



ECE 572 - Security, Privacy, and Data Analytics Project Report

Submitted by

Sannath Reddy Vemula	V00949217
Mona Malik	V00935324
Venish Patel	V00949366
Nikita kapoor	V00949256
Nishra Shah	V00936027

Abstract:

The report focuses on the Data Analytics applied on network security dataset which primarily can be considered as a data of binary classification. This project demonstrates the ability to use different methodologies and programming languages to detect the malicious connections entering into a network. The report mainly draws the attention of readers onto the concepts like classification, regression, clustering and correlation which are considered significantly important prior to applying an algorithm/technique to detect anomaly behaviors/patterns in network security datasets such as IDS or IPS log files. The basic design of the project is around analyzing the data logs which were collected by an organization in which different categories of connections like Non-malicious connections, SQL Injection attacks, DoS attacks and other attacks may include. This report sheds light on the patterns or features that a certain class of connections may have that can be used to recognize and segregate the future anomaly connections from regular connections entering a private network. The major work on the dataset deals with finding the important features that can uniquely identify a class and also provides the possible ways of building a binary classifier with ease over such network security datasets. Few of the methodologies such as supervised and unsupervised Machine Learning algorithms, Text Mining and Neural Networks have been implemented in programming languages which are widely considered for such data analysis work. Different aspects like features and performance measures like accuracy comparing the implemented methodologies gives an insight about the algorithm/technique to be used depending upon the dataset. Within the Malicious class of the dataset, there exists furthermore classes which are dealt and classified using text mining.

Summary:

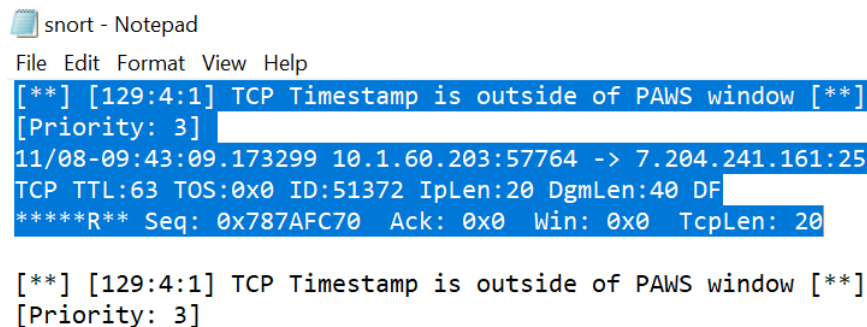
The analysis starts with preliminary steps like data conversion and preprocessing which serve as the basis for implementing the Machine Learning detection algorithms. The initial work is around converting the details of the records in the log file into CSV with designated features/columns. The CSV which has been normalized and is reduced in terms features by selecting the important features using Feature Extraction/Reduction Algorithm called Principal Component Analysis. Different ways of short-listing important or topmost features are demonstrated in each of the methodology which help in visualizing the complex high dimensional data in simple lower dimensions. Concepts like correlation are implemented to pick the subset of the features which ensure minimal loss of information. The report gives insights about such concepts that are considered to be the important aspects of data analysis which are utmost required before considering and working on any dataset. One of the supervised machine learning algorithms, K-Nearest Neighbors, has been implemented to show the clusters of binary classes(malicious and non-malicious connections) in a dataset. Similarly, results and insights from one of the unsupervised machine learning algorithms, K-Means clustering, were represented in the report. We have also presented the results of neural networks in classifying the records along with the accuracy and error rate. All of the above-mentioned work was done in two different programming languages, Python and MATLAB, which provide better packages for data analysis. We have concluded the report by showcasing the results of text mining that was performed within the malicious class which was helpful in identifying the type of attack that the connection in the log file was intending to execute on the private network.

Introduction:

We picked IDS log files dataset and demonstrated the use of different languages, tools and packages for a binary class dataset. For most of our work, we considered binary classifiers and to classify the attacks within the malicious class, text mining was conducted which clearly gave an insight about the number of attacks in different categories of attack that were executed on the private network. Discussing the scope of the project, we can apply different machine learning algorithms, binary classification techniques, and multi-class classifiers for this dataset. Since most of the features are categorical, the challenging part was to generate the data clouds/ clusters for malicious and non-malicious connections.

About the Dataset:

The dataset we worked on for the project is SNORT log file which is the data captured from the National Security Agency (NSA) provided by the United State Military Academy Westpoint[1]. The file maintains all the records of connections entering or leaving an organization's private network which are logged by making use of the mechanism called Intrusion Detection System (IDS). A single record in the log file holds the details of a connection like Date and Time of connection, Source IP address, Destination IP address, Source Port Number, Destination Port Number, Protocol, Flags, TCP Length, IP Length, Type of Service, etc. For our data analysis work on network security dataset, we took these details of the connections as features and parsed the text log file into CSV file as our first step. The CSV file is created such that all the important features are separated and represented accordingly. For instance, one of the fields in a log file represents the flags which are set for the connection and this field is of 8-bit length where each bit is reserved for each of 8 flags (2,1,U,A,P,R,S,F). While parsing the text file to CSV file this field is divided into 8 columns to make sure which flag is set for the connection. Below can be seen one of the records in snort file and its respective CSV file:



```
[**] [129:4:1] TCP Timestamp is outside of PAWS window [**]  
[Priority: 3]  
11/08-09:43:09.173299 10.1.60.203:57764 -> 7.204.241.161:25  
TCP TTL:63 TOS:0x0 ID:51372 IpLen:20 DgmLen:40 DF  
*****R** Seq: 0x787AFC70 Ack: 0x0 Win: 0x0 TcpLen: 20
```

Fig.1 : Record in Snort Log File

A	B	C	D	E	F	G	H	I	J	K	L	M	N
Priority	UnixTimeStamp	Source port	Destination port	Protocol	TTL	TOS	Datagram Length	A	P	R	S	F	TCP length
3	-1	0.763490101	-0.999236769	3	-0.5059761	1	-1	0	0	1	0	0	1
3	-0.999000281	-0.999236769	0.531834348	3	-0.49800797	1	-1	0	0	1	0	0	1
3	0.000000000	0.000000000	0.000000000	3	0.000000000	1	-1	0	0	1	0	0	1

Fig.2 : Same record parsed to CSV with expanded features

(Note: Data in the CSV represents the same data as text file but the features in CSV have been normalized and numerically represented which was done as a part of data pre-processing.)

In the above screenshot of the CSV file, every field of log file has been represented as features (columns) and the features which account for uniquely identifying the record are preserved and others have been discarded. Important features among all the features has been further discussed in the feature reduction section. Snort basically observes network packet traffic which can be configured to log or report on any information that is available from the network packet. Snort CSV logs do not include a header row, so we had to create a separate csv file to name each column.

The parsed csv file consists of 25741 rows and 21 columns

The following are the attributes of the generated csv file:

1. **Action Performed** - gives description about the connection
2. **Connection classification** - Primarily we have two different classes, connection which are malicious and non- malicious. Within non-malicious, we have values each explaining different types of cyber-attacks.
3. **Priority**- Tells us the order for the actions to be taken for the malicious connections. A priority of 1 (high) is the most severe and 4 (very low) is the least severe.
4. **Date-Time** - Date and time at the connection was logged
5. **Source IP**: This is the IP address of the system which generates a request and directs it towards the destination system.
6. **Source Port**: The port from which the request was made to the network
7. **Destination IP**: This is the IP address of the system to which the request or response is directed.
8. **Destination Port** : The port to which the request is made.
9. **Protocol**: We have 4 classes under this feature - IP,TCP,UDP,ICMP. It gives information about the rules which are being used in establishing the connection.
10. **TTL(Time to Live)** - Time to live (TTL) or hop limit is a mechanism that limits the lifespan or lifetime of data in a computer or network. This takes numbers from 0 to 255.
The Time to Live (TTL) field keeps a counter that decrements every time a packet crosses a router.
11. **TOS(Type of Service)** - The Type of Service field is used for prioritizing traffic for performance. It is a 8-bit string which tells about precedence,delay, reliability, minimum cost and throughput. In the dataset this feature takes 6 values (61,64,240,0x0,0x10,0xC0).
12. **ID** - A unique identifier for each record.
13. **IP length** - This specifies the IP packet header length which is 20 bytes for IPv4.
14. **Datagram Length** - IP datagram consists of a header part and text part. It specifies the total length of the datagram in bytes.
15. **Flags** - It is denoted by a 8-bit string which tells which flags are set for the connection. Each bit indicates a distinct flag.
16. **Seq** - The seq keyword is used to check for a specific TCP sequence number.
17. **Ack** - This option checks for a particular acknowledgment number.
18. **TCP length** - It specifies the size of the TCP header in 32 bits.

Other details like Win, NOP WS, SackOS TS, MSS are not consistent and are not significantly used for the analysis.

Data Preprocessing:

Under Data preprocessing, we converted the parsed CSV into numerical data and normalized the features. For instance, features like 'Date and Time' would not have much to contribute in its original format. For this reason, the Date and Time feature of the dataset was converted into a UNIX timestamp which expresses the Date and Time of the record in terms of seconds from 1st January, 1970. This would help to analyze the DoS attacks and extract the records which tried to connect to the private network in quick succession.

Below provided is the screenshot to shed light on the connection which has the potential to be DoS attacks. This helps to minimize the range of search for such attacks.

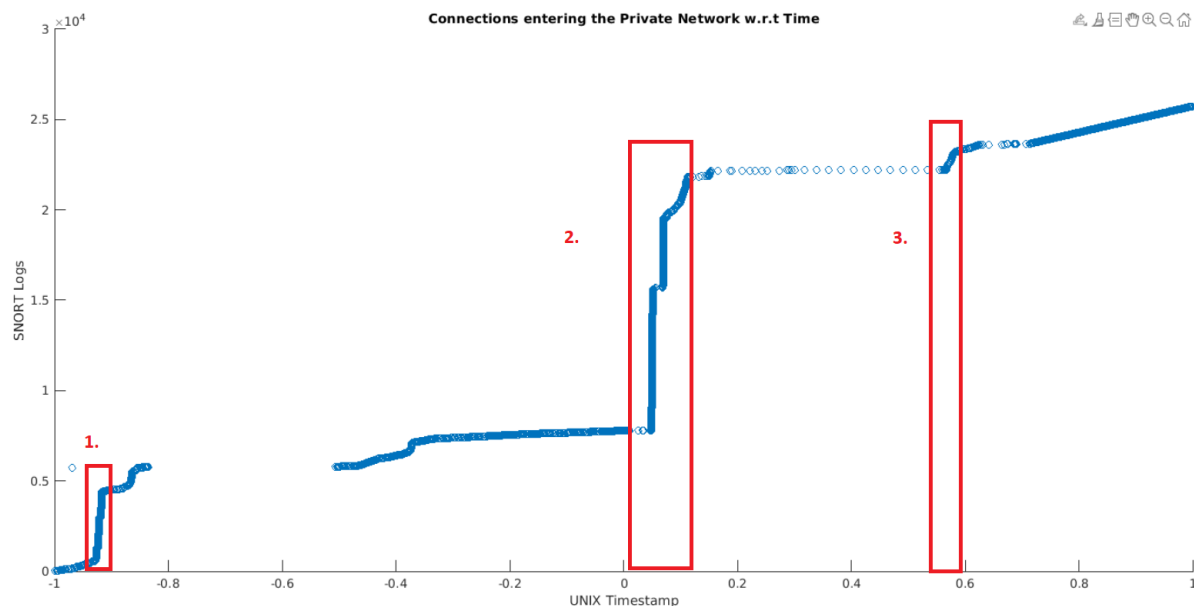


Fig.3: Multiple requests made to the private network in quick succession

The areas highlighted in the above graph indicate that multiple connection requests were made in quick succession to the private network and can be treated as suspicious connections for further analysis. This gives an insight about the type of connections entering the network and speeds up the process of detecting anomaly connections. There are other such features which have been converted to formats that are suitable for the analysis of the malicious connections.

Upon converting the dataset which can be understood by the Machine learning algorithms, we have selected the features which have different ranges of possible values and which have the potential to uniquely identify observations from a class. We finally made use of 15 features which are normalized and pre-processed.

Feature Extraction Algorithm: Principal Component Analysis (PCA) in MATLAB

Considering the CSV file which has been normalized and preprocessed, we performed PCA on it to represent the high dimensional data in the lower dimensions of 2D and 3D. This was done by extracting the top 3 features which contribute most to represent the dataset. Ranking the features based on the weightage in representing the original data has been accomplished by making use of statistical properties like Covariance, EigenValues and EigenVectors.

- Covariance - is the metric which gives directional information between two features. It gives the extent of spread of data with respect to 2 features.
- EigenVectors - The orthogonal axes in which the data can be better represented. They are the shifted coordinates with the mean of the dataset as the center.
- EigenValues - They denote the scaling factor of the eigenvectors. The vectors which have the larger eigenvalues indicate that the spread of data (or covariance) with respect to the feature in the direction is greater than other features.

So, the entire data can be converted by making use of the above 3 concepts to transfer and plot the complex higher dimensional data onto lower dimensions. From the below graph, TCP options - 70% approx., TCP length - 10% approx. and Flag F - 5% approx. are the topmost features which convey about 85% of the data.

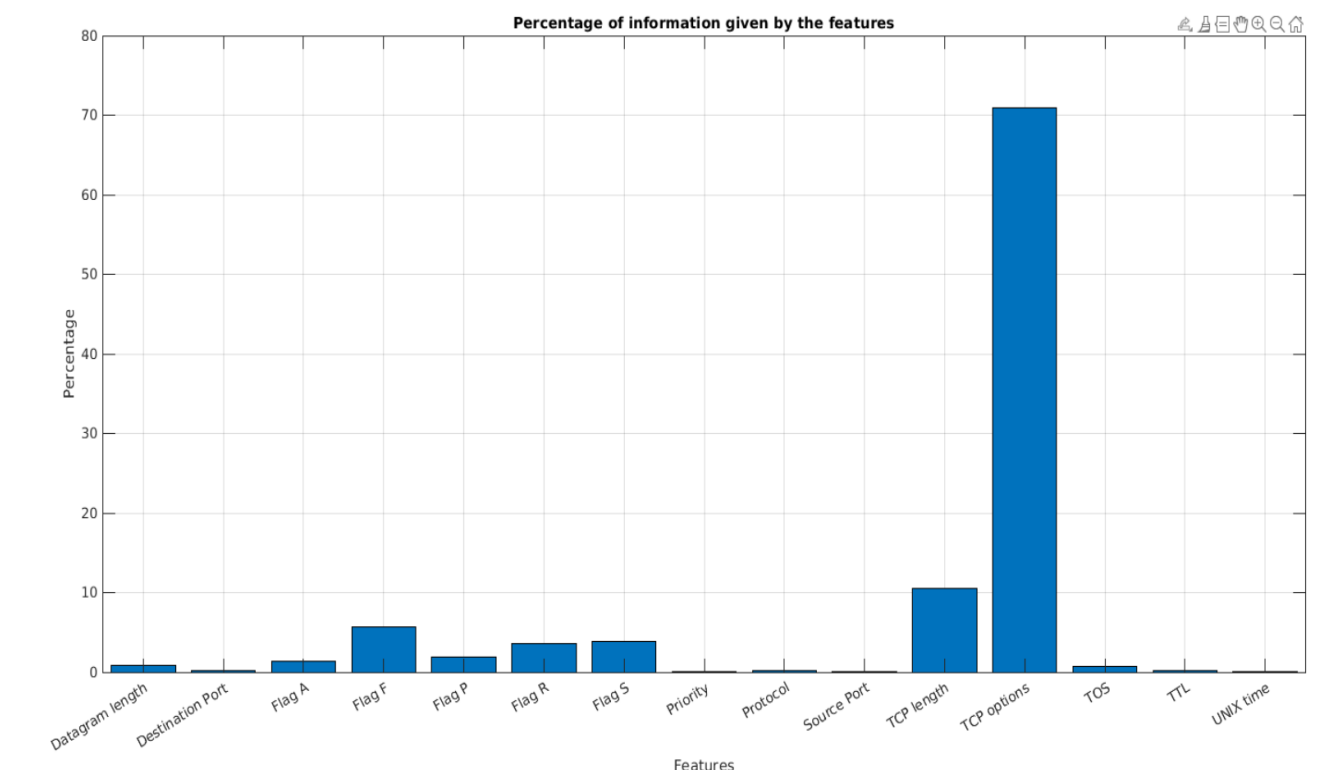


Fig.4: Weightage of Features in representing the dataset

Below is the table which gives the weightage of all the features:

Feature	EigenValue	Percentage
TCP Options	4.812	70.94%
TCP Length	0.7117	10.49%
Flag F	0.3886	5.72%
Flag S	0.2582	3.8%
Flag R	0.2411	3.55%
Flag P	0.1275	1.87%
Flag A	0.0948	1.39%
Datagram Length	0.0538	0.79%
Type of Service	0.0486	0.71%
Time to Live	0.016	0.23%
Protocol	0.0126	0.18%
Destination Port	0.0107	0.15%
Source Port	0.0041	0.06%
UNIX timestamp	0.0028	0.04%
Priority	0.0001	0.0018%

Table.1: Features ranked in descending order in terms of EigenValues.

Using the above highlighted features to plot the entire dataset in lower dimensions, below are the 2D and 3D graphs visualized:

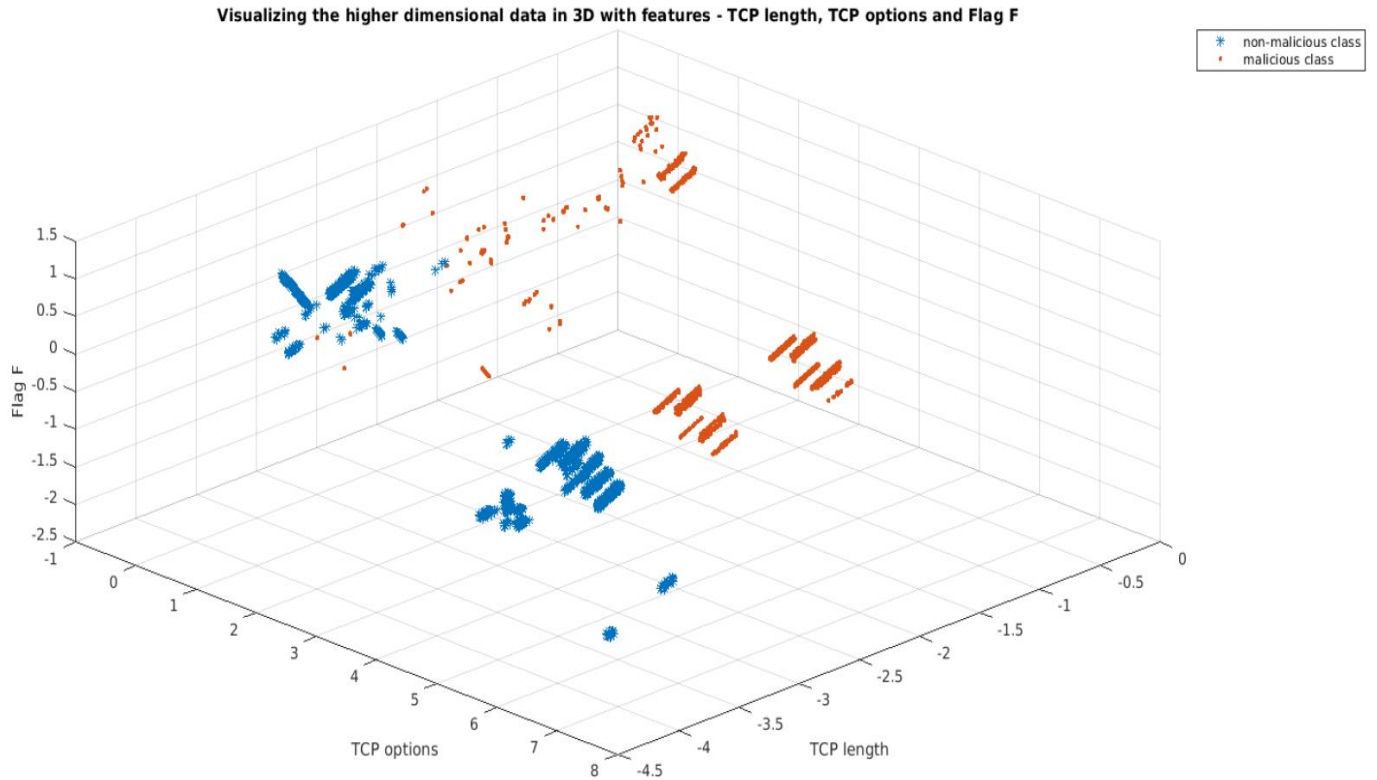


Fig.5: Projecting the dataset onto 3D with features- TCP length, TCP options and Flag F

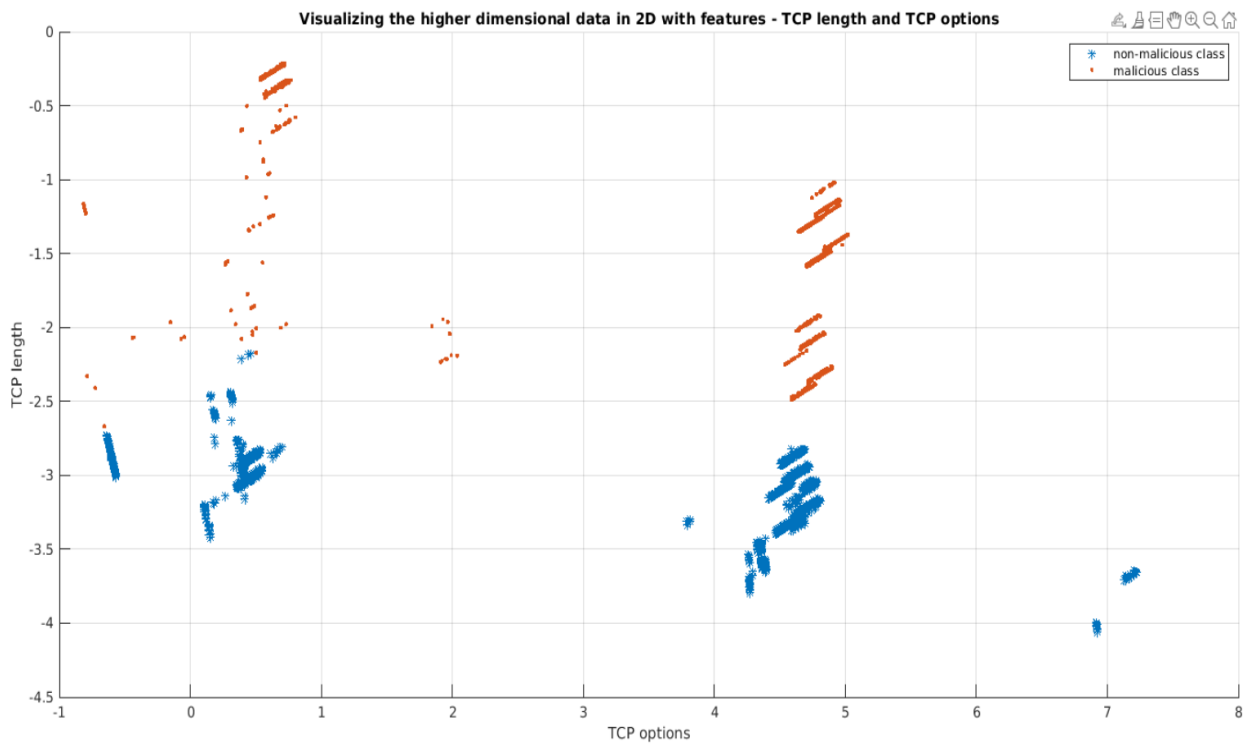


Fig.6: Projecting the dataset onto 2D with features- TCP length and TCP options

Insight: From the above plots, we can see that the data points are clearly separable in 2D and 3D which gives us the insight that TCP options, TCP length and Flag F are the features which can solely identify if a future connection is malicious or non-malicious. Both the plots also indicate that it is a linear data and a linear regression can also be performed on similar lines for accuracy.

Implementation of Neural Networks in MATLAB:

We have implemented a Neural network to see the classification capability on the dataset without excluding any of the 15 features. The entire dataset is divided into train and test sets by considering 75% of the data for training the neural network and 25% of the data to test the performance of the trained neural network. The performance of the neural network on the dataset is reported below in terms of confusion matrix, ROC curve. Below the screenshots of confusion matrix and ROC curve show that the neural network was successfully able to identify every connection and classify them accordingly. As we can see that there are no type-1 or type-2 errors in both the metrics. This proves that, given the dataset and the features of the connections, the neural network is capable of categorizing the connections accurately.

This was implemented in MATLAB by making use of a neural network pattern recognition tool which is again a basic tool to explore and conduct such analysis. The trained neural network was tested for its accuracy by feeding 25% of data as a test dataset which proved that the neural network was trained aptly for classification the connections with the features mentioned in the above sections.

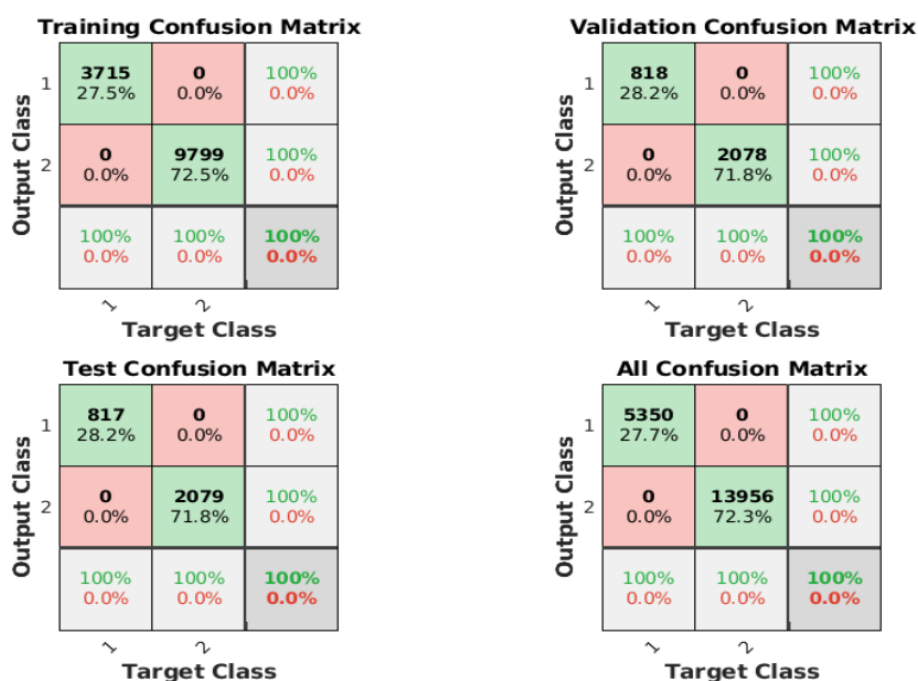


Fig.7: Confusion Matrix for Malicious and Non-Malicious classified by Neural Network

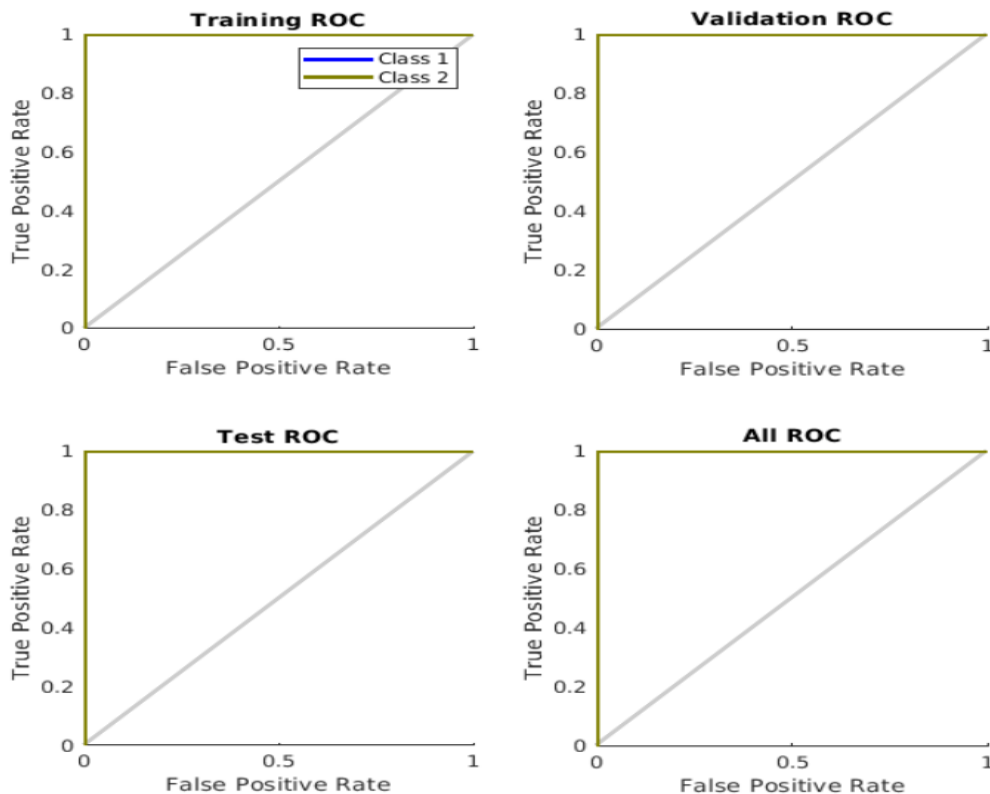


Fig.8: ROC curve for the neural network classifier

Supervised Machine Learning Algorithm: K-nearest neighbors in Python

Supervised learning is the machine learning task of learning a function that maps an input to an output based on training data labels. A supervised learning algorithm analyzes the training data and produces a classifier, which can be used for mapping new examples.

K-nearest neighbors is a supervised machine learning algorithm which works by assuming that similar things exist in close proximity. First, we find the Euclidean distances between points and select the specified number points (K) closest to the entering point and then averages the labels for the cluster.

Thereafter, the correlation of features in the dataset with respect to the given class or labels (i.e malicious or non-malicious) is taken into consideration for selecting three highest correlated features measurements to train the KNN classifier which will be later used for prediction of unseen test data.

In the figure below, we see that feature measurements - Priority, Protocol, and Destination Port have the highest correlation with respect to Classes. This indicates that these 3 features have the potential to predict the class of the unknown future connections.

<i>Features</i>	<i>Correlation</i>
Priority	0.862247
R	0.371830
Protocol	0.291782
Destination port	0.288368
A	0.264573
F	0.215099
TOS	0.212674
P	0.165469
Source port	0.159474
Unix TimeStamp	0.135101
TCP options	0.129982
TCP length	0.123938
S	0.031568
Datagram Length	0.024056
TTL	0.012675

Table.2: Correlation of 15 Features with the class labels

We are using sklearn package for implementing K nearest neighbors on our dataset. To train our classifier we are using three highly correlated features which are Priority, Protocol, and Destination Port. For measuring the accuracy of this classifier, we are using train test split to split our data into 75% training data and 25% test data. We are using 5 nearest neighbors to cluster the points and Euclidean distance to measure the distance between the points.

In order to measure the accuracy of our classifier, we plotted a confusion matrix between predicted labels and actual labels. Here we see that there are no type 1 and type 2 errors.

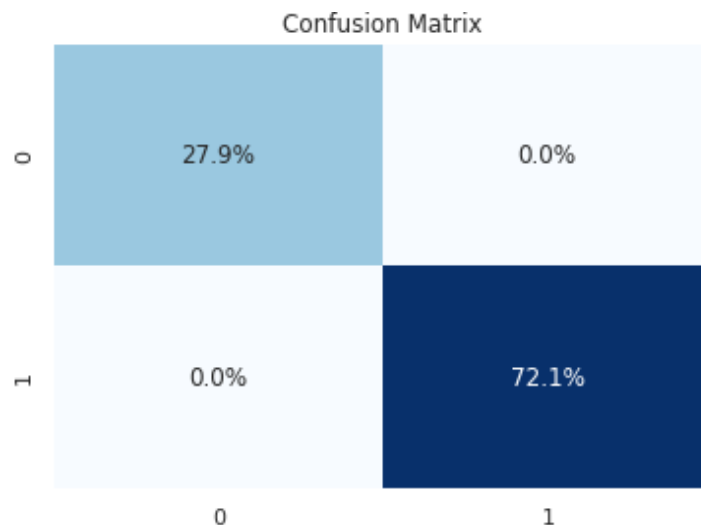


Fig.9: Confusion matrix for KNN classifier

Plotting against destination port and priority features, we got the above confusion matrix and were precisely able to classify malicious and non-malicious connections into red and green clusters.

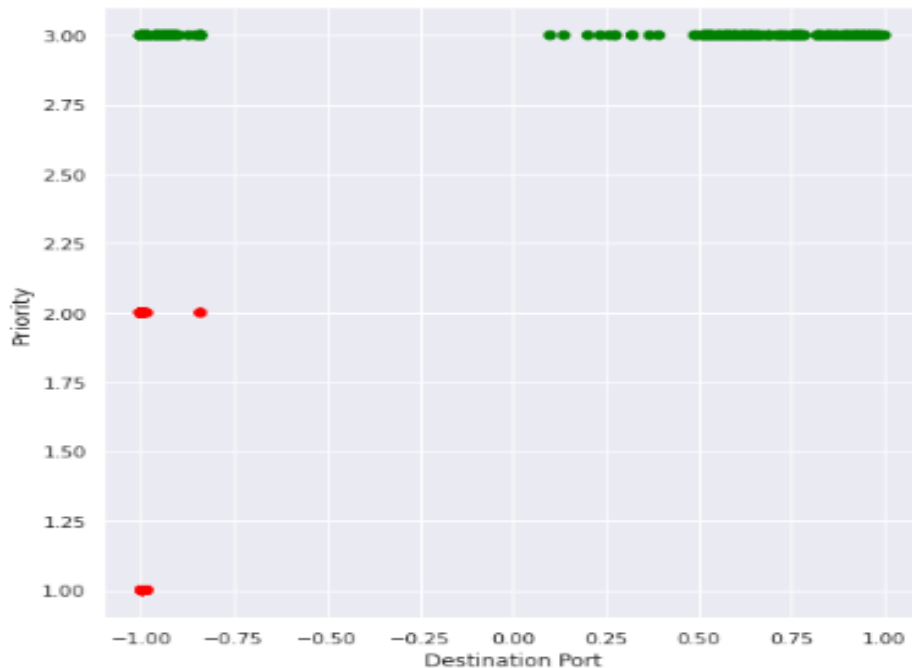


Fig.10: 2D Scatter plot with top features - Priority and Destination Port

For the malicious class, the majority of the connections have made a request to destination ports - 80 and 25 (where port 80 is HTTP and port 25 is SMTP).

Insight: This gives an insight that cross-checking the connections which are making requests to HTTP and SMTP ports will minimize the search space in detecting anomaly behaviors.

Unsupervised Machine Learning Algorithm: K-Means Clustering in Python

K-means clustering is a type of unsupervised learning, which is used when you have unlabeled data (i.e., data without defined categories or groups). The goal of this algorithm is to find clusters in the data based on the hidden patterns, with the number of clusters represented by the variable K. The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity.

For finding the hidden patterns of the malicious and non-malicious classes we have implemented k means from scratch on our snort logs. For implementing the K means to find a hidden insight we have used fairly high correlated features Destination port and Acknowledgement flag. In the snort log alert messages will be generated when you receive a TCP packet with the A flag set and the acknowledgement contains a value of 0.

The results of the K-means clustering algorithm are:

- The centroids of the K clusters, which can be used to label new data
- Labels for the training data(each data point is assigned to a single cluster) - we got two clusters which are segregating the malicious and non-malicious data points.

Insight: From the insight we can see that we have two clearly defined clusters around centroid 1 and centroid 2. From the above analysis we discover that port 25 and 80 are malicious ports. Correlating these facts, we can assume that in future any point falling in cluster 1 has high chances of being malicious and requires immediate attention.

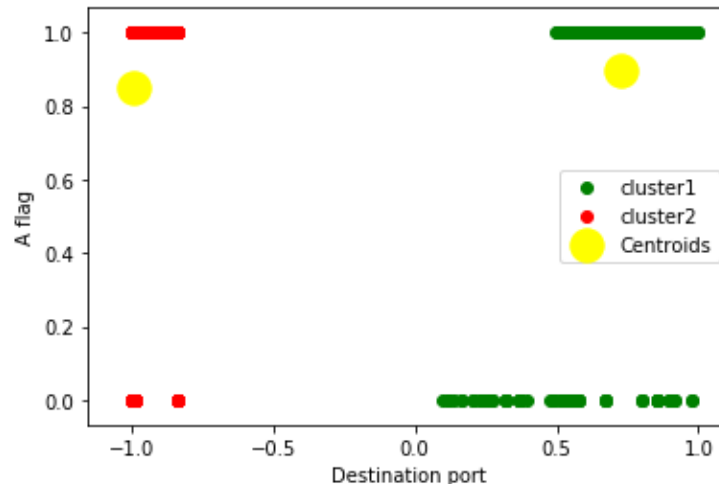


Fig.11:Centroids produced for malicious and non-malicious classes with KMeans clustering

Text Mining within Malicious class of dataset:

We started by recognizing the patterns/features that each kind of attack follows and searched for the records which exhibit similar features. In this way, we classified the malicious class further into 5 main classes which include the attacks like 'Web application attack', 'Attempted Denial of Service', 'Access to potentially vulnerable web application', 'Attempted user privilege gain', 'Executable code detected'. We divided the malicious class into 75-25 train and test sets. The features which represent these attacks are grouped and denoted by a string which was used to track the other such attacks. The features like priority, UNIX timestamp, Destination port, protocol played a major role in most of the attacks and the difference in weightage of each feature represented an attack. Below is the bar graph which shows the number of connections that executed each of the above-mentioned attacks.

For instance, a connection with priority as '1' and destination port as '80' is almost every time a web application attack. One of the details given in the snort log file was about the link to refer about the possible attack the connection can execute. This feature was web-scraped and integrated with the keywords like 'web attacks', 'SQL injection', 'phishing', 'man-in-the-middle attack' to confirm the attack. This approach almost classified every attack within malicious class correctly and below is the bar graph shown for which the analysis was done.

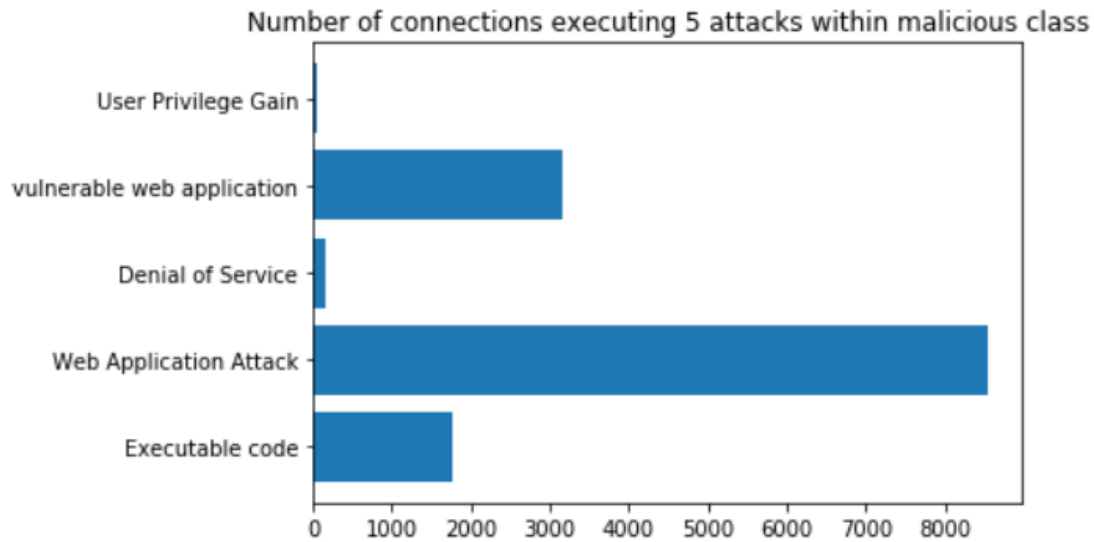


Fig.12: Number of connections executing the 5 attacks within malicious class

(Note: The above represented attacks are not the top 5 attacks but, these are the attacks which the features in the dataset were able to define.)

Conclusion:

With the majority of the report representing the techniques and algorithms in machine learning to detect the malicious connections, we also demonstrated the technique to classify multi-class dataset within the malicious class. We have worked on data normalization, data preprocessing, feature extraction techniques, different classification methodologies and their implementation in two different languages.

References:

- [1]<https://www.westpoint.edu/centers-and-research/cyber-research-center/data-sets>
- [2]<https://medium.com/machine-learning-algorithms-from-scratch/k-means-clustering-from-scratch-in-python-1675d38eee42>