Securing web applications using machine learning-driven firewall

Submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

Gaddam Harshavardhan
(38110197)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,

CHENNAI – 600119, TAMILNADU

MAY - 2022

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Gaddam Harshavardhan (38110197)** who carried out the project entitled "**Securing web applications using machine learning-driven firewall**" under my supervision from November 2021 to May - 2022.

Internal Guide

Dr M Maheshwari M.E., (PhD)

Head of the Department

Dr.Lakshmanan L, M.E, PhD.

Submitted for Viva voce Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

I, Gaddam Harshavardhan (38110197) hereby declare that the Project Report entitled "**Securing web applications using machine learning-driven firewall**" done by me under the guidance of Dr M Maheshwari M.E., (PhD) is submitted in partial fulfilment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph.D.**, **Dean**, School of Computing , **Dr.S.Vigneshwari M.E., PhD, and Dr.L.Lakshmanan M.E., PhD,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr M Maheshwari M.E., (PhD)** for her valuable guidance, suggestions and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

	TABLE OF CONTENTS	
CHAPTER No.	**TITLE**	**PAGE No**

ABSTRACT

Internet is a most wonderful tool that is created by a human being. All the information that we require is just a few clicks away. Yet, it is a wonderful tool there are a lot of security issues associated with the internet. A firewall is a tool that prevents applications from cyber security attacks. Yet, there are powerful firewalls still, cyber-attacks are happening around the world. Most of the cyber-attacks going on in the world are due to man-made errors.

So, it is important for us to understand there is a need for a powerful yet fast processing firewall that is driven by machine learning algorithms in order to improve security. There is a common pattern in attacking a web application or web server by hackers. The same pattern can be used to train a machine learning model and add it to a web application to attain maximum security.

A logistic regressing machine learning model is more fitted for these types of machine learning applications which can be trained and tested against the Kaggle dataset. Kaggle dataset consists of more than 12 lakhs security data which also has data from previous cyber attacks.

ML model which is being used in this project is built on sklearn framework and web-part is fully built on the nodejs framework which are Java script frameworks most popularly used for the backend process in web development.

I will be using HTML, CSS, and bootstrap for the front end to create a user interface for users, Nodejs for the backend process and flask for the automation process to get real-time security in the web.

CHAPTER 1

INTRODUCTION

1.1 PROBLEM STATEMENT:

To develop a firewall using machine learning that can detect web attacks efficiently and fastly. Also, it needs to classify the type of attack and protect the application from being hacked.

1.2 OBJECTIVE:

The main objective of this project is to show how normal firewalls fail to give maximum security to a web application or a network server, There are a lot of web servers that are being attacked by hackers. Many web servers are prone to data breaches and attacks by hackers when they are still secured by firewalls. So it is important to understand that there is a strong need for a firewall, which can secure websites and networks much more secure.

We are using a firewall that is being defined by a few predefined rules during its programming. All the attacks/requests that did not satisfy any of those conditions will be restricted by the firewall. but, there is a strong need for a firewall that uses come common phrases in hacking to get trained if there are any requests that involve any of these phrases in them then those requests will be blocked.

1.3 NECESSITY:

Though there are many companies that use very strong firewall systems and still get attacked by hackers. Few hackers steal critical and confidential data from servers which include contact details and payment data, which not only impact the organization but also people who registered on that site. So there is a strong need for a better firewall. Here ML comes in to give better security.

ML-based firewall got trained using ml model using a strong dataset which is a collection of a very large dataset that consists of more the 12lakh attack payloads which are being used by many hackers in past 15 years which is collected from web server data and classed according to its severity. This firewall which is powered by the ML model monitors network packets or web requests and if any request is found to be causing any harm to the webserver or application firewall will immediately block that request.

1.4 OUTLINE

Request and attacks data from web servers were collected. Used data science techniques to extract needed information and remove unwanted information from web requests and form a dataset from it. Using the dataset to train the ML model with high speed and better accuracy.ML model trained using dataset is tested using few test matrics and scripts. An API is developed to monitor website traffic which is also useful as a firewall.

1.5 WEB DESIGN OVERVIEW:

The web design process started with a visual concept by designing in Figma software. Then, I used HTML and CSS to build the website. HTML handles the basic structure of the page, while CSS handles the style and appearance.

CHAPTER 2

LITERATURE SURVEY

In the years gone by, research on the topic of firewalls using machine learning and deep learning algorithms to detect Cyber attacks on web servers and their analysis have taken place widely. This is due to the demand in understanding the deeper relationship between payload and hacking type, and also the relationship between the payload involved themselves.

ROBSON V. MENDONÇA proposed a deep convocational neural network model that is fast in its way and can classify payloads more accurately that uses deep CNN for faster detection. This model is published in Intrusion Detection System Based on Fast Hierarchical Deep Convolutional Neural Network journal 2021.

TAHA SELIM USTUN proposed random forest and decision tree ML models which is used as a firewall for IEC 61850 which is also called the internet of things(IoT) which is connecting physical devices with the internet. Detection is done using symmetric and asymmetric faults. This method is published in Artificial Intelligence Based Intrusion Detection System for IEC 61850 Sampled Values Under Symmetric and Asymmetric Faults 2021.

Dilara Gümü¸sba¸s proposed a few AI models like data encoders, CNN to detect attacks on a web server. It also discusses various datasets available for training AI models in cyber-attack detection. This paper also discusses the importance of data encoding during the training of AI models for better accuracy. This is published in the journal A Comprehensive Survey of Databases and Deep Learning Methods for Cybersecurity and Intrusion Detection Systems in 2020.

Dennis Appelt gave an overview of SQL Injection and how it can be very dangerous in its own way. He also suggested an algorithm to detect SQL injection efficiently. His work is published in the paper Behind an Application Firewall, Are We Safe from SQL Injection Attacks? in 2015.

Beibei Li suggested a few measures to secure physical systems from cyber attacks. Physical systems in industries are important for the industry securing them from cyber attacks is crucial for industries. Beibei Li also suggested a few measures to maintain privacy in industrial physical systems. This work is published in DeepFed: Federated Deep Learning for Intrusion Detection in Industrial Cyber-Physical Systems in 2020.

Gustavo Betarte used a multi-class classifier to detect OWASP top 10 vulnerabilities in web applications. He used a multi-class classifier in the web to parse requests and detect attack/ Hacking payload in the request. He also 3 major datasets to test it. His work is published in Web Application Attacks Detection Using Machine Learning Techniques in 2018.

Dennis Appelt in his paper about testing firewall systems gave an approach to test firewalls in a way that is effective. He also proposed an ML model to generate attack payloads like SQLInjection in testing firewall systems. His work is published in the paper A Machine-Learning-Driven Evolutionary Approach for Testing Web Application Firewalls in 2018.

CHAPTER 3

AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AIM:

The main aim of this project is to make it much more secure than ever. At the same time, it should be fast and effective. Even though almost 100% of servers are being secured by firewalls, there are a lot of web attacks are happening daily and most of the time firewalls are not able to secure networks or web servers from a new type of attack.

Most of the time hackers try to exploit web servers using various web attacking techniques like SQL injection, Cross-site scripting, remote code execution and web server attacks and many more. A firewall that is being used in a web server is trained in a way that it can detect many of these types of attacks. Ml model uses Sklearn logistic regression model which is a simple, yet fast and effective machine learning model for machine learning.

Dataset used in this ML model consists of more than 12lakh data which is collected from web servers and classified according to the superiority of attack. ML model used in this firewall has attained an efficiency of 97.5% which is specifically trained for injection, XSS, RCE and directory transversal based attacks. It also achieved a great speed of predictions of 1,00,000+ requests per second.

3.2 SCOPE OF THE PRESENT INVESTIGATION:

With the reference from a survey by HackTheBox(HTB), A leading hacking practice platform in association with Bugcrowd community, A leading community of ethical hackers throughout the world in the year 2020, came to know that most the biggest drawback which makes breaching servers easy when the same type of attack can be done to many web servers. Till then many individuals and corporate companies started creating AI-based firewalls but no one got satisfied with that because of its slow nature and less the 50% accuracy in detecting a web attack.

Also, they came to know that from the community of developers and hackers, the firewall they are using utilises a lot of computational power and resources like RAM and network from their machine which also is one of the drawbacks of it.

This firewall use as less as 10GB of storage for training and maintenance of the ML model and is used as less than 1GB ram to compute 1lakh+ of predictions. Which is quite small compared to previous firewalls. It can be further trained to detect DDoS attacks and authentication attacks so that web servers are much more secure.

3.3 SYSTEM STUDY

Using traditional firewalls came to know that the biggest drawback which makes breaching servers easy is when the same type of attack can be done to many web servers. Till now so many companies started creating AI-based firewalls but no one got satisfied with that because of its slow nature and less the 50% accuracy in detecting a web attack. The firewall they are using utilises a lot of computational power and resources like RAM and network from their machine which also is one of the drawbacks of it.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

4.1 INTRODUCTION TO MACHINE LEARNING:

Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) with data, without being explicitly programmed.

The name machine learning was coined in 1959 by Arthur Samuel. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or unfeasible; example applications include email filtering, detection of network intruders, and computer vision.

Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter sub-field focuses more on exploratory data analysis and is known as unsupervised learning.

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and

results" and uncover "hidden insights" through learning from historical relationships and trends in the data.

4.2 TRAINING THE DATA:

There are basically two widely-used types of training that can be done to create a model:

i. Supervised Learning

ii. Unsupervised Learning

4.2.1 Supervised Learning:

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labelled training data consisting of a set of training examples. In supervised learning, each example is a pair consisting of an input object (typically a vector) and the desired output value (also called the supervisory signal). A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples. An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.

4.2.2 Unsupervised Learning:

Unsupervised machine learning is the machine learning task of inferring a function that describes the structure of "unlabeled" data (i.e. data that has not been classified or categorized). Since the examples given to the learning algorithm are unlabeled, there is no straightforward way to evaluate the accuracy of the structure that is produced by the algorithm—one feature that distinguishes unsupervised learning from supervised learning and reinforcement learning.

The type of training used in this model is **SUPERVISED LEARNING.**

4.3. METHODS IN SUPERVISED LEARNING:

Supervised Learning mainly consists of two methods,

- **Classification**
- **Regression**

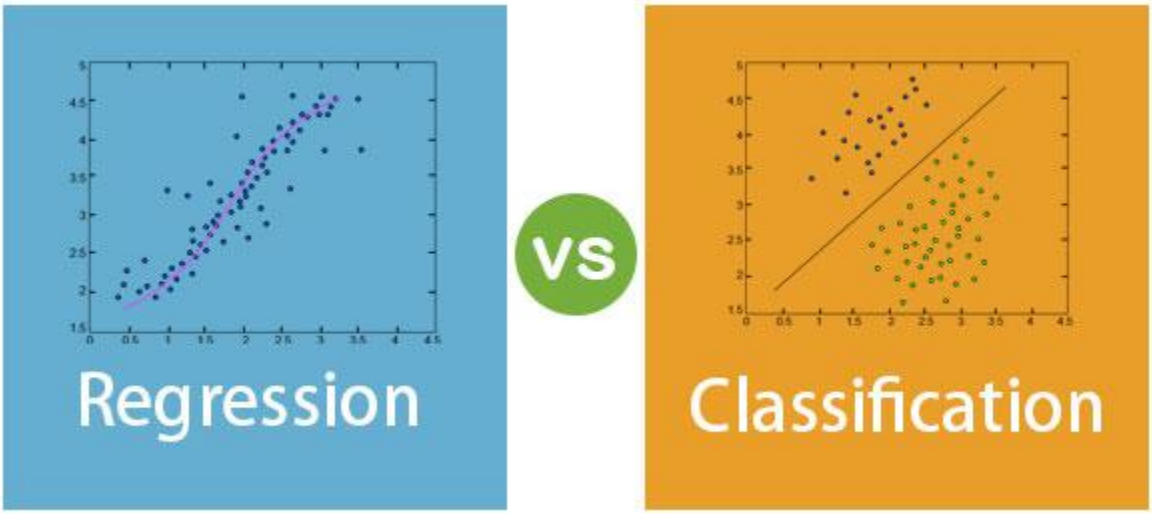


Fig 4.1 Classification vs Regression

4.3.1 Classification:

In machine learning, classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. Examples are assigning a given email to the "spam" or "nonspam" class, and assigning a diagnosis to a given patient based on observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms, etc.). Classification is an example of pattern recognition. An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The corresponding unsupervised

procedure is known as clustering and involves grouping data into categories based on some measure of inherent similarity or distance.

4.3.2 Regression:

Regression analysis estimates the conditional expectation of the dependent variable given the independent variables – that is, the average value of the dependent variable when the independent variables are fixed. Less commonly, the focus is on a quantile or other location parameter of the conditional distribution of the dependent variable given the independent variables. In all cases, a function of the independent variables called the regression function is to be estimated. Regression analysis is widely used for prediction and forecasting. It is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships.

In restricted circumstances, regression analysis can be used to infer causal relationships between the independent and dependent variables. However, this can lead to illusions or false relationships, so caution is advisable; for example, correlation does not prove causation. The type used in this model is CLASSIFICATION and so, more focus will be given to it.

4.4 APPROACHES IN CLASSIFICATION:

There are different approaches when it comes to the Classification method, they are:

1. Linear classifiers
    - 1.1 Fisher's linear discriminant
    - 1.2. Logistic regression
    - 1.3. Naive Bayes classifier

1.4. Perceptron

2. Support vector machines

2.1. Least squares support vector machines

3. Quadratic classifiers

4. Kernel estimation

4.1. k-nearest neighbour

5. Boosting (meta-algorithm)

6. Decision trees

6.1. Random forests

7. Neural networks

8. Learning vector quantization

We have used **logistic regression** and **support vector machines** in this project.

4.4.1 LOGISTIC REGRESSION

In statistics, the logistic model (or logit model) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object is detected in the image would be assigned a probability between 0 and 1, with a sum of one.

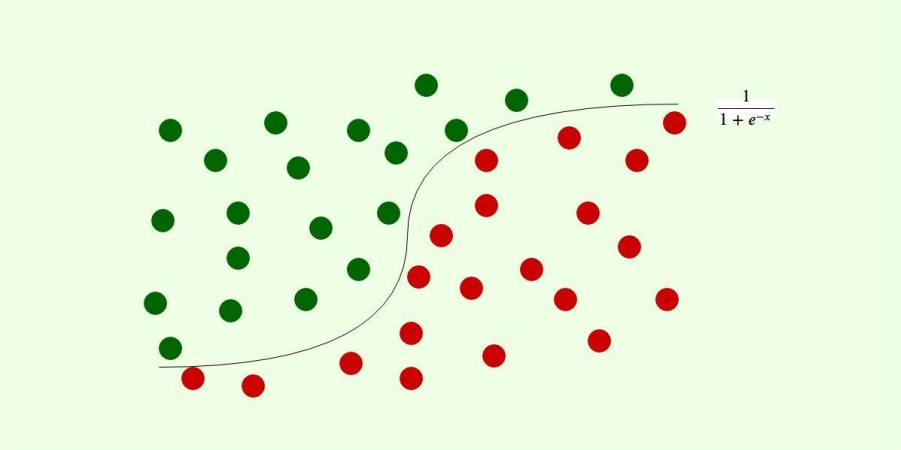


Fig 4.2 Logistic regression

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

4.4.2 SUPPORT VECTOR MACHINES

In machine learning, **support vector machines** (**SVMs**, also **support vector networks**) are supervised learning models with associated learning algorithms that analyze data for classification and regression analysis. Developed at AT&T Bell Laboratories by Vladimir Vapnik with colleagues SVMs are one of the most robust prediction methods, is based on statistical learning frameworks or VC theory proposed by Vapnik (1982, 1995) and Chervonenkis (1974).

Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier (although methods such as Platt scaling exist to use SVM in a probabilistic classification setting).

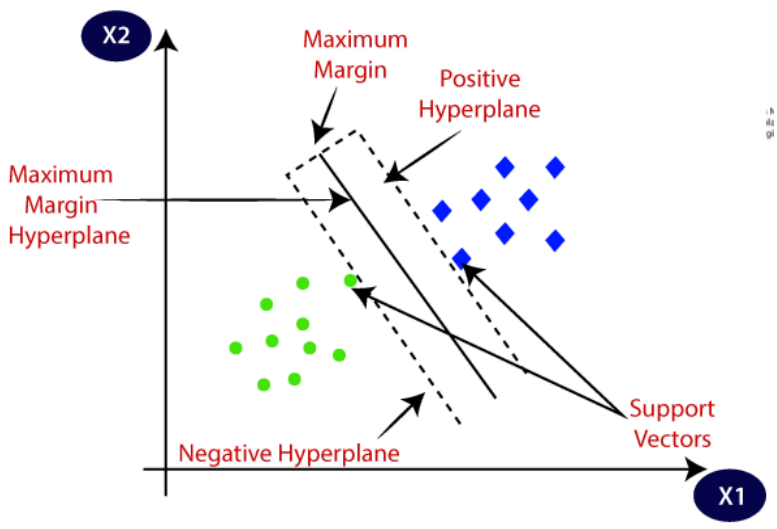


Fig 4.3 Support vector machines

SVM maps training examples to points in space so as to maximise the width of the gap between the two categories. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

4.5. PACKAGES:

The packages used in this model include:

- Pandas
- NumPy
- Scikit-learn
- Scikit-plot
- Matplotlib's
- pyplot
- Seaborn

4.5.1 DATA MANIPULATION PACKAGES:

- **Pandas**
  - pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time-series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open-source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.
  - The two primary data structures of pandas, Series (1- dimensional) and DataFrame (2-dimensional) handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, DataFrame provides everything that R's data. the frame provides and much more. pandas are built on top of NumPy and are intended to integrate well within a scientific computing environment with many other 3rd party libraries.
  - Written in: Python, Cython and C
- **NumPy**
  - NumPy is a library for Python, adding support for large, multidimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy is open-source software and has many contributors.
  - NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms are written for this version of Python often run much slower than compiled equivalents. NumPy addresses the slowness problem partly by providing

multidimensional arrays and functions and operators that operate efficiently on arrays, requiring rewriting some code, mostly inner loops using NumPy.

- Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars.
- In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality.
- Written in: Python and C

4.5.2 MODEL BUILDING PACKAGE:

➢ **Scikit-learn**

- Scikit-learn (formerly scikits.learn) is a free software machine learning library for Python programming language. It features various classification, regression and clustering algorithms including support vector machines(SVM), random forest, gradient boosting, K-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
- It has been built on top of NumPy, SciPy and matplotlib.
- Written in: Python, Cython, C and C++

4.5.3 DATA VISUALIZATION PACKAGES:

➢ **Scikit-plot**

- Scikit-plot is the result of an unartistic data scientist's dreadful realization that visualization is one of the most crucial components in the data science process, not just a mere afterthought.
- Gaining insights is simply a lot easier when you're looking at a coloured heatmap of a confusion matrix complete with class labels rather than a single-line dump of numbers enclosed in brackets. Besides, if you ever need to present your results to someone 15 (virtually any time anybody hires you to do data science), you show them visualizations, not a bunch of numbers in Excel.
- All in all, it is an intuitive library used to add plotting functionality to a scikit-learn object.
- Written in: Python, Cython, C and C++.

➢ **Matplotlib's pyplot**

- Matplotlib is a Python 2D plotting library that produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.
- It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt or GTK+.
- matplotlib.pyplot provides a MATLAB-like plotting framework.
- pylab combines pyplot with numpy into a single namespace. This is convenient for interactive work, but for programming, it is recommended that the namespaces be kept separate.
- Written in: PythonSYSTEM REQUIREMENTS SPECIFICATION

4.6 AN OVERVIEW OF OWASP TOP 10 ATTACKS

4.6.1 SQL INJECTION:

SQL injection is a vulnerability that contains malicious SQL query which triggers the database. It may lead to the leak of data that the user does not intend to see such as user data, private customer details and many others.

Fig 4.4 SQL Injection

4.6.2 XSS:

XSS stands for cross-site scripting which is a vulnerability that takes a piece of code(JS code especially) through user input. The code that is injected will propagate to other users and that code can give access to cookies, session tokens, or other sensitive information stored in the browser and used with that site.

4.6.3 CSRF:

Cross-Site Request Forgery (CSRF) is an attack that forces an authenticated end-user to send an unwanted request to a web application in which they're currently authenticated. With some techniques such as social engineering, an attacker sends a fake web page to the user using SMS or email. If a CSRF attack is successful then that request can perform many actions like Sending a message, changing the passwords and sometimes it can transfer funds.

4.6.4 DDOS:

A distributed denial-of-service (DDoS) attack is an attempt to push abnormal traffic to a targeted server or network by using multiple compromised computers operated by viruses or internet bots. Requests from these computers make servers unavailable or slow. Now actual users feel that the server is slow or sometimes the server is not available to serve the request.

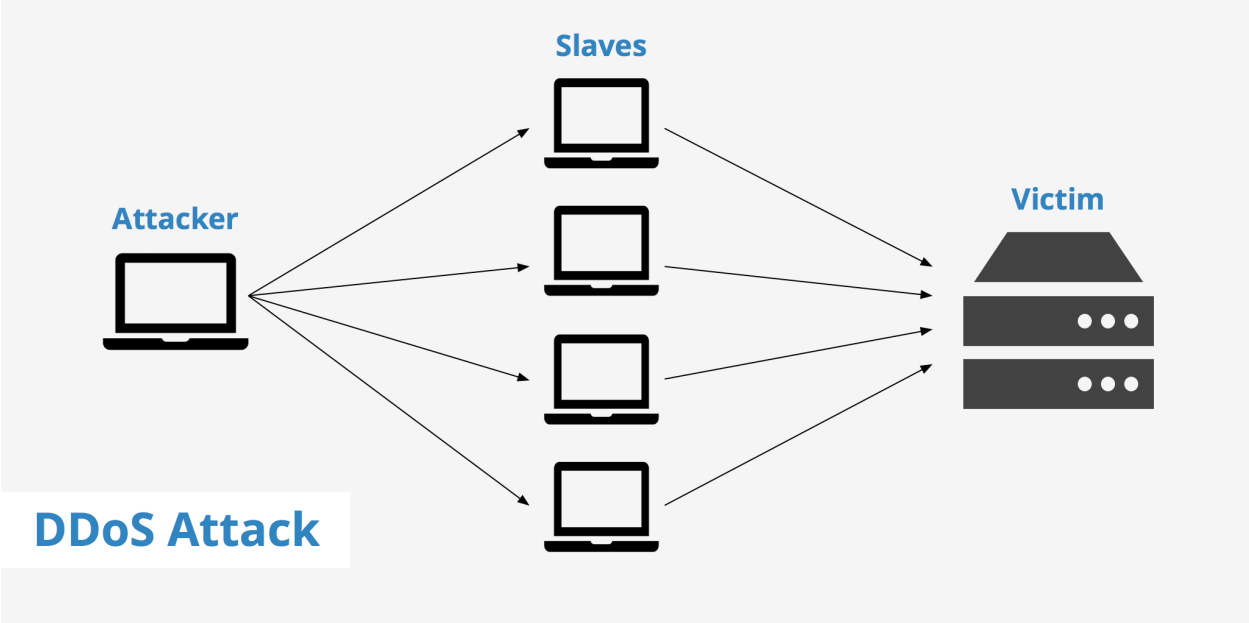


Fig 4.5 DDOS Attack

4.6.5 BRUTE FORCE:

A brute force attack is an attack that tries all combinations of words, numbers or cryptographic keys to hack into systems. In simple words, it's a trial and error method that forces applications to get access. This technique is a quite popular technique in cracking 2 Factor authentication. This method is not an efficient technique that may take a lot of time in cracking. It can take anywhere from a few seconds to years depending on the length and complexity of the password or a key.

4.6.6 IDOR:

Insecure direct object reference(IDOR) is an attack that gives access to other users' documents that are not supposed to be visible. There are some parameters such as id, user id, session id that are used by web servers in identifying users uniquely or data packets used in networks. When a hacker sees these parameters in a request tries to change the values of those parameters and gain information about other users. If the application is not secured well these types of vulnerabilities can lead to data compromise.

4.7 HARDWARE REQUIREMENTS:

Processor	: i5/i7 processor
RAM	: 4GB
Hard Disk	: 50GB
Keyboard	: Multimedia / Standard Keyboard
Mouse	: Optical Mouse
Monitor	: 17'' LCD Monitor

4.8 SOFTWARE REQUIREMENTS:

Front End	: HTML, CSS, Bootstrap
Back End	: Python
Machine learning	: Sklearn, pandas, NumPy, Seaborn
Server	: Flask web server
Operating System	: Windows 10
Web Browser	: Google Chrome (Preferred)
Text Editor	: ATOM

CHAPTER 5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

5.1 RESULTS:

The result of the classification model prepared using Python may be divided into three parts in itself:

I. Pre-model Data Visualization

II. Model Information Visualization

III. Post-model Data Visualization

DATA SET:

The data set which was used to train and test the model was taken from CICIDS which is one of the most popular datasets for ML-based firewalls. This dataset has a size of 2830743 which is in form of CSV or PCAP data fires which has been requested in the webserver.

CICIDS dataset has full request as attributes such as to request size, source, data in it and many more which are vital in their own ways in detecting the cyber attack. It has a total of 152 features. This dataset also has data for protocols like HTTP, HTTPS, FTP, SSH, and email protocols. Removing a few of the features or decreasing the size of the dataset will reduce the accuracy of the model.

5.1.1 PRE-MODEL DATA VISUALIZATION:

The pre-model data visualization section consists of all the data visualizations or example visualizations made with the pure data set to understand the features and the spread of data in the data set.

Fig 4.1 shows the distribution of the dataset into good and bad queries where 0 represent bad queries that have at least a payload that can is responsible for vulnerability and 1 represents good queries that do not have any hacking payload.

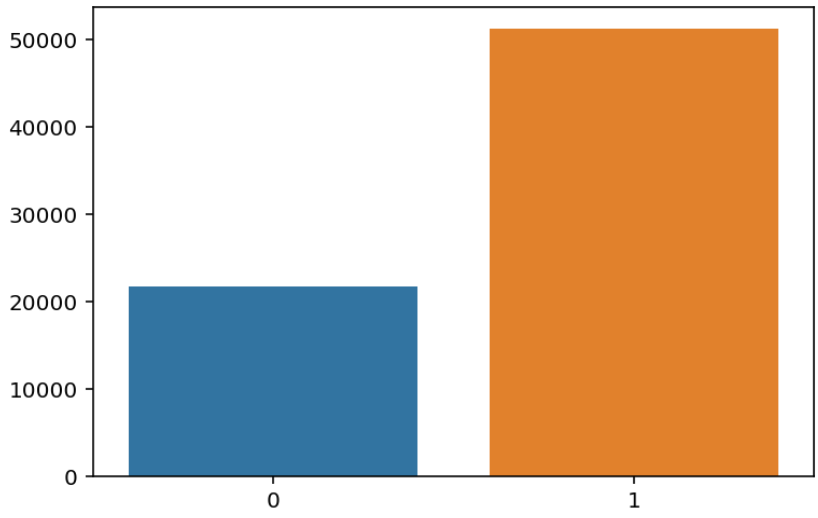


Fig 5.1 Model division

Figure 4.2 shows the distribution of attack payloads in the dataset.

Fig 5.2 Payload distribution

5.1.2 MODEL INFORMATION VISUALIZATION

In this part of the chapter, the visualizations with respect to the model will be discussed. Here is the CNN model that is responsible for recognizing payload in request.

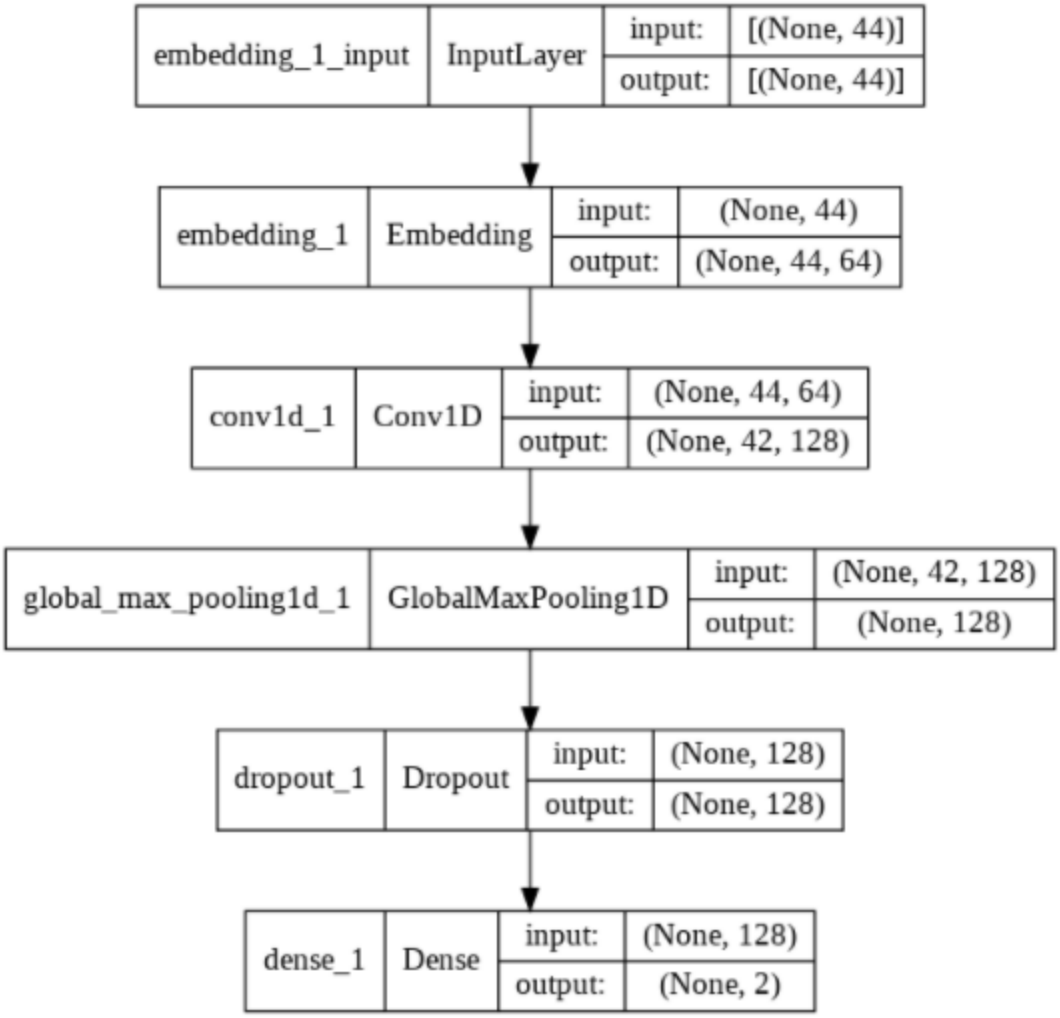


Fig 5.3 CNN Model

Classification report:

	precision	recall	f1-score	support
0	0.99	0.90	0.94	6841
1	0.92	1.00	0.96	8644
accuracy			0.95	15485
macro avg	0.96	0.95	0.95	15485
weighted avg	0.95	0.95	0.95	15485

Fig 5.4 Classification report

The above figure represents all the necessary factors taken to determine whether a model is good or bad. They are technically known as Model Evaluation Metrics.

Precision:

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$= \frac{True\ Positive}{Total\ Predicted\ Positive}$$

Fig 5.6 Precision formula

Precision is defined as the fraction of the examples which are actually positive among all the examples which we predicted positive. It is also defined as the number of true positives divided by the number of true positives plus the number of false positives. False positives are 32 cases the model incorrectly labels as positive that are actually negative, or in our example, individuals the model classifies as terrorists that are not.

Recall:

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

$$= \frac{True\ Positive}{Total\ Actual\ Positive}$$

Fig 5.7 Recall formula

The metric our intuition tells us we should maximize is known in statistics as recall, or the ability of a model to find all the relevant cases within a dataset. The precise definition of recall is the number of true positives divided by the number of true positives plus the number of false negatives. True positives are data points classified as positive by the model that actually are positive (meaning they are correct), and false negatives are data points the model identifies as negative that actually are positive (incorrect).

F1-Score:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

Fig 5.8 F-1 Score formula

Traditionally, it is defined to be the harmonic mean of precision and recall. It is also called the F Score or the F Measure. Put another way, the F1 score conveys the balance between precision and recall. It is considered to be a better measure than Precision and Recall individually as the trade-off between the two is hard to achieve.

Support:

Support is pretty simply put just the number of data rows supporting the particular class in the trained set. It is generally given to show out of what number the other metrics have been calculated.

5.2 PERFORMANCE ANALYSIS

The performance of python is optimised to maximum using python to utilize less space and memory which helps in predicting data in a faster way and to provide users very secure way to use the website for all types of attacks, all the python modules like Flask and Sklearn and NumPy etc are chosen very carefully in order to gain very efficiency in space and speedy execution of the program in order to make the user feel safe and comfortable in using a firewall. Also, it uses as less as 10GB of storage for training and maintenance of the ML model and is used as less than 1GB ram to compute 1lakh+ of predictions, which is quite small compared to previous firewalls. It can be further trained to detect DDOs attacks and authentication attacks so that web servers are much more secure.

5.3 DISCUSSION

A lot of discussions has done among cyber security researchers and many hackers and Cyber security professionals and research on thousands of security data breaches has been done in the field of cyber security before building this tool and dataset. It is very efficient in storage and speed even for small servers or websites. It is the combined product of many security professionals, researchers, many machine learning developers and data scientists.

CHAPTER 6
SUMMARY AND CONCLUSION

6.1 SUMMARY

This project was created to further create new opportunities for detecting cyber attcks in early stages or even before its occurrence.But due to the difficulties faced for collecting proper and reliable data to process and produce optimal results have been a major drawback for the research.

To summarise, we have developed a machine learning-based firewall which is being used to secure networks and web servers in a most effective way Ml model is trained using a dataset which consists of 12lakh+ web server request data which is being collected from web servers from data breaches and web server attacks. ML model has attained an accuracy of 97.5% and can compute 1,00,000+ request predictions in less than a second. It is specifically trained to prevent SQL injection, RCE, XSS and directory-based attacks.

The difficulties faced has caused this efficient idea to be only researched upon in limited terms. The success of this research, included that reliable data is sought, can produce revolutionary results and may change the course of how cyber attacks is being detected in the future.

6.2 FUTURE SCOPE:

I am thankful for this opportunity to work on this project. This project is one of my long thought research ideas which may turn to be useful. The future of this project has several opportunities. This project, by my plan, is to be extended for further research with respect to my goal of establishing newer, more reliable and flexible methods of

detecting cyber attacks. This inter-disciplinary approach may yield a great result if treaded rightly.

6.3 CONCLUSION

Machine learning model for the firewall is developed, deployed and ready to use by web servers. ML model achieved an accuracy of 97.5% and can detect most of the server attacks and injections into web servers. API has been developed to use the ML model easy. Further, it can be trained to detect DDOS, Suspicious login and monitoring user activity.

REFERENCES

1. G. De Carvalho Bertoli et al., "An End-to-End Framework for Machine Learning-Based Network Intrusion Detection System," in IEEE Access, vol. 9, pp. 106790-106805, 2021, doi: 10.1109/ACCESS.2021.3101188.
2. G. Betarte, Á. Pardo and R. Martínez, "Web Application Attacks Detection Using Machine Learning Techniques," 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), 2018, pp. 1065-1072, doi: 10.1109/ICMLA.2018.00174.
3. Mishra, Archana & Agrawal, Abhishek & Ranjan, Rajeev. (2011). Artificial intelligent firewall. 10.1145/2007052.2007094.
4. Q. Zhao, J. Sun, H. Ren and G. Sun, "Machine-Learning Based TCP Security Action Prediction," 2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE), 2020, pp. 1329-1333, doi: 10.1109/ICMCCE51767.2020.00291.
5. Anaconda Free Distribution - https://www.anaconda.com/
6. Jupyter Notebook - https://jupyter.org/
7. Data Science - https://en.wikipedia.org/wiki/Data_science
8. Machine Learning - https://en.wikipedia.org/wiki/Machine_learning
9. Scikit-learn - https://scikit-learn.org/stable/
10. Pandas - https://pandas.pydata.org/
11. Matplotlib - https://matplotlib.org/
12. Seaborn-https://seaborn.pydata.org/
13. Random Forest - https://en.wikipedia.org/wiki/Random_forest
14. R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning for Network Intrusion Detection," 2010 IEEE Symposium on Security and Privacy, 2010, pp. 305-316, doi:10.1109/SP.2010.25.
15. Lin, Wei-Chao & Ke, Shih-Wen & Tsai, Chih-Fong. (2015). CANN: An Intrusion Detection System Based on Combining Cluster Centers and Nearest Neighbors. Knowledge-Based Systems. 78.10.1016/j.knosys.2015.01.009.
16. Elhag, Salma & Fernández, Alberto & Bawakid, Abdullah & Alshomrani, Saleh & Herrera, Francisco.(2015). On the combination of genetic fuzzy systems and pairwise learning for improving detectionrates on Intrusion Detection Systems. Expert Systems

with Applications. 42. 193–202.10.1016/j.eswa.2014.08.002. O. Can and O. K. Sahingoz, "A survey of intrusion detection systems in wireless sensor networks,"

17. 2015 6th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO),
18. 2015, pp. 1-6, doi: 10.1109/ICMSAO.2015.7152200.
19. Razzaq, A., Latif, K., Ahmad, H., Hur, A., Anwar, Z. and Bloodsworth, P., 2014. Semantic security
20. against web application attacks. Information Sciences, 254, pp.19-38.
21. Lambert, Glenn M. II, "Security Analytics: Using Deep Learning to Detect Cyber Attacks"
22. (2017). UNF Graduate Theses and Dissertations. 728. https://digitalcommons.unf.edu/etd/728
23. G. Creech, “Developing a high-accuracy cross platform Host-Based Intrusion Detection System capable of reliably detecting zero-day attacks”, 2014. Khraisat, A., Gondal, I., Vamplew, P. and Kamruzzaman, J., 2019. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity, 2(1).
24. Gaddam, H. and Maheshwari, M., 2022. SQL Injection-Biggest Vulnerability of the Era. [online] Easychair.org. Available at: <https://easychair.org/publications/preprint/mptV> [Accessed 12 January 2022].
25. Zhang, X., Zhou, Y., Pei, S., Zhuge, J. and Chen, J., 2020. Adversarial Examples Detection for XSS Attacks Based on Generative Adversarial Networks. IEEE Access, 8, pp.10989-10996.
26. Dong, S., Abbas, K. and Jain, R., 2019. A Survey on Distributed Denial of Service (DDoS) Attacks in SDN and Cloud Computing Environments. IEEE Access, 7, pp.80813-80828. Siddiqui, M. and Verma, D., 2011. Cross site request forgery: A common web application weakness. 2011 IEEE 3rd International Conference on Communication Software and Networks,. Salamatian, S., Huleihel, W., Beirami, A., Cohen, A. and Medard, M., 2020. Centralized vs Decentralized Targeted Brute-Force Attacks: Guessing With Side-Information. IEEE Transactions on Information Forensics and Security, 15, pp.3749-3759.
27. En.wikipedia.org. 2022. Insecure direct object reference - Wikipedia. [online] Available at:<https://en.wikipedia.org/wiki/Insecure_direct_object_reference> [Accessed 12 January 2022].

28. Savas, O., & Deng, J. (Eds.). (2017). Big Data Analytics in Cybersecurity (1st ed.). Auerbach Publications. https://doi.org/10.1201/9781315154374
29. Pacheco, F., Exposito, E., Gineste, M., Baudoin, C. and Aguilar, J., 2019. Towards the Deployment of Machine Learning Solutions in Network Traffic Classification: A Systematic Survey. IEEECommunications Surveys & Tutorials, 21(2), pp.1988-2014.
Icsdweb.aegean.gr. 2022. AWID - Aegean Wi-Fi Intrusion Dataset. [online] Available at:<http://icsdweb.aegean.gr/awid/features.html> [Accessed 12 January 2022].
30. “Intrusion Detection Evaluation Dataset,” Canadian Institute for Cybersecurity, 2017. [Online].Available: https://www.unb.ca/cic/datasets/ids2017.html
31. Blog, S., 2022. SIGKDD : KDD Cup 1999 : Computer network intrusion detection. [online] Kdd.org.Available at: <http://www.kdd.org/kdd-cup/view/kdd-cup-1999> [Accessed 12 January 2022].
32. M. Tavallaee, E. Bagheri,W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 dataset,” in Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl., 2009, pp. 1–6.
33. Takakura.com. 2022. Traffic Data from Kyoto University's Honeypots. [online] Available at:<http://www.takakura.com/Kyoto_data> [Accessed 12 January 2022].
34. N. Moustafa and J. Slay, “UNSW-NB15: A comprehensive data set for network intrusion detection systems,” in Proc. Mil. Commun. Inf. Syst. Conf., 2015, pp. 1–6.

APPENDIX

SOURCE CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
import warnings
warnings.filterwarnings('ignore')
data = pd.read_excel("dataset.xlsx")
data = data.dropna(axis = 0)
data.shape
data.head()
data_pos=data[data["type"].isin([1])]
data_pos.head()
data_pos.tail()
data_neg=data[data["type"].isin([0])]
data_neg.head()
data_neg.tail()
data.type.value_counts()
sns.barplot(x=data.type.value_counts().index,y=data.type.value_counts().values)
from sklearn.model_selection import train_test_split
X_train_data,x_test_data,Y_train_data,y_test_data=train_test_split(data["query"],data["type"],test_si
ze=0.3)
Y_train_data.head()
X_train_data.head()
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
tfidf_vector = TfidfVectorizer()
X_train_data = [str (item) for item in X_train_data]
tfidf_vector.fit(X_train_data)
print(tfidf_vector.get_feature_names()[0:20])
print(tfidf_vector.get_feature_names()[-20:])
X_train_data_new=tfidf_vector.transform(X_train_data)
X_train_data_new.shape
x_test_data = [str (item) for item in x_test_data]
x_test_data_new=tfidf_vector.transform(x_test_data)
```

```
predictions = dict()
from sklearn.linear_model import LogisticRegression
lr_model = LogisticRegression()
lr_model.fit(X_train_data_new,Y_train_data)
predictions["LogisticRegression"] = lr_model.predict(x_test_data_new)
from sklearn.metrics import classification_report,accuracy_score,confusion_matrix
accuracy_score(y_test_data,predictions["LogisticRegression"])
print(classification_report(y_test_data, predictions['LogisticRegression'], target_names = ["Good",
"Bad"]))
matrix = confusion_matrix(y_test_data, predictions['LogisticRegression'])
matrix_normalized = matrix.astype('float') / matrix.sum(axis=1)[:, np.newaxis]
sns.heatmap(matrix_normalized)
plt.ylabel('Actual')
plt.xlabel('Predicted')
print_results = {}
for k,v in predictions.items():
    print_results[k] = accuracy_score(y_test_data,v)
print_results
result_table=pd.DataFrame(list(print_results.items()), columns=["Model","Accuracy"])
result_table
plt.figure(figsize= (10,8))
sns.barplot(x = "Model", y = "Accuracy", data = result_table)
plt.title("Model accuracy")
plt.xticks(rotation = 90)
```

CNN MODEL:

```
import matplotlib.pyplot as plt
import os
import re
import shutil
import string
import tensorflow as tf
from tensorflow.keras import regularizers
from tensorflow.keras import layers
from tensorflow.keras import losses
from collections import Counter
import pandas as pd
```

```
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from tensorflow.keras import preprocessing
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
import pydot
def clean_text(text1):
    textArr= text1.split()
    text2 = ' '.join([w for w in textArr if (not w.isdigit())])
    return text2.lower()
train_data= pd.read_excel("train_dataset.xlsx")
train_data.dropna(axis = 0, how ='any',inplace=True)
train_data['Num_words_text'] = train_data['query'].apply(lambda x:len(str(x).split()))
train_data['query'] = train_data['query'].apply(lambda x: str(x))
print('-------Train data--------')
print(train_data['type'].value_counts())
print(len(train_data))
print('-------------------------')
max_train_sentence_length  = train_data['Num_words_text'].max()


test_data= pd.read_excel("test_dataset.xlsx")
test_data.dropna(axis = 0, how ='any',inplace=True)
test_data['Num_words_text'] = test_data['query'].apply(lambda x:len(str(x).split()))
test_data['query'] = test_data['query'].apply(lambda x: str(x))
max_test_sentence_length  = test_data['Num_words_text'].max()
print('-------Test data--------')
print(test_data['type'].value_counts())
print(len(test_data))
print('-------------------------')

test_data['query'] = test_data['query'].apply(clean_text)

print('Train Max Sentence Length :'+str(max_train_sentence_length))
print('Test Max Sentence Length :'+str(max_test_sentence_length))
```

```
num_words = 200000

tokenizer = Tokenizer(num_words=num_words,oov_token="unk",lower=False)
tokenizer.fit_on_texts(train_data['query'])

# print(str(tokenizer.texts_to_sequences(['xyz how are you'])))
X_train, X_valid, y_train, y_valid = train_test_split(train_data['query'].tolist(),\
                                        train_data['type'].tolist(),\
                                        test_size=0.2,\
                                        stratify = train_data['type'].tolist(),\
                                        random_state=0)



print('Train data len:'+str(len(X_train)))
print('Class distribution'+str(Counter(y_train)))
print('Valid data len:'+str(len(X_valid)))
print('Class distribution'+ str(Counter(y_valid)))


x_train = np.array( tokenizer.texts_to_sequences(X_train) )
x_valid = np.array( tokenizer.texts_to_sequences(X_valid) )
x_test  = np.array( tokenizer.texts_to_sequences(test_data['query'].tolist()) )



x_train = pad_sequences(x_train, padding='post', maxlen=44)
x_valid = pad_sequences(x_valid, padding='post', maxlen=44)
x_test = pad_sequences(x_test, padding='post', maxlen=44)

print(x_train[0])

le = LabelEncoder()

train_labels = le.fit_transform(y_train)
train_labels = np.asarray( tf.keras.utils.to_categorical(train_labels))
#print(train_labels)
valid_labels = le.transform(y_valid)
```

```
valid_labels = np.asarray( tf.keras.utils.to_categorical(valid_labels))

test_labels = le.transform(test_data['type'].tolist())
test_labels = np.asarray(tf.keras.utils.to_categorical(test_labels))
list(le.classes_)


train_ds = tf.data.Dataset.from_tensor_slices((x_train,train_labels))
valid_ds = tf.data.Dataset.from_tensor_slices((x_valid,valid_labels))
test_ds = tf.data.Dataset.from_tensor_slices((x_test,test_labels))

print(y_train[:10])
train_labels = le.fit_transform(y_train)
print('Text to number')
print(train_labels[:10])
train_labels = np.asarray( tf.keras.utils.to_categorical(train_labels))
print('Number to category')
print(train_labels[:10])
count =0
print('======Train dataset ====')
for value,label in train_ds:
    count += 1
    print(value,label)
    if count==3:
        break
count =0
print('======Validation dataset ====')
for value,label in valid_ds:
    count += 1
    print(value,label)
    if count==3:
        break
print('======Test dataset ====')
for value,label in test_ds:
    count += 1
    print(value,label)
    if count==3:
```

```
        break
max_features =200000
embedding_dim =64
sequence_length = 44

model = tf.keras.Sequential()
model.add(tf.keras.layers.Embedding(max_features +1, embedding_dim,
input_length=sequence_length,\
                              embeddings_regularizer = regularizers.l2(0.0005)))

model.add(tf.keras.layers.Conv1D(128,3, activation='relu',\
                             kernel_regularizer = regularizers.l2(0.0005),\
                             bias_regularizer = regularizers.l2(0.0005)))



model.add(tf.keras.layers.GlobalMaxPooling1D())

model.add(tf.keras.layers.Dropout(0.5))

model.add(tf.keras.layers.Dense(2, activation='sigmoid',\
                            kernel_regularizer=regularizers.l2(0.001),\
                            bias_regularizer=regularizers.l2(0.001),))




model.summary()
model.compile(loss=tf.keras.losses.CategoricalCrossentropy(from_logits=True), optimizer='Nadam',
metrics=["CategoricalAccuracy"])
epochs = 25
# Fit the model using the train and test datasets.
history = model.fit(train_ds.shuffle(2000).batch(128),
                    epochs= epochs ,
                    validation_data=valid_ds.batch(128),
                    verbose=1)
```

WEB CODE:

```
from flask import Flask,request,render_template,redirect, url_for, jsonify
app = Flask(__name__,template_folder='templates')
import json

# load the model from disk
import pickle

# filename = 'model.pkl'
clf = pickle.load(open('model.pkl', 'rb'))
cv=pickle.load(open('tranform.pkl','rb'))

@app.route('/',methods=["GET"])
def home():
  return render_template('home.html')

@app.route('/',methods=["POST"])
def querycheck():
  query = [request.form['query']]
  vect = cv.transform(query).toarray()
  opt = clf.predict(vect)
  opt = list(opt)
  print(opt)
  if(opt[0]==1):
    return render_template('good.html')
  else:
    return render_template('sus_act.html')

@app.route('/api',methods=["POST"])
def check():
  requestData = request.data
  requestData = json.loads(requestData.decode('utf-8'))
  print(requestData)
  # data=list(requestData.values())
  data = requestData

  if(len(data)==0):
```

```
      response = app.response_class(
      response=json.dumps({"Response":False}),
      status=200,
      mimetype='application/json'
      )
      return response

    vect = cv.transform(data).toarray()
    opt = clf.predict(vect)
    opt = list(opt)
    print(opt)
    if 0 in opt:
      response = app.response_class(
      response=json.dumps({"Response":True}),
      status=200,
      mimetype='application/json')
      return response
    else:
      response = app.response_class(
      response=json.dumps({"Response":False}),
      status=200,
      mimetype='application/json'
      )
      return response



if __name__ == '__main__':
    app.run(debug=True)

const express =require('express');
const router=express.Router({mergeParams: true});
const sqlConn = require('../utils/dbConfig').sqlConnection;
const api = require("../utils/apiConf");



router.get("/",(req,res) => {
    const id = req.query.id;
```

```
    sqlConn.query("SELECT * FROM posts WHERE id ='"+id+"' ;", (err, postResults, fields) => {
        if(err) {
            console.log("Posts error "+err);
            return err;
        }
        sqlConn.query("SELECT * FROM comments WHERE postid ='"+id+"' ;", (err, commentResults,
fields) => {
            if(err) {
                console.log("Comments error "+err);
                return err;
            }
            var data = {
                "post": postResults,
                "comments" : commentResults,
            };
            if(postResults.length > 0){
                res.render("postsPage",{"data":data});
            }else{
                res.send("Post not found");
            }
        });
    });
});

router.post("/comment/add",async(req,res)=>{
    const id = req.query.id;
    const comment = req.body.comment;
    var result = await api([comment]);
    var result = result.data.Response;
    console.log(result)
    if(!result){
        sqlConn.query("INSERT INTO `comments` (`postid`, `comment`) VALUES
('"+id+"','"+comment+"');",(err,results, fields) => {
            if(err) {
                console.log("Comment add error "+err);
                return err;
            }
```

```
            res.redirect("back");
        });
    }else{
        res.render("sus");
    }
});

// Export models
module.exports=router

const express =require('express');
const router=express.Router({mergeParams: true});
const sqlConn = require('../utils/dbConfig').sqlConnection;
const api = require("../utils/apiConf");

router.get("/",(req,res)=>{
    res.send("Admin home");
})

router.get("/login",(req,res)=>{
    res.render("login");
});

router.post("/login",async (req,res)=>{
    const uname = req.body.username;
    const pass = req.body.pass;
    var inpData = [uname,pass];
    var result = await api(inpData);
    var result = result.data.Response;
    console.log(result)
    if(!result){
        if(uname && pass){
            sqlConn.query("SELECT * FROM admin WHERE username ='"+uname+"' AND
pass='"+pass+"' ;", (error, results, fields) => {
                console.log(results);
                if(results.length > 0){
                    // Authenticate the user
```

```
                req.session.loggedin = true;
                res.redirect("/");
            }else{
                res.send("Failed to login");
            }
        });
    }else{
        res.send("Please enter username and pass");
    }
  }else{
    res.render("sus");
  }

});

// Export models
module.exports=router

const express = require('express');
const app = express();
app.use(express.static('public'));
app.set("view engine","ejs");
const bodyparser=require("body-parser");
app.use(bodyparser.urlencoded({extended:true}));
app.use(express.json());
const session = require('express-session');

// Session secret
app.use(session({
  secret: 'secret',
  resave: true,
  saveUninitialized: true,
}));

// Importing database connection
const dbConnection = require("./utils/dbConfig").sqlConnection;
```

```
const apiConnection = require("./utils/apiConf");

// Impoting routes
const adminRoute = require("./routes/admin");
const postsRoute = require("./routes/posts");



app.use("/admin",adminRoute);
app.use("/post",postsRoute);



// Routes
app.get("/",(req,res)=>{
  dbConnection.query('SELECT * FROM posts ORDER BY time DESC;', function (error, results,
fields) {
    if (error) throw error;
    var data = {
      "session":req.session.loggedin,
      "posts":results,
    }
    res.render("home",{"data":data});
    });
})




app.listen(3002,() => {
  console.log("Running on port 3002")
})
const axios = require('axios').default;

const api = async(data)=>{
    try{
        const result = await axios.post("https://payload-clasify.herokuapp.com/api",data);
        return result
    }
    catch(error){
```

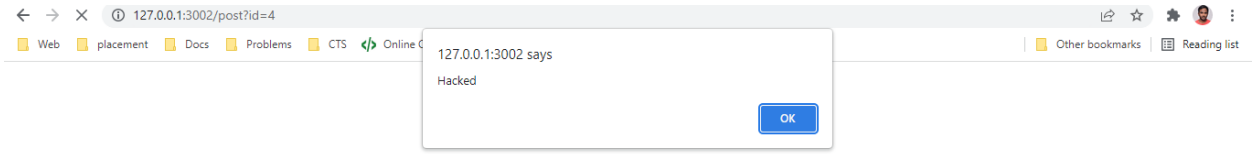
```
        console.log(error)
        return error
    }
}

module.exports = api;
const mysql = require('mysql');

var config = {
    sqlConnection : mysql.createConnection({
        host     : 'localhost',
        user     : 'root',
        database : 'finalproject'
    }),
};


module.exports = config;
```

SCREENSHOTS



Suspecious activity detected😕