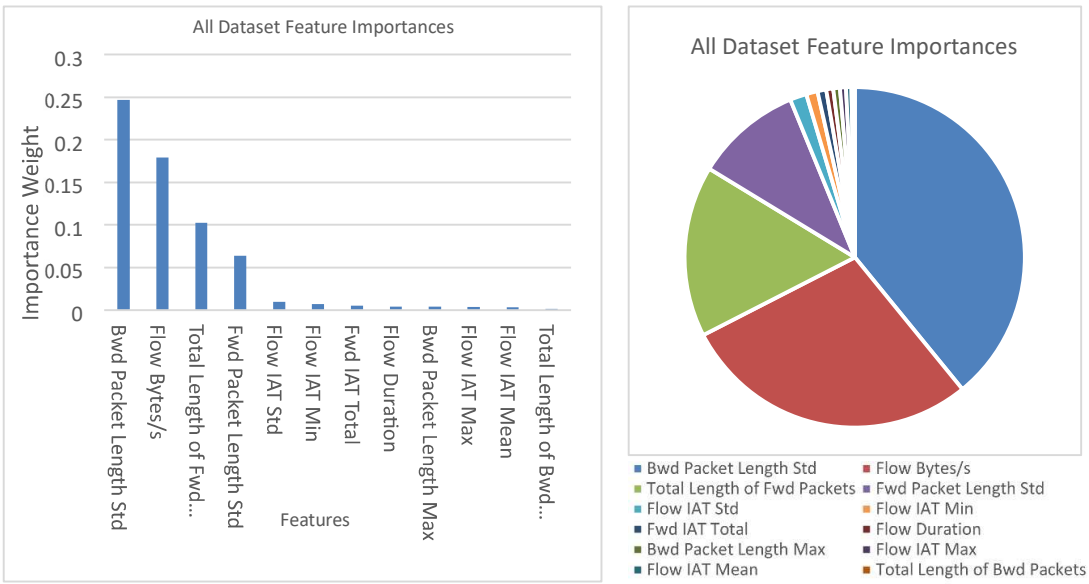


Figure 16. Graphs of Feature Importance Weights According to Attack and Benign Labels

Implementation of Machine Learning Algorithms

Naïve Bayes

Naïve Bayes is a machine learning algorithm that is simplified with the addition of the independence condition on Bayes' theorem.

Bayes' theorem is expressed by the following equation:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \qquad \textbf{2.1}$$

P (A | B): The probability of occurrence of A in the case of B occurrences.
P (B | A): The probability of occurrence of B in the case of A occurrences. P(A) and P(B): Prior probabilities of A and B.

Naive Bayes is a network of probabilities consisting of a parent node representing the unobserved state and multiple child nodes representing the observed states.

In this statistical network, it is assumed that the child nodes are independent of each other. Although this hypothesis is the basis of Naive Bayes' theory, the assumption that sub nodes are independent of one another is often not correct. This is the underlying reason for the Naive Bayes method to achieve lower accuracy when compared to other machine learning methods. Despite this disadvantage, it is one of the most preferred machine learning methods because the training period is very short, and the computational cost is very low.

Decision Trees

Decision trees are one of the popular classifiers used in machine learning methods. In this approach, the rules used are straightforward and understandable.

Each decision tree consists of nodes (root-node and sub-nodes), branches and leaves. Within each node, there is a decision statement. According to the result of this decision, the algorithm chooses one of the two branches in the next step (the number of branches may be more than two in some sub-algorithms). This selected branch takes the algorithm to the next node. This process ends with the last element, the leaf.

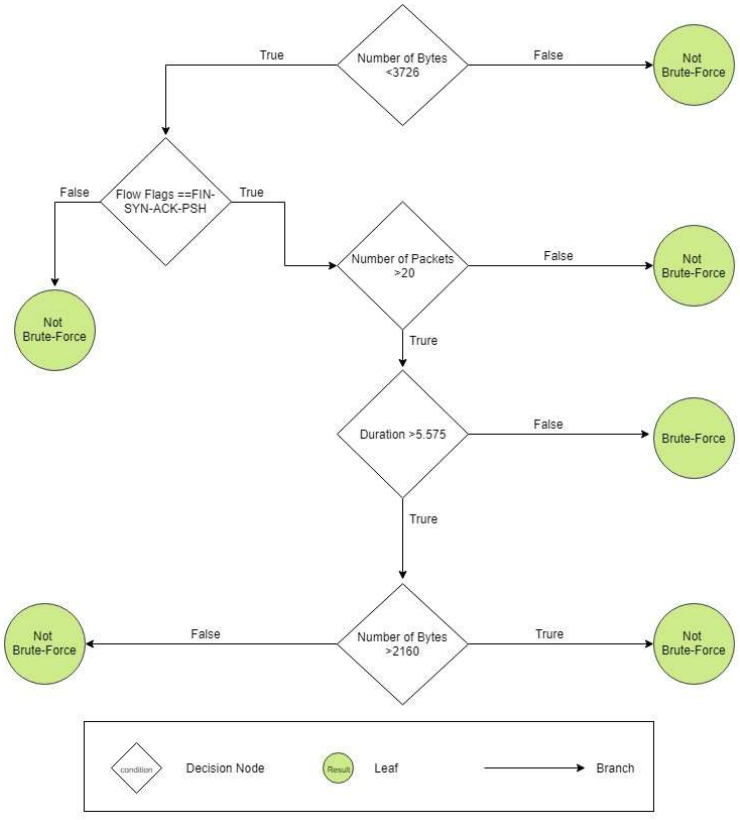


Figure 7. Detection of brute-force attack using decision trees.

Decision trees method applies the divide-and-conquer strategy. It makes very large and meaningless data smaller and group them into a meaningful one. It is therefore widely used for the classification of large and complex data.

Unfortunately, these complex trees lead to overfitting[1]and prevent getting fair results. Another disadvantage is that a slight misclassification of the data at the beginning of decomposition can lead to different branches and misleading results. To cope with this, usually, the data to be used must go through pre-processing.

In the implementation phase, a decision tree algorithm, ID3(Iterative Dichotomiser 3), is used.This algorithm is suitable for situations where the training set contains many features. It also stands out with its remarkable features such as giving a reasonable value without doing too much computation and connecting more than two branches to decision nodes.

Random Forest

Random forest is a machine learning approach that uses decision trees. In this method, a "forest" is created by assembling a large number of different decision tree structures which are formed in different ways.

Random forest generates N decision trees from training set data. During this process, it randomly resamples the training set for each tree. Thus, n decision trees are obtained, each of which is different from the other. Finally, voting is performed by selecting new estimates from estimates made by N trees. The value with the highest rating is determined as the final value.

The random forest has many advantages. These can be listed as follows:

- This algorithm can work well with very large and complicated datasets.
- The overfitting problem frequently encountered by decision trees is very rare in this algorithm.
- It can be applied to a lot of kind of machine learning problem.
- It is also good to deal with missing values in the data set. It replaces these lost values with their own created values.
- In addition, it calculates and uses the importance level of the variables when making the classification. Thanks to this, it is also used for feature selection in the machine learning.

On the other hand, the structure of Random Forest is quite complicated because it is made up of many decision trees. Another disadvantage is that It is pretty difficult to understand its functioning.

K Nearest Neighbor

KNN (K Nearest Neighbor), which is a sample-based method, is one of the most used machine learning algorithms with its simple and fast structure. This algorithm depends on the assumption that the examples in a dataset will exist close to the examples with similar properties in another dataset.

In this context, KNN identifies the class of new data that is not classifiable by using training data of known class type. This determination is made by observing the nearest neighbors of the new sample, for which no classifications are specified.

In a plane with N properties, the number of neighbors to be looked at for an unclassified sample is specified by the number K. For the unknown sample, the distances to the neighbors are calculated and the smallest K numbers are chosen from these distance values. The most repeated property within the K values is assigned as the unknown instance property. In Figure 5, a visual is created for the values 2 and 5 of K in a 3-dimensional plane (N = 3).

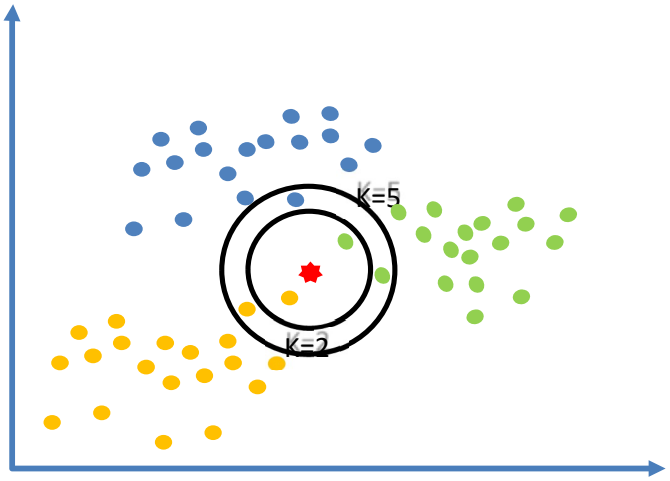


Figure 8. Operation of KNN algorithm for K = 2 and K = 5 values.

KNN, which provides good performance over multidimensional data and is a fast algorithm during the training phase, is relatively slow in the estimation stage [43].

AdaBoost

AdaBoost (Adaptive Boosting), a boosting method, is a machine learning algorithm developed to improve classification performance. The basic working principle of Boosting algorithms can be explained as follows: The data are first divided into groups with rough draft rules. Whenever the algorithm is run, new rules are added to this rough draft rules. In this way, many weak and low performance rules called "basic rules" are obtained.

Once the algorithm has been working many times, these weak rules are combined into a single rule that is much stronger and more successful. During this process, the Algorithm assigns a weighting coefficient to each weak rule, giving the highest coefficient value to the lowest error rate. These weight values come into play when final rules are selected. The final rule is created by giving priority to the high scored weak rules.

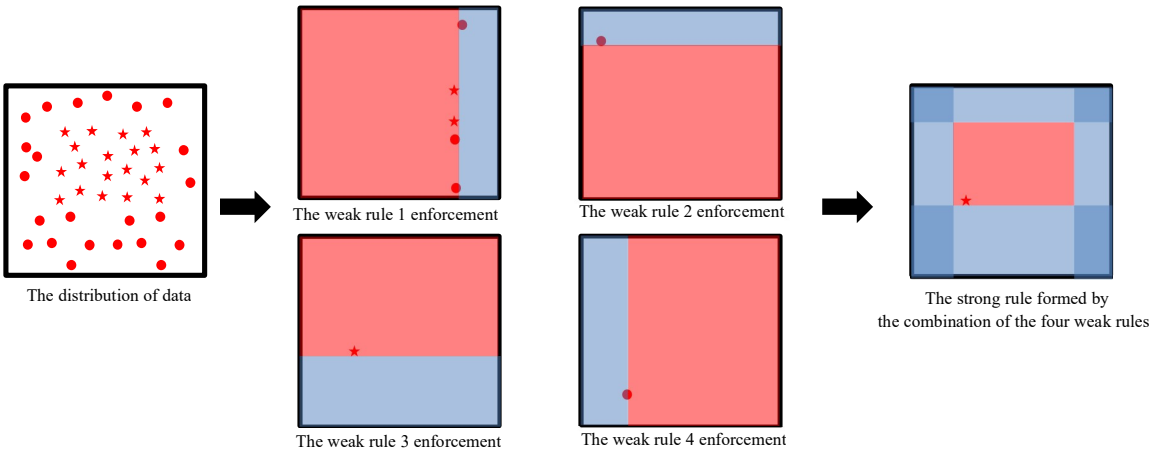


Figure 9. Demonstration of the operation of the AdaBoost algorithm.

AdaBoost algorithm has many advantages. These can be listed as follows:

- There is no need for variable transformation to use it in this algorithm.
- It can perform operations on too many weak rules.
- The overfitting problem very rare in AdaBoost.
- It is also good to deal with missing values in the data set.

On the other hand, it can be mentioned as disadvantages that it is weak against noise and extreme values, and that the predictive value does not have the highest values when compared to other algorithms.

MLP

MLP (Multi-Layer Perceptron) is a genre of artificial neural networks. Artificial neural networks (ANN) is a machine learning method that takes inspiration from the way the human brain works. The intention of this method is to imitate the properties of the human brain, such as learning, decision making, and deriving new information. While the human brain is made up of interconnected cells called neurons, artificial neural networks are made up of interconnected hierarchical artificial cells.

MLP consists of three stages. These stages are the Input layer, the hidden layer, and the output layer. The input layer is the stage of the MLP that is responsible for receiving data. No information processing is performed on this layer. Only the received information is transmitted to the next layer, the hidden layer. Each neuron in this step bonds with all the neurons in the hidden layer.

In the hidden layer, the data sent from the input layer is processed and transmitted to the next layer, the output layer. In MLP, the number of hidden layers or the number of artificial neurons in this layer may change. For example, the MLP used in this study has a 3-step hidden layer and 13 neurons in each layer. The change in the layer count and neuron number directly affects MLP performance and yield. By increasing the number of layers and neurons, MLP can deal with more complex problems. However, the increase in the number of these elements will also lead to an increase in the duration of the algorithm.

In the output layer, which is the last layer, each cell is tied to all the cells in the hidden layer, and the results of the processed data in the hidden layer are served at this stage.

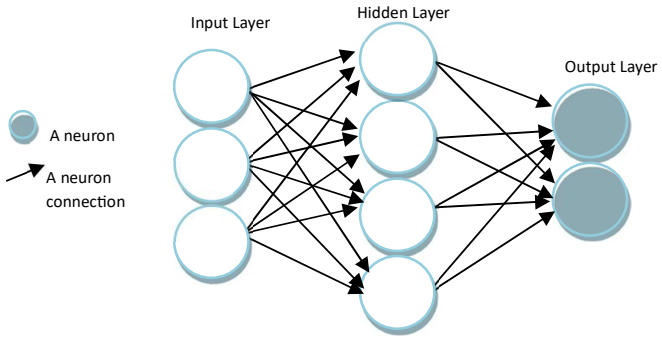


Figure 10. Demonstration of a three-layer MLP

The advantages MLP provides are as follows:

- It is good at coping with complicated problems.
- It can work with missing data.
- It can generalize after the learning process. Thus, it serves a wider area than the other machine learning algorithms.

But from the other side:

- It is difficult to build the network structure.
- The user should decide the appropriate network structure.
- Overfitting problems may be encountered.
- It is difficult to interpret and understand.

QDA

QDA (Quadratic Discriminant Analysis) is a discriminant analysis method. Discriminant Analysis is a statistical technique for assigning a measured data to one group among many groups. When this assignment is made, the observed data must be assigned to the group to which it belongs. If it is assigned a group that does not belong to it, an error occurs.

This mistake is known as "the error rate". The purpose of discriminant analysis is to perform the assignment process with a minimum error rate. Assuming that the data group has normal distribution and the variances are equal, the analysis is called Linear Discriminant Analysis. However, if the assumption of equality of groups' variances is not correct, a Quadratic Discriminant Analysis is obtained.

In order to be able to apply the Quadratic Discriminant Analysis, the number of samples observed must be greater than the number of groups.

Two different approaches have been used to apply machine learning algorithms to the dataset. In the first method, the files created in the Feature Selection section and the attributes obtained in the same section are used. These files contain 30% attack and 70% benign data and each of

them named by the type of attack it contains. The seven machine learning methods are applied to each file 10 times, resulting in a separate outcome for each attack type. With this method, it is aimed to observe the effectiveness and performance of different machine learning methods on different attack types.

In the second approach, the entire data set is used as a single file. All attacks contained in this file are collected under a single common name; "attack". So, the data in this file contains only the attack and benign tags. The set of features to be used consists of combining the 4 features with the highest importance-weight achieved for each attack in approach 1 under a single roof. Thus, 4 features are obtained from each of the 12 attack types, resulting in a pool of features consisting of 48 attributes. After the repetitions are removed, the number of features is 18. The list of these features can be seen in Table 5

Bwd Packet Length Max	Flow IAT Mean	Fwd Packet Length Min
Bwd Packet Length Mean	Flow IAT Min	Fwd Packet Length Std
Bwd Packet Length Std	Flow IAT Std	Total Backward Packets
Flow Bytes/s	Fwd IAT Total	Total Fwd Packets
Flow Duration	Fwd Packet Length Max	Total Length of Bwd Packets
Flow IAT Max	Fwd Packet Length Mean	Total Length of Fwd Packets

Table 5. The feature list created for all attack types.

Results and Discussion

In this section, the results of the project done in the implementation section are presented. In this context, in the assessment carried out, the evaluation criteria are presented via the data of the F-measure.

The performance evaluation procedures are repeated 10 times for each machine learning algorithm. The numbers given in the tables are the arithmetic mean of these 10 processes. Box and whisker graphs are created to illustrate the consistency of the results and the change between them.

4.1 Approach 1 - Using 12 Attack Types

Seven different machine learning methods are applied to 12 different attack types and the obtained results are presented in Table 7.

Among the results presented in Table 7, the biggest and smallest achievements are emphasized as follows: The greatest scores are bold, the smallest scores are underlined and italics. In the results of the algorithms, if there is an equality in F-measure, the following values are examined in order to eliminate equality, respectively: accuracy, precision, recall, and time (In Table 7, only F-Measure values are given.).

Attack Names	**F-Measures**						
	NB	RF	KNN	ID3	AB	MLP	QDA
Bot	*0.54*	0.96	0.95	0.96	**0.97**	0.64	0.68
DDoS	0.77	0.96	0.92	**0.96**	0.96	0.76	*0.34*
DoS GoldenEye	0.81	0.99	0.98	**0.99**	0.99	*0.64*	0.71

DoS Hulk	*0.23*	0.93	0.96	**0.96**	0.96	0.95	0.36
DoS Slowhttptest	*0.35*	0.98	0.99	0.98	**0.99**	0.78	0.38
DoS slowloris	*0.37*	0.95	0.95	**0.96**	0.95	0.74	0.46
FTP-Patator	**1.00**	1.00	1.00	1.00	1.00	1.00	1.00
Heartbleed	**1.00**	0.99	1.00	0.95	0.93	*0.66*	1.00
Infiltration	0.78	0.92	0.88	0.89	**0.92**	*0.52*	0.83
PortScan	*0.39*	1.00	1.00	**1.00**	1.00	0.61	0.85
SSH-Patator	*0.33*	0.96	0.95	**0.96**	0.96	0.83	0.41
Web Attack	0.74	0.97	0.93	**0.97**	0.97	*0.60*	0.84

Table 7. Distribution of results according to type of attack and machine learning algorithm.

When looking at the results, it is noticed that Random Forest, KNN, ID3 and Adaboost algorithms, have achieved over 90% success in detecting almost all attack types. Among these four algorithms, ID3, which is the most successful, has completed 7of 12 tasks with the highest score. In fact, in the 6 of these 7 tasks (DDoS, DoS GoldenEye, DoS Hulk, PortScan, SSHPatator and Web Attack) ID3 shares the highest score with at least one algorithm. However, low processing time puts it ahead of other algorithms.

Another point of interest is that all machine learning algorithms achieve a full score in the FTPPatator attack. This could be due to the fact that the features used to describe the FTP-Patator attack may become a characteristic that can be easily distinguished between normal and attack data, trapped in a very narrow range relative to normal data (See Figure 17).

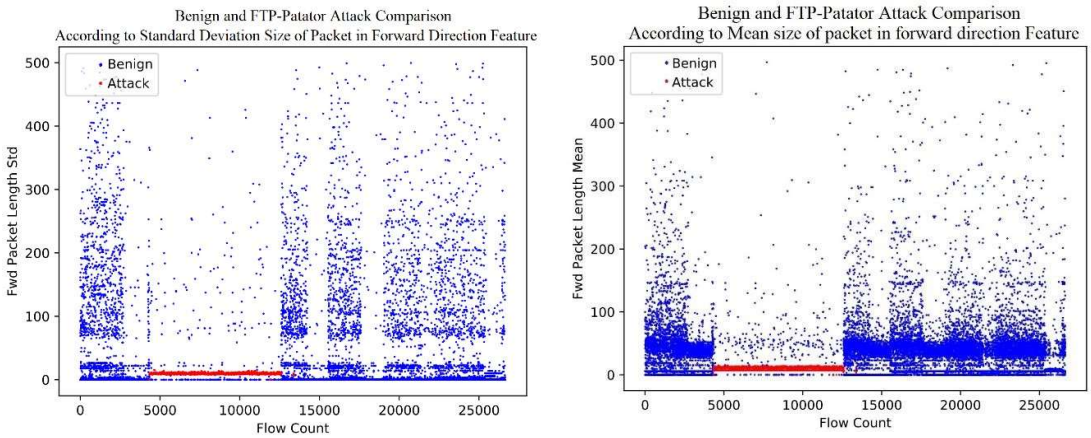


Figure 17. "Fwd Packet Length Std" and "Fwd Packet Length Mean" features in FTP-Patator and Benign flow.

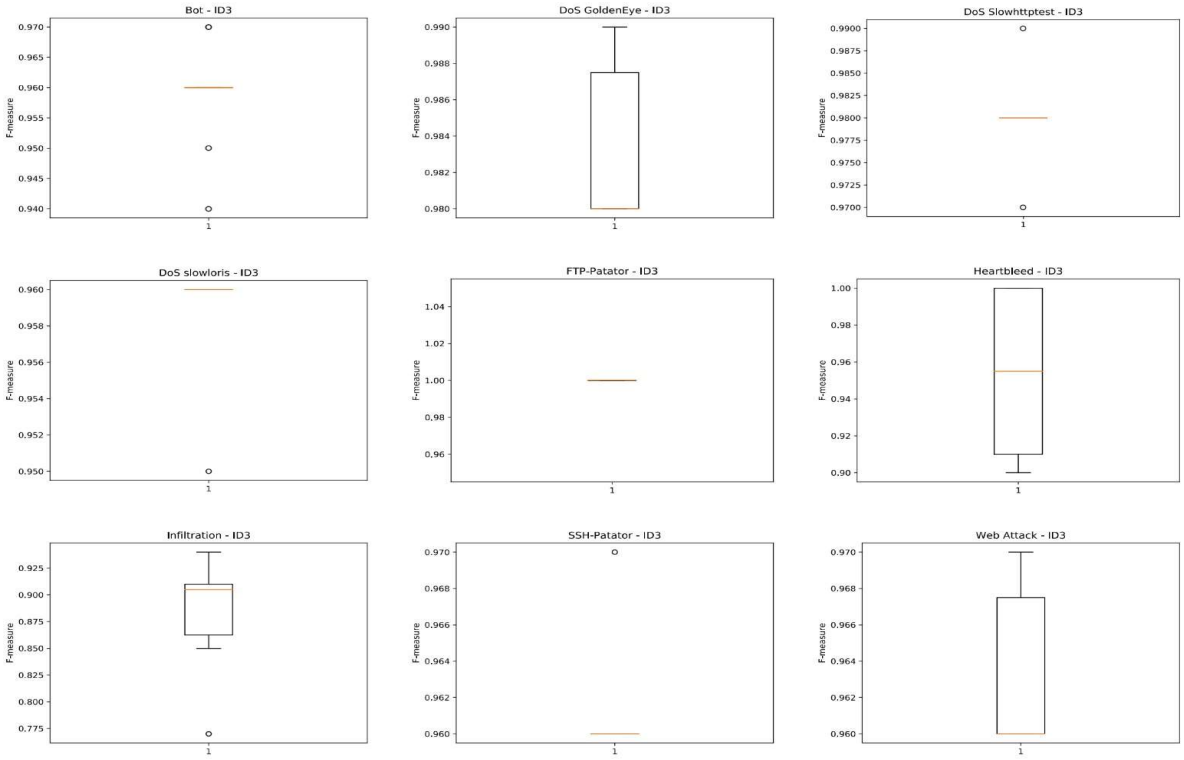


Figure 19. Box and whisker graphics containing the results of applying the ID3 algorithm to various attack types.

4.2 Approach 2 - Using Two Groups: Attack and Benign

In this section, the entire data set is used as a single dataset file. All attacks contained in this file are collected under a single common name, "attack". Seven different machine learning methods are applied to this dataset. In this approach, the 18 features obtained in the Feature Selection According to Attack or Benign section are used.

Using Features Selection for All Dataset.

An implementation that uses an alternative feature selection method can be seen in Table 9. In this method, the 18 features obtained in the Feature Selection According to Attack or Benign section are used. The changing parts are highlighted in red. When Table 8 and Table 9 are compared, there is no significant change in the algorithms of Random Forest, ID3, AdaBoost, and MLP based on F-measure. However, In Naive Bayes and QDA an increase of 2 and 11 points were observed respectively.

Machine Learning Algorithms	**Evaluation Criteria**				
	F-Measure	Precision	Recall	Accuracy	Time
Naive Bayes	0.81	0.8	0.82	0.82	1.6258
QDA	*0.41*	0.83	0.38	0.38	1.925
ID3	0.95	0.95	0.95	0.95	11.552
AdaBoost	0.94	0.94	0.94	0.94	144.166
MLP	0.79	0.815	0.84	0.84	51.799
K Nearest Neighbours	0.97	0.97	0.97	0.97	1038.253

Table 9. Implementation of features obtained using Random Forest Regressor for All Dataset.

	NaiveBayes					Randomforrest					KNN					ID3					Adaboost					MLP					QDA				
	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time	Acc	F1	Pr	Rc	Time
Bot	0.56	0.55	0.82	0.56	0.003	0.97	0.97	0.97	0.97	0.030	0.96	0.96	0.96	0.96	0.014	0.97	0.97	0.97	0.97	0.008	0.98	0.98	0.98	0.98	0.170	0.69	0.62	0.61	0.69	0.125	0.68	0.68	0.84	0.68	0.003
DDoS	0.77	0.76	0.76	0.77	0.045	0.96	0.96	0.97	0.96	0.430	0.93	0.93	0.93	0.93	1.361	0.96	0.96	0.97	0.96	0.200	0.96	0.96	0.96	0.96	2.818	0.77	0.74	0.76	0.77	4.962	0.42	0.35	0.80	0.42	0.054
DoS GoldenEye	0.82	0.80	0.82	0.82	0.011	0.99	0.99	0.99	0.99	0.096	0.98	0.98	0.98	0.98	0.093	0.98	0.98	0.98	0.98	0.043	0.98	0.98	0.98	0.98	0.674	0.62	0.61	0.72	0.62	0.711	0.95	0.95	0.95	0.95	0.013
DoS Hulk	0.34	0.23	0.80	0.34	0.298	0.94	0.93	0.94	0.94	3.738	0.96	0.96	0.96	0.96	254.4	0.96	0.96	0.96	0.96	0.909	0.96	0.96	0.96	0.96	22.16	0.94	0.94	0.94	0.94	25.63	0.41	0.36	0.81	0.41	0.319
DoS Slowhttptest	0.41	0.36	0.73	0.41	0.006	0.98	0.98	0.98	0.98	0.056	0.99	0.99	0.99	0.99	0.058	0.98	0.98	0.99	0.98	0.020	0.99	0.99	0.99	0.99	0.313	0.72	0.71	0.83	0.72	0.387	0.42	0.38	0.74	0.42	0.006
DoS slowloris	0.42	0.36	0.80	0.42	0.006	0.95	0.94	0.94	0.94	0.055	0.95	0.95	0.95	0.95	0.035	0.96	0.96	0.96	0.96	0.022	0.95	0.95	0.95	0.95	0.372	0.77	0.76	0.80	0.77	0.494	0.48	0.46	0.79	0.48	0.008
FTP-Patator	1.00	1.00	1.00	1.00	0.006	1.00	1.00	1.00	1.00	0.057	1.00	1.00	1.00	1.00	0.214	1.00	1.00	1.00	1.00	0.014	1.00	1.00	1.00	1.00	0.412	1.00	1.00	1.00	1.00	2.683	1.00	1.00	1.00	1.00	0.008
Heartbleed	1.00	1.00	1.00	1.00	0.004	1.00	1.00	1.00	1.00	0.011	1.00	1.00	1.00	1.00	0.002	0.95	0.95	0.98	0.95	0.001	0.95	0.94	0.94	0.95	0.003	0.52	0.47	0.47	0.52	0.011	1.00	1.00	1.00	1.00	0.005
Infiltration	0.82	0.79	0.82	0.82	0.002	0.93	0.93	0.95	0.93	0.011	0.85	0.86	0.87	0.85	0.003	0.91	0.91	0.94	0.91	0.002	0.90	0.90	0.93	0.90	0.051	0.59	0.53	0.53	0.59	0.008	0.83	0.82	0.85	0.83	0.002
PortScan	0.44	0.39	0.80	0.44	0.185	1.00	1.00	1.00	1.00	2.554	1.00	1.00	1.00	1.00	54.72	1.00	1.00	1.00	1.00	0.784	1.00	1.00	1.00	1.00	15.01	0.72	0.61	0.63	0.72	13.77	0.84	0.84	0.89	0.84	0.205
SSH-Patator	0.41	0.34	0.80	0.41	0.008	0.96	0.96	0.96	0.96	0.069	0.96	0.96	0.96	0.96	0.045	0.96	0.96	0.97	0.96	0.027	0.96	0.96	0.97	0.96	0.411	0.87	0.87	0.88	0.87	0.324	0.47	0.43	0.80	0.47	0.006
Web Attack	0.73	0.75	0.86	0.74	0.005	0.97	0.97	0.97	0.97	0.029	0.93	0.94	0.94	0.94	0.014	0.96	0.96	0.96	0.96	0.009	0.97	0.96	0.96	0.97	0.170	0.69	0.64	0.68	0.69	0.111	0.83	0.84	0.89	0.84	0.003

Machine Learning Implementation Results (for Attack Files)

Evaluation

In this section, the final results of the implementation (See Table 11) are evaluated.

The F-measure was determined as the main evaluation criterion. However, in cases where equality prevailed in the results, Recall and Precision were assessed respectively. The timing of the execution was not included in the evaluation, considering that it may be misleading, even though it is shared in both works.

Machine Learning Algorithms	**Our Study**			**Sharafaldin et al[4].**		
	F-Measure	Precision	Recall	F-Measure	Precision	Recall
Naive Bayes	**0.86**	0.86	0.87	0.84[2]	0.88	0.84[7]
QDA	0.86	0.87	0.88	**0.92**	0.97	0.88
Random Forest	0.94	0.94	0.94	**0.97**	0.98	0.97
ID3	0.95	0.95	0.95	**0.98**	0.98	0.98
AdaBoost	**0.94**	0.94	0.94	0.77	0.77	0.84
MLP	**0.83**	0.82	0.87	0.76	0.77	0.83
K Nearest Neighbours	**0.97**	0.97	0.97	0.96	0.96	0.96

Table 12. Comparison of the performance of two exercises against the evaluation criteria.

Table 12 and Figure 23 show the comparison of the results obtained. When the results obtained in them are examined, it is seen that Random Forest, ID3 and QDA algorithms of Sharafaldin and his colleagues [4] are 0.3,0.3,0.6 points higher than our study, respectively. This big difference in QDA is quite remarkable. Because, in our study, this algorithm has a maximum of 0.86 as a result of attempts to increase performance.

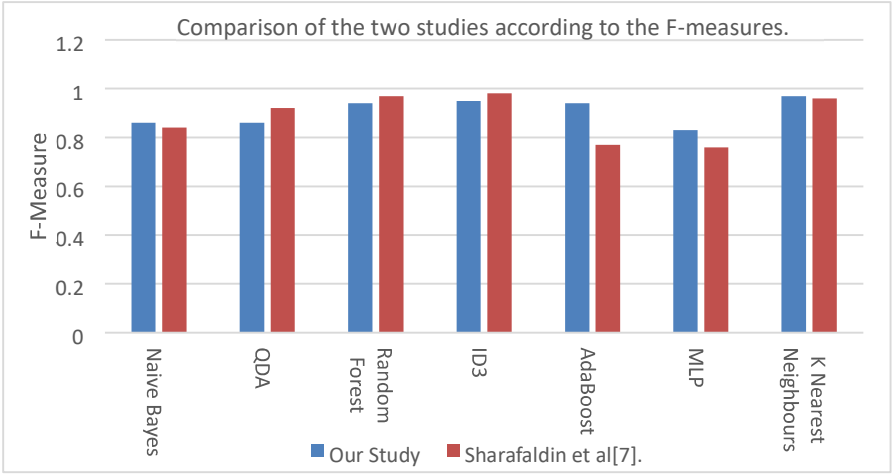


Figure 23. Comparison of the performance of the two studies compared to the F-measures.

On the other hand, the F-criterion score of our study is 0.1 and 0.2 points higher in the KNN and Naive Bayes algorithms, respectively. In the Adaboost and MLP methods, the difference is remarkably high (0.17 and 0.7) In both studies, MLP is the method with the lowest success rate. However, it is possible to increase performance by increasing the hidden layer size and depth in MLP. In our study, an MLP with a 3-step hidden layer and 13 neurons in each layer

was used, but in the other study, the comparison is not possible since no information is given on this topic.

The highest performance was in the KNN algorithm with 0.97 points in our Project whereas in the other study ID3 with 0.98 points. When speed is considered, in both studies Naive Bayes is the fastest algorithm, while the KNN is the slowest.

6 Conclusion and Future Work

Conclusion

In this Project, it is aimed to detect network anomaly using machine learning methods. In this context, the CICIDS2017 has been used as dataset because of its up-to-datedness, wide attack diversity, and various network protocols (e.g. Mail services, SSH, FTP, HTTP, and HTTPS). This dataset contains more than 80 features that define the network flow. During the application, the importance weight calculation was made with the Random Forest Regressor algorithm to decide which of these features will be used in machine learning methods. Two approaches have been used when making these calculations. In the first place, importance weights are calculated separately for each attack type. In the second method, all the attacks are collected under a single group and the importance weights for this group are calculated. That is, the common properties that are important for all attacks are determined. Finally, seven machine learning algorithms, which are widely used and have different qualities, have been applied to this data. These algorithms and the achieved performance ratios according to F-measure are as follows (F-measure takes a value between 0 and 1): Naive Bayes: 0.86, QDA: 0.86, Random Forest: 0.94, ID3: 0.95, AdaBoost: 0.94, MLP: 0.83, and K Nearest Neighbours: 0.97.

Future Work

In this project, various machine learning methods were applied independently from each other and experimental results were obtained. However, this method has a weak practical applicability in real life. In order to deal with this problem, a multi-layered / hierarchical machine learning structure can be designed.

References

[1] I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark dataset," *Software Networking,* vol. 1, no. 1, pp. 177200, 2017.

[2] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications,* vol. 60, pp. 19-31, 2016.

[3] "NSL-KDD dataset," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/nsl.html. [Accessed 06 Aug 2018].

[4] "Intrusion Detection Evaluation Dataset (CICIDS2017)," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: http://www.unb.ca/cic/datasets/ids-2017.html. [Accessed 08 Aug 2018].

[5] *J. P. Mueller and L. Massaron, "Machine Learning For Dummies Cheat Sheet," dummies. [Online]*. Available: https://www.dummies.com/programming/bigdata/data-science/machine-learning-dummies-cheat-sheet/. [Accessed: 14 Aug 2018].

[6] "sklearn.preprocessing.LabelEncoder," *1.4. Support Vector Machines - scikit-learn 0.19.1 documentation.* [Online]. Available: http://scikitlearn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html . [Accessed: 19-Aug-2018].

[7] "train_test_split," *scikit-learn 0.19.1 documentation.* [Online]. Available: http://scikitlearn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. [Accessed: 20-Aug-2018].

[8] *J. Brownlee, "Feature Importance and Feature Selection With XGBoost in Python," Machine Learning Mastery, 10-Mar-2018. [Online]*. Available: https://machinelearningmastery.com/feature-importance-and-feature-selection-withxgboost-in-python/. [Accessed: 19-Aug-2018].

[9] R. Alshammari and A. N. Zincir-Heywood, "A flow based approach for SSH traffic detection," in *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, 2007, pp. 296-301: IEEE.

[10] H. Choi, H. Lee, and H. Kim, "Fast detection and visualization of network attacks on parallel coordinates," *computers & security,* vol. 28, no. 5, pp. 276-288, 2009.

[11] "sklearn.ensemble.RandomForestRegressor," *scikit-learn.* [Online]. Available: http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html. [Accessed: 19-Aug-2018].