



## Project Title

# Real-Time IP Traffic Classification in Virtual Private Network using Machine Learning

Prepared by

Youssef S. Ezz-Eldeen	120160657
Sohil F. Barhoom	120160336
Albaraa S. Almughaiyer	120162728
Ameer A. Abu Mhady	120160604

Supervised by

**Dr. Shadi Abudalfa**

A Graduation Project Submitted to the Engineering Department in partial fulfillment of the requirements of Bachelor of Information Security Engineering

Palestine, Gaza

{5/7/2021}

## I. ACKNOWLEDGMENT

In the beginning, praise be to God, who honored us with the completion of this project and its completion to the fullest.

We thank everyone who stood by us and helped us complete this scientific project, and we especially mention Department of Engineering and Planning at the University College of Applied Sciences for the facilities, assistance and guidance they provided for us over the five years Moreover, we thank the supervisor of the graduation project, Dr. Shadi Abudalfa, for his patience Guidance throughout the semester. In addition, moreover, it is our duty to thank everyone Members of the Projects Committee for their fruitful discussions and encouragement.

Finally, we would like to thank everyone who helped, supported and encouraged us to do so Complete our graduation project. In addition, we give a lot of thanks and gratitude Our wonderful families who were and still are the foundation and sources of strength and support us all

## II. DECLARATION

We hereby certify that this material, which we are now offering for assessment in the program of study leading to a Bachelor of Science in Information Security Engineering, is entirely our work, and that we have taken reasonable care to ensure that the work is original, and does not, to the best of our knowledge, infringe any copyright law.

### **Signed:**

Youssef S. Ezz-Eldeen	120160657
Sohil F. Barhoom	120160336
Albaraa S. Almughaiyer	120162728
Ameer A. Abu Mhady	120160604

**Date: 5, July 2021**

### III. ABSTRACT

With the terrible technological acceleration and the spread of websites for companies and institutions and the opening of some of their services to the digital world, it was necessary to protect the internal systems of these companies, we notice the development in the field of protection and information security is very large, but the bad guys are able to hide their personal identity using new methods such as VPN, which invade the world Very large, so the idea of this project relies on artificial intelligence, machine learning, to classify people who have access to institution systems.

When a new service is made available to users via the Internet, the computer team in the company must study the risks and protections for the service and systems in general, with the spread of the Coved-19 epidemic in the year 2019-2020, became all jobs online, and with e-learning, access to services for institutions has become very large Human beings will not be able to distinguish whether a person at this time is using a VPN or not, and this made the idea grow greatly among our team.

## IV. TABLE OF CONTENTS

I. ACKNOWLEDGMENT.....	2
II. DECLARATION.....	3
III. ABSTRACT.....	4
IV. TABLE OF CONTENTS.....	5
LIST OF TABLES .....	7
LIST OF FIGURES.....	<b>Error! Bookmark not defined.</b>
<b>1. INTRODUCTION .....</b>	<b>8</b>
1.1 OVERVIEW .....	8
1.2 PROBLEM STATEMENT .....	9
1.3 OBJECTIVES .....	9
1.4 IMPORTANCE OF THE PROJECT .....	9
1.5 SCOPE .....	10
1.6 OVERVIEW OF DOCUMENTATION .....	10
<b>2. BACKGROUND.....</b>	<b>11</b>
2.1 REAL-TIME IP TRAFFIC CLASSIFICATION IN VIRTUAL PRIVATE NETWORK USING MACHINE LEARNING .....	11
2.1.1 OVERVIEW .....	11
2.2 TYPES OF VPN IMPLEMENTATIONS.....	12
2.3 OPENVPN.....	12
2.4 MACHINE LEARNING .....	14
2.4.1 Machine Learning Process .....	14
2.4.2 Machine learning Techniques .....	19
2.5 GOOGLECOLABORATORY ENVIRONMENT .....	20
2.6 RAPIDER MINER.....	21
2.7 CICFLOWMETER.....	21
<b>3. RELATED WORKS.....</b>	<b>22</b>
3.1 LITERATURE SURVEY .....	22
3.2 RELATED IP TRAFFIC CLASSIFICATION SYSTEMS AND SOFTWARE .....	23
3.3 TRAFFIC CLASSIFICATION USING MACHINE LEARNING .....	23
3.4 LIBPROTOIDENT .....	23
3.5 SPID (STATISTICAL PROTOCOL IDENTIFICATION) .....	24
3.6 L7-FILTER .....	24

3.7	DJANGO .....	24
3.8	PYTHON.....	24
<b>4.</b>	<b>DATASET AND METHODOLOGY .....</b>	<b>26</b>
4.1	DATASET .....	26
4.1.1	Dataset Description.....	26
4.1.2	Dataset Analysis .....	28
4.1.3	Preprocessing for dataset .....	29
4.1.4	Training Dataset .....	29
4.1.5	Test Dataset.....	29
4.2	MACHINE LEARNING CLASSIFIER .....	29
4.2.1	4.2.1 Feature Engineering .....	29
4.2.2	Classification Model .....	30
<b>5.</b>	<b>EXPERIMENTAL SETUP, RESULTS AND ANALYSIS .....</b>	<b>31</b>
5.1	EXPERIMENTAL SETUP .....	31
5.1.1	Machine learning Model.....	31
5.1.2	VPN/ non-VPN Classifier Web Application.....	36
5.2	RESULT AND ANALYSIS .....	37
5.2.1	Result for Machine Learning Model.....	37
5.2.2	Result for Classification in VPN/non-VPN Web Application .....	40
<b>6.</b>	<b>CONCLUSION AND FUTUURE WORK.....</b>	<b>41</b>
6.1	OVERVIEW .....	41
6.2	6.2 LIMITATIONS .....	41
6.3	FUTURE WORK .....	41
6.4	CONCLUSION .....	41
6.5	REFERENCES .....	42

## LIST OF TABLES

Table 3-1 Comparison of Network VPN papers and research.....	25
Table 4-1Features in ISCXVPN2016 dataset .....	28

## LIST OF FIGURES

Figure 2.1 Vpn Architecture Design .....	14
Figure 2.2 Machine Learning Process .....	15
Figure 4.1 describe infrastructure for ISCXVPN2016 dataset .....	26
Figure 5.1 CICFlowMeter Tool .....	34
Figure 5.2VPN and non-VPN file for feature .....	34
Figure 5.3KNN algorithm experiment results .....	37
Figure 5.4 SVM algorithm experiment results .....	38
Figure 5.5 NB algorithm experiment result .....	38
Figure 5.6 LR algorithm experiment results.....	39
Figure 5.7 GBTs algorithm experiment results.....	39
Figure 5.8 RF algorithm experiment results.....	40
Figure 5.9 Result for Classification in VPN/non-VPN Web Application.....	40

## *Chapter One*

# **1. INTRODUCTION**

This chapter is about a general introduction providing an overview of the topic, problem statement, and the project description followed by the objectives and the importance of this project. This chapter ends with a section that states the contents of the next chapters.

## **1.1 OVERVIEW**

With the terrible technological acceleration and the spread of websites for companies and institutions and the opening of some of their services to the digital world, it was necessary to protect the internal systems of these companies, we notice the development in the field of protection and information security is very large, but the bad guys are able to hide their personal identity using new methods such as VPN, which invade the world Very large, so the idea of this project relies on artificial intelligence, machine learning, to classify people who have access to institution systems.

When a new service is made available to users via the Internet, the computer team in the company must study the risks and protections for the service and systems in general, with the spread of the Coved-19 epidemic in the year 2019-2020, became all jobs online, and with e-learning, access to services for institutions has become very large Human beings will not be able to distinguish whether a person at this time is using a VPN or not, and this made the idea grow greatly among our team.

Traffic classification can be categorized based on its final purpose: associating traffic with encryption (e.g., encrypted traffic), protocol encapsulation (e.g., tunneled through VPN or HTTPS); according to specific applications, (e.g., Skype), or according to the application type (e.g., Streaming, Chat), also called traffic



characterization. Some applications (e.g., Skype, Facebook) support multiple services like chat, voice call, file transfer, etc. These applications require identifying both the application itself and the specific task associated with it.

## **1.2 PROBLEM STATEMENT**

A lot of institutions are hacked and the Hacker is not known to use VPN. This always puts the systems in the hands of hackers even if the system is developed without deterring the actor, so the idea of the project came and the supervisor presented the idea to us and it was accepted.

## **1.3 OBJECTIVES**

The goal of the project is to classify the next package for the organization as follows: vpn or non vpn, and this makes detecting hackers faster and addressing the error before it occurs.

This objective reveals some objectives as follows:

- ✚ Collection and processing of data for Real-Time IP Traffic Classification in Virtual Private Network.
- ✚ Apply different machine learning techniques to the data set.
- ✚ Evaluate the performance of various machine learning techniques applied to Real-Time IP Traffic Classification in Virtual Private Network and correct the error before it happens.
- ✚ Implementation of a Real-Time IP Traffic Classification in Virtual Private Network system based on machine learning Technique.
- ✚ The project worked on all systems and could be developed in the future to be API

## **1.4 IMPORTANCE OF THE PROJECT**

The importance of this project is to identify and deal with suspicious traffic on the network before anything goes wrong, and information security professionals may take steps before hackers to close the targeted vulnerabilities.

## **1.5 SCOPE**

For any system, users or companies that have sensitive data need protection and want to secure their networks and protect their servers or computers from being attacked.

## **1.6 OVERVIEW OF DOCUMENTATION**

The remaining chapters of the documentation are organized as follow:

- Chapter 2: Background: this chapter provides a brief background about our project.
- Chapter 3: Related literature reviews: it provides an overview of related research and applications related to ours. It presents a comparison between related applications.
- Chapter 4: Methodology: this chapter covers the detailed explanation of methodology that is being used to make the Real-Time IP Traffic Classification in Virtual Private Network system complete and work well.
- Chapter 5: Experimental Setup and Results and Analysis: It contains a testing plan and the result of that. Also, it includes evaluation of the project in terms of cost, environmental and social impact, manufacturability, health and safety, and sustainability.
- Chapter 6: The final chapter presenting conclusions and future ideas to develop the Real-Time IP Traffic Classification in Virtual Private Network System.

## **2. BACKGROUND**

In this chapter, we will present an overview of the VPN, and its definition, what are its uses in our time and why do we need it and the impact of machine learning technology on development in this field

### **2.1 REAL-TIME IP TRAFFIC CLASSIFICATION IN VIRTUAL PRIVATE NETWORK USING MACHINE LEARNING**

In this section of background, we describe information about Virtual Private Network.

#### **2.1.1 OVERVIEW**

Network (VPN) is a service which hides real traffic by creating SSL protected channel between the user and the server. Every internet activity is then performed under the established SSL tunnel.

User inside the network with malicious intent or to hide his activity from the network security administration of the organization may use VPN services. Any VPN service may be used by users to bypass the filters or signatures applied on network security devices. These services may be the source of new virus or worm injected inside the network or a gateway to facilitate information leakage.

In this paper we have proposed a novel approach to detect VPN activity inside the network. The proposed system analyses the communication between user and the server to analyse and extract features from network, transport and application layer which are not encrypted and classify the incoming traffic as malicious i.e., VPN traffic or standard traffic.

It should also be noted that while VPN's may be constructed to address any number of specific business needs or technical requirements, a comprehensive VPN solution provides support for dial-in access, multiple remote sites connected by leased lines (or other dedicated means), the ability of the VPN service provider to "host" various services for the VPN customers (e.g., web hosting), and the ability to support not just intra-, but also inter-VPN connectivity, including connectivity to the global Internet. [1]

## **2.2 TYPES OF VPN IMPLEMENTATIONS**

There are two types of VPN, namely Remote Access VPN and Site to Site VPN

### **a) Remote access VPN**

VPDN is a type of user-to-LAN connection, the connection that connects a mobile user with a Local Area Network (LAN). This means that the user can access the private network from anywhere. Usually, this VPDN utilized by employees who are out of the Office and require a connection to the Office network of the company. Usually, companies that want to make this type of VPN network will work closely with Enterprise Service Provider (ESP). ESP will provide a Network Access Server (NAS) for the company. ESP will also provide special software to computers used by employees of the company

### **b) Site-to-site VPN**

Site-to-site VPN used to connect various areas that already fixed or fixed, this device utilizing a dedicated VPN are connected through the internet. Site-to-site VPN is divided into two, namely, extranet and intranet. An intranet that is where VPN is used only to connect various locations that are still one agency or one company. Like the Central Office is connected to the Branch Office. In other words, administrative control in control. While the extranet is where VPN is used to connect the company with other companies, such as partners, suppliers, or customers. In other words, administrative control is under the control of some of the relevant agencies.

## **2.3 OPENVPN**

OpenVPN is an open-source application for Virtual Private Networking (VPN), where the application can create a connection point to point tunnel that has been encrypted. OpenVPN using private keys, certificate, or username or password to perform authentication in building connections.

## Where to use OpenSSL encryption?

### a) **Layer 2 and Layer 3 VPN**

OpenVPN offers two basic modes, which operates as both a VPN layer 2 or layer 3 tunnel so that OpenVPN can also run-on Ethernet Frames, IPX packets and Windows Network (NETBIOS) Packet Browsing, all of which is a problem in the VPN solution

### b) **Protecting field workers with the internal firewall.**

Users who connect to the VPN Server will make the tunnel and turn the laptop/computer network settings, so that network traffic is sent through the tunnel. If the tunnel is established then the firewall from the VPN Server will be able to protect your laptop/computer connected, even though it is not the local machine.

### c) **OpenVPN connections can be tunneled through almost every firewall tunnel.**

OpenVPN can work on sites that use HTTPS protocol.

### d) **Proxy support and configuration.**

OpenVPN has proxy support and can be configured to run as a service and TCP or UDP as server or client. As the OpenVPN server, just wait until the client connection requests, while as a client, it tries to make the connection that corresponds to the configuration.

### e) **Only one port in the firewall must be opened to allow incoming connections.**

Since the OpenVPN 2.0, a special server mode allows connection of multiple incoming TCP or UDP port of the same, while still using a different configuration for every single connection.

### f) **Virtual interfaces allow very specific networking and firewall rules.**

All regulations, restrictions, and forwarding mechanisms concepts like NAT can be used with OpenVPN tunnel.

### g) **High flexibility with extensive scripting possibilities.**

OpenVPN offers many starting points for individual scripts. This script can be used for a variety of purposes from authentication to failover or more.

**h) Transparent, high-performance support for dynamic IPs.**

OpenVPN no longer needs to use static IP'S on both sides of the tunnel. The second point is the end of the tunnel can have cheap DSL, access with dynamic IP users will rarely see the IP changes on both sides. the second session is the Windows Terminal Server and the Secure Shell (SSH) will only be "hang" for a few seconds, but it will not stop the requested demand after a brief pause.

**i) No problems with NAT.**

Both the OpenVPN server and clients can be in a network that uses private IP addresses only. Any firewall can be used to send traffic to another tunnel.

## **2.4 MACHINE LEARNING**

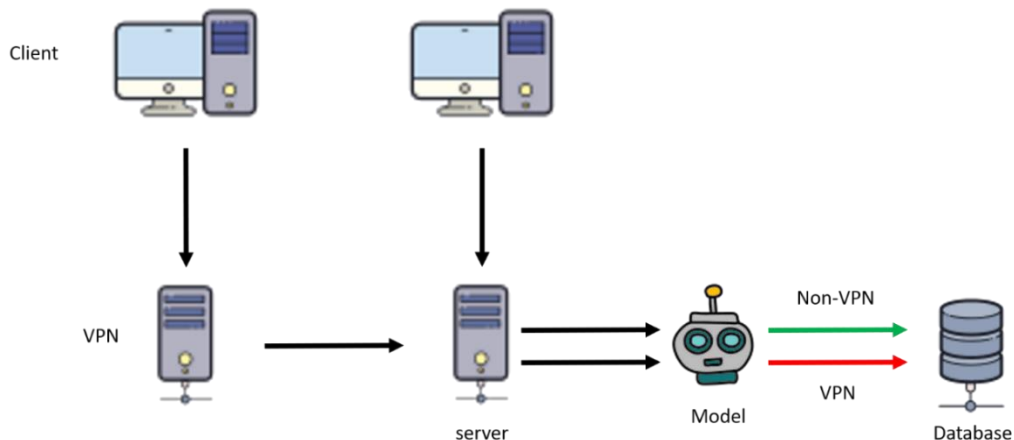
This section of background describes Machine learning overview, followed by the description of techniques relevant to this study, the next is Machine Learning Process and Performance Measures.

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide.

### **2.4.1 Machine Learning Process**

There are 6 stages for Processing Machine Learning, As show in Figure 2.1



*Figure 2.1 Vpn Architecture Design*

## 1. Data collection

ML techniques require representative data, to build an effective ML model for a given networking problem. Data collection is an important step, in general, data collection can be achieved in two phases (offline and online).

Offline data collection allows to gather a large amount of historical data that can be used for model training and testing.

Real-time network data collected in the online phase can be used as feedback to the model, or as input for re-training the model.

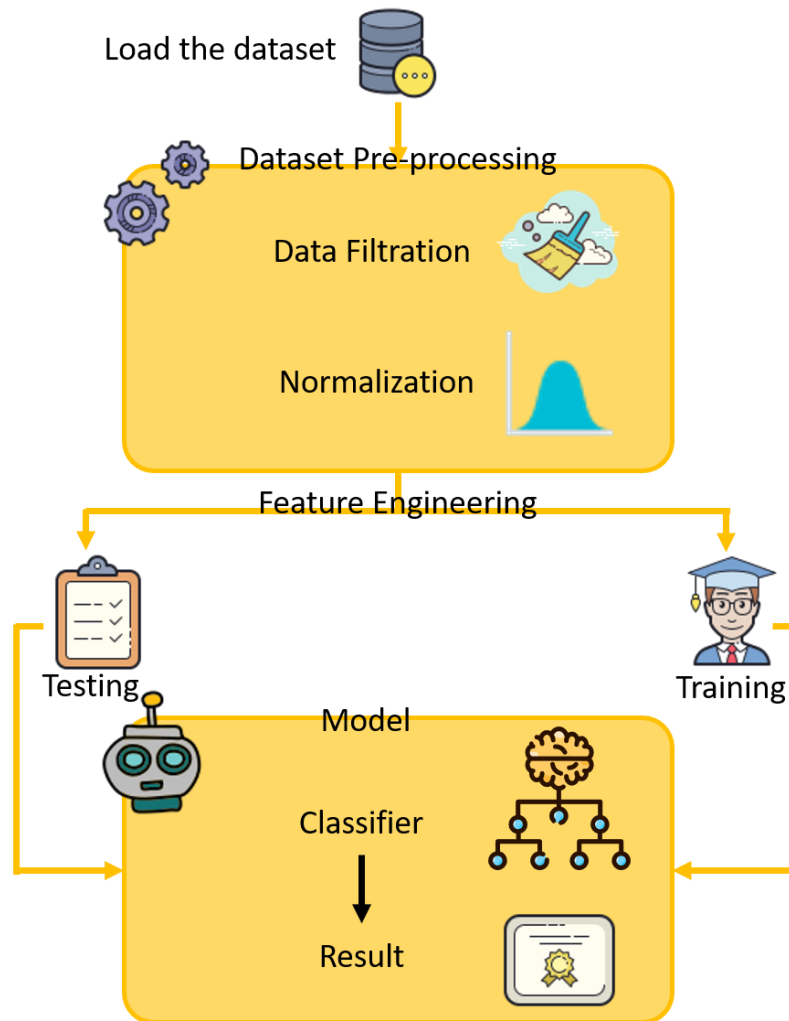


Figure 2.2 Machine Learning Process

## 2. Dataset pre-processing:

It's considered an essential step in data mining and machine learning, Large-scale datasets usually contain noisy, incomplete, inconsistent and redundant

data. So, the dataset is passed through a group of pre-processing operations to get suitable format for building strong model like:

- Data filtration:

Because of heterogeneity, the dataset usually contains anomalous and redundant instances. Which reduce the classification accuracy? We need to solve the problem by removing these records from dataset before starting process.

- Data transforming (Label encoding):

The utilized datasets contain symbolic, continuous, and binary values. And because many classifiers take only numerical values, it's good to convert them for improving accuracy (converting dataset's non-numerical features into a numerical amount).

- Normalization

Different scales among features can degrade the classification performance. so it is important to normalize the values in each attribute and maps features onto a normalized range. so that the minimum value in each attribute is zero and the maximum being one.

This provides more homogeneous values to the classifier while maintaining relativity among the values of each attribute.

- Feature extraction

The collected raw data may be noisy or incomplete. Before using the data for learning, it must go through a pre-processing phase to clean the data. Another important step prior to learning, or training a model, is feature extraction.

At the finest level of granularity, packet-level features are simplistically extracted or derived from collected packets, e.g., statistics of packet size, including mean, root mean square (RMS) and variance, and time series information, such as hurst. The key advantage of packet-level statistics is their insensitivity to packet sampling that is often employed for data collection and interferes with feature characteristics.

On the other hand, Flow-level features are derived using simple statistics, such as mean flow duration, mean number of packets per flow, and mean number of bytes per flow.

Whereas, connection-level features from the transport layer are exploited to infer connection-oriented details. In addition to the flow-



level features, transport layer details, such as throughput and advertised window size in TCP connection headers, can be employed. Though these features generate high quality data, they incur computational overhead and are highly susceptible to sampling and routing asymmetries.

### **3. Feature selection**

It's the removal of features that are irrelevant or redundant. Irrelevant features increase computational overhead with marginal to no gain in accuracy, while redundant features promote over-fitting. Feature extraction is often a computationally intensive process of deriving extended or new features from existing features, using techniques, such as entropy, Fourier transform and principal component analysis (PCA).

Feature's selection and extraction can be performed using tools, such as NetMate and WEKA. However, in this case, the extraction and selection techniques are limited by the capability of the tool employed.

### **4. Model Training**

Training your model is the bulk of machine learning. The objective is to use your training data and incrementally improve the predictions of the model.

### **5. Score/Test Model**

The model that was built or trained during step 4 is tested using the test data set, and the produced result is used for building a new model, that would consider previous models, i.e., "learn" from them.

### **6. Performance measures and evaluation**

We will evaluate the performance of the classifier or machine learning model and the system by using: confusion matrix analysis, Accuracy, Precision, Recall and F1-Score evaluation techniques [2]. which detailed below:

Several widely used metrics to measure the performance of the models can be computed based on the confusion matrix.

**Accuracy:** It estimates the ratio of the correctly recognized connection records to the entire test dataset. If the accuracy is higher, the machine learning model is better.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

**Precision:** It estimates the ratio of the correctly classification vpn/non-vpn connection records to the number of all traffic connection records. If the Precision is higher, the machine learning model is better.

$$precision = \frac{TP}{TP + FP}$$

**Recall:** proportion of test samples labelled as positive that are correct.

$$Recall (R) \text{ or sensitivity} = \text{true positive fraction} = \frac{TP}{TP + FN}$$

**F1-Score:** F1-Score is also called as F1-Measure. It is the harmonic mean of Precision and Recall. If the F1-Score is higher, the machine learning model is better.

$$F1 - Score = 2 \times \left( \frac{Precision \times Recall}{Precision + Recall} \right)$$

**True Positive Rate (TPR):** It is also called as Recall. It estimates the ratio of the correctly classified traffic connection records to the total number of traffic connection records. If the TPR is higher, the machine learning model is better.

$$TPR = \frac{TP}{TP + FN}$$

**False Positive Rate (FPR):** It estimates the ratio of the Normal connection records flagged as vpn/non-vpn to the total number of Normal connection records. If the FPR is lower, the machine learning model is better.

$$FPR = \frac{FP}{FP + TN}$$

#### 2.4.2 Machine learning Techniques

There are two types of Machine Learning Techniques Supervised and Unsupervised

**Learning In Supervised learning,** learning is based on labelled data. A label is a representation to one class included in the dataset. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it contained the object [3].

- 1) **Support Vector Machines (SVMs):** a classification method based on maximum margin linear discriminants, that is, the goal is to find the optimal hyperplane that maximizes the gap or margin between the classes. Further, we can use the kernel trick to find the optimal nonlinear decision boundary between classes, which corresponds to a hyperplane in some high-dimensional “nonlinear” space [4].
- 2) **logistic regression (LR):** is to predict the probability of the response variable values based on the independent variables. Logistic regression is in fact a classification technique, that given a point  $x_j \in \mathbb{R}^d$  predicts  $P(c_i | x_j)$  for each class  $c_i$  in the domain of  $Y$  (the set of possible classes or values for the response variable) [4].
- 3) **Naive Bayes (NB):** methods Are a set of supervised learning algorithms based on applying Bayes’ theorem with the “naive” assumption of conditional independence between every pair of features given the value of the class variable. Is simple of "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features (see Bayes classifier) [4].

- 4) **The k-nearest neighbours (KNN):** algorithm Is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. And is a supervised machine learning algorithm being one that relies on labelled input data to learn a function that produces an appropriate output when given new unlabelled data [4].
- 5) **Random Forest (RF):** is a popular ensemble method based on a standard machine learning technique called "decision tree". In a decision tree, a pair of variable-value is chosen that will split in such a way to generate “best” two child subsets. Then again, the algorithm is applied on each branch of the tree. As an input enters the root of the tree and traverses down the tree and gets bucketed into smaller sets. Random forest chooses a subsample of the feature space in each split and makes the trees de-correlated.

Random forest also prunes the tree by setting a stopping criterion for the node splits. "Max features" is a parameter used to tune the maximum number of features used to split the node [4].

**In unsupervised learning**, the algorithm builds a mathematical model from a set of data which contains only inputs and no desired output labels. Unsupervised learning algorithms are used to find structure in the data, like grouping or clustering of data points. Unsupervised learning can discover patterns in the data, and can group the inputs into categories [3]

## **2.5 GOOGLECOLABORATORY ENVIRONMENT**

Google Colaboratory is a free, in-the-browser, collaborative programming environment that provides an interactive and easy to use platform for deep learning researchers and engineers to work on their data science projects. providing a free cloud service based on Jupiter Notebooks that supports free GPU.

Not only is this a great tool for improving your coding skills, but with Colab you can import an image dataset, train an image classifier on it, and evaluate the model, all in just a few lines of code. it also allows absolutely anyone to develop deep learning applications using popular libraries such as PyTorch, TensorFlow, Keras, and OpenCV [5].

## **2.6 RAPIDER MINER**

RapidMiner blends ease of use with data science rigor. Your friendly neighbourhoods' data science tool has put on a business suit and is looking good in it. Enterprise AI teams composed of seasoned data scientists and data-savvy engineers, analysts, and business users will value RapidMiner's blend of expansive functionality and attention to statistical rigor with ease of use and automation. It has some of the most productivity-enhancing capabilities for automated data preparation (Turbo Prep) and model development (Auto Model) in the multimodal market, along with one of the most comprehensive visual tools for building data and ML pipelines. RapidMiner also supports Python-loving data scientists who prefer notebooks, and it enables everyone to operationalize and manage their models on a common platform in the near term, RapidMiner might not just have something for everyone; it could have everything for everyone. But first it needs to round out its already impressive capabilities supporting notebooks and Model Ops and finish transitioning its IDE to the web [6].

## **2.7 CICFLOWMETER**

The CICFlowMeter is an open-source tool that generates Biflows from pcap files, and extracts features from these flows.

CICFlowMeter is a network traffic flow generator available from [here](#). It can be used to generate bidirectional flows, where the first packet determines the forward (source to destination) and backward (destination to source) directions, hence the statistical time-related features can be calculated separately in the forward and backward directions. Additional functionalities include, selecting features from the list of existing features, adding new features, and controlling the duration of flow timeout [7].

## *Chapter Three*

### **3. RELATED WORKS**

In this chapter to show and explain the related papers and projects on our project.

#### **3.1 LITERATURE SURVEY**

There were many papers and researches that were interested in this field due to its importance in analysing and extracting data from the network and classifying it. Now, here we review some of the research that we benefited from and talked about the topic and we quote some topics from them, as follows:

Malware behaviour is as old as the internet. From this historical perspective, the competition between malware creators and defenders of companies, governments and ordinary users keeps malware constantly changing and evolving. With the growing usage of the Internet and its increasing importance for our daily lives, security plays a more profound and essential role than ever before. While security companies have developed methods to protect users against attackers, many challenges keep making this task difficult to achieve in real life. One of the most important challenges in security is encryption of the internet traffic, as it requires a fine balance between respecting the privacy of users as much as possible and at the same time gathering data from the traffic for detecting malicious behaviour with the best accuracy. This problem opens the doors for new approaches and methods of Artificial Intelligence [6].

Here we talk about a way to use neural networks to classify encrypted traffic. We are hiring Two classification algorithms based on supervised learning, most of them Used for traffic classification research, it is multi-layered perceptron (MLP) and recurrent neural network (RNN). MLP As a learning method, it learns instantly Special features of the initial traffic in the first step. traffic movement the classification is detected layer by layer, high level Features such as the introduction of the activation function. In the second step, if the MLP does not identify the traffic classes, we present RNN to discover the labels. The technique is the end-to-end approach, widely utilized in technologies of deep learning. It could also gain knowledge, especially the nonlinear relationship among the raw traffic input and the predicted performance label, rather than splitting a sophisticated issue into the meta-problems [7].

### **3.2 RELATED IP TRAFFIC CLASSIFICATION SYSTEMS AND SOFTWARE**

In this section we mention some of the most popular related IP Traffic Classification systems and software. Then followed by (Table 2) that presents a comparison between number of software depending on: developed by whom, free or not, the language of building it, the operating system that supports the functionality.

### **3.3 TRAFFIC CLASSIFICATION USING MACHINE LEARNING**

A team from the Institute for Web Technologies & Applications at the University of Applied Sciences Kufstein to get our Application Control module ready for the future. Machine learning (ML) is to be used to classify data traffic as part of a project funded by the state.

Application Control uses Deep Packet Inspection (DPI) to both monitor and filter data traffic. It is possible to ensure that certain applications or protocols are allowed to pass entirely or with limited throughput, or not allowed to pass at all.

Application Control only partially detects encrypted connections and rates them unknown they cannot be blocked, nor can their bandwidth be limited. Encrypted connections (mainly HTTPS) will become more common in future, so Application Control needs support [8].

### **3.4 LIBPROTOIDENT**

Is a library that performs application layer protocol identification for flows. Unlike many techniques that require capturing the entire packet payload, only the first four bytes of payload sent in each direction, the size of the first payload-bearing packet in each direction and the TCP or UDP port numbers for the flow are used by libprotoident.

Libprotoident features a very simple API that is easy to use, enabling developers to quickly write code that can make use of the protocol identification rules present in the library without needing to know anything about the applications they are trying to identify. Libprotoident supports over 400 different application protocols and this number will continue to grow over the course of future releases! [9]

### **3.5 SPID (STATISTICAL PROTOCOL IDENTIFICATION)**

The project is based on statistical analysis of network flows to identify application traffic. The SPID algorithm can detect the application layer protocol (layer 7) by signatures (a sequence of bytes at a particular offset in the handshake), by analysing flow information (packet sizes, etc.) and payload statistics (how frequently the byte value occurs in order to measure entropy) from pcap files. It is just a proof-of-concept application and currently supports approximately 15 application/protocols such as eDonkey Obfuscation traffic, Skype UDP and TCP, BitTorrent, IMAP, IRC, MSN, and others [10].

### **3.6 L7-FILTER**

Is a classifier for Linux's Netfilter that identifies packets based on application layer data? It can classify packets such as Kazaa, HTTP, Jabber, Citrix, Bittorrent, FTP, Gnucleus, eDonkey2000, and others. It classifies streaming, mailing, P2P, VOIP, protocols, and gaming applications. The software has been retired and replaced by the open source Netify DPI Engine [11].

### **3.7 DJANGO**

Django was developed in a fast-paced newsroom environment, it was designed to make common Web-development tasks fast and easy. Here's an informal overview of how to write a database-driven Web app with Django. The goal of this document is to give you enough technical specifics to understand how Django works, but this isn't intended to be a tutorial or reference – but we've got both! When you're ready to start a project, you can start with the tutorial or dive right into more detailed documentation [14].

### **3.8 PYTHON**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed [15].



Table 3-1 Comparison of Network VPN papers and research

Name	Developed by	Language	Free or not	Functionality	Website	Installed on
Traffic-Detector	Panagiotis Yialouris	java	Free	Analysing traffic from six different cases of some of the most mainstream protocols and applications.	[6]	Linux Windows
Tstat	mPlane FP7 European project		free personal and research	A passive sniffer able to: - provide insight into traffic patterns. - give details and statistics for numerous applications and protocols.	[10]	Linux systems. Mac OS. support for compilation for Android.
PACE 2	ipoque	C	not free	- Real-time IP traffic classification. - Metadata parameters such as bandwidth usage, traffic volume, etc. - Encrypted traffic intelligence by combining advanced techniques and behavioural and statistical analysis as well as machine learning (ML) to classify.	[11]	Windows
Cisco NBAR	Cisco		not free	Mission critical applications including ERP and workforce optimization applications can be intelligently identified and classified using Network Based Application Recognition (NBAR).	[12]	Cisco NBAR

## Chapter Four

### 4. DATASET AND METHODOLOGY

This Chapter covers the explanation details of methodology and dataset that are being used in this project.

This methodology is used to achieve the objective of the project that will accomplish a perfect result.

#### 4.1 DATASET

In this section, we are going to use this set of data to make the project stronger, and produce better results, so we will rely on the dataset ISCXVPN2016, you can see more about the data set from the source [14].

##### 4.1.1 Dataset Description

In the beginning, we tried to collect dataset using two simple networks that represent victims and attacker's machines. An effective way to collect data from traffic as csv files is by using the CICFlowMeter tool.

The dataset we would collect won't be enough, and won't be containing a lot of attacks because of the lack of resources. So, we also went to use the ISCXVPN2016 dataset.

A group of researchers created a huge data set to classify the traffic into a VPN or Non-VPN, where they worked on a small software code to simulate traffic, and the code worked automatically and randomly to send a VPN or Non-VPN pack, and also used Linux to schedule the task by Cron is very simple to deal, you only need to modify the /etc /crontap file, and you can learn more about it from the source [15] [16].

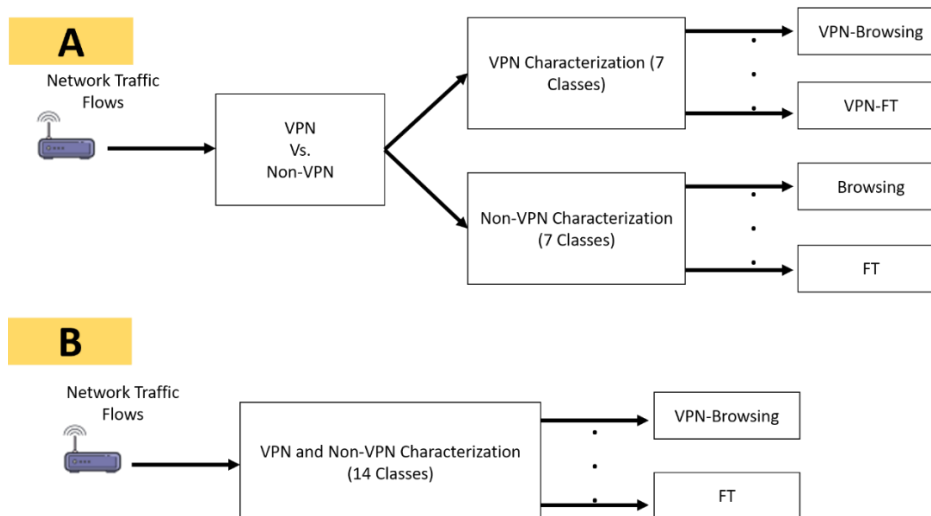


Figure 4.1 describe infrastructure for ISCXVPN2016 dataset

this dataset is captured a regular session and a session over VPN, and this includes the 14 traffic categories: VOIP, VPN-VOIP, P2P, VPN-P2P, etc.

We also give a detailed description of the different types of traffic generated:

**Browsing:** Under this label we have HTTPS traffic generated by users while browsing or performing any task that includes the use of a browser.

**Email:** The traffic samples generated using a Thunderbird client, and Alice and Bob Gmail accounts. The clients were configured to deliver mail through SMTP/S, and receive it using POP3/SSL in one client and IMAP/SSL in the other.

**Chat:** The chat label identifies instant-messaging applications. Under this label we have Facebook and Hangouts via web browsers, Skype, and IAM and ICQ using an application called pidgin.

**Streaming:** The streaming label identifies multimedia applications that require a continuous and steady stream of data. We captured traffic from Youtube (HTML5 and flash versions) and Vimeo services using Chrome and Firefox.

**File Transfer:** This label identifies traffic applications whose main purpose is to send or receive files and documents. For our dataset we captured Skype file transfers, FTP over SSH (SFTP) and FTP over SSL (FTPS) traffic sessions.

**VoIP:** The Voice over IP label groups all traffic generated by voice applications. Within this label we captured voice calls using Facebook, Hangouts and Skype.

**TraP2P:** This label is used to identify file-sharing protocols like Bittorrent. To generate this traffic, we downloaded different .torrent files from a public a repository and captured traffic sessions using the uTorrent and Transmission applications.

#### ISCXVPN2016 Network Traffic Dataset content

Traffic: Content

Web Browsing: Firefox and Chrome

Email: SMTPS, POP3S and IMAPS

Chat: ICQ, AIM, Skype, Facebook and Hangouts

Streaming: Vimeo and YouTube

File Transfer: Skype, FTPS and SFTP using FileZilla and an external service

VoIP: Facebook, Skype and Hangouts voice calls (1h duration)

P2P: uTorrent and Transmission (Bittorrent)

The features in ISCXVPN2016 dataset are described in Table 4.1 that contains Feature Name, Description about this Feature and Datatype

Table 4-1 Features in ISCXVPN2016 dataset

Feature	Description	Type
Label	indicate whether the traffic is VPN or non-VPN	string
Dst Port	Destination port number	integer
Protocol	Protocol	integer
TimeStamp	Time Stamp of the flow	string
Flow Duration	Flow duration	integer
Tot Fwd/BwdPkts	Total packets in forward/backward directions	integer
TotLenFwd/BwdPkts	Total size of packets in forward/backward directions	integer
Fwd/BwdPkt - Len Max/Min/Mean/Std	Maxi/Mini/Average/Std. Dev. size of package in forward/backward directions	integer
Flow Byts/s & Flow Pkts/s	Flow byte rate, i.e., number of packets per seconds	float64
Flow IAT Mean/Std/ Max/Min	Average/Std. Deviation/Maxi/Mini time between two flows	float64
Fwd/Bwd IAT Tot/Mean/ Std/- Max/Min	Total/Average/Std. Deviation/Maxi/Mini time between two packets in forward/backward directions	float64
Fwd/Bwd PSH/URG Flags	Number of times the PSH/URG flag was set in packets in forward/backward direction	integer
Fwd/Bwd Header Len	Total bytes used for header in forward/backward direction	integer
Fwd/BwdPkts/s	Number of forward/backward packets per second	float64
Pkt Len Min/Max/Mean/Std	Maxi/Mini/Average/Std. Dev. length of a flow	integer
Pkt Len Var	Mini inter-arrival time of packet	float64
FIN/SYN/RST/PUSH/ACK/ URG/CWE/ECE Flag Cnt	Number of packets with FIN/SYN/RST/PUSH/ACK- /URG/CWE/ECE	integer
Down/Up Ratio	Download/upload ratio	integer
Pkt Size Avg	Average size of packets in forward/backward direction	float64
Fwd/BwdSeg Size/Byts/b/Blk Rate Avg	Average number of bulk rate/bytes bulk rate/packets bulk rate in forward/backward directions	float64
SubflowFwd/BwdPkts/Byts	The average number of bytes/packets in a sub flow in forward/backward direction	integer
InitFwd/Bwd Win Byts	Number of bytes sent in initial window in forward/backward directions	integer
Fwd Act Data Pkts	Number of packets with at least 1 byte of TCP data payload in forward	integer
FwdSeg Size Min	Minimum segment size observed in forward	integer
Active Mean/Std/Max/Min	Maxi/Mini/Average/Std. Dev. a flow was active before becoming idle	float64

#### 4.1.2 Dataset Analysis

In this section, we make data analysis of the ISCXVPN2016 dataset. the aims of this analysis to answer the following questions:

- How many total network flows are contained in the dataset?
- How many VPN network flows are contained in the dataset?
- How many non-VPN network flows are contained in the dataset?

After data analysis, we found this information:

1. The basic information of the dataset reveals that the whole dataset consists of 46071 network flows and it contains missing values (null values and infinity values). so, we need data pre-processing.
2. The number of VPN network flows are.
3. The number of non-VPN network flows are.

#### 4.1.3 Preprocessing for dataset

In this section, we explain how to clean, clear, and pre-process the Dataset.

These steps have been taken to Pre-processing the dataset: -

1. Delete noisy features and missing value.
2. Format data into standard datatype.
3. To reduce the size of the datasets, reduce the unnecessary accuracy of the float numbers by dropping digits after the decimal point.
4. change the "Infinity" and "NaN" values to zero.
5. Remove columns that were repeated in the dataset by code written by java

#### 4.1.4 Training Dataset

We used part of ISCXVPN2016 dataset at the rate of 70% for training and learning the machine learning model.

#### 4.1.5 Test Dataset

We used part of ISCXVPN2016 dataset at the rate of 30% for testing the machine learning model.

### 4.2 *MACHINE LEARNING CLASSIFIER*

In this section, we explain how implementation Machine Learning Classifier, such as how make feature engineering and how implement our model

#### 4.2.1 4.2.1 Feature Engineering

As the first step in feature engineering, we note a Low Amount of Variance for some features. so, all features with zero amount of variation are removed as those features will not have any influence on the prediction of the target variable.

In the second step, we removed undesired features such as Timestamp.

In the next step, Features having a high correlation amongst each other are removed.

Those features won't bring any additional predictability but may introduce noise. Finally, the remaining number of features dataset after feature engineering is 76 features.







#### 4.2.2 Classification Model

In our model, the main task is getting a set of statistics information of traffic flow, and identify if the flow traffic is VPN or Non-VPN based on learning on a set of already labelled data containing both the VPN and Non-VPN traffic.

we split the data into two parts, in part one the training at the rate of 70%, and part two the testing at the rate of 30% from the total of the samples. After generating training and testing datasets we applied Several algorithms with different values from supervised classification algorithms.

As we will notice in Chapter 5, [Section 5.2](#).

The supervised machine learning classification models used for building are listed below:

-  support vector machine (SVM)
-  Naïve Bayes (NB)
-  k-nearest neighbour (KNN)
-  logistic regression (LR)
-  Gradient Boosting Trees (GBTs)
-  Random Forest (RF)

## 5. EXPERIMENTAL SETUP, RESULTS AND ANALYSIS

### 5.1 EXPERIMENTAL SETUP

In this section, we explain the experimental setup to implement the project such as implementation Machine Learning Model and build Web Application.

#### 5.1.1 Machine learning Model

##### 5.1.1.1 Pre-processing Dataset

- I. In the following code, we Import all packages needed in pre-process dataset

```
import os
import sklearn
from sklearn.utils import shuffle
from sklearn.neighbors import KNeighborsClassifier
import pandas as pd
import numpy as np
from sklearn import linear_model, preprocessing
```

- II. In the following code: All data set files have been combined into one file to start splitting for the training and testing part. Changed the value of the label to Vpn or non-Vpn.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintWriter;
import java.util.Scanner;

public class CsvOneFile {
    // NeedManualLabel == > default name for label

    public static void main(String[] args) throws
    FileNotFoundException {
        String pathFile = "C:\\Users\\youssef
        ezzeldeen\\Desktop\\skypeFile";
        File sourceFile = new File(pathFile);
        File[] files = sourceFile.listFiles();

        int x = 0;
        File output = new File("C:\\Users\\youssef
        ezzeldeen\\Desktop\\outputskypeFile2.csv");
        PrintWriter printWriter = new PrintWriter(output);
        for (int i = 0; i < files.length; i++) {
            File[] CsvFiles = files[i].listFiles();
            for (File CsvFile: CsvFiles

        ) {
```

```

        Scanner
s = new
Scanner(CsvFile.getAbsoluteFile());
while (s.hasNextLine()) {
x++;

System.out.println(files[i].getAbsolutePath());
String line = s.nextLine();
if (line.contains("Flow ID")) {
line = "";
}
if (i == 0) {

printWriter.append(line.replace("NeedManualLabel", "non-vpn") +
"\n");
} else {
printWriter.append(line.replace("NeedManualLabel", "vpn") + "\n");
}
}
}
}
System.out.println(x);
System.out.println("finshed is programe");
}
}

```

III. In the next code, the dataset was spited by taken 70% from each label to train data and 30% for each label to test data.

```

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Scanner;
// split datasets to 30% testing and 70% traning
public class SplitDatasetForML {

    public List<Integer> numbers;
    public final String pathFile = "C:\\\\Users\\\\youssef
ezzeldeen\\\\Desktop\\";
    private final Scanner s;
    private final File f;
    private final PrintWriter printWriter;
    private final PrintWriter pw;

    public SplitDatasetForML() throws FileNotFoundException {
        numbers = new ArrayList<>();
        setArray();
        f = new File(pathFile + "outputskypeFile.csv");
    }
}

```



```

        s = new Scanner(f);
        printWriter = new PrintWriter(
            new File(pathFile + "testingskypeFile.csv"));
        pw = new PrintWriter(
            new File(pathFile + "truningskypeFile.csv"));
    }

    public void run() throws IOException {
        System.out.println("run");
        for (int i = 0; i < numbers.size(); i++) {
            String line = getLineInFile(numbers.get(i));
            if (i >= 483) {
                printWriter.close();
                writeTurningFile(line);
            } else {
                writeTestingFile(line);
            }
        }
        pw.close();
    }

    private String getLineInFile(int randomNumber) throws
    IOException {
        return Files.readAllLines(Paths.get(pathFile +
        "outputs skypeFile.csv")).get(randomNumber);
    }

    private void setArray() {
        for (int i = 0; i < 1611; i++) {
            numbers.add(i);
        }
        Collections.shuffle(numbers);
        System.out.println(numbers);
    }

    private void writeTestingFile(String line) {
        printWriter.append(line + "\n");
        System.out.println("writeTestingFile");
    }

    private void writeTurningFile(String line) {
        pw.append(line + "\n");
        System.out.println("writeTurningFile");
    }
}

```

IV. In the following code, the model was trained on a simple sample of 700 records from the dataset and tested on 300 records to verify the ratio of each model. The ratios were clarified in Chapter 4

```

x_train, x_test, y_train, y_test =
sklearn.model_selection.train_test_split(X, y, test_size=0.3)

```

```
model = KNeighborsClassifier(n_neighbors=5)
model.fit(x_train, y_train)
return model
```

### 5.1.1.2 Feature Engineering

Here we explain how make Feature Engineering in our model.

- ✚ Extracting features from PCAP files was done with the helper program CICFlowMeter

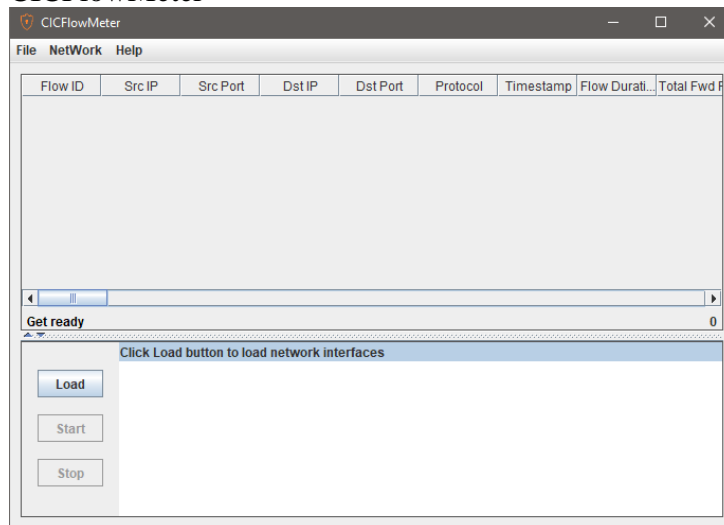


Figure 5.1 CICFlowMeter Tool

- ✚ All the operations that were done on the features we have done manually with help codes in search and filtering

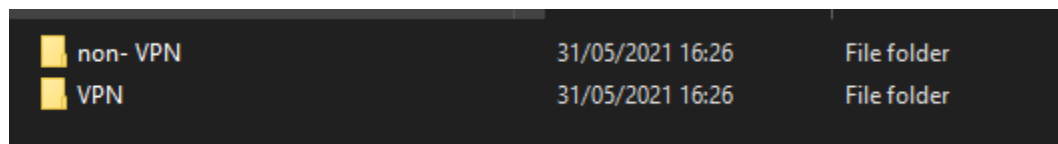


Figure 5.2VPN and non-VPN file for feature

### 5.1.1.3 Classifier Model

Sklearn was used to implement Machine Learning algorithms, we have an example to explain how to implement the model for KNN and also for the other algorithms.

- I. In the following code, we import all packages needed in Machine Learning Model

```
import os
import sklearn
from sklearn.utils import shuffle
from sklearn.neighbors import KNeighborsClassifier
import pandas as pd
import numpy as np
from sklearn import linear_model, preprocessing
```

- II. In the code below, we read preprocessed all data, it contains 76 features.

```
data = pd.read_csv('meetup/ML/output.csv')
```

- III. We split the dataset into training and testing, as in the following code.

```
x_train, x_test, y_train, y_test =
sklearn.model_selection.train_test_split(X, y, test_size=0.3)
```

- IV. The Accuracy is calculated and the result was good at  $k = 5$ .

```
x_train, x_test, y_train, y_test =
sklearn.model_selection.train_test_split(X, y, test_size=0.3)
model = KNeighborsClassifier(n_neighbors=5)
acc = model.score(x_test, y_test)
print(acc) #91.5 is the best
```

- V. We trained the model and saved all its properties, so as not to repeat the training every time the user runs the program.

```
model.fit(x_train, y_train)
filename = 'finalized_model.sav'
pickle.dump(model, open(filename, 'wb'))
```

- VI. We did a test for a set of features obtained from the Internet. The result was correct for most of the values

```
def predict(listFeatures):
    model = KNN.KNNModel()
    names = ["non-vpn", "vpn"]
    predicted = model.predict(listFeatures)
    return names[predicted[0]]
```

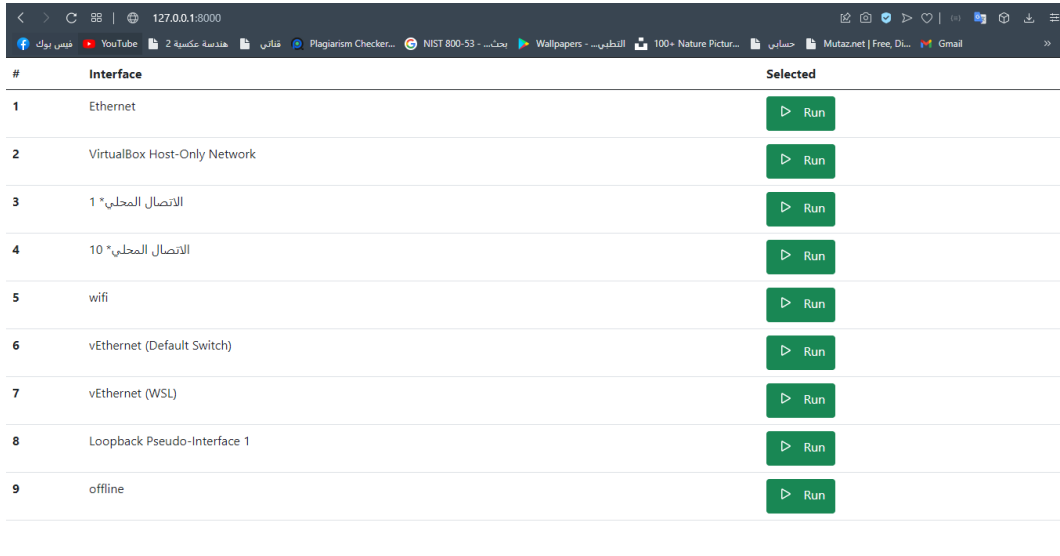
### 5.1.2 VPN/ non-VPN Classifier Web Application










We build web Application by using django python framework, in this Section, we explain classes that used.

#### Simple interface

##### ○ Index.html

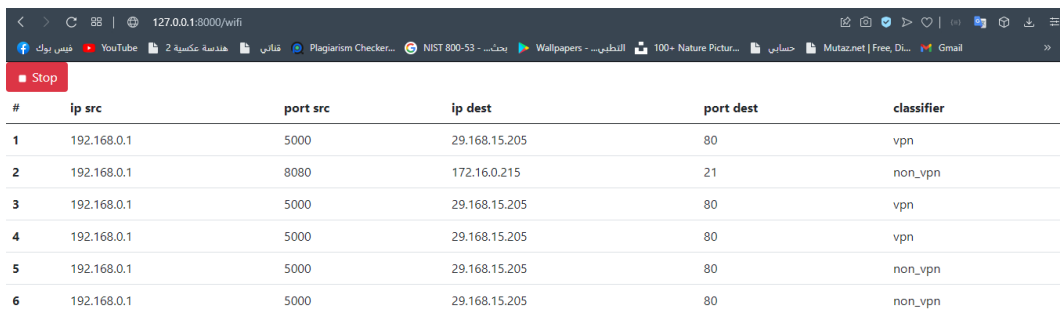
Fetch all network interfaces and select one to run packets capture



#	Interface	Selected
1	Ethernet	
2	VirtualBox Host-Only Network	
3	الاتصال المحلي * 1	
4	الاتصال المحلي * 10	
5	wifi	
6	vEthernet (Default Switch)	
7	vEthernet (WSL)	
8	Loopback Pseudo-Interface 1	
9	offline	

##### ○ Ruselt.html

After capturing packets from the network, it shows the results of the captured packets. it is either a VPN or non-VPN



#	ip src	port src	ip dest	port dest	classifier
1	192.168.0.1	5000	29.168.15.205	80	vpn
2	192.168.0.1	8080	172.16.0.215	21	non_vpn
3	192.168.0.1	5000	29.168.15.205	80	vpn
4	192.168.0.1	5000	29.168.15.205	80	vpn
5	192.168.0.1	5000	29.168.15.205	80	non_vpn
6	192.168.0.1	5000	29.168.15.205	80	non_vpn

- Off Line.html  
When a user has pcap files and wants to categorize them as vpn or non-vpn, we provide this service to him and it may be in the future API "Under construction"

## 5.2 RESULT AND ANALYSIS

In this Section, we explain the result of project, result for machine learning and Web Application.

### 5.2.1 Result for Machine Learning Model

We explain Confusion matrices and Classification report for the machine learning algorithm on the dataset:

In first practical work, we built a model for each attack in the dataset by using different classifier algorithms, in this Figures we display Confusion matrices for the best classifier Algorithms that used.

#### 5.2.1.1 *k*-nearest neighbor (KNN)

Work was done on more than value for *k* and the results were different. The best it was at a value of 5

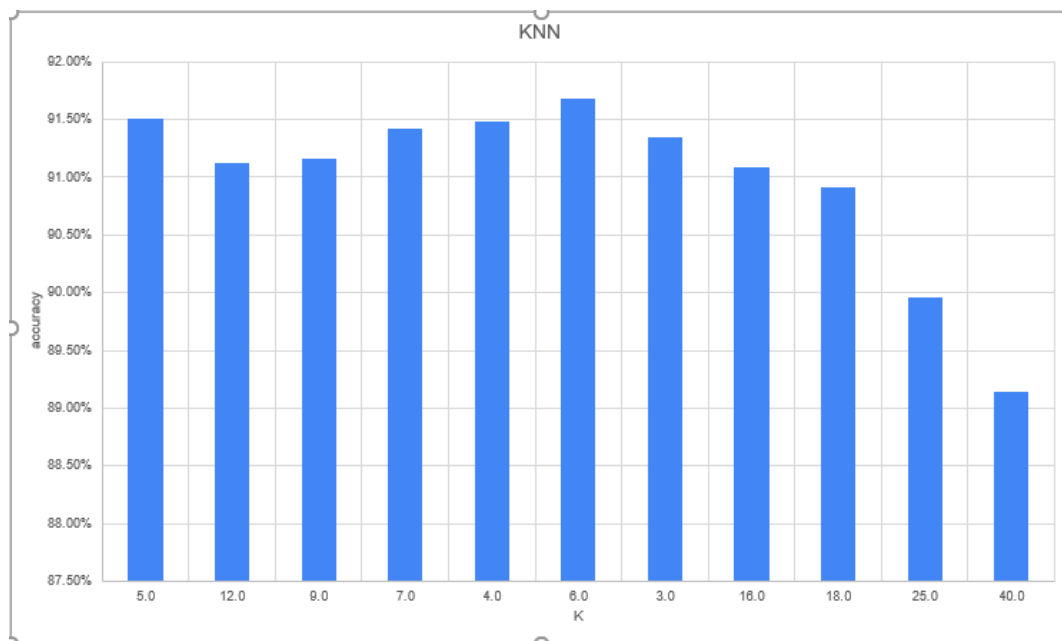


Figure 5.3KNN algorithm experiment results

### 5.2.1.2 support vector machine (SVM)

It took a very long time to train and the results were very poor and this made her excluded in our project.

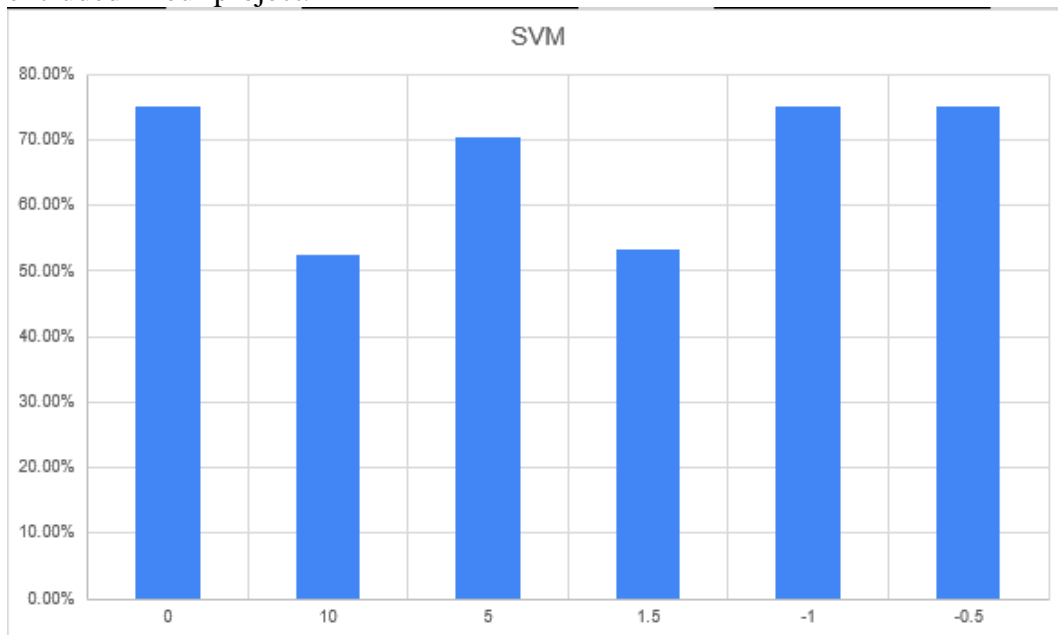


Figure 5.4 SVM algorithm experiment results

### 5.2.1.3 Naïve Bayes (NB)

The results were average and the algorithm was fast in training

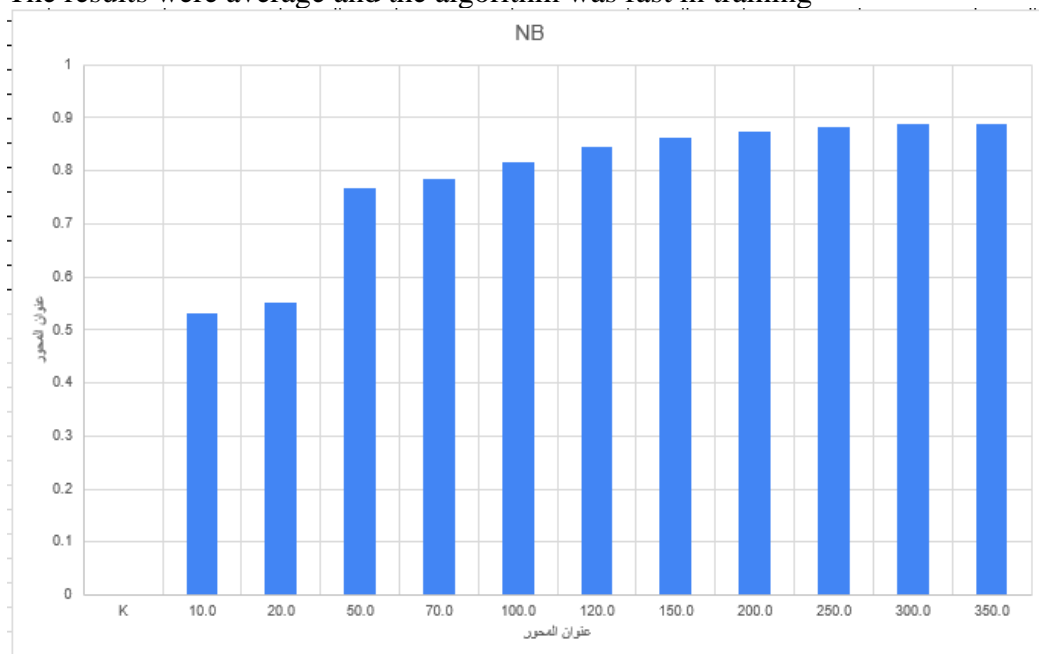


Figure 5.5 NB algorithm experiment result

#### 5.2.1.4 logistic regression (LR)

As we can see from the following figure, the results are very weak

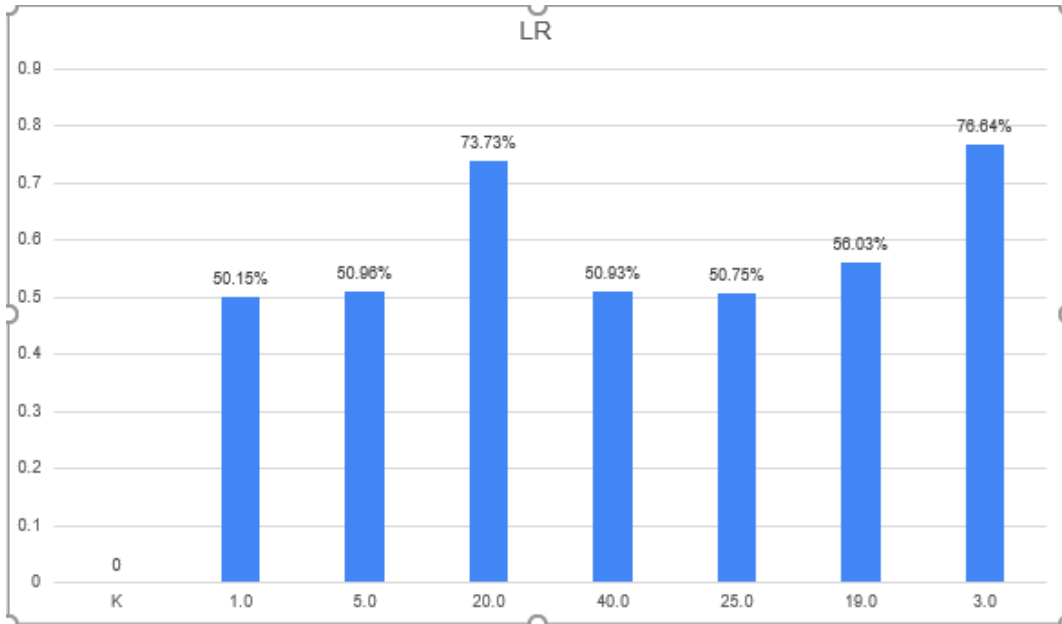


Figure 5.6 LR algorithm experiment results

#### 5.2.1.5 Gradient Boosting Trees (GBTs)

We note that the results were very impressive, which are required to start working on the project in the practical part of it

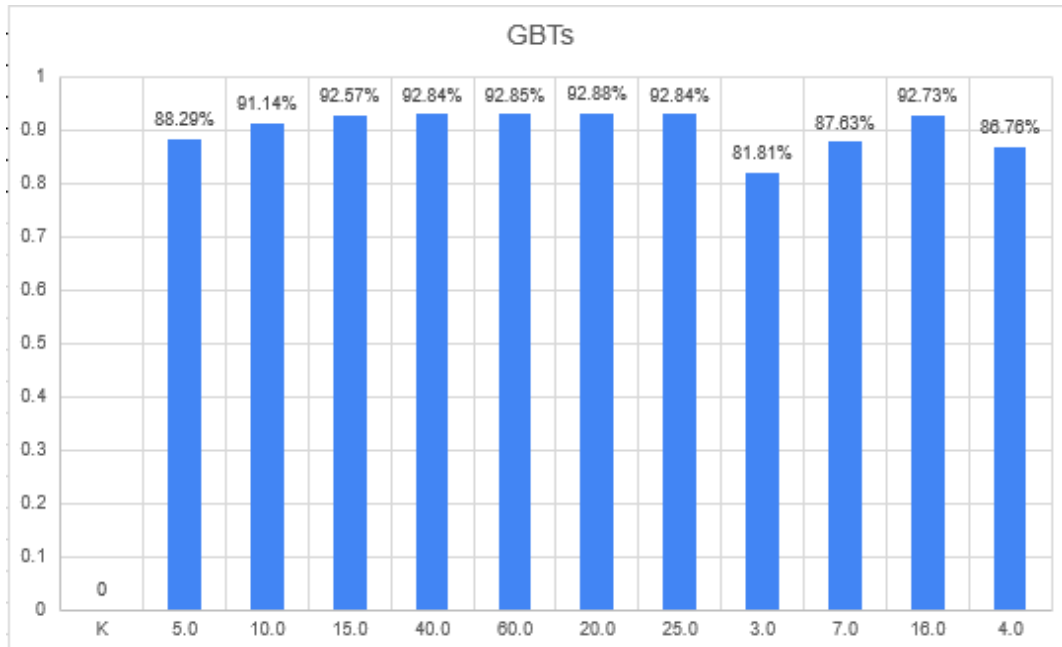


Figure 5.7 GBTs algorithm experiment results

### 5.2.1.6 Random Forest (RF)

We note that the results were very impressive, but the algorithm uses randomness too much.

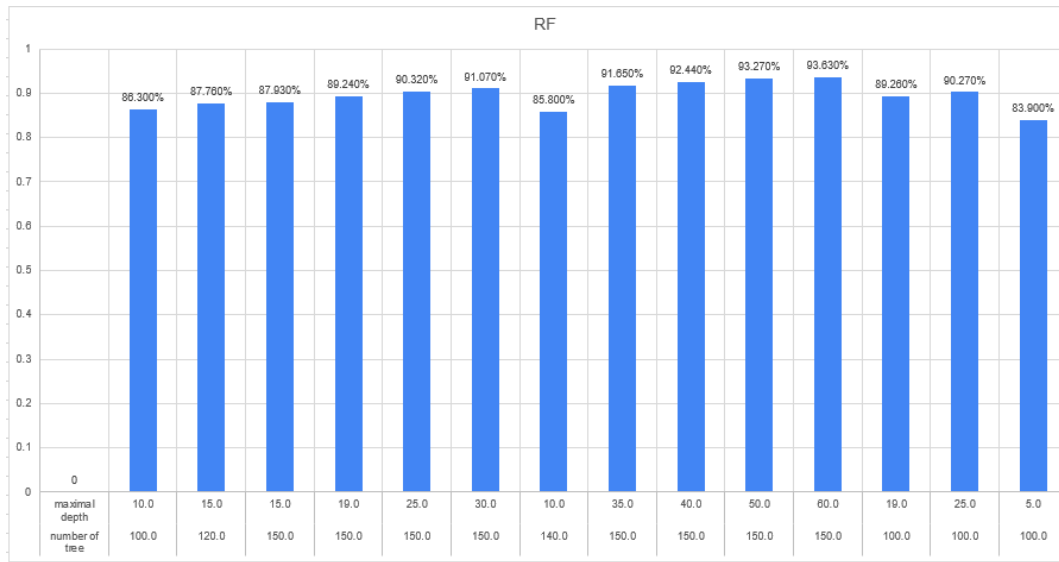


Figure 5.8 RF algorithm experiment results

### 5.2.2 Result for Classification in VPN/non-VPN Web Application

In this section, we explain the web application score, packet capture results, and rating in VPN/non-VPN.

Verify that the Wi-Fi interface pcap file is running as expected and converts pcap files to CSV files by the CICFlowmeter library, as shown in Figure 5.9

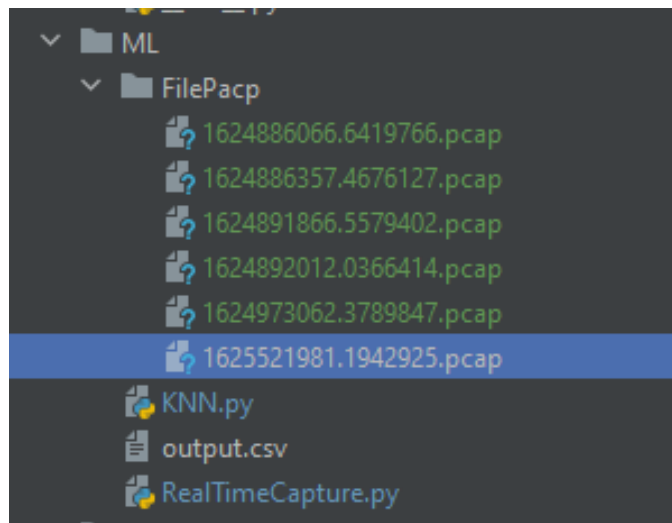


Figure 5.9 Result for Classification in VPN/non-VPN Web Application



## **6. CONCLUSION AND FUTUURE WORK**

### **6.1 OVERVIEW**

This chapter summarizes the project and shows the future work. At its end, it gives a conclusion for the project with the results which have been reached, providing some important recommendations for developers on the system in the future.

### **6.2 6.2 LIMITATIONS**

Unfortunately, there are some limitations that facing in the project:

Recourse Limitations To classifier VPN/non-VPN, our ML must capture, store, and analyse large volumes of data, in real-time. and the size of data passing over network can be large amount, so the system that will run this classifier VPN/non-VPN need large space for it.

The VPN traffic is very similar to the non-Vpn traffic, which made a lot of false positive in the classifier, that means it requires more advance work, analysis and coding than the other traffic to achieve high accuracy in classify it. so, we consider it as future work later.

### **6.3 FUTURE WORK**

Much more can be added to this system. Future work concerns with deepening the analysis of particular mechanisms and try the semi supervised classification

- ✚ We can expand the system to detect suspicious vpn traffic that affect on the network.
- ✚ Also, as a future work for us, extract additional features from the dataset that improve the quality of the machine learning model more.
- ✚ We also need to follow up and make updates to the system in order to keep abreast of what is new in the world of cyber security. As well we need to upgrade the system to make it being prevention system not only detection system.

### **6.4 CONCLUSION**

Our project is considered the first line of defence in companies because of its great benefit in knowing the threat before it occurs, and detecting suspicious traffic that is encapsulated in the VPN.

When you have a very large network in your company that receives thousands of inquiries every day, it will be difficult for you to distinguish suspicious requests, and this reduces the possibility of identifying the risk before it occurs.

Our saviour project came to these companies to make them safer and not to be exploited by hackers or competing companies to make the competition fair.

It seems like our project needs a lot of data to be trained on and to be in different parts of the world, and that makes it more accurate.

## 6.5 REFERENCES

- [1] M. Zain ul Abideen, S. Saleem and M. Ejaz, "VPN Traffic Detection in SSL-Protected Channel," hindawi, Islamabad, Pakistan, 2019.
- [2] R. Basnet, R. Shash, L. Walgren and T. Doleck, "Towards Detecting and Classifying Network Intrusion," 2019.
- [3] R. Nagar and Y. Singh, "A literature survey on Machine Learning Algorithms," jetir, 2019.
- [4] M. J. ZAKI and W. MEIRA, JR, DATA MINING AND MACHINE LEARNING, 2020.
- [5] T. Pessoa, R. Medeiros, T. Nepomuceno, G.-B. Bian, V. Albuquerque and P. P. R. Filho, "Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications," research Gate, 2018.
- [6] M. Gualtieri and K. Carlsson, "The Forrester Wave™: Multimodal Predictive Analytics And Machine Learning, Q3 2020," ibm, 2020.
- [7] ArashLashkari, "CICFlowMeter," 7 6 2021. [Online]. Available: <https://github.com/ahlashkari/CICFlowMeter/blob/master/ReadMe.txt>.
- [8] P. Dimou, J. Fajfer, N. Müller, E. Papadogiannaki, E. Rekleitis and F. Střasák, "ENCRYPTED TRAFFIC ANALYSIS Use Cases & Security Challenges," European Union Agency for Cybersecurity (ENISA), 2019.
- [9] A. Parchekani, S. N. Naghadeh and V. Shah-Mansouri, "Classification of Traffic Using Neural Networks by Rejecting," ResearchGate, Tehran, 2020.
- [10] "IAC Box," 17 12 2019. [Online]. Available: <https://www.iacbox.com/en/blog/detail2/news/detail/traffic-classification-using-machine-learning/>. [Accessed 15 01 2021].
- [11] "Network Research Group," 05 04 2019. [Online]. Available: <https://research.wand.net.nz/software/libprotoident.php>. [Accessed 08 01 2021].
- [12] E. Hjelmvik and W. John, "Statistical Protocol IDentification with SPID: Preliminary Results," sjalander, 2020.
- [13] "source Forge," 07 01 2009. [Online]. Available: <http://l7-filter.sourceforge.net>. [Accessed 07 01 2021].
- [14] p. employees, "django Documentation," Python, [Online]. Available: <https://docs.djangoproject.com/en/3.2/intro/overview/>.
- [15] p. employees, "Python documentation," python, [Online]. Available: <https://docs.python.org/3/>.
- [16] G. D. Gil, A. H. Lashkari, M. Mamun and A. A. Ghorbani, Characterization of Encrypted and VPN Traffic Using Time-Related Features, Rome, 2016, pp. 407- 414.

- [17] M. Drake, "Digital Ocean," 10 02 2020 . [Online]. Available:  
<https://www.digitalocean.com/community/tutorials/how-to-use-cron-to-automate-tasks-centos-8>. [Accessed 01 02 2021].
- [18] S. Miller, K. Curran and T. Lunney, "Multilayer Perceptron Neural Network for Detection of Encrypted VPN Network Traffic," Resear Chgate, 2018.