Al-Imam Mohammad ibn Saud Islamic University
College of Computer and Information Sciences
Computer Science Department

CS364: Machine Learning
2nd Semester 1443H / Spring 2022

Course Project

Name	ID	Email
Sarah Khalid Alaradi	440023365	skmaloridi@sm.imamu.edu.sa
Shoug Ali Alsuhaibani	440022732	ssuhaibani@sm.imamu.edu.sa

Section: 371

Supervised by: Dr. Wojdan BinSaeedan

Date of Submission 14/05/2022

Table of Contents

What is a DDoS? **3**
Implementation 3
Dataset 3
Work Environment 3
Machine Learning Technique 4
Libraries 5
Features 5
Notebook 5
Results 6
Resources **9**
References **9**

Table of Figures and Tables

Figure 1. DDoS attack. 3
Figure 2. Features independency results. 6
Figure 3. Model accuracy scores. 6
Figure 4. Model precision and recall scores. 7
Figure 5. The top 10 IP addresses causing DDoS. 8
Confusion matrix 1. for training set. 7
Confusion matrix 2. for test set. 7
GIF 1. Setting up the work environment. 4
Table 1. Dataset features. 5

What is a DDoS?

Denial of service (DoS) attacks users from accessing the network and makes services unavailable or only partially available. The attacker floods the targeted machine or resource with excessive requests. The main goal of a DoS attack is to exhaust the network with a high volume of traffic, thus denying services access to legitimate users. DoS attacks can be categories to two main categories: network/transport-level attacks and application-level attacks [1]. Network-level DoS attacks disable legitimate users' access by exhausting network resources. While in application-level, DoS attacks disable service by exhausting the server resources. In distributed denial of service (DDoS) attacks, the victim's incoming traffic originates from many different sources. Instead of using a single machine, attackers use many (remotely) controlled computers to attack the victim see **Figure 1** for more clarity.

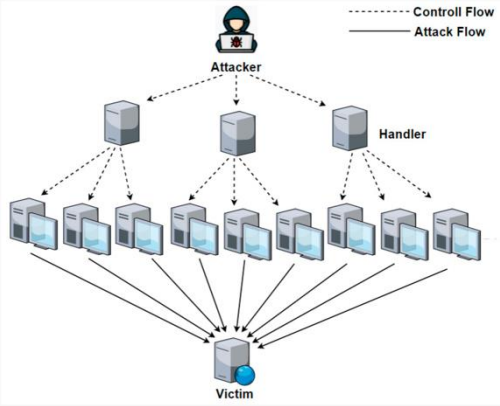


Figure 1. DDoS attack.

Implementation

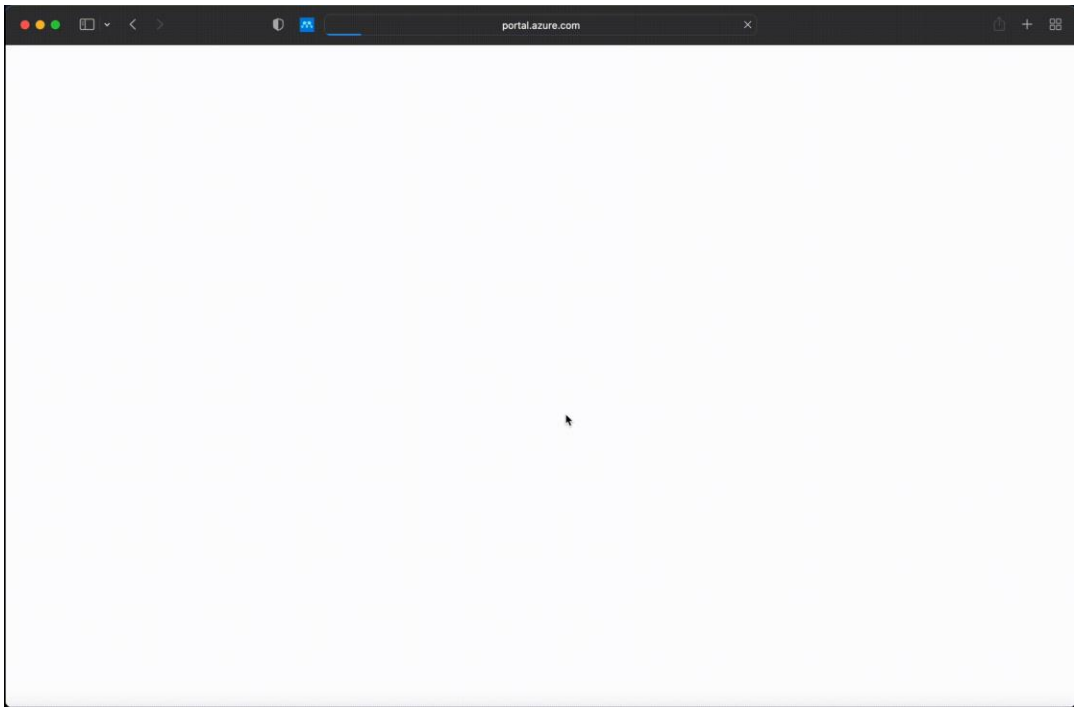
Dataset

There are no latest datasets found exclusively for DDoS in the Public domain, we found this dataset in Kaggle is extracted from different IDS datasets that were produced in different years and different experimental DDoS traffic generation tools. The dataset has more than 12,000,000 records (DDoS and benign) and 85 features.

*To see the dataset check resources

Work Environment

Due to our large dataset, when we used a local Jupyter notebook does not run normally instead of this we use one of Microsoft services azure ML. Azure Machine Learning is a cloud-based solution for ML workload and provides an end-to-end machine learning platform to enable users to build and deploy models faster on Azure.



GIF 1. Setting up the work environment.

An illustration of working in azure is shown down in **GIF 1** (open the document as .docx to see the GIF).
After creating a notebook for the project, we created a **compute instance** to execute the written code. The instance we chose had the following specifications Standard_E4a_v4 (4 cores, 32 GB RAM, 100 GB disk). To Know more about **Azure Machine Learning compute instances.**
*To try using azure services check resources

Machine Learning Technique

To deal with cloud DDoS attacks, several machine learning techniques have been developed. DDOS attacks have become so dynamic and robust, and lunch in different patterns, that one single solution is hard to implement. after several research works conducted on preventing and detecting DDOS attacks, they will be prevented and detected by using classification algorithms. They provide 17 types of ML techniques found all of these are applied to classify if the network traffic is normal or DDOS attacks[2].
In this notebook, we use the Naive Bayes technique is a powerful, extremely fast, easy-to-train classifier that allows it to work on very large datasets and provides better accuracy with fewer features needed. It works on the principle applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable[3].
Bayes theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

The Naive Bayes has three Naive Bayes variants based on the same number of different probabilistic distributions: Bernoulli, multinomial, and Gaussian. Bernoulli is valuable for a

model where feature vectors are binary (i.e. 0 and 1), multinomial suitable for feature vectors where each value represents the number of occurrences of a term or its relative frequency, Gaussian working with continuous values[4]. Depending on our dataset we use Gaussian distribution.

Libraries

• Ipaddress provides the capabilities to create, manipulate and operate on IPv4 and IPv6 addresses and networks.
• Seaborn is a data visualization library that provides a high-level interface for drawing attractive and informative statistical graphics.
• Pandas provides fast, flexible, and expressive data structures designed to make working with data both easy and intuitive.
• Matplotlib is a comprehensive library for creating static, animated, and interactive visualization.
• Sklearn supports supervised and unsupervised learning and provides various tools for model fitting, data preprocessing, model selection, model evaluation, and many other utilities.

Features

The dataset has 85 features that hugely impact the performance of our model, we want to reduce the number of features to reduce the computational cost and improve the performance. According to a survey [5] we have chosen the most useful features below:

Feature	Description
Flow Duration	Duration of the flow
Src IP	Internet Protocol Address (Source)
Src Port	Port (Source)
Dst IP	Internet Protocol Address (Destination)
Dst Port	Port (Destination)
Tot Fwd Pkts	Transmitted packets
Init Bwd Win Byts	Number of transmitted bytes
Protocol	Type of Protocol
Label	Attack classification Labels

Table 1. Dataset features.

Notebook

Preparing the Data: In this section, we import dataset and change datatypes of source, distention IP and Label features because we want to make it easy to deal with later. After convert IPs from object to string, Naïve bayes cannot working with String values because of this we are using Ipaddress to convert each IP to unique int values.

Data Analysis: In this section, we tested the independence of the features by using Seaborn library. In pie chart we investigated the proportion of DDoS and benign labels in the data set by using Matplotlib and Seaborn.

Building Naïve Bayes model: First, we separate features and label in two variables then splitting the data into test and train with test size of 20%. Second, we have built a **GaussianNB** classifier is trained using training data then we **fit()** classifier for training it. After

building a classifier, our model is ready to make predictions by using **predict()** method with test and train set features.

Results: In this section, we calculate the accuracy score for the test and train models, construct a confusion matrix for train and test, calculate precision and recall and top 10 source IP address in the dataset with DDoS label were obtained.

Results

Gaussian Naive Bayes Classifier is based on the Bayes Theorem, with the one assumption of the strong independence assumptions between the features. Thus, the classifier assume that the value of a particular feature is independent of the value of any other feature. So, we examined the independence of the chosen features before constructing the model to enhance the classification process in advance. **Figure 2** shows the result of features independency, where blue tones indicating high independence.

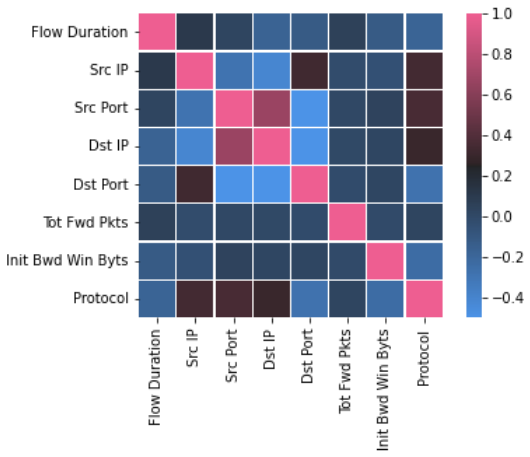


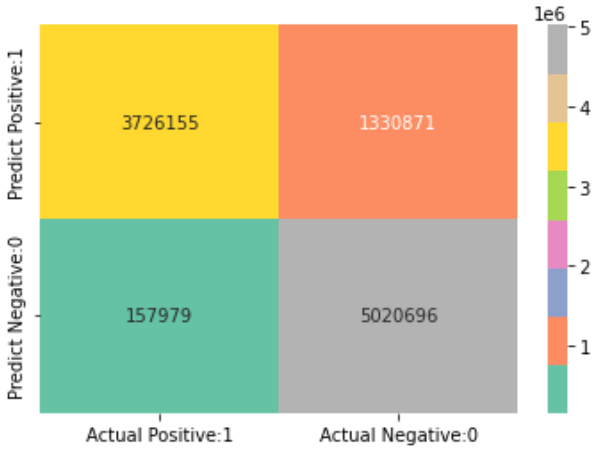
Figure 2. Features independency results.

After training and testing the model, the accuracy scores shown in **Figure 3** as follows:

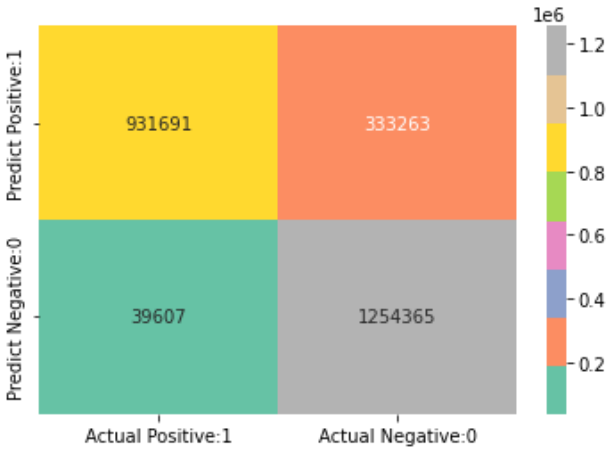
```
Training-set accuracy score: 85.4543%
Model accuracy score: 85.4287%
```

Figure 3. Model accuracy scores.

The following are the confusion matrixes for the training and testing sets, respectively:



Confusion matrix 1. for training set.



Confusion matrix 2. for test set.

Precision and recall performance measures were used to further evaluate the classification results. Precision is a metric that quantifies the number of correct positive predictions made. Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made. High scores for both show that the classifier is returning accurate results (high precision), as well as returning a majority of all positive results (high recall). The following **Figure 4** is an illustration of the results.

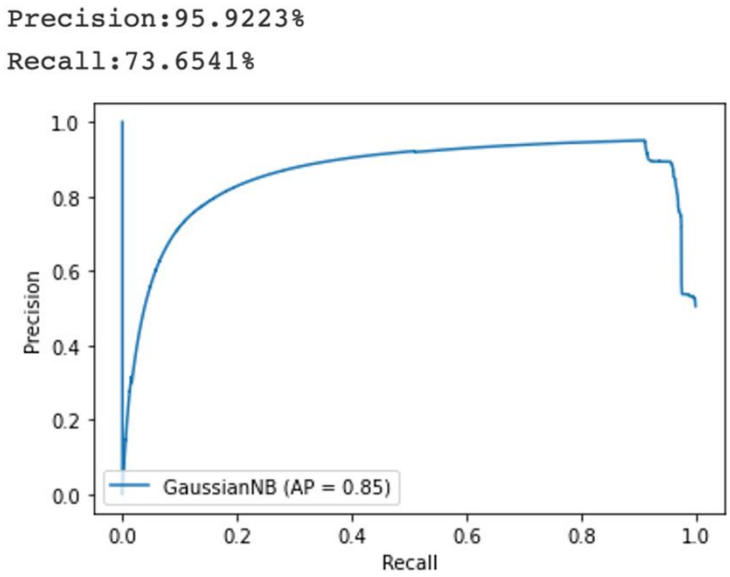


Figure 4. Model precision and recall scores.

The top 10 IP addresses that resulted in a DDoS attack were identified by the model, which are represented in **Figure 5** below.

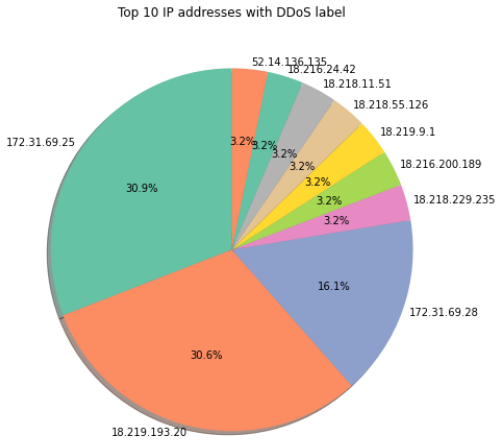


Figure 5. The top 10 IP addresses causing DDoS.

Resources

Dataset

Azure Machine Learning

References

[1] S. Ranjan, R. Swaminathan, M. Uysal, and E. Knightly, "DDoS-resilient scheduling to counter application layer attacks under imperfect detection," *Proc. - IEEE INFOCOM*, 2006, doi: 10.1109/INFOCOM.2006.127.

[2] I. Conference and E. Engineering, "Machine Learning for Cloud DDoS Attack Detection : A Systematic Review m au," 2020, doi: 10.1109/ICCCEEE49695.2021.9429678.

[3] S. Iot *et al.*, "Adaptive Machine Learning Based Distributed Denial-of-Services Attacks Detection and Mitigation System for SDN-Enabled IoT †," 2022.

[4] S. Raschka, "Naive Bayes and Text Classification I - Introduction and Theory," pp.