# Detection of Application Layer DDoS Attack Using Logistic Regression

MSc Research Project
M.Sc. Cybersecurity

Jerry Devassy
Student ID: x19220961

School of Computing
National College of Ireland

Supervisor: Liam McCabe

# National College of Ireland

# MSc Project Submission Sheet

# School of Computing

| | | | |
|---|---|---|---|
| **Student Name:** | Jerry Jockey Devassy | | |
| **Student ID:** | X19220961 | | |
| **Programme:** | M.Sc. Cybersecurity | **Year:** | 2021-2022 |
| **Module:** | MSc Research Project | | |
| **Supervisor:** | Liam McCabe | | |
| **Submission Due Date:** | 16/12/2021 | | |
| **Project Title:** | Detection of Application Layer DDoS Attack by using logistic Regression | | |
| **Word Count:** | 6063 | **PageCount:** | 21 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

**Signature:** Jerry Jockey Devassy

**Date:** 16th December 2021

**PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST**

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies) | □ |
| **Attach a Moodle submission receipt of the online project submission,** to each project (including multiple copies). | □ |
| **You must ensure that you retain a HARD COPY of the project,** both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | □ |

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Detection of Application Layer DDoS Attack by Using Logistic Regression

Jerry Devassy
X19220961

### Abstract

One of the biggest concerns with respect to online security to many online organizations is a distributed denial of service attack (DDoS). DDoS attacks have been a danger to network security for over a year and will remain tosbe in the foreseeable future. DDoS attacks in an application layer offer a significant issue to Application server these days. The primary goal of a Web server is to provide uninterrupted application layer services to its legit users. However, a DDoS attack in an application layer disrupts the web server's services to its normal customers, resulting in massive losses. Furthermore, performing an application layer DDoS attack takes extremely few resources. The techniques for detecting all sorts of application layer DDoS attacks are quite sophisticated. To develop a framework that would be effective for detecting application layer DDoS attacks for regular user should be that the browsing activity must be simulated in such a way that the legit users and the attacker can be distinguished. In this research, a technique for detecting application layer DDoS attacks that uses feature learning method such as co-relation coefficient to select the best features that are required to improve the efficiency of the model and reduce the size of the dataset. Later logistic regression is used as a classifier to test the model as well as train the model. The model successfully classifies the web traffic based on its nature as normal or attack traffic and after evaluating the model, the prediction percentage that was obtained from testing the dataset was high as compared to the available classification algorithms.

## Table Of Content:

## Table of Figures:

## Table of Tables:

# 1 Introduction

One of the greatest security concerns to online security is a distributed denial of service attack (DDoS), in which a large number of zombie devices overwhelm the web server with huge packets. According to ("Blog- Akamai Technologies," 2021.) there was a 51% rise in application layer attacks in a year from Q4, 2013 to Q4, 2014, and a 16% increase in three months from Q3, 2014 to Q4, 2014. The gravity of the application layer Ddos attack was highlighted in a blog post by ("Cyber Security Leader | Imperva, Inc.," 2021.). DDoS attacks can be launched at any tier of the protocol stack, including OSI and TCP/IP. For example, ARP flooding, ICMP flooding, TCP/UDP flooding, and HTTP flooding attacks are carried out in MAC, Network, Transport, and Application layers respectively. MAC, Network, and Transport layer attacks are launched by exploiting protocol stack or by using the IP spoofing technique. However, in order to get access to application layer services and launch an application layer DDOS attack, the user must establish a valid connection with the web server. Thus, the research question Does the use of Logistic Regression classification method help effectively for HTTP/HTTPs protocol in improving the accuracy while detecting the DDoS Attack in Application Layer?

Because of the increased range of qualities and techniques available to attackers, application layer cyberattacks have efficiently grown increasingly. The ability to identify a DDoS attack for the HTTP/HTTPs protocol is challenging since such attacks might appear to be legal requests at times. Failure to recognize malicious attack can result in the shutdown of services, obtaining database access, and acquiring important data, followed by a ransom demand to fix the situation. The cost of a DDoS attack involves not only financial impact, but also non-financial components such as customer loss, and administrative costs which relates to discover vulnerable networks and correcting the damage. To minimize these losses, organizations should put in place a system that can detect, categorize, and prevent DDoS attack traffic ("Reasons Why Every Business Need DDoS Protection | Indusface Blog," 2019).

## 1.1 Motivation

During a DDoS attack, the targeted machine is so full of external communication requests that it cannot respond to the correct vehicle. Such attacks often result in multiple servers. DDoS attacks are deliberate attempts to force target PCs to reclaim, or use their own resources, such as network bandwidth, bandwidth, and data processing systems; provide the services they need. To combat DDoS, attackers first developed a network of cracked computers used to build a large truck that wanted to deny the victim's legitimate staff. The attacker prepares the attack device inside the hacked part of the network attack. The owner of this attack is called a zombie and can carry out any attack under the control of the attacker. In addition, the attacker adjusts the firearm's network connection to spot the most difficult objects. Many existing systems cannot detect DDoS attacks from legitimate space waves. DDoS attacks are routinely carried out at the network layer. There have been several DDoS attacks on internet services and web applications recently such as the Github Incident in 2018 where the attacker used high traffic that sent 1.3 Tbps of traffic with 126.9 million packets of data each second. The attacker did not use any botnets to attack rather used memcaching method which means that a fake request is sent to a vulnerable server, which then floods the targeted victim with increased traffic. This attack caused the Github system's to be down for 20 minutes since Github used a DDoS mitigation technique which detected the attack and quick steps were taken to reduce the impact ("7 of the Most Famous Recent DDoS Attacks," 2020).

## 1.2 Contribution

Both supervised and unsupervised learning algorithms have been shown to be effective and dependable in detecting DDoS attacks on the web protocol. In this research, logistic regression which is a supervised learning approach based on classification is implemented to improve the speed and accuracy to detect a DDoS attack. The dataset used for this model is the CICIDS 2017 dataset ("IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB," n.d.) which has various DDoS attack traffic. This dataset has 79 total features and has four different DDoS attacks i.e., DoS Slowhttptest, DoS Hulk, DoS Goldeneye, DoS Slowloris. With 79 features present in the dataset, it gives a lot of computing difficulties and hence correlation co-efficient which is a feature selection method is used to select the best features that is required to train and test the model. The logistic regression method is then used to classify whether the incoming traffic is a normal traffic or an attack traffic.

A thorough literature review was undertaken to study and find the best supervised learning algorithm which can be implemented to detect a DDoS attack on the application layer. Guidelines for supervised learning which was utilized to detect DDoS attack traffic were established using previous research on supervised machine learning.

The sections that follows will be discussed in the rest of this article: Section 2 focuses on past research that has been done in a similar manner and compares their findings. Section 3 discusses the approach and methods utilized to develop the model. Section 4 contains the model's design specifications and explains how the model is implemented. Section 5 shows the hardware and software requirements and files that are required to implement the model. Section 6 will evaluate the model's output with various test cases that are implemented. Finally, Section 7 will bring the research to a summary with a conclusion and recommendations for future work.

# 2 Literature Review

A thorough literature study was undertaken to fully understand the context of research on DDoS attack detection and classification using different methods.

## 2.1 Related Work

Many existing programs attempt to detect an attack by considering the header file, the arrival of the packet, and more. Errors are corrected based on changes in IP authentication, such as a combination of IP addresses, TTL, and multiple characters. It has been suggested that there would be a solution where the received IP packets would be dropped if there was a significant difference between the cost of the hop and the cost listed first in the table. Authors (Li et al., 2008), mentioned that the sailor throws a ball at its destination then the victim can attach the

Stackpi logo to the IP address to determine the location of the IP address. In several filters against DDoS attacks (Martins, 2014), the author Martins relies heavily on scans to identify malicious packets. The plan can be adapted to vehicle handovers and efforts to improve efficiency. Authors Zhou et al, discussed in paper that use of integrated circuits to capture vehicle models and determine where and when DDoS attacks can occur. Also, the authors introduced the concept of a Secure Overlay Services (SOS) insurance network that provides efficient traffic(Zhou et al., 2014). SOS networks can modify topology to avoid DDoS and survive some malicious situations. Authors Kun Yang et al, mentioned in paper (Yang et al., 2020) that it is classical artificial neural network which is used for data representation in an unsupervised manner such as images with data present in it. The accuracy found in this method is not great as compared to other feature selection models. In paper (Ni et al., 2013), Authors Ni, discussed that based on the entropy of HTTP GET requests per source IP address, an unique technique to detecting application-layer DDoS attacks is proposed.

Regarding the Cloud DDoS attacks, authors ("Feature Selection Techniques Cloud DDOS Attack Detection," 2019) discussed in the paper that it creates an ideal network traffic feature set for network intrusion detection which gives a set of feature to detect a DDoS attack. Authors (Xie and Yu, 2009) introduced a strategy based on the popularity of the data using an unbalanced measurement method based on Markov's semi-hidden display to detect attacks in paper Large Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviours. The main problem with the Hidden Semi-Markov system is the complexity of the algorithm. In this paper, authors (Gu et al., 2013), has implemented an entropy-based layer-DDoS detection system that determines the number of sites and the same method used to remove click-through identification rates. The inclusion of entropy to say that the extracted identity is necessary to determine the uncertainty. Authors ("Self-similarity based DDoS attack detection using Hurst parameter," 2016) reported data theory-based analysis systems far from the package sent the behaviour of sceptical streams used in various flood attacks through legal entities.

Authors (Mirkovic et al., 2005) have identified the engines of different sites and devised ways to protect the DDoS application layer against the use of human-type behaviours and to target different types of DDoS bots from human users. In this paper, authors (Kandula et al., 2005) reported a system that protects websites from DDoS attacks from CAPTCHA which allows users to access services only. This system assumes that human users can detect distorted images, but the machine does not fail.

## 2.2 DDoS attack detection in cloud environments

The issue of protecting, detecting, and reducing DDoS has received significant attention and demand around the cloud. The problem of the DDoS attack sought to draw the attention of scientists. Scientists around the world continue to work to develop a variety of methods and tools to improve the results of DDoS attacks. Although many requests address the way and means to stop DDoS attacks, unfortunately, even today, the introduction of existing processes cannot prevent DDoS attacks from affecting the cloud environment. In fact, over time, the frequency of attacks and the frequency of attacks increase. One of the most common denominators is a lack of confidence in the final development and distribution because it is not possible to participate worldwide by mistake. The second reason may be the social issues behind international support. The third advantage stems from the nature of DDoS attacks, for example, there is no way to provide access and support to better provide protection against DDoS attacks.

Amazon Web Services reports that the biggest DDoS attack to date occurred in February 2020(Cimpanu, 2020). The maximum number of attacks is 2.3 Tbps. The Security Services Office (CLDAP) server handles the attacker's selection for the attack as an alternative to the LDAP and also scans the registration. Previously a 2.3 Tb/s DDoS outage in February 2020, the second largest DDoS attack was a 1.3 Tb/s DDoS outage - targeting GitHub with 126.9 million packets(“7 of the Most Famous Recent DDoS Attacks,” 2020).

### 2.3 DDoS attack using Supervised Machine Learning

To identify DDoS attacks, numerous supervised learning approaches are used; for example, in paper A system approach to network modelling for DDoS detection using a Naìve Bayesian classifier (Vijayasarathy et al., 2011), authors proposed that the Naïve Bayes machine learning algorithm was used to distinguish the attack data from the benign data. It looked at how important data pre-processing is for different-sized training datasets and feature sets. In paper Proactive detection of DDoS attacks utilizing k-NN classifier in an anti-DDoS framework(Nguyen and Choi, 2009), authors examines and analyses the attack architecture at various stages in order to accurately detect the DDoS attacks and reduce false positives. The examined data is also utilized to draw variables depending on the KNN algorithm's properties. Each part of the assault scenario is therefore built-in accordance within the standards, allowing the attack to be detected early on.

### 2.4 DDoS attack on Software Defined Network (SDN)

Authors investigated in the paper (Chaudhary et al., 2018), that the viability of SDN based on key characteristics that make cloud computing a viable networking solution. They have proposed a unique flow-table sharing strategy for mitigating overwhelming DDoS attacks in the flow table on the SDN-based cloud. They later devised a unique flow-table sharing strategy to protect the flow table against DDoS overloading attempts. Authors proposed a GE-based measure to identify low-rate DDoS attacks on the SDN control layer in the paper An early detection of low rate DDoS attack to SDN based data centre networks using information distance metrics(Sahoo et al., 2018). The traffic attack was identified at the controller using the employed ID. In terms of statistical information distance measures, the results showed greater detection accuracy than the usual Shannon entropy.

## 3 Research Methodology

To detect a DDoS attack in application layer, this research uses a CICIDS 2017 Dataset(“IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB,” 2017) which had 5 different DoS attacks and normal traffic that were captured in it. Since it had such a large number of features it was difficult to compute on the basis of all 79 features that were present in the dataset and hence Correlation Coefficient test was performed to find the best and the most important features. Later Logistic Regression was used to classify from the selected features that if the traffic belonged to normal or attack traffic. The next sections will explain briefly about the entire method.

## 3.1 Data Gathering and Selection

The attackers make use of bot-based technologies to launch DDoS attacks on a number of companies, and these companies refuse to release the log files or any other proof of the attack because of the company's reputation and data security. To understand and develop the dataset for this study, a complete review of well-known DDoS attack simulation tools such as Spybot, SDBot, and others was researched. Although, it is difficult to launch a DDoS attack on a Web application since it requires a well-equipped lab with web servers and data servers(“Quick Guide: Simulating a DDoS Attack in Your Own Lab,” 2021).

The dataset used for this research is obtained from the CICIDS(“IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB,” 2017) which has a total of 692703 values and 79 different features. Due to such large dataset computing the whole dataset was taking a long period of time and hence, 6000 values were taken randomly to train the model and 4000 random values were taken to test the model. The model would be evaluated based on the 4 different DoS attacks shown in Figure 1 were captured in the dataset i.e., DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS Slowhttptest and also the normal traffic (BENIGN) present in the dataset.

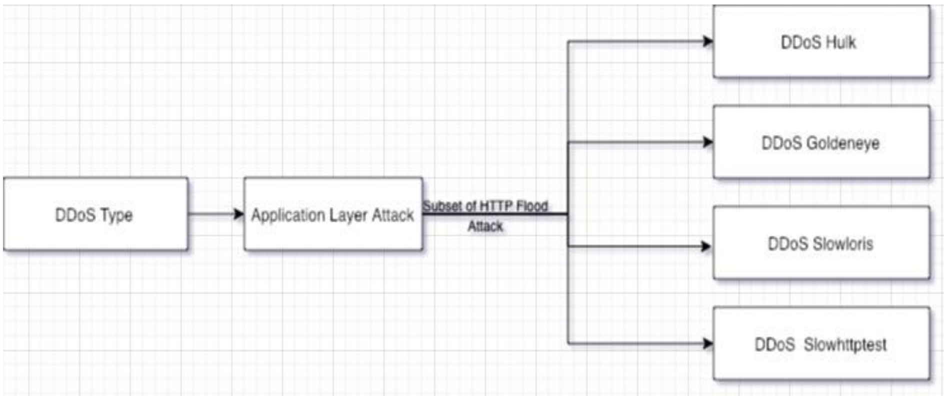


*Figure 1: DDoS Dataset Flow*

In Table 1, the count of traffic that was used to train and test the model is shown. The values selected in this were randomly selected.

| Attack Type | Count of Traffic |
|---|---|
| DDoS Hulk | 2500 |
| DDoS Goldeneye | 1500 |
| DDoS Slowloris | 500 |
| DDoS SlowHttptest | 500 |
| BENIGN (Legit traffic) | 5000 |

*Table 1: Dataset Records used for training and testing*

## 3.2 Data Pre-Processing

After successfully gathering the data, the whole data was stored in a CSV format which had normal traffic (BENIGN) and attack traffic (DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS slowHttptest). This was not enough as many had infinite values and NaaN (Not a Number) values present in it which could give inconsistent output and therefore efforts were made to solve the issue by using the python code. All the infinite values were rounded upto the maximum length of datatype and NaaN values were dropped successfully.

## 3.3 Data Encoding

Encoding categorical variables in machine learning is critical to ensure that the algorithm does not cluster similar results or entries near to one another in one branch while being trained. As with this architecture, encoding is achieved through the use of One-Hot Encoding. One-Hot Encoding was selected over Label Encoding because Label Encoding encodes the data and offers a grading system between the various values(Brownlee, 2020). The information regarding encoding values in the dataset is highlighted in the table 2.

| Dataset Type | Values |
|---|---|
| DDoS Hulk and BENIGN traffic | BENIGN - 0<br>DDoS Hulk - 1 |
| DDoS Goldeneye and BENIGN traffic | BENIGN - 0<br>DDoS Goldeneye - 1 |
| DDoS Slowloris and BENIGN traffic | BENIGN - 0<br>DDoS Slowloris - 1 |
| DDoS SlowHttptest and BENIGN traffic | BENIGN - 0<br>DDoS SlowHttptest -1 |

*Table 2: Information of Data Encoding*

## 3.4 Feature Learning

A correlation coefficient test was performed using Python code to examine the relationship between the various characteristics as well as how each feature would complement the others and the results are displayed in figure 3(Fern and O, 2021).

The result shows the positive and negative correlations between the variables, helping us to better understand their relationship. Because the number of variables employed in the correlation study was large, a separate function in Python was written to export the characteristics that had a strong link with the target variable 'Label.' The filtering threshold for these characteristics was set at 0.99. Table 3 lists the important features that are discovered using correlation analysis:

| Feature Name | Average Packet Size | Avg Bwd Segment Size | Avg Fwd Segment Size | Bwd Header Length | Fwd Header Length | Fwd IAT Max |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| Date Type | Float 64 | Float 64 | Float 64 | Int 64 | Int 64 | Int 64 |

| Feature Name | Idle Max | Idle Min | Idle Mean | Subflow Bwd Bytes | Subflow Bwd Packets | Subflow Fwd Bytes |
|---|---|---|---|---|---|---|
| Date Type | Int 64 | Int 64 | Int 64 | Int 64 | Int 64 | Int 64 |

| Feature Name | Total Backward Packets | Total Length of Bwd Packets | act_data_pkt_fwd | Fwd IAT Total | Fwd Packets/s | Subflow Fwd Packets |
|---|---|---|---|---|---|---|
| Date Type | Int 64 | Int 64 | Int 64 | Int 64 | Float 64 | Int 64 |

*Table 3: Extracted Features from Dataset*

### 3.5 Training and Testing of Data

Logistic regression, like other regression models is a predictive analysis in which the Logistic regression is used to describe data and explain how characteristics and classes are related.

The logistic regression Classifier is then used to train the encoded training subsets and after this, the training is done in batches which helps to keep track of the outcomes for each subgroup. The data is divided in a 70-30 format where 70% data is used for training the model and 30% data is used for testing. The output is generated in the form of a confusion matrix, from which we may calculate the model's accuracy.

## 4 Design Specification

This section discusses the structure of the constructed model. This section serves as a comprehensive tool for categorizing data flow depending on its type. The model may accept CSV files as inputs, which are then utilized for data preparation and feature selection using correlation coefficient analysis. After that, the retrieved features are encoded, and a subset of the dataset is generated. The freshly produced dataset is then used to classify the test data.

Figure 2 depicts the whole architecture of the model. CICIDS 2017 was chosen as the dataset because it comprises online traffic recorded in the form of a PCAP file and translated into a CSV file. Before splitting the dataset into subsets, it is subjected to feature extraction against the target variable 'Label,' because all of the attack groups have the same online traffic attributes. Correlation analysis is used to extract the characteristics, following which the dataset is separated into four subsets:

1) Benign and DDoS Hulk Dataset

2) Benign DDoS Goldeneye Dataset

3) Benign and DDoS Slowloris Dataset

4) Benign and DDOS SlowHttptest Dataset

The values of the target feature are encoded and then provided to the model for training.

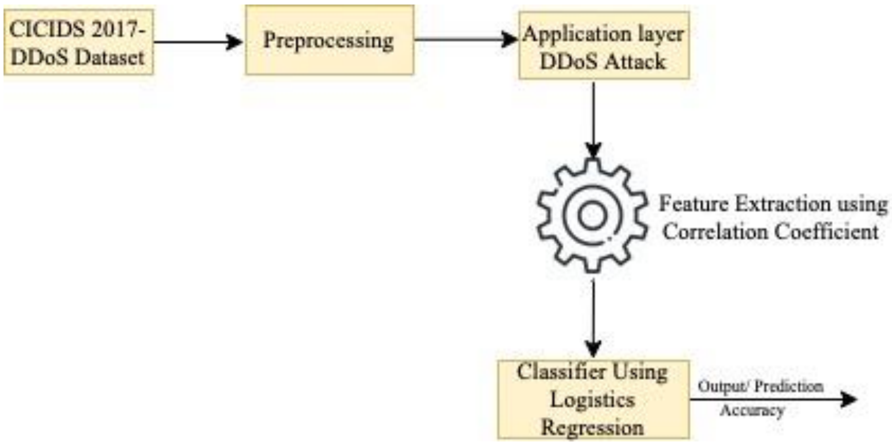


*Figure 2: Architecture of DDoS Attack*

Logistic regression separates labelled datasets into X and Y, with X having all independent factors and Y comprising the dependent binary variable to be predicted. When separating the train and test data, the test split is set to 30% of the total data, with the remaining 70% utilized for training. The accuracy of the model is calculated using a confusion matrix and an actual vs predicted matrix, both of which are generated as outputs.

### 4.1 Accuracy:

Accuracy is a statistical measuring scale used to evaluate a model. It is assumed to be the number of values that a model can efficiently calculate. It is possible to compute it as follows:

Accuracy = (True Positive + True Negative/ (True Positive + False Positive + False Negative)

True positives reflect true values, and the model is also true. In reality, negative values are represented by true negative values, and the model will likewise be predicted to be negative. False positive is the total of all truly negative values, and the model is true. False negative is the sum of true values, and the model will be predicted to be negative.

## 5 Implementation

This section will go through the steps taken to put the suggested concept into action and will showcase the hardware and software utilized, as well as the code structure.

### 5.1 Hardware Requirements

MacBook Air laptop was used to build the model which had the following hardware specifications:

- Processor: 1.1GHz dual-core Intel Core i3, Turbo Boost up to 3.2GHz, with 4MB L3 cache
- Memory: 8GB of 3733MHz LPDDR4X onboard memory
- Storage: 256GB PCIe-based SSD
- Graphics: Intel Iris Plus Graphics

## 5.2 Software Requirements

Software's that was used to complete this model in which data gathering and compilation was carried out. The model was built specifically on Mac OS.

- Development environment: Jupyter Notebook
- Development language: Python
- Libraries used: Pandas, Numpy, Seaborn, Sklearn, Os, LogisticRegression, Matplotlib

## 5.3 Data Files

**.ipynb:** This file contains the whole code required to create the model. Python 3 was used for the coding, which was done in juypter notebook.

**Dataset:** Various dataset were used while building this model and table 4 discusses the details of the dataset along with the description of when the dataset was used.

| Dataset File Name | Dataset Size (Row Count) | Stage in which Used |
|---|---|---|
| Dataset_DDoS | 225.9 MB (692703) | Feature Selection |
| Dataset_DDoS_10kvalues | 862 KB (10000) | Dataset sent to logistic regression for training and testing the model which is derived from the features that are selected |
| DoS Hulk-Benign_1kData | 65 KB (1000) | Dataset used for testing the model |
| DoS Goldeneye-Benign_1kData | 90 KB (1000) | Dataset used for testing the model |
| DoS Slowloris-Benign_1kData | 68 KB (1000) | Dataset used for testing the model |
| DoS SlowHttp-Benign_1kData | 70 KB (1000) | Dataset used for testing the model |

*Table 4: Datasets Used in Model Lifecycle*

## 5.4 Package Installation

To carry out the research, the following packages and libraries are installed:

- Pandas: Used to read the dataset
- Numpy: Used for array operation
- Metrics: Used to calculate and print the accuracy
- LogisticRegression: Used to train the model and classify the data

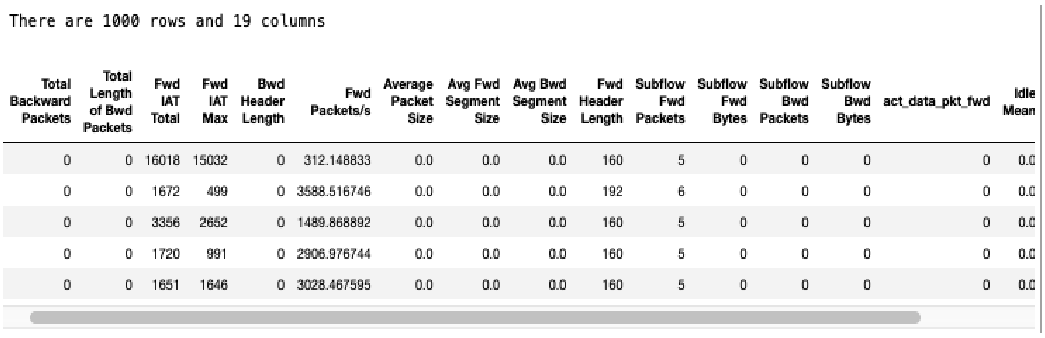
# 6 Evaluation

To assess the model's efficiency, four test scenarios were developed against which the model would be evaluated using the datasets and DDoS attack types. In the testing dataset, new values were selected randomly as compared to the training dataset. Along with the test situations, the results/observations are documented.

## 6.1 Model is tested using new dataset of DDoS attack - Hulk

- **State of the model**: The model was already trained on all four attacks (DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS Slowhttptest) and Benign data which was used to train the model.
- **Test scenario**: To test the model's accuracy, the same dataset of the same DDoS attack type– Hulk is used.
- **Classification model**: For testing, the model which was created earlier “model_lg” is used for classifying the legit and attack traffic from the dataset.
- **Dataset Name:** DoS Hulk-Benign_1kData.csv
- **Count of Web traffic used for testing:** 1,000
- **Result:** Random subset was created for testing and both benign and DDoS Hulk traffic is used to test the model. Figure 3 shows the dataset that includes 1000 traffic count having both legit and attack traffic. Since untrained data is sent to the model, the attack DDoS Hulk has features that are related and similar to other DDoS attacks which confuses the model to classify if the traffic is an legit or attack traffic and hence the accuracy score dropped to 79.6%. Figure 4 shows the accuracy score of the model obtained and gives the comparison of Actual or Predicted values.

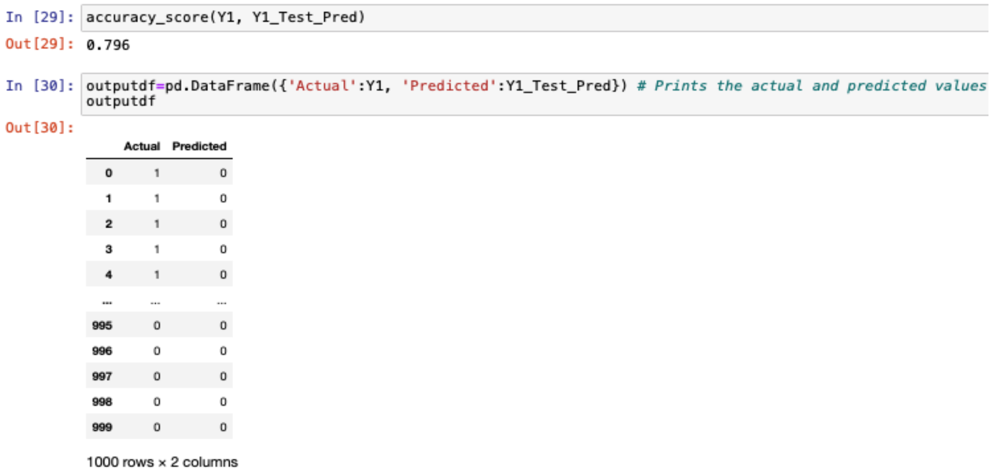
  Even then the model predicted an overall of 80% of the traffic as DDoS Hulk attack traffic as shown in the confusion matrix in Figure 5.



```
There are 1000 rows and 19 columns
```

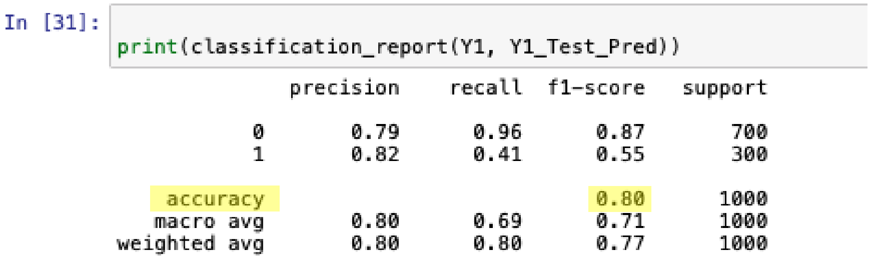
| Total Backward Packets | Total Length of Bwd Packets | Fwd IAT Total | Fwd IAT Max | Bwd Header Length | Fwd Packets/s | Average Packet Size | Avg Fwd Segment Size | Avg Bwd Segment Size | Fwd Header Length | Subflow Fwd Packets | Subflow Fwd Bytes | Subflow Bwd Packets | Subflow Bwd Bytes | act_data_pkt_fwd | Idle Mean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 16018 | 15032 | 0 | 312.148833 | 0.0 | 0.0 | 0.0 | 160 | 5 | 0 | 0 | 0 | 0 | 0.0 |
| 0 | 0 | 1672 | 499 | 0 | 3588.516746 | 0.0 | 0.0 | 0.0 | 192 | 6 | 0 | 0 | 0 | 0 | 0.0 |
| 0 | 0 | 3356 | 2652 | 0 | 1489.868892 | 0.0 | 0.0 | 0.0 | 160 | 5 | 0 | 0 | 0 | 0 | 0.0 |
| 0 | 0 | 1720 | 991 | 0 | 2906.976744 | 0.0 | 0.0 | 0.0 | 160 | 5 | 0 | 0 | 0 | 0 | 0.0 |
| 0 | 0 | 1651 | 1646 | 0 | 3028.467595 | 0.0 | 0.0 | 0.0 | 160 | 5 | 0 | 0 | 0 | 0 | 0.0 |

*Figure 3: Test Case 1 Dataset- Hulk DDoS*



```
In [29]: accuracy_score(Y1, Y1_Test_Pred)
Out[29]: 0.796

In [30]: outputdf=pd.DataFrame({'Actual':Y1, 'Predicted':Y1_Test_Pred}) # Prints the actual and predicted values
         outputdf
Out[30]:
            Actual  Predicted
       0         1          0
       1         1          0
       2         1          0
       3         1          0
       4         1          0
     ...       ...        ...
     995         0          0
     996         0          0
     997         0          0
     998         0          0
     999         0          0

       1000 rows × 2 columns
```

*Figure 4: Actual v/s Predicted Score*



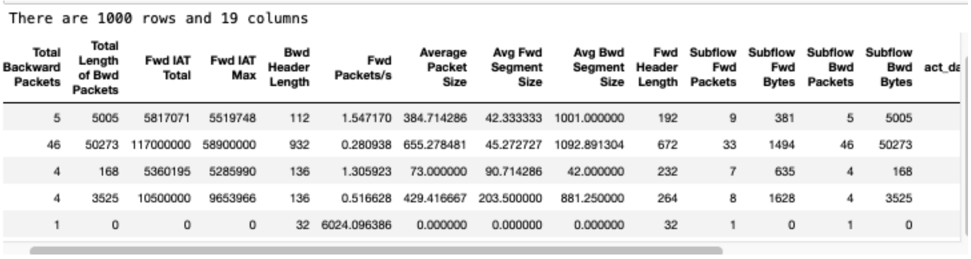
```
In [31]: print(classification_report(Y1, Y1_Test_Pred))

                  precision    recall  f1-score   support

               0       0.79      0.96      0.87       700
               1       0.82      0.41      0.55       300

        accuracy                           0.80      1000
       macro avg       0.80      0.69      0.71      1000
    weighted avg       0.80      0.80      0.77      1000
```

*Figure 5: Confusion matrix*

## 6.2 Model is tested using new dataset of DDoS attack - Goldeneye

- **State of the model**: The model was already trained on all four attacks (DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS Slowhttptest) and Benign data which was used to train the model.
- **Test scenario**: To test the model's accuracy, the same dataset of the same DDoS attack type– Goldeneye is used
- **Classification model**: For testing, the model which was created earlier “model_lg” is used for classifying the legit and attack traffic from the dataset.
- **Dataset Name:** DoS GoldenEye-Benign_1kData.csv
- **Count of Web traffic used for testing:** 1,000
- **Result:** Random subset was created for testing and both benign and DDoS Goldeneye traffic is used to test the model. Figure 6 shows the dataset that includes 1000 traffic count having both legit and attack traffic. The test produced the accuracy of 98.5% which detected most of the attack traffic that was tested in it from the model. Figure 7 shows the accuracy score and the Actual v/s Predicted values obtained from the model.

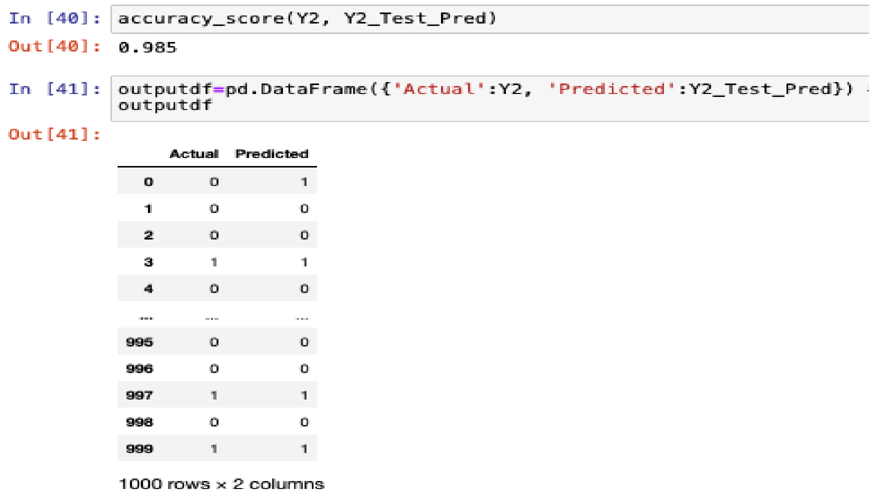
  The model predicted an overall of 98% of the traffic as DDoS Goldeneye attack traffic as shown in the confusion matrix in Figure 8.



```
There are 1000 rows and 19 columns
```

| Total Backward Packets | Total Length of Bwd Packets | Fwd IAT Total | Fwd IAT Max | Bwd Header Length | Fwd Packets/s | Average Packet Size | Avg Fwd Segment Size | Avg Bwd Segment Size | Fwd Header Length | Subflow Fwd Packets | Subflow Fwd Bytes | Subflow Bwd Packets | Subflow Bwd Bytes | act_da |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 | 5005 | 5817071 | 5519748 | 112 | 1.547170 | 384.714286 | 42.333333 | 1001.000000 | 192 | 9 | 381 | 5 | 5005 | |
| 46 | 50273 | 117000000 | 58900000 | 932 | 0.280938 | 655.278481 | 45.272727 | 1092.891304 | 672 | 33 | 1494 | 46 | 50273 | |
| 4 | 168 | 5360195 | 5285990 | 136 | 1.305923 | 73.000000 | 90.714286 | 42.000000 | 232 | 7 | 635 | 4 | 168 | |
| 4 | 3525 | 10500000 | 9653966 | 136 | 0.516628 | 429.416667 | 203.500000 | 881.250000 | 264 | 8 | 1628 | 4 | 3525 | |
| 1 | 0 | 0 | 0 | 32 | 6024.096386 | 0.000000 | 0.000000 | 0.000000 | 32 | 1 | 0 | 1 | 0 | |

*Figure 6: Test Case 2 Dataset- Goldeneye DDoS*

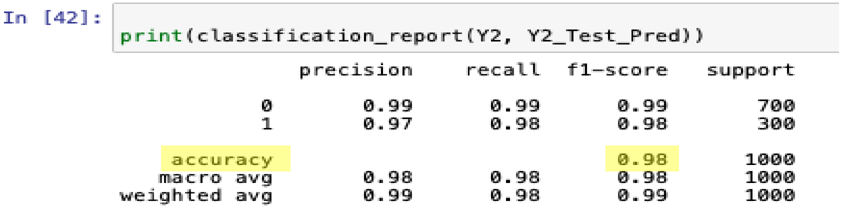


```
In [40]: accuracy_score(Y2, Y2_Test_Pred)
Out[40]: 0.985

In [41]: outputdf=pd.DataFrame({'Actual':Y2, 'Predicted':Y2_Test_Pred}) -
         outputdf

Out[41]:
```

| | Actual | Predicted |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |
| 2 | 0 | 0 |
| 3 | 1 | 1 |
| 4 | 0 | 0 |
| ... | ... | ... |
| 995 | 0 | 0 |
| 996 | 0 | 0 |
| 997 | 1 | 1 |
| 998 | 0 | 0 |
| 999 | 1 | 1 |

1000 rows × 2 columns

*Figure 7: Actual v/s Predicted Score*



```
In [42]: print(classification_report(Y2, Y2_Test_Pred))
                  precision    recall  f1-score   support

               0       0.99      0.99      0.99       700
               1       0.97      0.98      0.98       300

        accuracy                           0.98      1000
       macro avg       0.98      0.98      0.98      1000
    weighted avg       0.99      0.98      0.99      1000
```

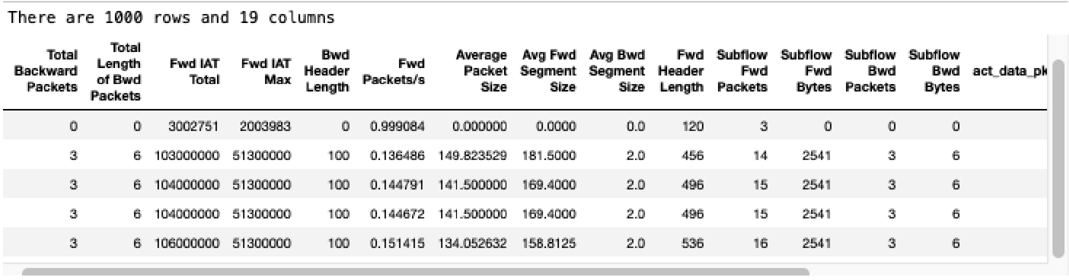
*Figure 8: Confusion Matrix*

## 6.3 Model is tested using new dataset of DDoS attack - Slowloris

- **State of the model**: The model was already trained on all four attacks (DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS Slowhttptest) and Benign data which was used to train the model.
- **Test scenario**: To test the model's accuracy, the same dataset of the same DDoS attack type– Slowloris is used
- **Classification model**: For testing, the model which was created earlier “model_lg” is used for classifying the legit and attack traffic from the dataset.
- **Dataset Name:** DoS Slowloris-Benign_1kData.csv
- **Count of Web traffic used for testing:** 1,000

- **Result:** Random subset was created for testing and both benign and DDoS Slowloris traffic is used to test the model. Figure 9 shows the dataset that includes 1000 traffic count having both legit and attack traffic. The output of the test gave an accuracy of 90.3% which detected most of the attack traffic that was tested in it from the model. Figure 10 shows the accuracy score and the Actual v/s Predicted values obtained from the model.

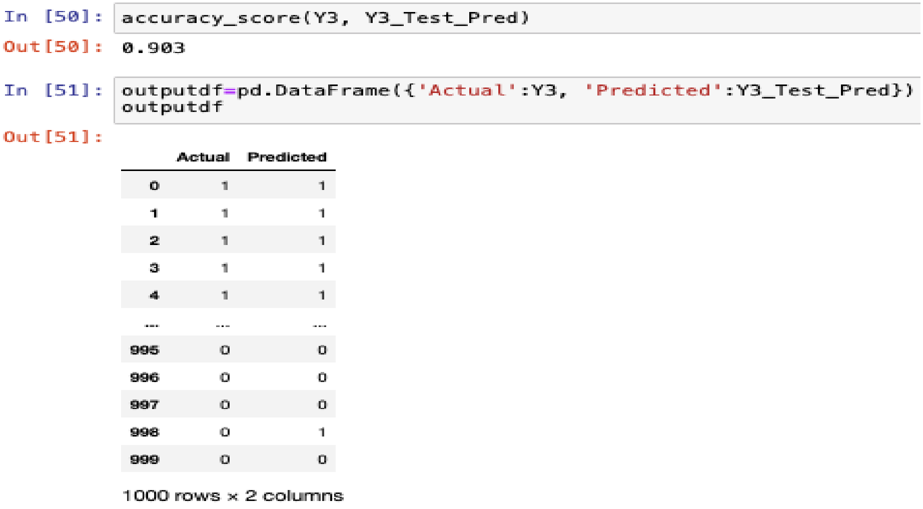
  The model predicted an overall of 90% of the traffic as DDoS Slowloris attack traffic as shown in the confusion matrix in Figure 11. The precision in Slowloris was dropped to 76% but the f1 score predicted was 90% as the recall value predicted was 100%



```
There are 1000 rows and 19 columns
```

| Total Backward Packets | Total Length of Bwd Packets | Fwd IAT Total | Fwd IAT Max | Bwd Header Length | Fwd Packets/s | Average Packet Size | Avg Fwd Segment Size | Avg Bwd Segment Size | Fwd Header Length | Subflow Fwd Packets | Subflow Fwd Bytes | Subflow Bwd Packets | Subflow Bwd Bytes | act_data_pk |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 3002751 | 2003983 | 0 | 0.999084 | 0.000000 | 0.0000 | 0.0 | 120 | 3 | 0 | 0 | 0 | |
| 3 | 6 | 103000000 | 51300000 | 100 | 0.136486 | 149.823529 | 181.5000 | 2.0 | 456 | 14 | 2541 | 3 | 6 | |
| 3 | 6 | 104000000 | 51300000 | 100 | 0.144791 | 141.500000 | 169.4000 | 2.0 | 496 | 15 | 2541 | 3 | 6 | |
| 3 | 6 | 104000000 | 51300000 | 100 | 0.144672 | 141.500000 | 169.4000 | 2.0 | 496 | 15 | 2541 | 3 | 6 | |
| 3 | 6 | 106000000 | 51300000 | 100 | 0.151415 | 134.052632 | 158.8125 | 2.0 | 536 | 16 | 2541 | 3 | 6 | |

*Figure 9: Test Case 3 Dataset- Slowloris DDoS*

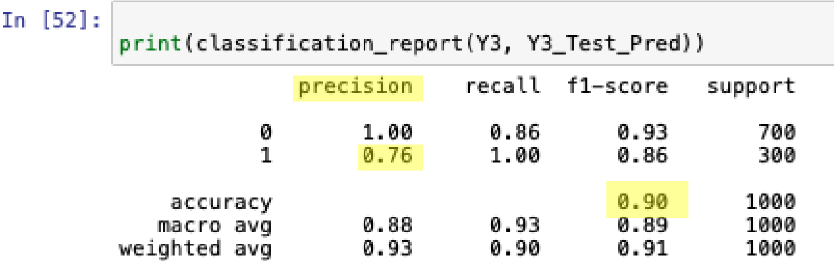


```
In [50]: accuracy_score(Y3, Y3_Test_Pred)
Out[50]: 0.903

In [51]: outputdf=pd.DataFrame({'Actual':Y3, 'Predicted':Y3_Test_Pred})
         outputdf
Out[51]:
```

| | Actual | Predicted |
|---|---|---|
| 0 | 1 | 1 |
| 1 | 1 | 1 |
| 2 | 1 | 1 |
| 3 | 1 | 1 |
| 4 | 1 | 1 |
| ... | ... | ... |
| 995 | 0 | 0 |
| 996 | 0 | 0 |
| 997 | 0 | 0 |
| 998 | 0 | 1 |
| 999 | 0 | 0 |

1000 rows × 2 columns

*Figure 10: Actual v/s Predicted Score*



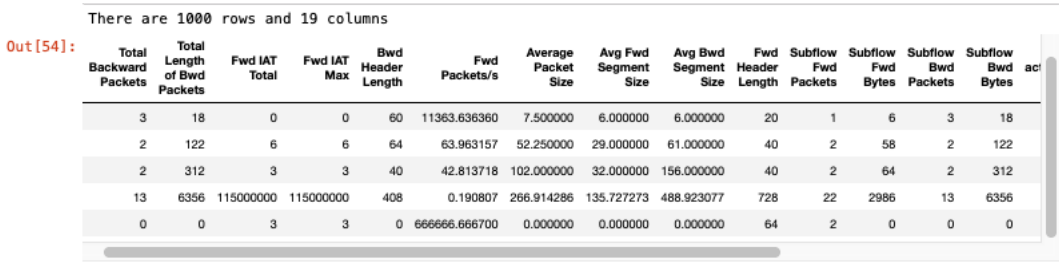
```
In [52]:
         print(classification_report(Y3, Y3_Test_Pred))

                       precision    recall  f1-score   support

                  0         1.00      0.86      0.93       700
                  1         0.76      1.00      0.86       300

           accuracy                             0.90      1000
          macro avg         0.88      0.93      0.89      1000
       weighted avg         0.93      0.90      0.91      1000
```

*Figure 11: Confusion Matrix*

## 6.4 Model is tested using new dataset of DDoS attack - SlowHttptest

- **State of the model**: The model was already trained on all four attacks (DDoS Hulk, DDoS Goldeneye, DDoS Slowloris, DDoS Slowhttptest) and Benign data which was used to train the model.
- **Test scenario**: To test the model's accuracy, the same dataset of the same DDoS attack type– SlowHttptest is used.
- **Classification model**: For testing, the model which was created earlier “model_lg” is used for classifying the legit and attack traffic from the dataset.
- **Dataset Name:** DoS SlowHttp-Benign_1kData.csv
- **Count of Web traffic used for testing:** 1,000
- **Result:** Random subset was created for testing and both benign and DDoS SlowHttptest traffic is used to test the model. Figure 12 shows the dataset that includes 1000 traffic count having both legit and attack traffic. SlowHttptest attack gave a accuracy of 91.9% which detected most of the attack traffic that was tested in it from the model. Figure 13 shows the accuracy score and the Actual v/s Predicted values obtained from the model.

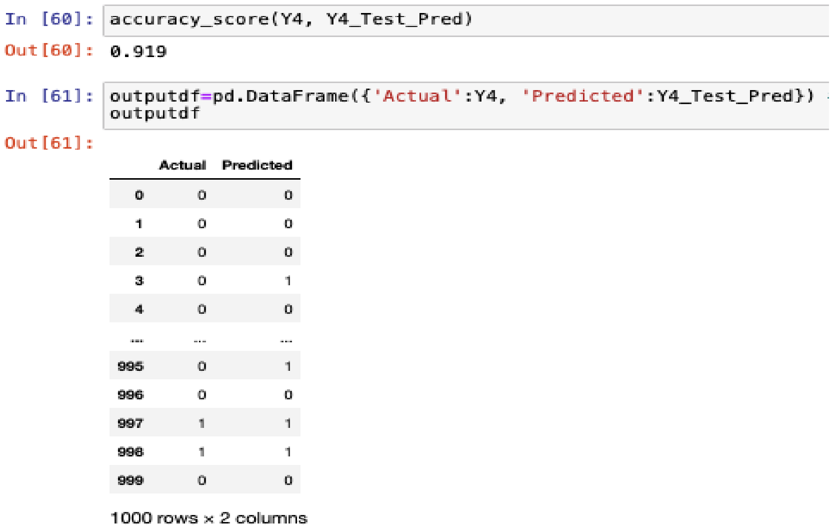
  The model predicted an overall of 92% of the traffic as DDoS SlowHttptest attack traffic as shown in the confusion matrix in Figure 14.



```
There are 1000 rows and 19 columns
Out[54]:
```

| Total Backward Packets | Total Length of Bwd Packets | Fwd IAT Total | Fwd IAT Max | Bwd Header Length | Fwd Packets/s | Average Packet Size | Avg Fwd Segment Size | Avg Bwd Segment Size | Fwd Header Length | Subflow Fwd Packets | Subflow Fwd Bytes | Subflow Bwd Packets | Subflow Bwd Bytes | ac |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 18 | 0 | 0 | 60 | 11363.636360 | 7.500000 | 6.000000 | 6.000000 | 20 | 1 | 6 | 3 | 18 | |
| 2 | 122 | 6 | 6 | 64 | 63.963157 | 52.250000 | 29.000000 | 61.000000 | 40 | 2 | 58 | 2 | 122 | |
| 2 | 312 | 3 | 3 | 40 | 42.813718 | 102.000000 | 32.000000 | 156.000000 | 40 | 2 | 64 | 2 | 312 | |
| 13 | 6356 | 115000000 | 115000000 | 408 | 0.190807 | 266.914286 | 135.727273 | 488.923077 | 728 | 22 | 2986 | 13 | 6356 | |
| 0 | 0 | 3 | 3 | 0 | 666666.666700 | 0.000000 | 0.000000 | 0.000000 | 64 | 2 | 0 | 0 | 0 | |

*Figure 12: Test Case 4 Dataset- SlowHttptest DDoS*

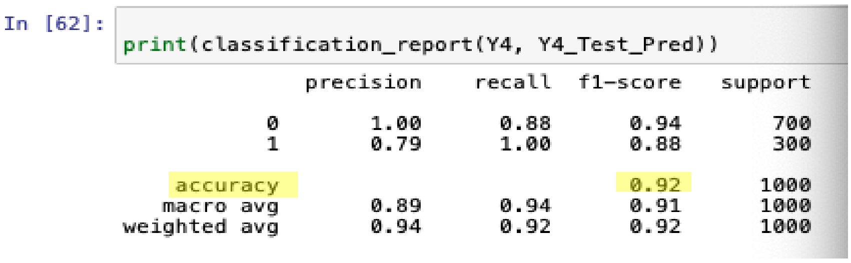


```
In [60]: accuracy_score(Y4, Y4_Test_Pred)
Out[60]: 0.919

In [61]: outputdf=pd.DataFrame({'Actual':Y4, 'Predicted':Y4_Test_Pred})
         outputdf
Out[61]:
```

| | Actual | Predicted |
|---|---|---|
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 2 | 0 | 0 |
| 3 | 0 | 1 |
| 4 | 0 | 0 |
| ... | ... | ... |
| 995 | 0 | 1 |
| 996 | 0 | 0 |
| 997 | 1 | 1 |
| 998 | 1 | 1 |
| 999 | 0 | 0 |

1000 rows × 2 columns

*Figure 13: Actual v/s Predicted Score*



```
In [62]: print(classification_report(Y4, Y4_Test_Pred))
              precision    recall  f1-score   support

           0       1.00      0.88      0.94       700
           1       0.79      1.00      0.88       300

    accuracy                           0.92      1000
   macro avg       0.89      0.94      0.91      1000
weighted avg       0.94      0.92      0.92      1000
```

*Figure 14: Confusion Matrix*

## 6.5 Summary

It was discovered that after training the model with several datasets and evaluating its efficiency, the model had an average prediction accuracy of 85% across all four test scenarios, with the lowest prediction accuracy rate of 80% and the highest prediction accuracy rate of 98%. The conclusion reached after examining the test cases was that, while the size of the dataset has minimal effect on accuracy, the kind of attack used by the dataset can be accurately predicted. Also, it can be concluded that while creating the dataset’s subset, the value which decides if the traffic is an attack or not should not be labelled together as the accuracy drops and if the values are shuffled randomly, it improves the accuracy of the model which also gives accurate data that the model is trained properly.

| **Test Case** | **Dataset on which model is trained** | **Testing done on Untrained Dataset** | **Accuracy Obtained** |
|---|---|---|---|
| Test Case 1 | DDoS Hulk | DDoS Hulk | 80% |
| Test Case 2 | DDoS GoldenEye | DDoS GoldenEye | 98% |
| Test Case 3 | DDoS Slowloris | DDoS Slowloris | 90% |
| Test Case 4 | DDoS SlowHttptest | DDoS SlowHttptest | 92% |

*Table 5: Evaluation of models implemented*

# 7 Conclusion and Future Work

The findings and evaluation show that the stated hypothesis of using logistic regression to identify DDoS attack traffic from legit traffic at the application layer is accurate. After using a variety of DDoS attacks for testing, maximum accuracy obtained by the model is 98%. The outcome of the research implies that the quantity of the dataset has no bearing on the model's accuracy prediction. It is also crucial to think about the kind and type of attack dataset that the model is given. Finally, predicting DDoS attack traffic using Correlation Coefficient for feature selection and logistic regression for classification is successful to a great extent.

### Future Work:

Due to time constraints and technology limitations, there could be a possibility to create a dataset for DDoS attacks by simulating the attack traffic using some tools on a web application and capturing the traffic using Wireshark. As the traffic that would be captured is newly created, it could be sceptical to predict the model's ability in finding the accuracy of the model. In the future, the use of larger, real-time datasets with a higher number of characteristics could be implemented to test the model's capabilities against other cyber-attacks, such as Malware or phishing attempts.

# References

7 of the Most Famous Recent DDoS Attacks [WWW Document], n.d. URL https://www.vxchnge.com/blog/recent-ddos-attacks-on-companies (accessed 12.13.21).

Blog [WWW Document], n.d. URL https://www.akamai.com/blog (accessed 12.11.21).

Brownlee, J., 2020. Ordinal and One-Hot Encodings for Categorical Data. Machine Learning Mastery. URL https://machinelearningmastery.com/one-hot-encoding-for-categorical-data/ (accessed 12.1.21).

Chaudhary, D., Bhushan, K., Gupta, B.B., 2018. Survey on DDoS Attacks and Defense Mechanisms in Cloud and Fog Computing. International Journal of E-Services and Mobile Applications (IJESMA) 10, 61–83.

Cimpanu, C., n.d. AWS said it mitigated a 2.3 Tbps DDoS attack, the largest ever [WWW Document]. ZDNet. URL https://www.zdnet.com/article/aws-said-it-mitigated-a-2-3-tbps-ddos-attack-the-largest-ever/ (accessed 11.21.21).

Cyber Security Leader | Imperva, Inc. [WWW Document], n.d. . Imperva. URL https://www.imperva.com/ (accessed 11.3.21).

DDoS attacks in Q3 2018 | Securelist [WWW Document], 2018. URL https://securelist.com/ddos-report-in-q3-2018/88617/ (accessed 11.17.21).

Feature Selection Techniques Cloud DDOS Attack Detection, 2019. . IJITEE 8, 1257–1260. https://doi.org/10.35940/ijitee.L3908.1081219

Feb 25, D.K.•, 2018, 2018. Kaspersky Lab Study: Average Cost of Enterprise DDoS Attack Totals $2M [WWW Document]. MSSP Alert. URL https://www.msspalert.com/cybersecurity-research/kaspersky-lab-study-average-cost-of-enterprise-ddos-attack-totals-2m/ (accessed 11.11.21).

Fern, J., O, n.d. Correlation Coefficient Definition [WWW Document]. Investopedia. URL https://www.investopedia.com/terms/c/correlationcoefficient.asp (accessed 11.12.21).

Gu, X., Wang, H., Ni, T., Ding, H., 2013. Detection of application-layer DDoS attack based on time series analysis: Detection of application-layer DDoS attack based on time series analysis. Journal of Computer Applications 33, 2228–2231. https://doi.org/10.3724/SP.J.1087.2013.02228

IDS 2017 | Datasets | Research | Canadian Institute for Cybersecurity | UNB [WWW Document], 2017. URL https://www.unb.ca/cic/datasets/ids-2017.html (accessed 12.9.21).

Kandula, S., Katabi, D., Jacob, M., Berger, A., 2005. Botz-4-Sale: Surviving Organized DDoS Attacks That Mimic Flash Crowds (Awarded Best Student Paper).

Li, Z.-L., Hu, G.-M., Yang, D., 2008. Global abnormal correlation analysis for DDoS attack detection. https://doi.org/10.1109/ISCC.2008.4625614

Martins, B.C., 2014. OVERMUNDO: UM CASO DE AUTORIA PEER-TO-PEER. P2P E INOVAÇÃO 1, 91–103. https://doi.org/10.21721/p2p.2014v1n1.p91-103

Mirkovic, J., Robinson, M., Reiher, P., Oikonomou, G., 2005. Distributed Defense Against DDOS Attacks.

Nguyen, H.-V., Choi, Y., 2009. Proactive detection of DDoS attacks utilizing k-NN classifier in an anti-DDoS framework. World Academy of Science, Engineering and Technology 39, 640–645.

Ni, T., Gu, X., Wang, H., Li, Y., 2013. Real-Time Detection of Application-Layer DDoS Attack Using Time Series Analysis. Journal of Control Science and Engineering 2013. https://doi.org/10.1155/2013/821315

Quick Guide: Simulating a DDoS Attack in Your Own Lab, 2021. 9. https://www.keysight.com/us/en/assets/7019-0414/technical-overviews/Simulating-a-DDoS-Attack-in-Your-Own-Lab.pdf (accessed 11.28.21).

Reasons Why Every Business Need DDoS Protection | Indusface Blog, 2019. . Indusface. URL https://www.indusface.com/blog/reasons-why-business-need-ddos-protection/ (accessed 11.30.21).

Sahoo, K.S., Puthal, D., Tiwary, M., Rodrigues, J.J.P.C., Sahoo, B., Dash, R., 2018. An early detection of low rate DDoS attack to SDN based data center networks using information distance metrics. Future Generation Computer Systems 89, 685–697. https://doi.org/10.1016/j.future.2018.07.017

Self-similarity based DDoS attack detection using Hurst parameter, n.d. https://doi.org/10.1002/sec.1639

Survey: DDoS Attacks Cause Loss of Customer Trust & Decreased Revenues, 2016. . Corero. URL https://corero.com/survey-ddos-attacks-cause-loss-of-customer-trust-decreased-revenues/ (accessed 12.10.21).

Vijayasarathy, R., Raghavan, S.V., Ravindran, B., 2011. A system approach to network modeling for DDoS detection using a Naìve Bayesian classifier, in: 2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011). Presented at the 2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011), pp. 1–10. https://doi.org/10.1109/COMSNETS.2011.5716474

Xie, Y., Yu, S.-Z., 2009. A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors. IEEE/ACM Transactions on Networking 17, 54–65. https://doi.org/10.1109/TNET.2008.923716

Yang, K., Zhang, J., Xu, Y., Chao, J., 2020. DDoS Attacks Detection with AutoEncoder, in: NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium. Presented at the NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium, pp. 1–9. https://doi.org/10.1109/NOMS47738.2020.9110372

Zhou, Wei, Jia, W., Wen, S., Xiang, Y., Zhou, Wanlei, 2014. Detection and defense of application-layer DDoS attacks in backbone web traffic. Future Generation Computer Systems 38, 36–46. https://doi.org/10.1016/j.future.2013.08.002