Anomaly Detection In IoT Using Machine Learning Algorithms

by

Aritro Roy Arko
15201020
Saadat Hasan Khan
15201013
Afia Anjum Preety
15301046
Mehrab Hossain Biswas
15201008

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
August 2019

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Aritro Roy Arko 15201020	Saadat Hasan Khan 15201013
Afia Anjum Preety 15301046	Mehrab Hossain Biswas 15201008

Approval

The thesis/project titled "Anomaly Detection In IoT Using Machine Learning Algorithms" submitted by

1. Aritro Roy Arko (15201020)
2. Saadat Hasan Khan (15201013)
3. Afia Anjum Preety (15301046)
4. Mehrab Hossain Biswas (15201008)

Of Summer, 2019 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on August 28, 2019.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty
Associate Professor
Department of Computer Science And Engineering
BRAC University

Program Coordinator:
(Member)

Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Mahbub Alam Majumdar
Chairperson
Department of Computer Science and Engineering
Brac University

Abstract

Internet of Things (IoT) is growing as one of the fastest developing technologies around the world. With IPv6 settling down, people have a lot of addressing spaces left that even allows sensors to communicate with each other while collecting data, leaving alone cars that communicate while travelling. IoT (Internet of Things) has changed how humans, machines and devices communicate with one another. However, with its growth, a very alarming topic is the security and privacy issues that are encountered regularly. As many devices exchange their data through internet, there is a high possibility that a device may be attacked with a malicious packet of data. In such cases, the security of the network of communication should be strong enough to identify malicious data. In other words, it is very important to create an intrusion detection system for the network. In our research, we propose a comparison between different machine learning algorithms that can be used to identify any malicious or anomalous data and provide the best algorithm for two data-sets. One dataset is on the environmental characteristics collected from sensors and another one is network dataset. The first data-set is developed from the data exchanged between the sensors in an IoT environment and the second dataset is UNSW-NB15 data which is available online.

Keywords: IoT (Internet of Things), Anomaly detection, Intrusion detection, Machine learning algorithms, Sensor data, Network Data, UNSW-NB15.

Acknowledgement

Firstly, all praise to the Almighty for whom we have completed our thesis in a smooth way and under complete guidance.
Secondly, we thank for the support and advice of Dr. Amitabha Chakrabarty sir in our work. He has pushed us to test our limits and bring out results in a sector we struggled to understand.
And finally, to our parents who have always motivated us and inspired us to work. Without their thorough support, our thesis would not have been complete. With their prayers, support, inspiration and investment, we are standing almost at the end point of our graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is IoT

Internet is the universal system of interconnected computer networks that connects devices globally by using Internet protocol suite (TCP/IP) . IoT is the short form for Internet of Things. Internet of things or internet of objects is basically the concept of connecting embedded devices, computers or sensors through wireless or wired network to the internet [1]. These devices can be monitored and controlled. They can also communicate and interact among themselves over the internet and can exchange data. The devices mentioned include everything from mobile phones, home security, smart television, vehicles etc [2] and are often called connected or smart devices as they can communicate through a process known as machine-to-machine (M2M) communication.

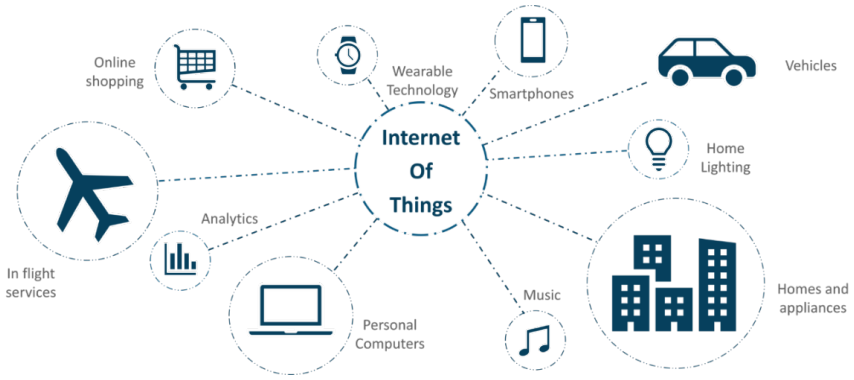

Figure 1.1: IoT in action

1.1.1 How does IOT work?

Smart-phones have been the most important object and plays a major role in the IoT as it has its own sensors which it can use to exchange data that it acquires but also control other IoT devices [3]. An example can be controlling a home security camera from any part of the world using internet connection. IoT devices contain sensors and at times also embedded processors that react to the data it collects from the surrounding. However, all IoT systems must have sensing, processing and actuating capabilities. Some systems are preferred to have a data storage for the

use of data in the long run. At times, IoT devices can be used to collect data that can assist machine learning works to build a system for our betterment.

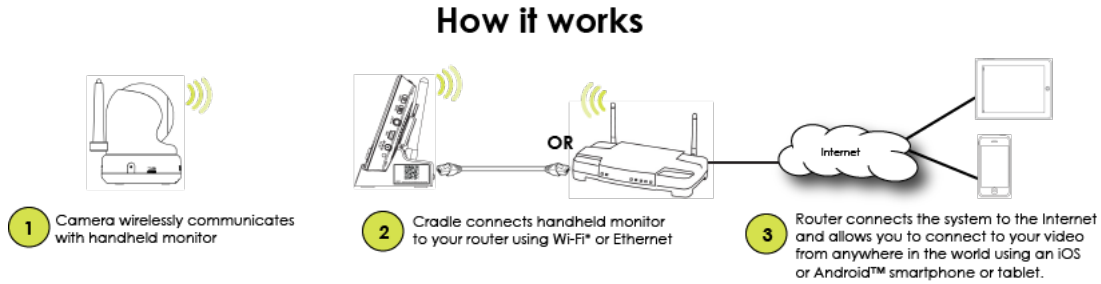

Figure 1.2: Mechanism of IoT

1.1.2 How big is the IoT?

Getting bigger every day, there are already around 8.4 billion IoT devices connected to the internet in 2017 and this is estimated to increase to 20.4 billion within the year 2020. Among this 8.4 billion devices, around 60 percent will be consumer products and rest will be used in the business sector like service or industries according to Gartner (January 2017) [4].

1.1.3 Application and advantages

IoT provides several day-to-day advantages not only for consumers sector but for business sector as well [5]. This technology allows us to view real time information that was not available before. Business can improve their production efficiency by reducing material waste and unforeseen downtime. Since Artificial Intelligence is used, it saves considerable human effort and time. Sensors can be used to build infrastructure, detect damage to infrastructure. Automated traffic control can help reduce road congestion and gadgets or sensors can be implemented on the outside to recognize change in the environment and warn us for any impending natural calamity. IoT makes our homes, offices and vehicles smarter, more measurable and this in turn makes our life more pleasing. Smart speakers like Amazon's echo or Google home makes it very easy to set timers, get information on the go or be it to play music. Through home security system, we can monitor our home from other places remotely and can see what is happening on the inside and outside or talk to visitors. Smart lighting systems can reduce unnecessary energy consumption as it stays off if there is no one in the house or smart air conditioners will turn on when it knows you are on your way to your home.

1.1.4 Disadvantages and Security Issues in IoT

Security in IoT is a double edged sword. Now that we have a system of interconnected devices on a network it makes the system somewhat secure and efficient. [3] Though connected devices generate colossal amount of data and are constantly being exchanged between devices so it in turn generates lots of internet traffic. These data can make the devices useful but this easily accessible nature of the internet brings up both security and privacy issues. These issues arise for many reasons.

Figure 1.3: Disadvantages of IoT described

Flaws in software is found very frequently because of the fact that IoT devices lack the capability to be patched or fixed. This makes them prone to security risks like DDoS attacks, one of the most common issues, which happens by setting the device's password to its default password which are easily crackable by hackers. Ransomware and Malware both relies on encryption to lock out the users devices completely and steals the user's data. Privacy remains among the largest issues of IoT as data is being transmitted, stored and processed and being harassed by large companies. This very data is then being sold to various other companies which is violating our rights for privacy and data security. Home Invasion, one of the scariest threats that IoT can possess as most of the homes and offices rely on automatic processes that leads to the use of IoT devices in these places. IP address of these devices are exposed to hackers who can use it to pinpoint the location and gain access to our personal data just to sell them to underground sites which are the hives of criminal activities.

1.2 Overview of Intrusion Detection Systems

Intrusion Detection System or Prevention Systems are systems that can be used to identify anomalies in information systems. Intrusion prevention system is considered to be a proactive method where as Intrusion Detection System is not. Therefore an Intrusion Prevention System would take necessary measures to eliminate the threat upon the discovery of one. However, an Intrusion Detection System would not.
An IDS (Intrusion Detection System) and an IPS (Intrusion Prevention System) can be network based or host based. This means they can be used to protect an entire network or just protect an end user. Intrusion Detection System were initially rule-based systems [6]. Intrusion Detection Systems have evolved to being subject to algorithms supported by machine learning where the machine can learn itself to categorize a threat. Over the years, development of IDS has taken place on several networks and were tested by several algorithms [7].
The question that comes first while applying any machine learning algorithm is whether it will be supervised or unsupervised. If the data is already labelled, then the IDS may work fine for that particular data set. But it would not be applicable to different data sets, specially the new intrusion type data sets [8]. Therefore,

automatic clustering should be done in datasets which are developing now and then. Unlabeled data should be clustered and labelled where data which are malicious should be identified and labelled as such. Data which are not malicious should be labelled like wise. Then, Machine learning algorithms can be applied to train the Algorithm and IDS systems can be developed. Algorithms for clustering include K-means clustering, IWC and machine learning algorithms have previously included Decision Tree C4.5 and C5.0 for implementing the IDS [9].

1.3 Machine Learning

1.3.1 Definition

American pioneer in the field of Artificial Intelligence, Arthur Samuel, is responsible for introducing the term machine learning in 1959. Machine learning works on prediction using computer and data and also associated with computational statistics. ML is a sub-category of AI. ML contains different theories, methods and application domains as it has a strong connection with mathematical optimization.
Machine Learning primarily includes the following four steps [10]:
• Choose the basis attributes or features or values for prediction. • Choose the appropriate machine-learning algorithm according to the dataset. For example, among classification algorithm or regression algorithm, which one to choose, high complexity or faster one. • Training and evaluating model performance based on different algorithms. • Classify or predict the unknown data, training model is used.

1.3.2 Supervised Learning

Algorithms for supervised learning are used to build a model from a set of data that has both the desirable inputs and outputs. The data used is training data and contains training examples that has single or multiple inputs and an output. The method used is known as supervisory signal.
When dealing with problems regarding supervised signal, first we begin with a set of training examples with correct labels associated with it. An example is when we train a supervised learning algorithm which takes thousands of pictures of handwritten digits with the correct labels containing the correct number for each image it represents. This enables the algorithm to classify handwritten digits and also learn the relationship of the images and their respective numbers and also use this relationship to classify new images without labels that has not been shown to the machine before.
Classification and regression is a part of supervised learning [11]. Algorithms for classifications are used in cases where outputs are restricted to sets of values which are limited. Algorithms for regression are used in cases when outputs may have values which are numerical within a given range.

1.3.3 Unsupervised Learning

Algorithms for unsupervised learning are given sets of data that contain inputs only and then the algorithm finds clustering of data points or finds structure in data. Therefore the algorithms learns from non-labeled, non-categorized or non-classified

test data. Unsupervised learning algorithms detect commonalities in the data which is based on the absence or presence of the commonalities in each new piece of data instead of a feedback approach.
Grouping, summarizing and finding underlying structure of unlabeled data in a useful manner is the function of unsupervised learning.

1.3.4 Anomaly Detection

Anomaly detection or outlier detection, in different words, is used for figuring out the rare observations in events or items, which lead to suspicions due to its differentiation from the majority of data.
Anomaly detection can be divided into three broad categories. From an unlabeled test data set, unsupervised anomaly detection algorithms identify anomalies under the assumption, which tells us that, most of the instances in the data provided are normal and by observing the data that are least fit to the remaining set of data. Supervised anomaly detection techniques need to be provided with a data-set which clearly identifies and differentiates between data that are normal and abnormal.

1.4 Aims and Objectives

With the widespread use of IoT systems, the number of malicious attacks has also increased [12]. According to Gartner Research [13], the number of IoT devices being used by 2020 will hit 26 billion. To ensure the system and data are only accessible to authorized entities such as individuals and process, logical controls are used as software shields. Intrusion detection and prevention system, passwords, access control etc. includes logical controls[14]. One logical control against IoT attacks is to implement an Intrusion Detection System (IDS). If there is a security breach or an attack on the IoT system, the IDS should be able to identify that an attack has taken place. The IDS should also be able to send alerts in such cases (Gordon, 2015). Moreover, an IDS is able to detect a fault using two techniques. They are: 1) Signature based detection 2) Anomaly based detection. Signature based detection methods are useful when detecting known attacks. Anomaly based detection are useful against unknown attacks or security breaches [15]. Usually anomaly based detection systems depend on AI or Machine Learning (ML) algorithms. The idea is to make the system learn and make it capable to distinguish between normal and abnormal behavior of the system. Conversely to detect specific patterns of the attacks that are known by investigating network data or traffic, signature based detection methods are powerful [16].

Chapter 2

Related Work

In 2015 the number of Internet of Things (IoT) connected devices was 15.41 billion worldwide. By the end of 2019 it will hit 26.66 billion. In 2025, the number of devices connected to the Internet will be 75.44 billion, causing the number to rise 5 times in a decade. [17]. These smart devices connected to the Internet are capable of generating data on its own based on behavior or expected result and share it over the Internet. This is what constitutes the concept of Internet of Things (IoT). IoT is a massive group of devices containing sensors or actuators connected together over wired or wireless networks.

IoT requires intelligent processing and reliable transmission within the network. Besides all the benefits and positive characteristics that the wireless medium has brought to IoT, there are vulnerabilities. There are different forms of traffic anomalies and attacks. The most common security threats are intruders, which is generally referred, as hacker or cracker and the other is virus. This comes with the risk of personal data being broadcasted to the world and cause more cyber-attacks like phishing, denial-of-service (DoS), Probe, Remote to Local (R2L) attack etc. Since these attacks are not common so normal data flow has a pattern which can be more or less the same. However, when another entity will try to distort or change the data, the pattern will change and this is where the Intrusion detection techniques comes into play in the context of internet and specifically in the context of Internet of Things.

To ensure the system and data are only accessible to authorized entities such as individuals and processes, logical controls are used as software shields. Intrusion detection and prevention system, passwords, access control etc. include logical controls [14].

To monitor and detect anomalies or any suspicious behavior, intrusion detection system (IDS) and intrusion prevention system (IPS) are being used. As diverse environments and latest technologies are prone to be maliciously attacked, machine-learning (ML) algorithms are able to detect, analyze and classify intruders in network accurately and rapidly [18].

Two types of intrusion detection system are there. They are: Anomaly based detection and Signature based detection. Anomaly based detection methods detects the attacks that are unknown by observing the whole system and the entities in it such as traffic, objects etc. Anything other than normal behavior is identified as a potential attack. Conversely to detect specific patterns of the attacks that are known by investigating network data or traffic, signature based detection methods

are powerful [16].
After researching on specific use cases, data characteristics and system needs, such a diversity of characteristics has confirmed that there is no unique method that would provide efficient and accurate intrusion detection for every dataset under the analysis [18].

2.1 Dataset

Obtaining labeled data in IoT for intrusion detection is quite difficult. KDD (Knowledge Discovery and Data Mining) Cup 1999 dataset published by MoD, as one of the freely available non-IoT dataset for intrusion detection [19]. But after a long period of researching and analyzing, it has been marked as full of error and outdated [9]. In [20] author points out differences in host and network-based intrusion detection techniques to demonstrate how the two can work together to provide additionally effective intrusion detection and protection. For that purpose, the main idea is to analyze the possible role of a set of chosen ML techniques in a way to construct strong defense mechanism for IoT environments based on dataset/flows for following two layers: I. Application Layer (Host based) II. Network Layer (Network Based)

2.2 Application Layer

The application layer uses machine learning, data mining, data processing, and other analytics to process information from the system and provide an output.
To work on the intrusion detection in application layer, [21] used a freely available unlabeled IoT dataset from Intel Barkley Research Lab. The dataset contains more than 2.3 million readings collected from 54 sensors in Intel Berkeley Research lab in 2004 [22].
To use the dataset further in anomaly detection research these unlabeled data require processing, as it is not primarily aimed for intrusion detection.

2.3 Network Layer

Even though cyber-security research is almost in its peak, there's a scarcity of proper dataset. Recently IoT eligible UNSW NB15 dataset has came to rescue modern intrusion based researches for network layer. This dataset has a huge number collection of normal network traffic and malicious network traffic instances [18].
It integrates normal behavior and latest attack instances in traffic. Malicious traffic is generated for a number of typical, up to date attack types. The lower number of malicious attack instances than the large number of normal traffic instances makes the UNSW-NB15 dataset unbalanced. This dataset is providing a good support for a wide range of networking IDS analysis [23].

2.4 Algorithm Analysis

Some of the relevant algorithms and their limitations are discussed in this section. In research paper [9], researchers have applied and compared the performances of

several supervised learning algorithms, including SVM (Supervised Vector Machine) and a range of ensemble classifiers. The ensemble algorithms combine basic classifiers, which are trained on different subsets of dataset. IDS (Intrusion Detection System) is capable of differentiating between abnormal behaviors to abnormal behaviors based on effective classification model built inside the IDS. An IDS is based on SVM (Support Vector Machine) and C5.0, which is a widely used implementation of Decision tree with each nodes, aim to provide an answer with regards to the specific attributes in the input record. Generally existing IDS fails to detect the incensement in attacks in critical cases because it is unfamiliar with such attacks as those attacks are new. Using the combination of C5.0 and SVM, data can be divided into classes and after being trained; SVM will be able to map data into high dimensional space [24].

In [9], researcher has talked about an intrusion detection model, which uses MapReduce to process large structured data set into key pairs. It combines SVM with Fuzzy C-Means (FCM). For clustering data it uses FCM and SVM for classifying data. It is mentioned in [25] that FCM is useful because unlike non-FCM it allows data points to belong to multiple clusters. It is also proposed a hybrid intrusion detection system using two data mining techniques, K-means and Naïve Bayes, for clustering and classifying [9].

IWC is an amplified version of k-means algorithm. As an input of it, the updated version of Intel Lab IoT dataset can be used. IWC algorithm clusters data in the dataset into two sections based on similarities between data records labeled as "normal" and "abnormal". After that the input data is split into two parts; training data (70 percent) and testing data (30 percent). The training data is used to build decision tree, and for predicting the testing data is used. To implement the algorithms and test the system MATLAB is used [22].

Chapter 3

Data Collection And Processing

3.1 Network Data Pre-processing

59.166.0.9	44113	149.171.12	6146	tcp	FIN	0.037533	3614	49590	31	29	7	24	-	757520.1	10399595	60	62	255	255
59.166.0.9	6592	149.171.12	43897	tcp	FIN	0.02199	2750	31360	31	29	7	17	-	977899.1	11171260	44	48	255	255
59.166.0.3	37603	149.171.12	14720	tcp	FIN	0.005324	3880	2456	31	29	7	7	-	5507138	3486101	18	18	255	255
59.166.0.4	10950	149.171.12	53	udp	CON	0.001073	130	162	31	29	0	0	dns	484622.5	603914.3	2	2	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0
175.45.176	0	149.171.12	0	ospf	INT	1.213061	3024	0	254	0	0	0	-	19230.69	0	28	0	0	0

Figure 3.1: Network Data

Labeled dataset for intrusion detection in internet of things (IoT) is something which is very rare and difficult to find. For our research, we are working with anomaly detection in IoT for both the application layer and network layer. We are using UNSW-NB15 dataset to detect anomaly in the network layer [26].
According to the UNSW-NB15 Testbed figure, the traffic generator IXIA Perfect-Storm tool was used to create a hybrid of modern normal and synthetic contemporary attacks of network layer in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS). The dataset has total 2,540,044 numbers of records divided and stored in four CSV files. IXIA is divided into two networks I1 (10.40.85.30) and I2 (10.40.184.30) which connects total 3 servers. Server 3 is for spreading malicious attacks in traffics; in contrary server 1 and 2 is for spreading normal traffics. There are two routers connected other devices of clients and firewall. To capture 100GB of network traffic data (Pcap files), Tcpdump tools have been used. Each Pcap file contains 1000MB of traffic. 12 algorithms have been developed and The Argus, Bro-

IDS tools to generate total number of 49 features with the class label. UNSW-NB15 data set were classified into nine types attacks as in the following points: Fuzzers Analysis, Backdoor, DoS, Exploit, Generic, Reconnaissance, Shellcode, Worm.
The UNSW-NB 15 data set are characterized into six groups as follows:
1) Flow features: The identifier attributes between hosts, such as client-to-server or server-to-client are mentioned here.
2) Basic features: This group includes the attributes that are responsible for protocols connectivity.
3) Content features: This includes the attributes of TCP/IP and some attributes of http services.
4) Time features: Attributes associated with time are present here. This includes arrival time, start/end packet time and round trip time of TCP protocol.
5) Additional generated features: This category can be further divided into two categories: (1) General purpose features (no. 36 - 40) in which the feature tries to protect the service of protocols. (2) Connection features (no 41- 47) are created from the flow of 100 record.
6) Labeled Features: This section presents the label of each record.
The dataset contains categorical features. There are a total of 49 columns. The data types are different. They range from being integer, float to time-stamp and nominal values. It is very important to convert them into numerical values to be used for machine learning purposes. Machine Learning algorithms work only on numeric values.
First, the UNSW-NB15 csv files are read into separate dataframes because the dataset is huge and reading the whole dataset into a single dataframe is very time consuming. The separate dataframes are then concatenated into a single dataframe. The dataframe columns that are of integer type is converted to numeric values using the convert-objects method. In the same way, float type, binary type columns are converted to their appropriate numeric values. The next step is removing NaN values for columns that contain them. NaN values are transformed to zero. The nominal features are then dealt with. Label encoding converts the nominal type values into their appropriate classes that are represented using numeric values. All the features are then normalized because UNSW-NB15 contains attributes which have a mixture of scales. Normalization is a type of scaling where all the values are converted into a range of 0 and 1. Normalization is done to ensure that all the attributes follow the same scale. Some algorithms like kNN etc work best when a scale of 0-1 is followed for all the features. The processed data is then stored in a separate data.h5 file which can be imported before applying the machine learning algorithms for intrusion detection.

3.2 Application Layer Data Collection & Experimental Setup

On the other hand, we have created our very own dataset for the application layer. This is a replication of the Intel Lab Data. The whole data collection process is done in the CSE Research Labs of BRAC University and our own houses for a period of 2 months. We set up 10 boards in different parts for greatest amount of variation possible. However, most of the data are similar because the conditions

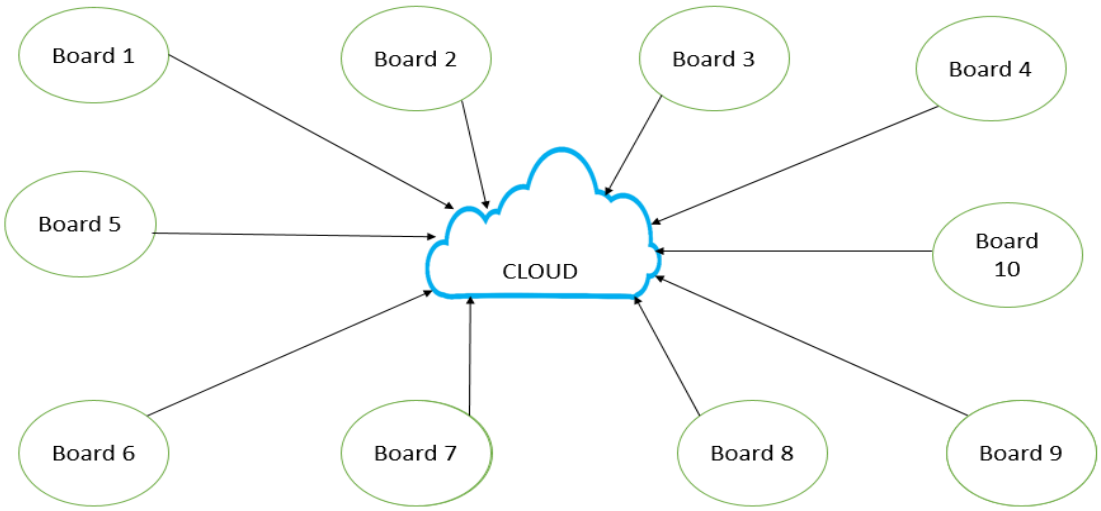

Figure 3.2: Block Diagram Of Board Interconnections

are very much the same in each place. Figure. 3.2 shows the block diagram of the connection between 10 boards placed in 10 different places and the cloud.
Figure. 3.3 represents the block diagram of hardware setup of a single board. Each board consists a total of 4 sensors which includes temperature sensor, humidity sensor, light sensor and voltage between batteries. The temperature is in degree Celsius, humidity is represented as percentage between 0-100 percent, light intensity in terms LUX where the highest possible intensity is approximately 400 LUX in a very bright room and lastly, the voltage varied between 2-3 V.
The system is designed in such a way that it takes data as input from the sensors and then passes the data to the Node MCU (ESP32) respectively. The microcontrollers then compute and convert these electrical readings to scaled values like LUX, humidity (percentage), temperature (degree Celsius) and voltage (Volts) which are understandable by humans. For measuring the temperature and humidity, we have used the DHT11 module. The sensor produces a voltage according to the environment in which it is operating. This is then converted to suitable temperature and humidity values. The values are processed within the microcontroller. The light intensity is detected using a LDR based sensor module which is used to determine the amount light in the room in terms of LUX. Furthermore, raw voltage readings from the battery are passed on to the ESP32 directly. The voltage varies from 2V-3V. All these data are accumulated and sent to the Adafruit cloud using ESP32 connected to a WIFI connection. The Adafruit cloud has a maximum data transfer rate of 30 data/minute. The data gets stored in 4 separate feeds from a single board for the 4 features at the same time. For our research, we collected the data for a period of 2 months and then merged all the data in a single CSV file for applying different anomaly detection algorithms.
For the data collection process, we have used the following hardware devices: 1). ESP32 2). DHT11 sensor 3). LDR Light Sensor Module 4) Battery (3V)

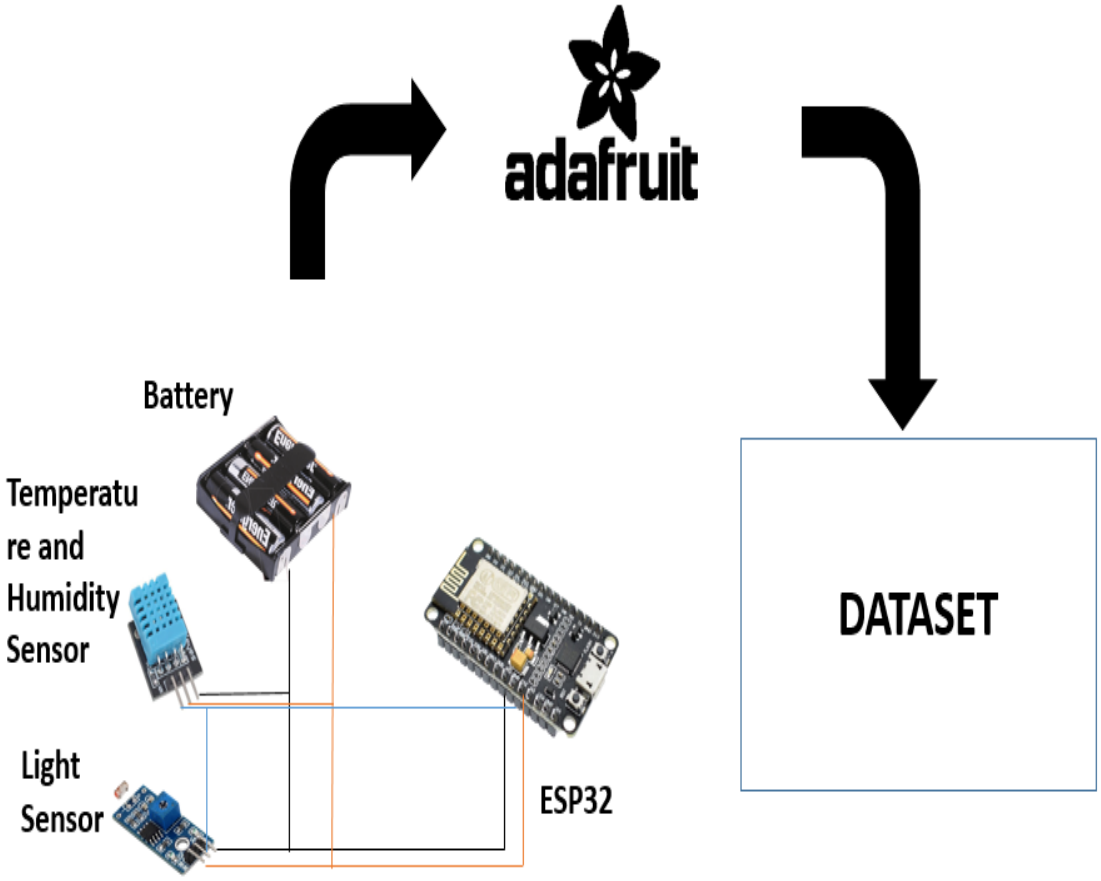

Figure 3.3: Block Diagram Of A Single Board Connected To Cloud

3.3 Hardware Connections

Figure. 3.4 shows the original connections of the hardware devices in a single board. ESP32 is a NodeMCU which is a low cost system on a chip microcontroller with WIFI and Bluetooth integrated in it. It is primarily used for IoT based projects that require internet connection for communication of data. As we are sending data to Adafruit cloud, we are using this particular device for communication. We are implementing a total of 10 ESP32 microcontrollers, which are placed in separate parts of the research labs and our houses. One ESP32 microcontroller has connections with a DHT11 sensor, LDR light sensor module and battery. This chip has a total of 48 pins which includes Ground (Gnd) and a VCC pin of 3.3V instead of 5V. There are a total of 18 ADC (Analog to Digital converter) pins and only 2 DAC (Digital to Analog converter) pins. The input channel are of 12 bits, so values range from 0-4095. 0 refers to 0V and 4095 refers to 3.3V. For our research, we have pins 2, 32 and 35 for connecting DHT11 sensor (temperature and humidity sensor), battery and LDR respectively.
DHT11 is a device which is used to measure temperature and humidity at the same time. Temperature readings are given in terms of degree Celsius and humidity readings are given in terms of percentage. There are mainly 3 pins: 1) GND 2) VCC 3) DATA. The ground is connected to the GND pin of the ESP32, VCC to the 3.3V and DATA pin to pin 2 of the NodeMcu. The operating voltage of DHT11 is from 3.5V to 5.5V.

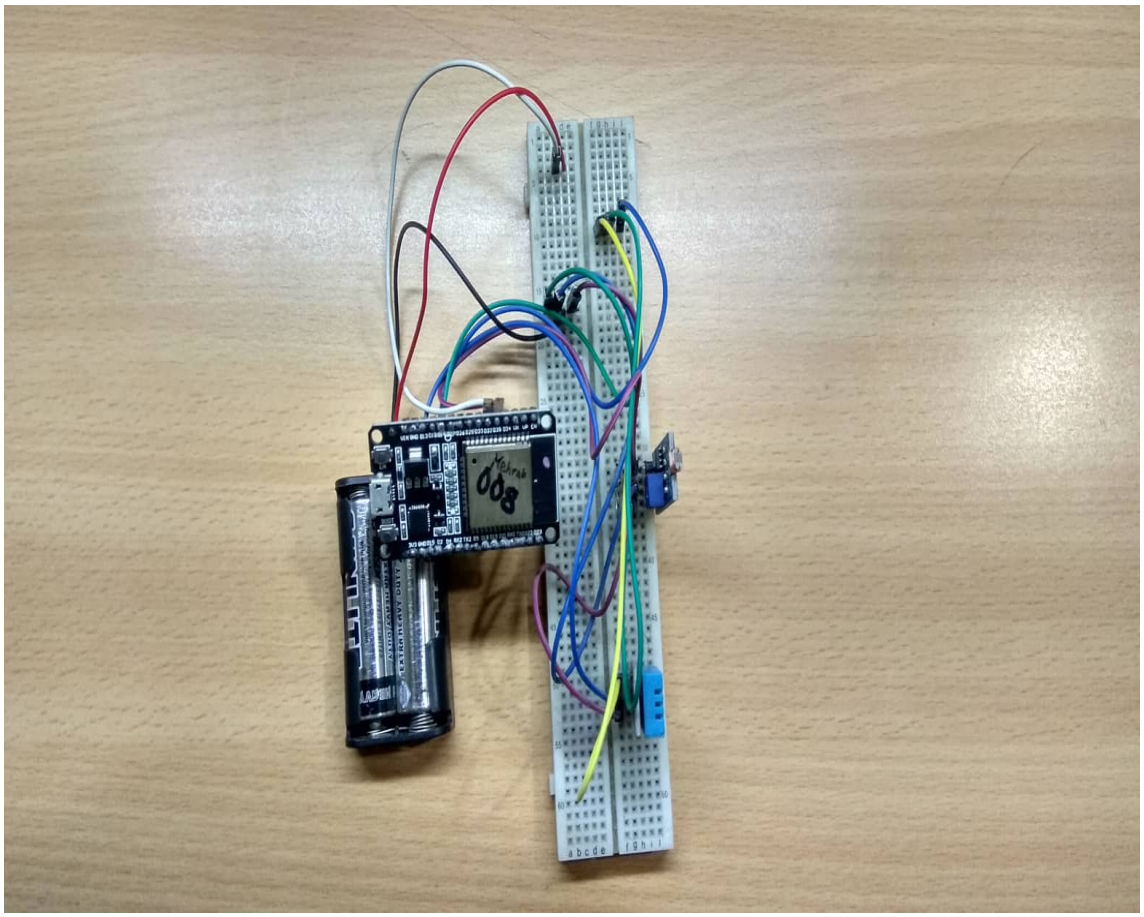

Figure 3.4: Top View Of A Single Board

The light sensor is a photoresistor which can have both digital and analog outputs. It has parts/pins similar to the DHT11 sensor module. They are : 1) GND 2) VCC 3) OUTPUT. The ground is connected to the GND pin of the ESP32, VCC to the 3.3V and OUTPUT pin to pin 35 of the NodeMcu. It is very important to note that we have represented light in terms of LUX. As a result, we needed some calculations to convert the analog reading into LUX. The formulae used are as follows:

$$VoltLDR = 0.0008057 * analogReading \tag{3.1}$$

$$ResistLDR = (10000 * (3.3 - VoltLDR))/(VoltLDR) \tag{3.2}$$

$$LUX = (500 * ResistLDR)/64881.00 \tag{3.3}$$

3.4 Sensor Data Pre-processing

Our data set has 4 features. They are: 1) Temperature 2) Humidity 3) Light Intensity 4) Voltage. Labeling approximately 70,000 data manually is very time-consuming. As a result, we used K-means clustering to add appropriate labeling. There may be subgroups in a data set and the subgroup data points remain close and form clusters. In our research, there are two types of data. They are 1) Normal 2) Abnormal. As a result, we formed two clusters. The normal cluster is formed using data that we collected using the hardware setup. There are also abnormal data

id	Temperature	Humidity	Light Intensity	Voltage
0E3XMZQF2E6J29R	31.6	47	246.68	2.65
0E3XMZTH04HC0GI	31.7	47	227.92	2.52
0E3XMZXJY7515X7	31.7	47	235.78	2.63
0E3XN00MVDCS1YI	31.7	47	226.45	2.6
0E3XN03PT1NYTD3	31.7	47	214.94	2.68
0E3XN06RQG8G12T	31.7	47	268.29	2.66
0E3XN09TN96NWNI	31.7	47	247.02	2.56
0E3XN0CWN4A88FC	31.6	47	264.92	2.54
0E3XN0FYGMRM1P	31.7	47	246.68	2.69
0E3XN0K0EG7DYK2	31.7	47	263.45	2.7
0E3XN0P2CN5GQA4	31.7	47	219.88	2.54
0E3XN0S4A4GYJGX	31.7	47	217.66	2.61

Figure 3.5: Dataset Before Labeling

from the hardware setup. Furthermore, we have also added synthetic data that are not within normal range and hence they are abnormal too. The abnormal cluster is formed from these wrong synthetic data and hardware collected data that are far beyond the range of being described as normal. The value of 'k' for our dataset is two, since we are forming two clusters of 'Normal' and 'Abnormal'. The algorithm tries to keep the subgroups as far away from one another to avoid overlapping of data, represented with a centroid for each cluster.

Before applying the machine learning algorithms, we needed to process the data. First, the data is filtered to a two column dataset, where Light intensity and Voltage values are discarded. This is because the Light Intensity and Voltage values have no relation with the other features of the dataset. We then studied the data and deduced that a correlation exist between Humidity and Temperature of the data. While that being said, the data also forms clusters between abnormal and normal values when Humidity against Temperature scatter diagram is plotted. This shows that forming clusters is only possible based on the relationship shared between humidity and temperature and not others, as there are no clusters when scatter diagrams are plotted with other features. Based on this logic, K-means clustering is done to make clusters. The data that we got from the sensors are classified as normal. The data that are synthetically generated are labelled as abnormal. After that, the discarded features are also included back in the dataset for machine learning purposes. Inclusion of the other features increases the number of unique rows. This helps in better training and testing and ensures that training and testing is done on different and unique rows of data. We then filtered out rows which are similar to each other. We just kept the unique rows. The Figure. 3.6 shows clusters formed identifying them as normal, abnormal and synthetic (also falls under

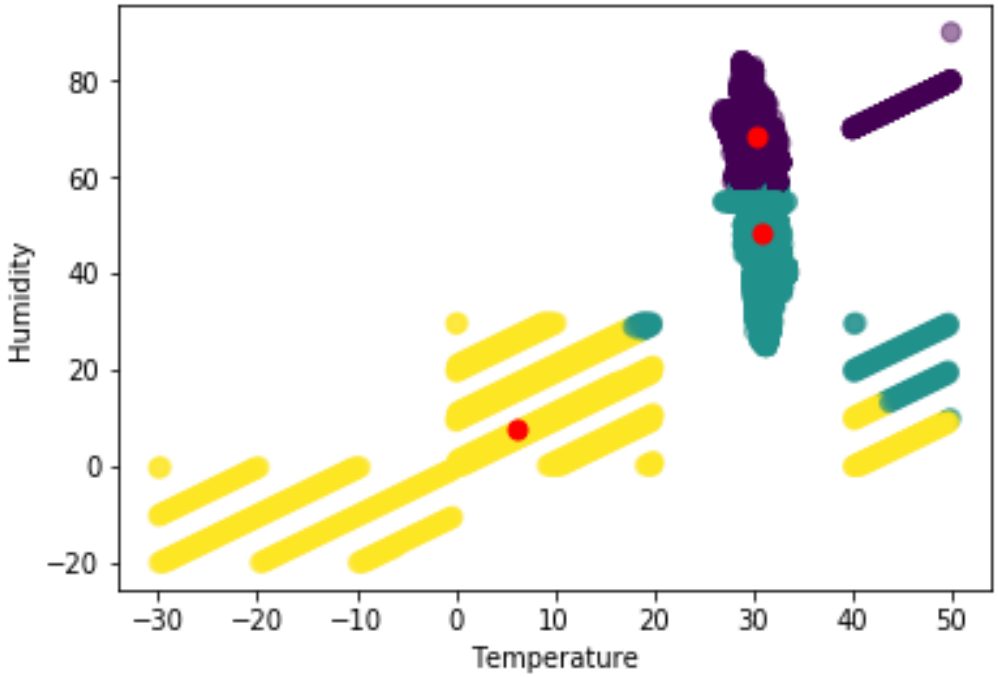

Figure 3.6: Scatter diagram between Temperature and Humidity

Temperature	Humidity	Light Intensity	Voltage	Output
31.6	47	246.68	2.65	1
31.7	47	227.92	2.52	1
31.7	47	235.78	2.63	1
31.7	47	226.45	2.6	1
31.7	47	214.94	2.68	1
31.7	47	268.29	2.66	1
31.7	47	247.02	2.56	1
31.6	47	264.92	2.54	1
31.7	47	246.68	2.69	1
31.7	47	263.45	2.7	1
29.9	59	206.03	2.51	0
29.9	59	221	2.51	0

Figure 3.7: Dataset after labelling

abnormal class). Figure. 3.7 represents the dataset after labelling is done. The label '1' means normal data and label '0' refers to an abnormal value.

Chapter 4

Research Methodology

Collection of raw data is very hard to do. Moreover, working with raw data is tough as it contains many repetitive rows and anomalous values which do not reflect the true situation. Therefore, raw data has to be filtered and moulded into a dataset that can be utilized according to the researcher's purpose. Similarly, our data is filtered to remove rows that are exactly same. Synthetic values are introduced in order to reflect better results on the algorithms. Some works such as removing similar rows, splitting dataset into test and train, are common before running each algorithm. Each algorithm runs in different conditions and different environments too.

4.1 Handling Data and Splitting

4.1.1 Sensor Data

Raw data is filtered many times to remove unusually different and abnormal values. The filtered dataset is then obtained with 76755 rows of data. The 76755 rows of data contain one of each of Humidity, Temperature, Light Intensity and Voltage values. Therefore, the set of 4 values defines the uniqueness of each set of data. There can be many values of temperature alone which could be the same. However, a row can only be unique if all the 4 values of Temperature, Humidity, Light Intensity and Voltage together resemble a unique set. Moreover, repetition of rows of data would cause problems when testing an algorithm because same rows of data would mean that an algorithm would have a higher chance of being trained and tested with the same data at the same time. Therefore, it shall give us high yet absurd and unreliable accuracy. This is what the theory of "Problem of Overfitting" explains. Overfitting can occur when an algorithm is trained and tested with the same data leading to a very high accuracy achieved by the algorithm. The figures below show how data would differ after removing identical sets of data.
Removal of repetitive rows is necessary for better results. After removal of repetitive data, 69261 data forms. The next process for all the algorithms is the Train - Test split of data. This is same for all them. We never test on the data that we have trained upon. We always try to maintain separate data for training and testing. 80 percent of the entire data is used for training and 20 percent of the entire data for testing. Train and Test split can be obtained automatically by using a built in command in python. However, train and test data can also be stored in separate

Temperature	Humidity	Light Intensity	Voltage	Output
31.6	47	246.68	2.65	1
31.7	47	227.92	2.52	1
31.7	47	268.29	2.66	1
31.7	47	226.45	2.6	1
31.7	47	226.45	2.6	1
31.7	47	268.29	2.66	1
31.7	47	247.02	2.56	1

Figure 4.1: Data having identical rows

Temperature	Humidity	Light Intensity	Voltage	Output
31.6	47	246.68	2.65	1
31.7	47	227.92	2.52	1
31.7	47	235.78	2.63	1
31.7	47	226.45	2.6	1
31.7	47	214.94	2.68	1
31.7	47	268.29	2.66	1
31.7	47	247.02	2.56	1
31.6	47	264.92	2.54	1

Figure 4.2: Data after removal of identical rows

CSV files which could be used while running the algorithm. After these background work, each algorithm is tested and different metrics are obtained which points out how successful each algorithm is.

4.2 Support Vector Machine

Support Vector Machine algorithm is also a classification algorithm more commonly known as SVM. The SVM algorithm classifies the data into different groups which maximizes the utility for any researcher. Based on the different groups, we can separate the data belonging to one class from data belonging to another class. SVM has been applied to places such as time series prediction, face recognition to biological data processing [27]. The SVM algorithm draws a hyper plane which is said to separate the two classes. The hyper plane is just not any ordinary line, but a line which tends to separate the two classes in such a way, such that the distance from the line to the closest point of both the classes remain as large as possible. In other words, it maximizes the distance from the hyper plane to each of the nearest data point of both the classes. The following equations describe the process on how this hyper plane is drawn and two different classes are identified.

$$w * x - b = 0 \tag{4.1}$$

The equation is of the hyperplane which separates the two classes equally. Here w is the normal vector to the hyperplane and x can vary for each set of data, b is the distance between the nearest point to the hyperplane and the hyperplane itself.

$$w * x - b >= 1 \tag{4.2}$$

If, however the result to this equation is positive and equal to one, then we get the hyperplane of the positive class, and any value greater than 1 would give data points of the positive class.

$$w * x - b <= -1 \tag{4.3}$$

Similarly, if the result is equal to -1, then it gives the hyperplane of the negative class and any value less than -1 would be for data points belonging to the negative class.
The margin that is drawn can sometimes be a large margin, or a narrow margin which depends upon the equation below [28]. If the value of C is very high , then it becomes a narrow margin. The narrower the margin the better the separation. If C ever

$$min||w||^2 + C * \sum_{i}^{N} \epsilon i \tag{4.4}$$

If the value of C is very high , then it becomes a narrow margin. The narrower the margin the better the separation. If C is ever infinite, then it is called a hard margin. Thus it means, with variation of C, the degree to which constraints are being maintained can be measured. Thus C is called the Regularization Parameter.

4.2.1 Implementation of SVM on Application Layer Data set

SVM is tested in our application layer data set. Since we have a lot of data, it is arranged in a continuous manner. The SVM algorithm can not work very nicely if the data is very continuous, meaning even with a very narrow margin , it would still make mistakes. Our specific kernel has been tested and the kernel that supports the best with the given set of data has been applied for the implementation of data. For this data set we have used a sigmoid kernel to draw the hyper plane. The figure 4.3 shows how our data would look after applying SVM classifier algorithm.

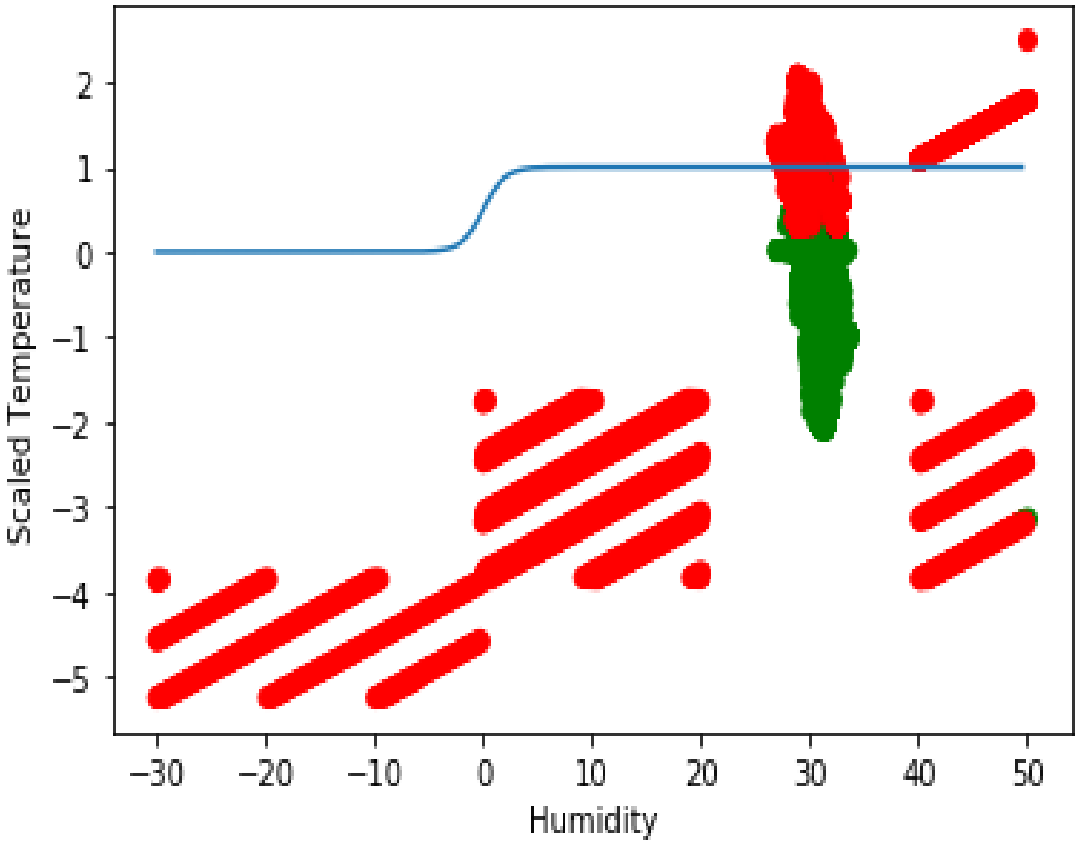

Figure 4.3: Implementation of SVM on our Data

4.2.2 Implementation of SVM on Network Layer Data set

SVM creates another hyperplane on the data for the network layer. Just like the application layer data set, the network layer data set is also run using the sigmoid curve. The sigmoid curve separates the data into two classes. A pre-processed normalized data set was used in order to run the algorithm, as computation power required to run the algorithm on the actual data set would be really high. Based on the input features, the hyperplane is drawn to separate the maximum between the two classes, attacking and non-attacking. The input features include source and destination IP addresses and port numbers, protocols and many more.

4.3 Logistic Regression.

Logistic Regression uses multi-variable analysis which uses multiple variables as features to predict a single outcome [29] In logistic regression, the dependent variable is a binary variable. In simple words, if a function of Y is developed in terms of X, then logistic regression simply finds the probability of P(Y=1) in terms of X. For that the logistic regression uses a sigmoid function to map these data to a probability score. The probability scores are placed in a distribution graph which identifies the two fields of data. The figure 4.4 shows the difference between linear and logistic regression and how it basically operates. The sigmoid function is also known as the

logistic function. The logistic function is calculated by the equation given below.

$$logit(Y) = \pi/1 - \pi \tag{4.5}$$

where pi represents the probability of any outcome and on the basis of this function, a threshold value of probability is set which acts as the border to different classes. Based on the multiple features

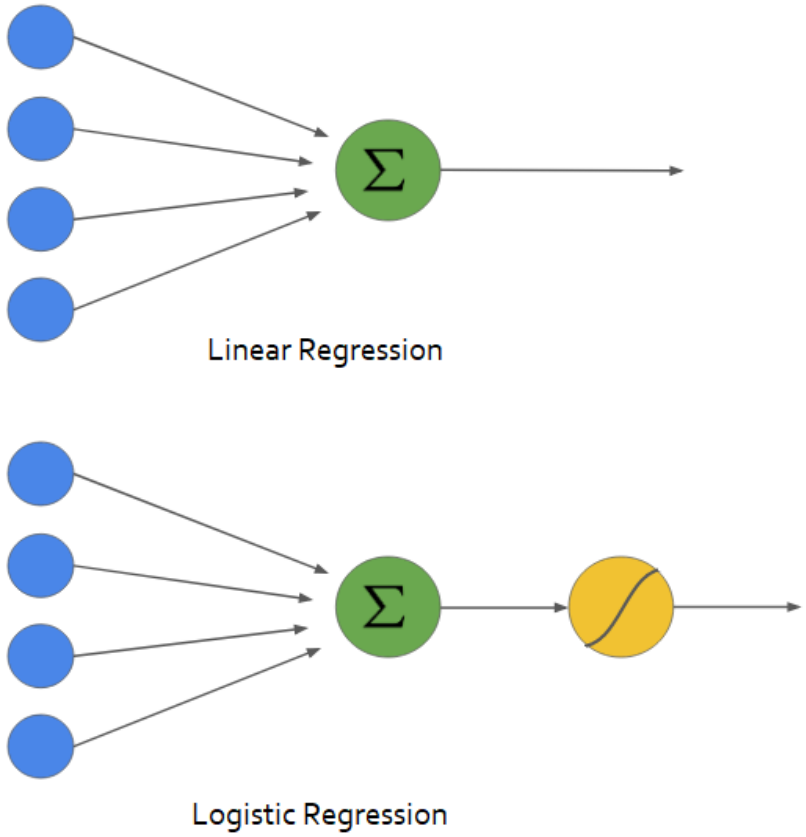

Figure 4.4: Difference between Linear and Logistic Regression

4.3.1 Implementation of Logistic Regression on Application Layer Data

Logistic Regression is used as a key algorithm to correctly identify abnormal and normal values of application layer data. The success of Logistic Regression tells us the fact that it is trained with multiple features in our data set which are the 4 columns Temperature, Humidity, Light Intensity and Voltage. In our data set, there are two classes. The logistic regression model is trained using the logistic function. Therefore, a threshold value in probability is set which ultimately determines the two classes. The figure 4.5 shows how Logistic Regression is applied in our data set and how it separates the two classes in our data set.

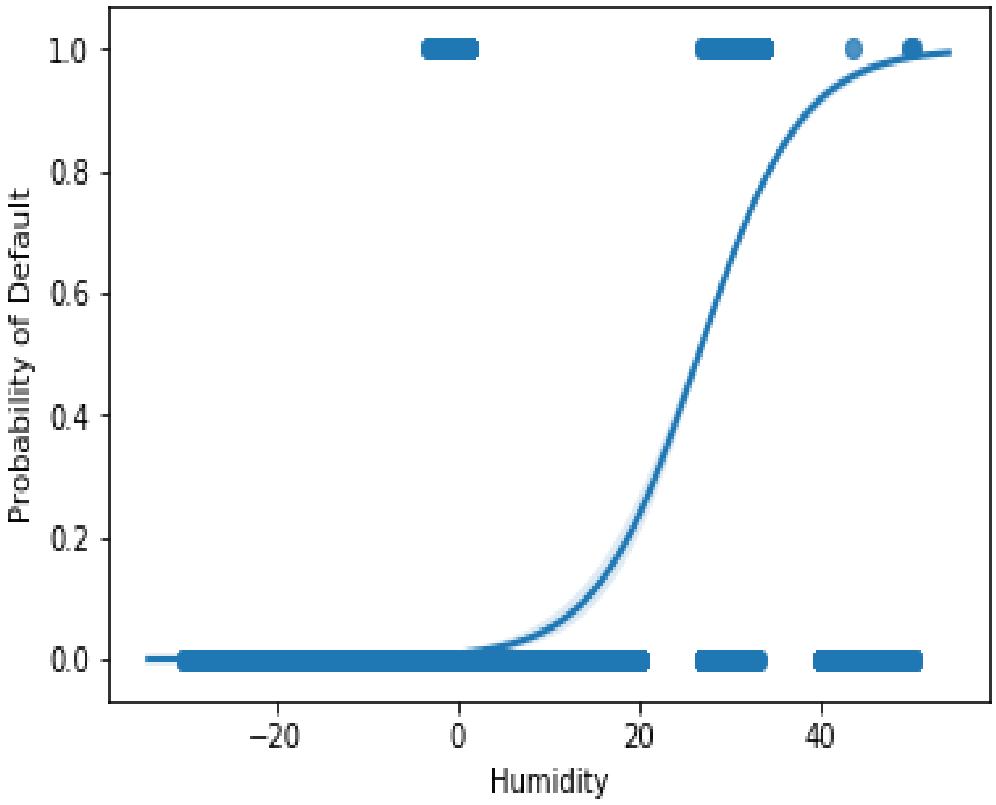

Figure 4.5: Implementation of Logistic Regression on Application layer Data set

4.3.2 Implementation of Logistic Regression on Network Layer Data

Just like the application layer data, data from the network layer is used to build a logistic regression function. The logistic regression uses multiple features to train itself on and then decides whether it is an attack or not an attack. The features to train upon include IP addresses, Protocols, Port numbers and others. And the result column just predicts whether this is an attacking packet, or this is a normal packet. To work with the data set, we have first imported a pre-processed normalized data set. The actual data set is very tough for implementing any sort of work because it is really heavy to process.

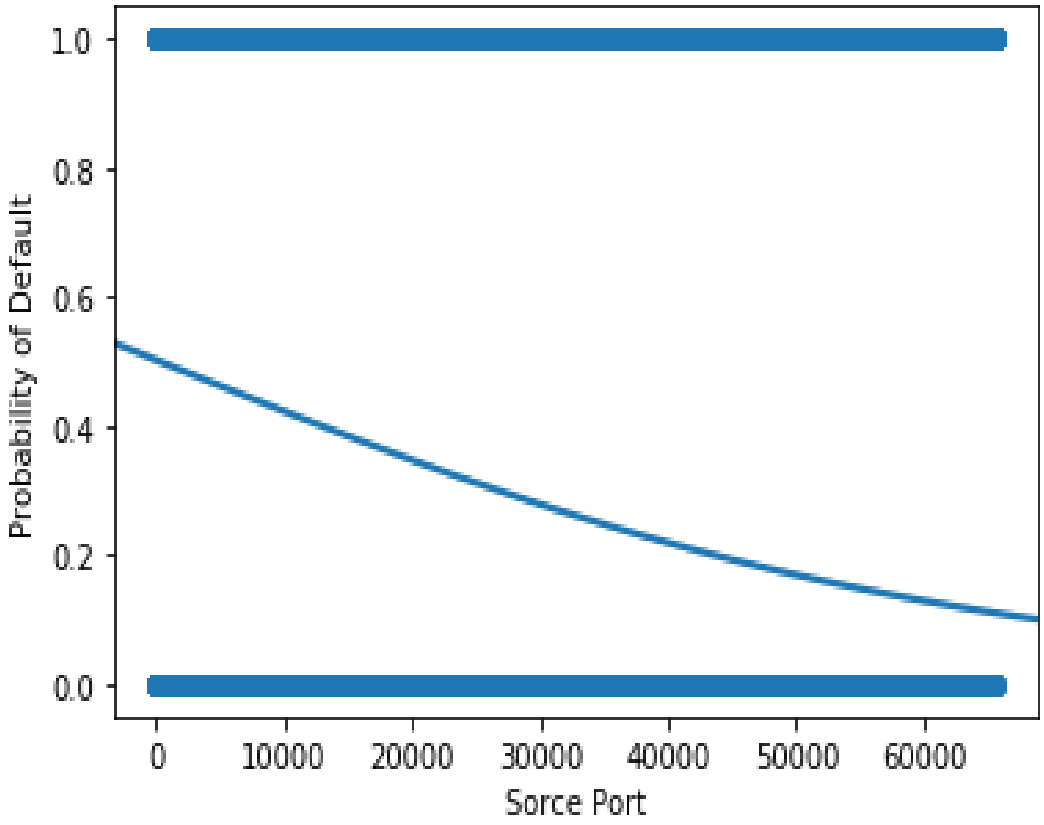

Figure 4.6: Implementation of Logistic Regression on Network layer Data set

The Figure. 4.6 shows how the sigmoid curve is plotted on the last 50,000 data of our data set. The last 50000 data are chosen because it contains a fair equal numbers of attack and non-attack packets. The entire data set could not be used as it takes too much of CPU power to compute.

4.4 K-Nearest Neighbour

kNN is an easy to implement supervised algorithm used in machine learning. It is useful in solving problems related to regression, prediction and classification. It is supervised machine learning algorithm that depends on labelled data taken as input to learn a function and then produces an appropriate output when a new unlabelled data is given. kNN algorithm works well across all parameters of considerations. It is mostly used for ease of interpretation and its low time of calculation. K nearest neighbour algorithm assumes that similar things exist in close proximity and works based on minimum distance from the query instance to the training samples to determine the nearest neighbours. The kNN prediction of the query instance is based on how close it is to the instances of the nearest neighbours. kNN algorithm stores all available cases and classifies new or upcoming cases by a majority vote of its k neighbours and compiles them into well-defined groups.

4.4.1 Implementation of kNN on Application Layer Dataset

Determining the value of k plays a significant role in determining the efficiency of the model. Value of K determines how well the data can be segregated to generalize the result of the kNN algorithm. A higher value of k has advantages that lowers the variance due to outlier values of data. The cons or side effect is developing a bias as the learner tends to ignore smaller patterns which may have useful data. For our dataset, we first found the value of 'k' that best suited our case. We initially classified with kNN using 'k' as 5. However, this had no real logic behind it. As a result, we decided to find the optimal value of 'k' that will give better and correct accuracy. This is done using GridSearchCV. It works by training our model on a range parameters and then finds out the best result based on comparison. In our case, we provided a range of values of k neighbors from 1 to 25 and checked which value of k neighbor provided the most accurate and reliable output. It is seen that k=2 best suited our data set.
After determining the value of k, we load the data set. Duplicate rows with same values for all the four features are dropped to avoid training the same data over and over again. It also prevents testing and training with same dataset, avoiding accuracy levels that are unrealistic. Next, we split the data into train and test. 80 percent of the sample is train set and 20 percent of the data is test set. The 80 percent data is trained with kNN classifier with k initialized as 2. We then test the 20 percent data. It is done by calculating the Euclidean distance between instances of the clusters or classes formed. The distances are then sorted in ascending order. The top k rows are then selected and the most frequent class of these rows are then returned. Finally, we calculated the accuracy and it was very high.

4.4.2 Implementation of kNN on Network Data

For implementing the kNN algorithm, we first imported the pre-processed normalized dataset. The most important part during the implementation of kNN is determining the value of 'K'. Initially, we set the value to 5 and started our experiment. We got an accuracy of 90.8 percent. However, this is not a logical approach. As a result, we decided to find the optimal value of 'k' that will give a credible accuracy. This is done using GridSearchCV. It works by training our model on a range parameters and then finds out the best result based on comparison. In our case, we provided a range of values of k neighbors from 1 to 25 and checked which value of k neighbor provided the most accurate and reliable output. It is seen that k=16 best suited our data set. We received a better accuracy when we used k=16 and it was 93.3 percent.

4.5 Decision Tree

Decision Tree is a supervised machine-learning algorithm. It can be used for both classification and regression. It works on the basis of building a tree. Decision of a decision tree is based on the principle of dividing the data into smaller parts until everything falls under the same category. In our case, the data sets are continuous. As a result, the tree itself converts them into discrete values before building up the tree. First, the entropy of target is determined. Next, the dataset is split on the

attributes and the sum of entropy for all the classes in the attributes is calculated. The resulting entropy is then subtracted from the target entropy before splitting. This is called the information gain. The attribute with the highest information gain is chosen to split the dataset. This process is repeated over and over again until we get a pure classification. The entropy is calculated using the following formula:

$$Entropy(S) = \sum_{i=1}^{c} -p(i)logp(i) \tag{4.6}$$

Here, "c" is the number of classes of an attribute. "pi" is the fraction of examples of the class "i".

$$Gain(S, A) = Entropy(S) - Entropy(A) \tag{4.7}$$

Here, S is the parent and A is the attribute that we want to split. Information gain is basically the decrease in entropy due to partitioning of data set based on an attribute. Decision tree is split based on attributes with higher information gain.

4.5.1 Implementation of Decision Tree on Application Layer Data

First, the root of the tree is built by placing the best attribute. For our dataset, it can be seen that humidity is the best attribute. The training set is broken into subsets. We repeated the process over and over again until we reach the leaf node which contains the exact classification. The attribute with highest priority is chosen as the root and the priority of attribute decreases as we go down the tree. This priority is based on the information gain which is dependent on the entropy. Initially, we got highest information gain for the attribute humidity. As a result, the algorithm start building the tree using humidity as the root node. As we go down the tree, the priority of the attributes decrease until we get the exact classification which is either normal or abnormal.

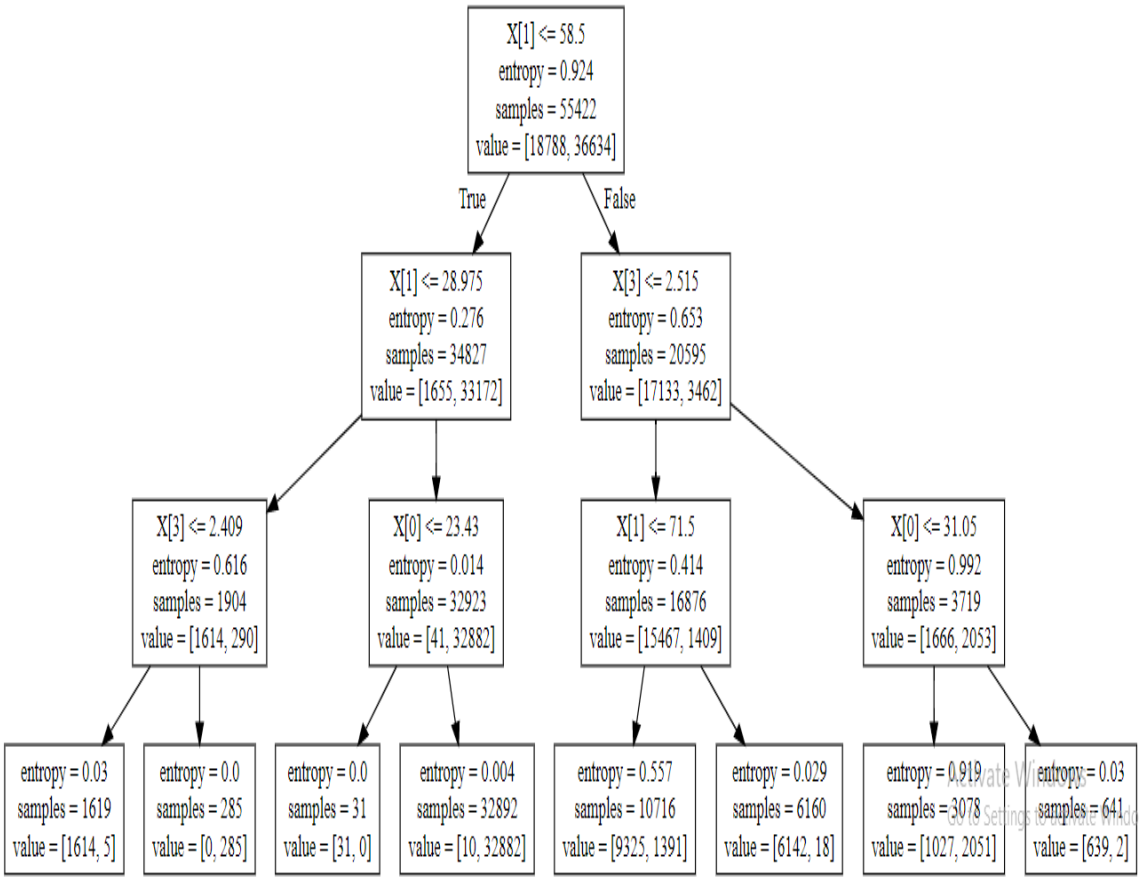

Figure 4.7: Decision Tree for Application Data

4.5.2 Implementation of Decision Tree on Network Data

The figure 4.8 represents the decision tree that is generated for our dataset. In our case, the splitting started with the attribute X [0] which is the first column, showing that it is the attribute with highest priority. The decision continued growing with the other attributes. It grew until we reached a leaf node which represented the class Normal/Non-Attack or Abnormal/Attack. The root of the tree is built by placing the best attribute.The training set is broken into subsets. We repeated the process over and over again until we reach the leaf node which contains the exact classification. The attribute with highest priority is chosen as the root and the priority of attribute decreases as we go down the tree.

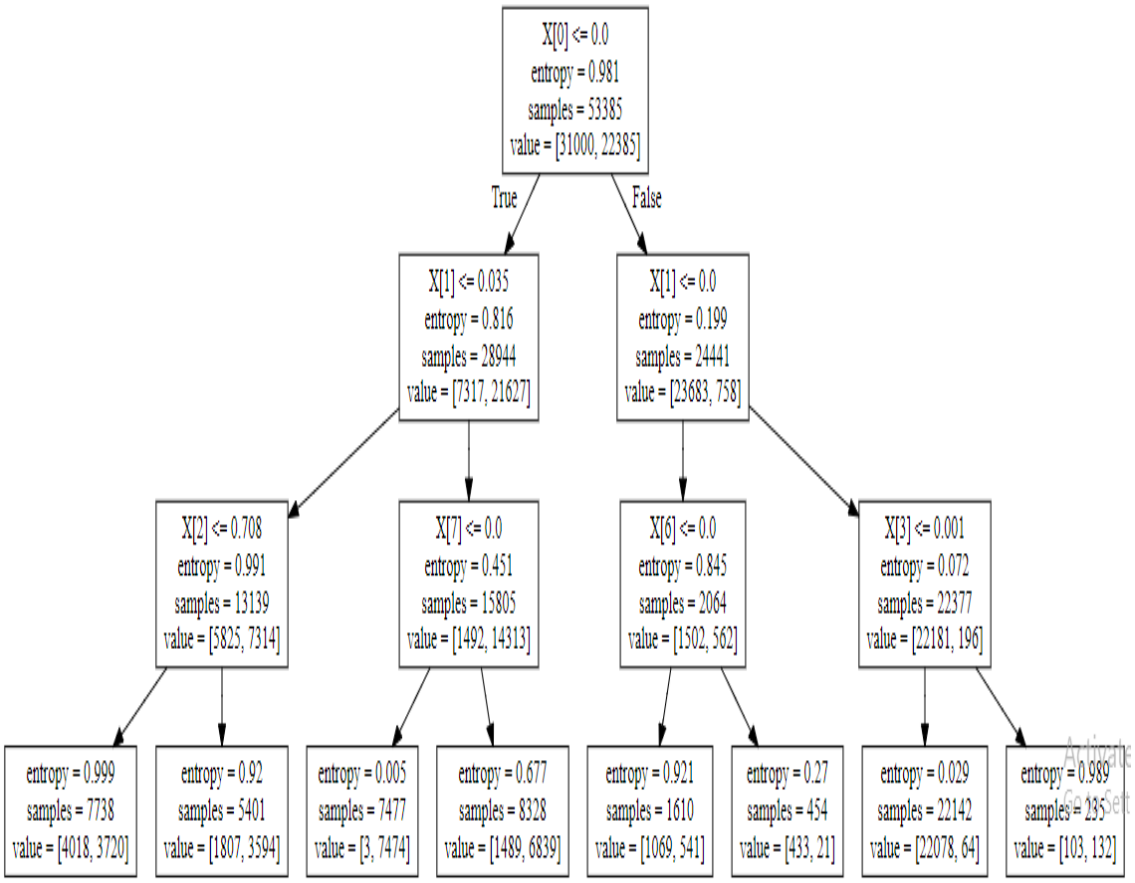

Figure 4.8: Decision Tree for Network Data

4.6 Random Forest

Random Forest is a simple, yet powerful, supervised machine-learning algorithm. It can be used for both classification and regression. In random forest, we grow multiple trees as opposed to a single tree. To classify a new object based on attributes, each tree gives a classification. The forest chooses classification which is the most supported (higher in number) over others in the forest. Additionally, in case of regression it takes average of the output by different trees. Random forest works in 4 basic steps. First of all, samples form the dataset are selected randomly. Next, decision tree is generated for each sample. From each sample, a prediction is made by each decision tree. A vote is performed between all the decisions. The prediction with the highest is vote is the outcome.

4.6.1 Implementation of Random Forest on Application Layer

For our paper, the algorithm first grows multiple trees as opposed to a single tree. To classify whether the data is normal or abnormal based on the attributes, each tree gives a classification. The forest chooses classification which is most supported (higher in number) over others in the forest. We first read the data set using the built in read method. Next,we split the data into train and test. 80 percent of the sample is train set and 20 percent of the data is test set. The data is trained with

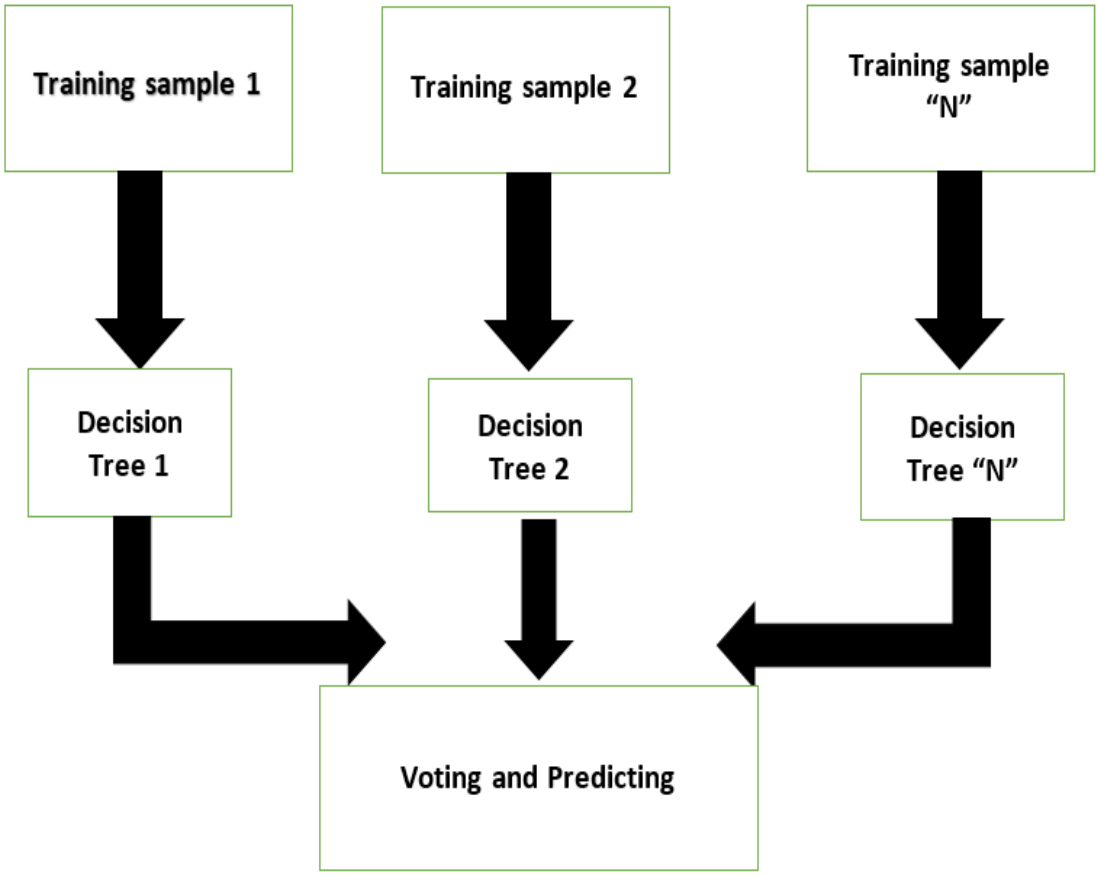

Figure 4.9: Random Forest Block Diagram

Random Forest Classifier and later the accuracy is tested. Using Random Forest, we have received the best accuracy among all the other algorithms used.

4.6.2 Implementation of Random Forest on Network Data

Just like how the algorithm grew numerous decision trees on the application layer data, here it will do the same thing. Voting occurs among the decisions made by each tree and the classification with the highest vote is chosen as the final outcome. All these give a more reliable, credible and accurate output for Random Forest. In our research, one of the challenges was finding algorithms which can handle large datasets. Random forest algorithm can handle large dataset with higher dimensions, because the higher the number of trees, the higher is the accuracy. Another important advantage of random forest algorithm is that it can handle missing values and maintains accuracy for missing data. This characteristic comes handy while handling huge dataset.

Chapter 5

Result Analysis

5.1 Machine Learning Algorithms on Application Layer

Algorithm	Accuracy	Time
SVM	72 percent	45 s
Logistic Regression	90.5 percent	0.27 s
Decision Tree	94.97 percent	0.28 s
kNN	97.7 percent	0.3 s
Random Forest	99.2 percent	3.8 s

Table 5.1: Comparison Between Algorithms on Application Layer Data

We have considered two parameters while analyzing the different machine learning algorithms on the sensor dataset.They are accuracy in terms of percentage and time in terms of seconds. The experiment is carried using Spyder (Python 3.6)in Windows environment on 4GB RAM and 2.4 GHz Intel Core i5. After carrying out several supervised algorithms, we have come to present all our findings here. We carried out Logistic Regression, SVM, k-NN, Random Forest and Decision Tree algorithms on our data-set. The results are described in the table. We have done the test procedure with 20 percent data which is equal to 13583 data.
From the table, we can see that the poorest algorithm for anomaly detection on this dataset is SVM. First of all, its accuracy is very low compared to other algorithms. Moreover, the execution time is also very large. As the complexity of the kernel increases, the algorithm begins to get slower. The other algorithms are more or less same in terms of their execution time. Random Forest has the second highest execution time due to the fact that it builds numerous tree before coming to a conclusion or decision.

5.1.1 Analysis of SVM

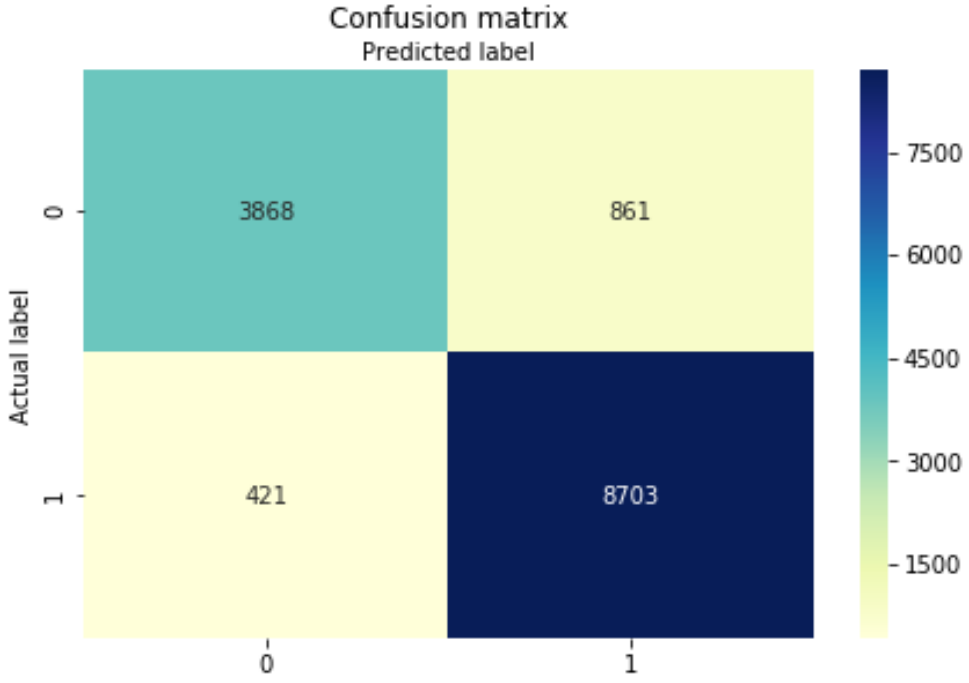

Figure 5.1: The confusion matrix for SVM

The figure 5.1 shows the confusion matrix for SVM. Out of the total data, 80 percent is used to train the data. The remaining data is used to test the data. From the figure, we can see that 8703 data are True positive. These are the data rows whose labels are labelled as normal or 1 and are also identified as normal or 1. Similarly 3868 data are true negative. These are data that are identified as abnormal or 0 and are actually labelled as 0 or abnormal. The 861 data which are actually abnormal or 0 are predicted to be normal or 1. Similarly, 421 data that are actually normal or 1 are predicted to be abnormal or 0. This gives SVM a lower accuracy rate compared to the other algorithms, which is about 72 percent.
The reason for this low accuracy is due to the fact that the hyperplane that is drawn can not separate the two classes accurately. We have used a sigmoid hyperplane to separate these data into their according classes for which the sigmoid line does not perform very well at separating the abnormal and normal data. Other kernels of Linear lines will not perform well because our data is very continuous rather than discrete. Since it is spread out everywhere randomly, we preferred to choose a sigmoid curve.

5.1.2 Analysis of Logistic Regression

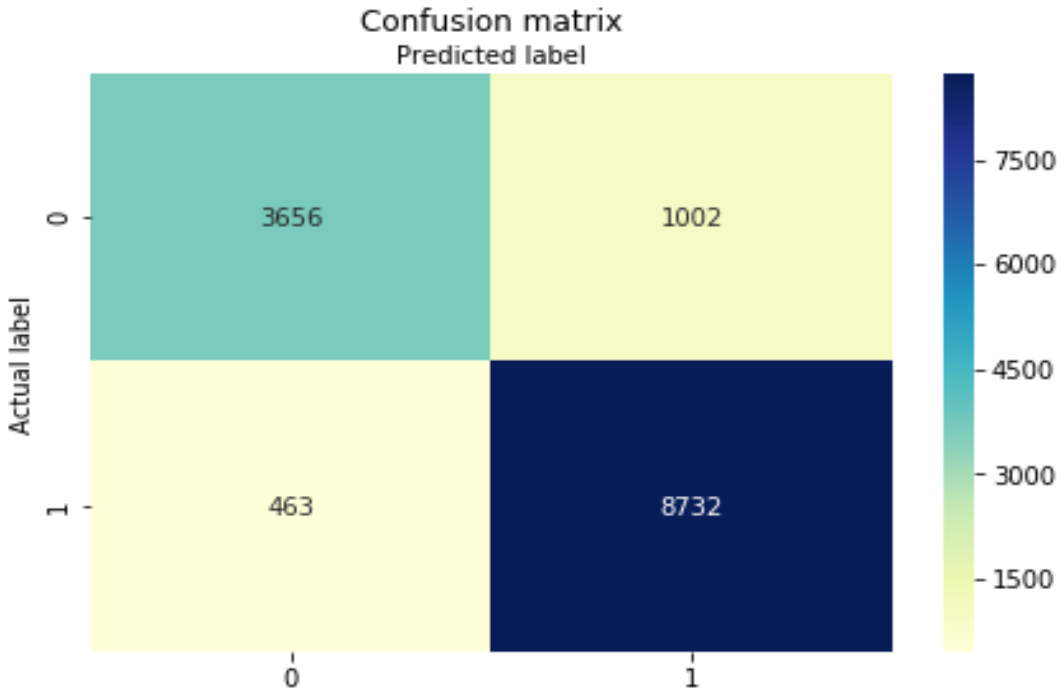

Figure 5.2: Confusion Matrix of Logistic Regression

Logistic Regression works using multiple classifying features to map into a single decision. The features are all fit into a Sigmoid function which then plots the probability of outcome belonging to each case. The figure 5.2 shows how logistic regression performs as an algorithm on our application layer data set. The total data set is divided into a 80-20 percent split, where 80 percent of the data is used to train the algorithm and 20 percent of the data is used to test the data set. 13583 sets of data is used to test the algorithm. Out of 13583 data, 8732 data is identified as True positive, which means, that they are originally labelled as Normal or 1 and are also identified as normal or 1. 3656 data are originally labelled as abnormal and 0, and are also identified as abnormal or 0. Therefore, it gives us a very high accuracy of about 90.4 percent.
Logistic regression works better with multiple features as input. Since our data set has 4 classes of features, the algorithm works well. The accuracy also depends on how many classes the logistic regression algorithm has to convert it to. In our case, the logistic function has to map between two classes. In binary logistic regression implementation, the value set as the threshold value for the boundary between two classes can be high which means that it can separate between different sets of data more accurately.On the other hand, there would have been multiple thresholds for each set of output classes if there were more than 1 group of outputs. Thus, each threshold would have been lesser in value and the difference between one threshold value for one class and another threshold value of another class would also be lower. This can eventually lead into more errors. Thus, for binary logistic regression, the chances of making mistakes is lower. Therefore, logistic regression in our case has a high value of accuracy.

5.1.3 Analysis of Decision Tree

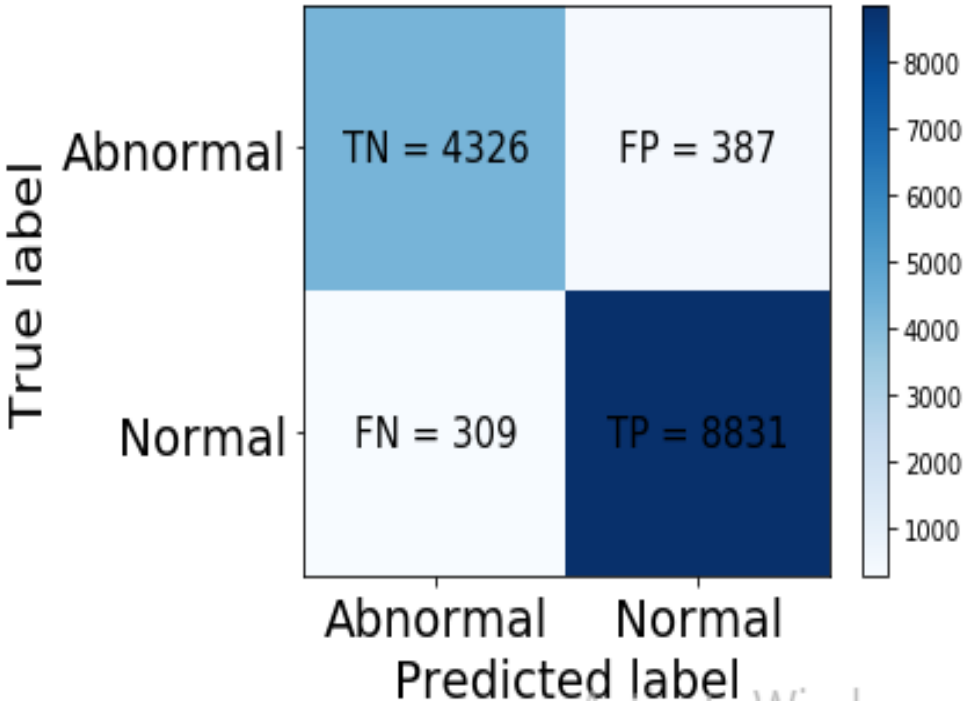

Figure 5.3: The confusion matrix for Decision Tree

Figure. 5.3 represents the confusion matrix for Decision Tree. Out of the 13583 data, 4326 data are true negative. These are data whose true labels are abnormal and they are identified correctly. 8831 data are true positive which means that they are predicted as normal and the predictions are correct. Furthermore, a total of 696 data are false positive and false negative respectively.
Decision Tree gives an accuracy of 94.97 percent. The reason for this accuracy is because decision tree assigns specific values to outcomes and decisions. Moreover, the high priority attributes are chosen for making decisions. In our case, the feature humidity has a very high priority, which is why we can see an impressive accuracy. There is a significant decrease in ambiguity and increase in reliability and accuracy.

5.1.4 Analysis of K-Nearest Neighbour

Figure. 5.4 represents the confusion matrix of kNN algorithm. From the Table 1, we can see that kNN has an impressive accuracy of 97.7 percent which is more than that of SVM, Decision Tree and Logistic Regression. When the dataset is well separable into their respective classes, kNN works very well in those situations. Figure. 5.5 shows the scatter diagram of our dataset. It depicts that the data is very well separated in their respective classes and hence the accuracy of kNN is very high compared to SVM. Moreover, the reason why kNN is having a greater accuracy than decision tree is due to the fact that the dataset is continuous rather than discrete or categorical. All the features have continuous values. kNN groups together data well when the dataset is continuous. For discrete data, decision tree generally works better.

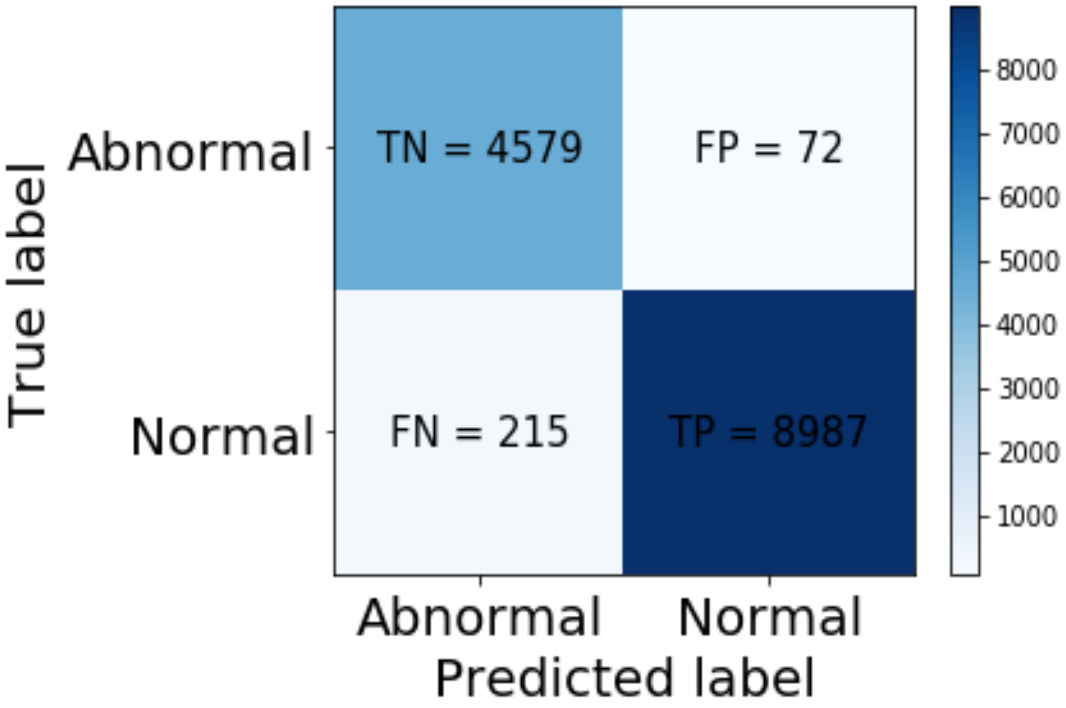

Figure 5.4: The confusion matrix for kNN

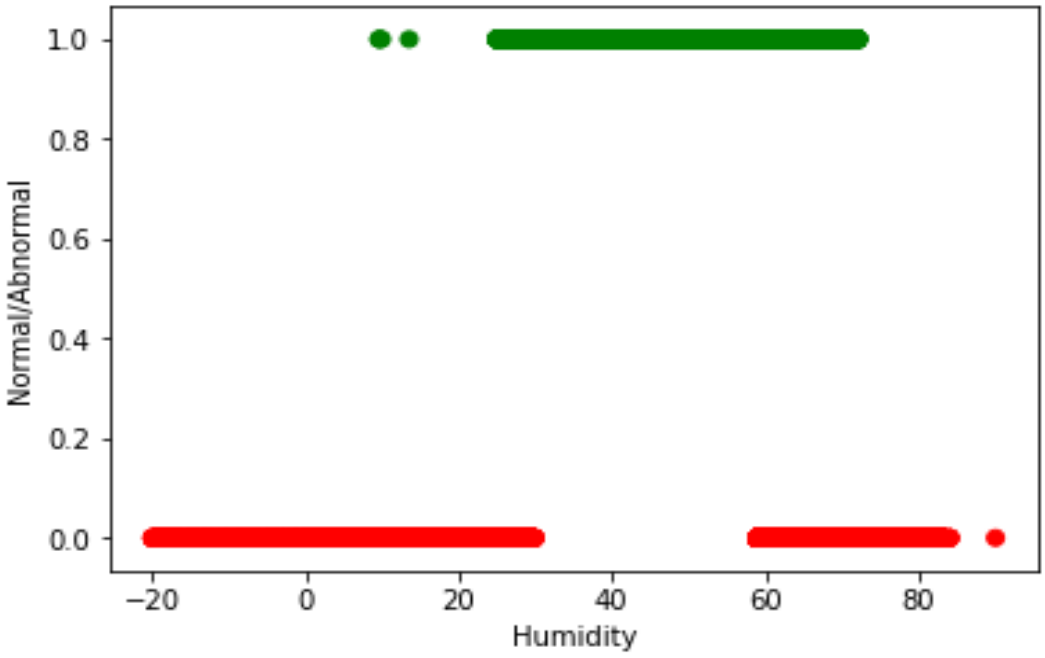

Figure 5.5: Scatter diagram

5.1.5 Analysis of Random Forest

In the case of Random Forest, we can see that we have achieved the highest accuracy of around 99 percent. There is a reason behind this. Random Forest is the

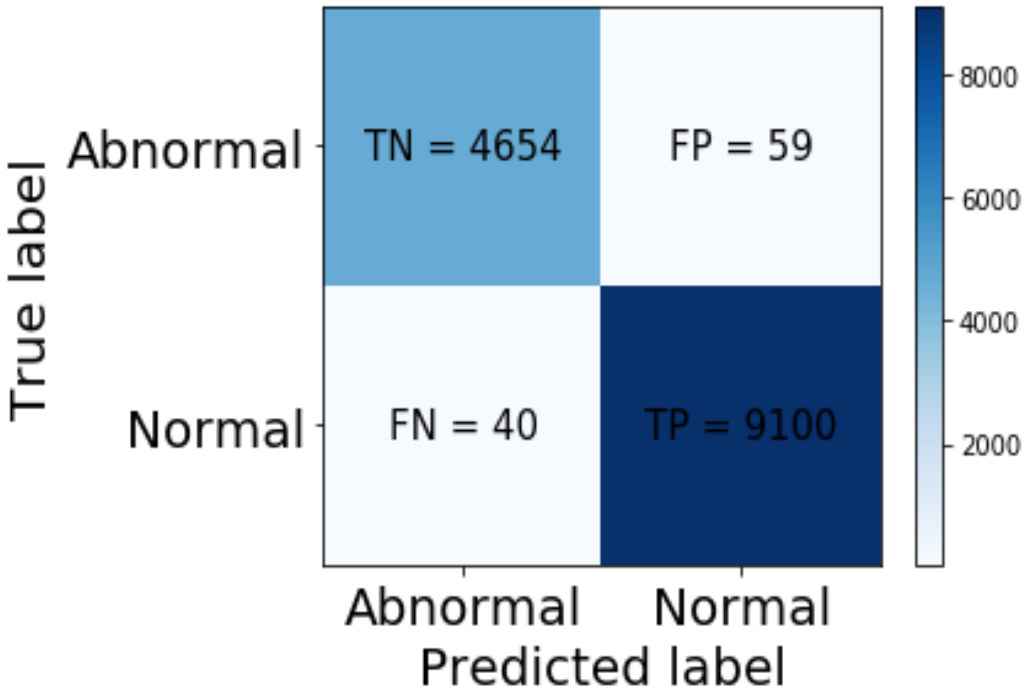

Figure 5.6: The confusion matrix for Random Forest

extension of Decision Tree. In Decision Tree, only a single tree is drawn for making a classification or decision. However, in the case of random forest, several trees are drawn. The decision from each tree is taken and the decision with the highest vote from the trees is chosen as the output and hence, the highest accuracy. Figure. 5.6 represents the confusion matrix for Random Forest.
In our research, one of the challenges was finding algorithms which can handle large data sets. We have used Random forest algorithm because it can handle large data set with higher number of dimensions, because the higher the number of trees, the higher is the accuracy. Another important reason of using random forest algorithm is that it can handle missing values and maintains accuracy for missing data. This characteristic comes handy while handling huge data set. After classifying our own application layer data set using random forest algorithm, we got 99.2 percent accuracy rate.

5.2 ML Algorithms on Network Layer

Algorithm	Accuracy	Time
SVM	93.8 percent	505 s
Logistic Regression	94.7 percent	1.32 s
Decision Tree	89.6 percent	1.86 s
kNN	93.3 percent	20.2 s
Random Forest	98.2 percent	23.5 s

Table 5.2: Comparison Between Algorithms on Network Data

We have considered two parameters while analyzing the different machine learning algorithms on the network dataset.They are accuracy in terms of percentage and time in terms of seconds. The experiment is carried using Spyder (Python 3.6) in Windows environment on 4GB RAM and 2.4 GHz Intel Core i5. Moreover, for the network data, if we see the labels, there are two labels, '0' and '1'. Unlike the application layer data, here 0 refers to Normal/Non-attack and 1 refers to Abnormal/Attack.

5.2.1 Analysis of SVM

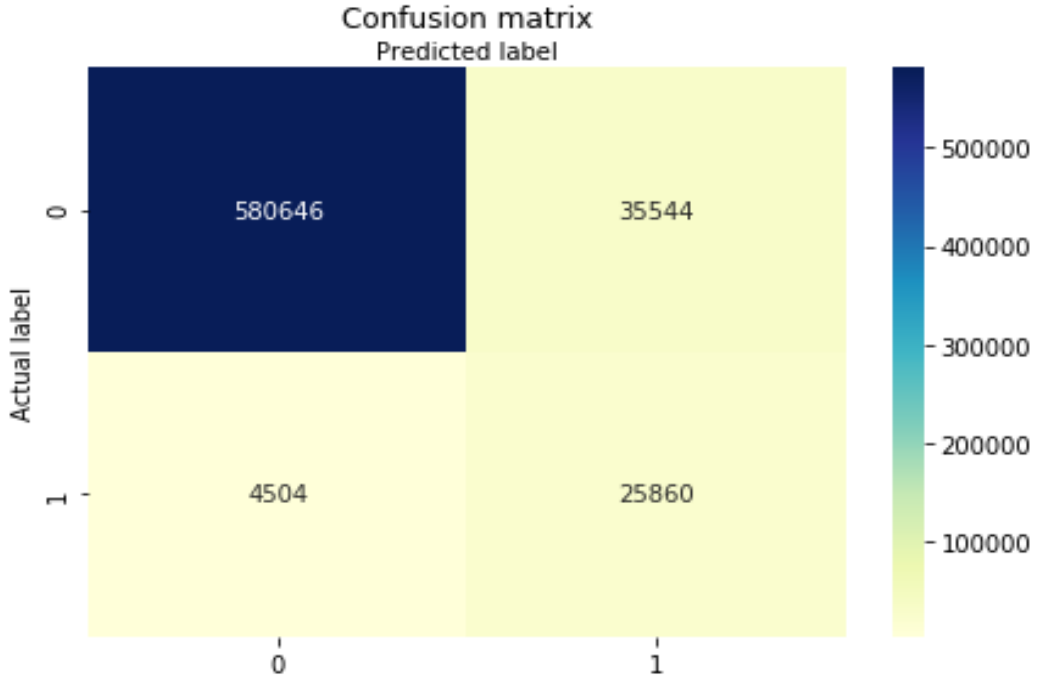

Figure 5.7: The confusion matrix for SVM

SVM tries to separate the classes of groups by drawing a hyperplane in the middle. Just how the application layer was analyzed, we have used a Sigmoid function as a hyperplane to separate the data of two classes. The SVM algorithm gives us an

accuracy of 93.8 percent while using a sigmoid kernel. This means that the data can be accurately grouped into two classes of probability. Moreover, the classes that they are grouped to, are mostly correctly defined. Out of 646554 data used to test, 580646 which are assigned as 0 are correctly identified as 0. Similarly, 25860 data are correctly identified as 1, which are originally assigned as 1. Therefore, from this high accuracy, we can say that the SVM algorithm has correctly discovered and grouped data into classes as they needed to be. However, SVM's problem is that it has a very high execution time. Even with a very high accuracy, the poor execution time is a major drawback for the algorithm. The reason why the algorithm is slow is because our dataset is huge with a lot features and dimensions. The higher the complexities, the more is the execution time. Furthermore, we have used the sigmoid kernel which increased the time complexity even further.

5.2.2 Analysis of Logistic Regression

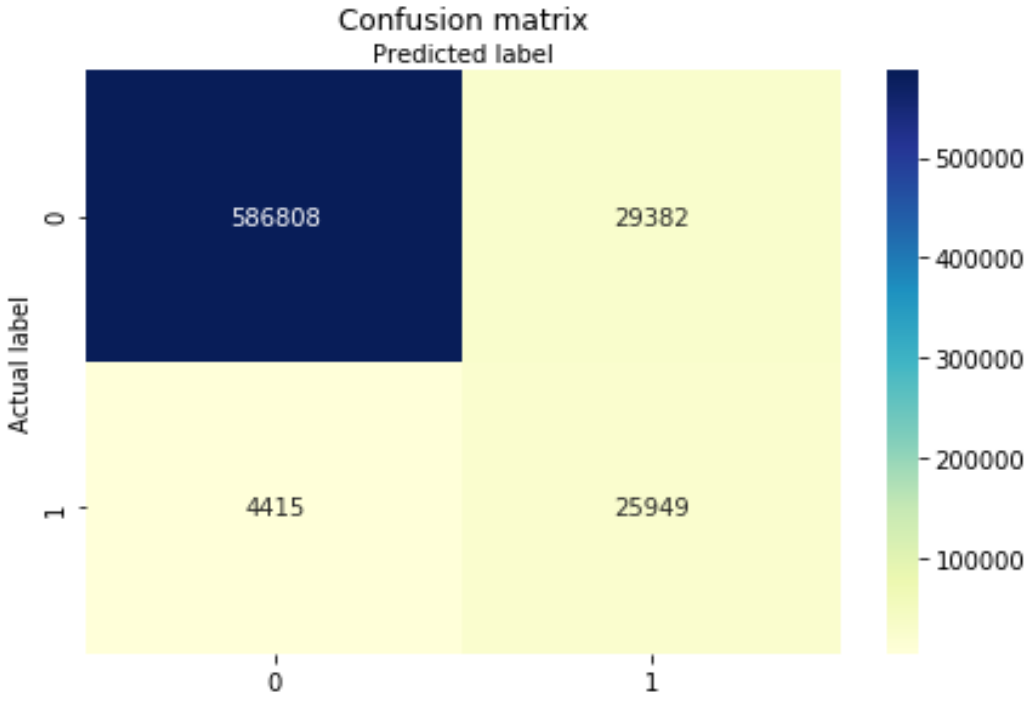

Figure 5.8: The confusion matrix for Logistic Regression

Logistic Regression takes multiple classes as input features and then models them into logistic function which is basically a sigmoid function. The more the features the better the logistic function will work. Since there are plenty of features to be trained upon and many training data, the logistic function is trained and set in such a way that it performs very well, giving us an accuracy of about 94.7 percent. From the figure above, we can see that 586808 data that are labelled as 0 are correctly identified as 0 by the algorithm. 25949 data that are labelled as 1, are correctly labelled as 1. Moreover, it takes just 1.32 seconds to run the entire algorithm. This gives us a very good accuracy compared to how fast it takes to execute. The time taken is less as it is not very tough to compute and build a logistic function from

the given data with so many input features. Thus, it is an algorithm which balances both accuracy and time in a favorable position.

5.2.3 Analysis of Decision Tree

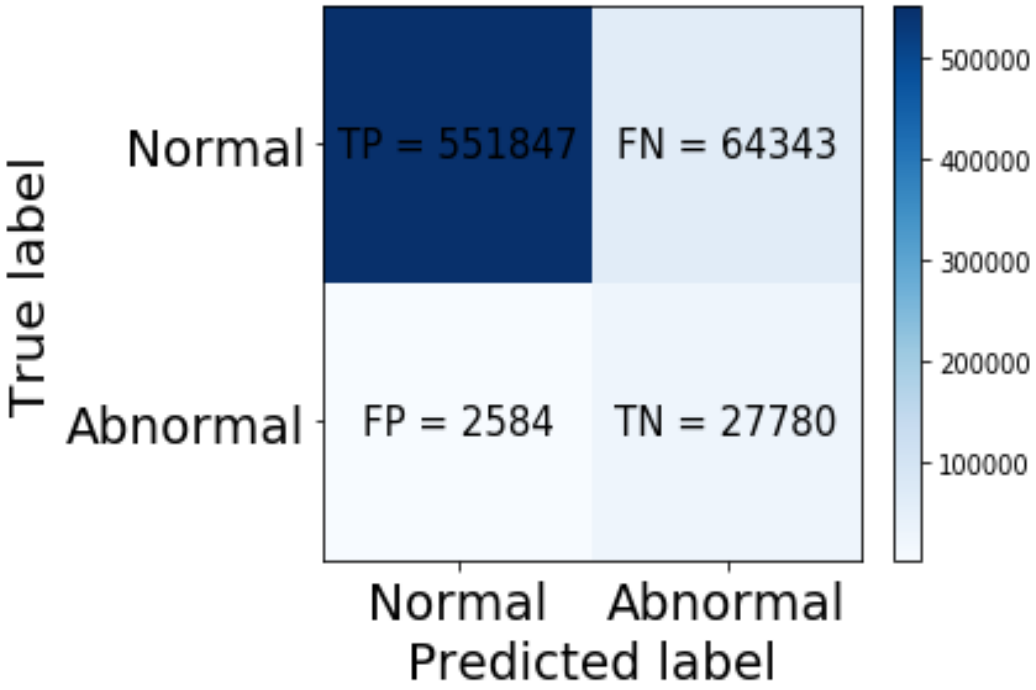

Figure 5.9: The confusion matrix for Decision Tree

Decision Tree has an accuracy of nearly 90 percent. The reason for this accuracy is because decision tree assigns specific values to outcomes and decisions. Moreover, the execution time is only 1.86 seconds, which is second only to Logistic Regression. The time of execution completely depends on how fast the tree can be grown. If the combination of dataset and algorithm requires a huge tree to be built, then it will take a lot of time. However, a smaller tree will mean that the execution time decreases by a huge amount. In our case, the dataset and number of features are huge, but the tree which is built is small, meaning that the execution time decreases by a big amount and this is what is reflected in the table. Furthermore, the algorithm is greedy as it has a tendency of lowering the cost. The confusion matrix of Decision Tree shows us that 551847 data are actually normal and predicted correctly. 27780 data are true negative data. The rest of the data in the test procedure are predicted wrongly. There is a total of 66927 wrong predicted data.

5.2.4 Analysis of kNN

kNN has an accuracy of 93.3 percent which is very good. However, when we consider the execution time, it is high. On average, the time is 20.2 seconds. It is not suitable for huge datasets as the time complexity can be expensive. kNN does not learn using functions rather it memorizes everything which is why it has big execution

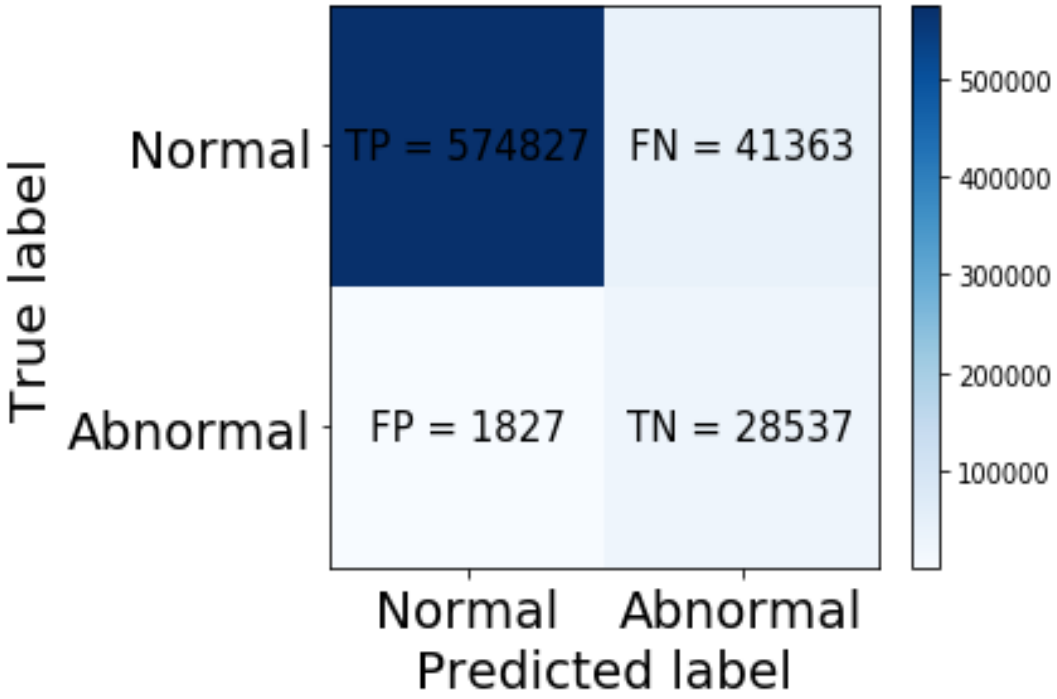

Figure 5.10: The confusion matrix for kNN

time. Figure. 5.10 represents the confusion matrix. 574827 data are true positive. 28537 data are actually abnormal and predicted correctly. The rest of the data that underwent the test are wrongly predicted.

5.2.5 Analysis of Random Forest

Although Random Forest is the best in terms of accuracy, but it has the second most poor execution time. It is 23.5 seconds. This is very poor in terms of large datasets. The reason for this huge delay is due to the fact that random forest builds numerous tree at a time. Then the decision of these trees are checked and the decision with the highest vote is selected. In the process of building these huge number of trees and then completing the voting procedure, a lot of time is wasted. This is why we can see a very large execution time. The confusion matrix shows us that out of all the test data, only a total of 11758 data are wrong predicted. They fall under the false negative and false positive classes.

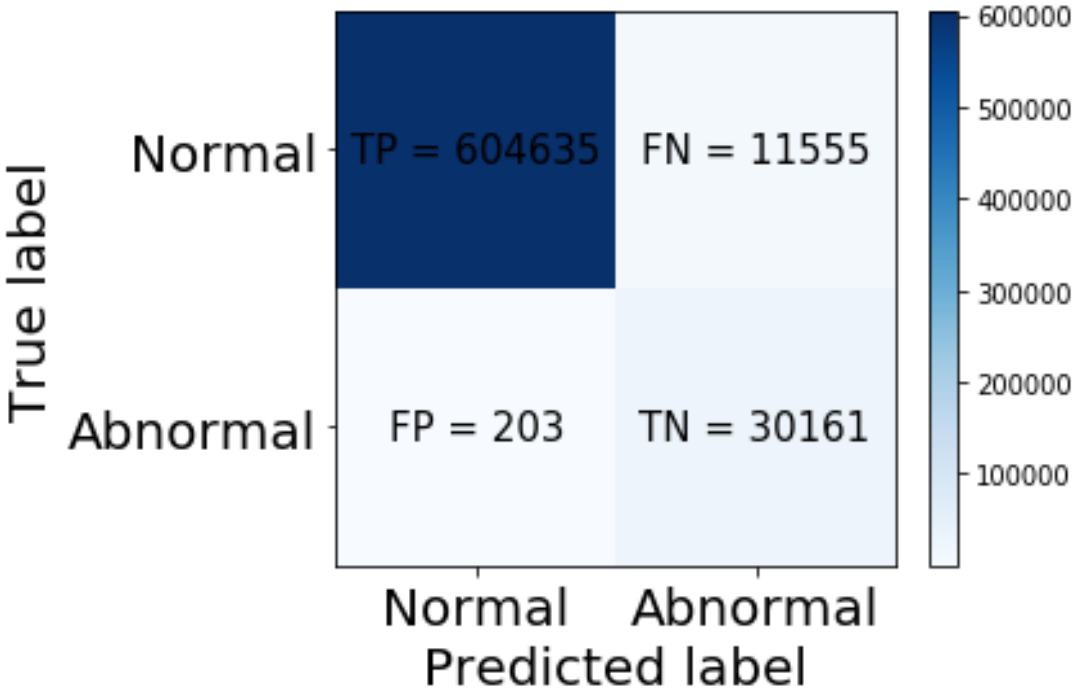

Figure 5.11: The confusion matrix for Random Forest

Chapter 6

Conclusion

6.1 Conclusion And Future Direction

Internet of things or objects is where devices which has embedded processing capabilities or even basic sensors are connected to one another through the internet which could be both wireless and wired. The fundamental feature of IoT is that these devices that are connected to one another can communicate among themselves, exchanging data over the internet. This process is also called machine-to-machine communication. With the rapid growth of technology and introduction of smart phones along with the gradual shift from IPv4 to IPv6, the number of IoT devices connected to the web is greater than the entire world's population. With IPv4, it was only possible to assign 4.2 billion addresses but with IPv6 in to action, the address space becomes quite large and for this almost all smart devices to come in the future can be connected to the internet. Now, these devices work through acquiring data from sensors and process them with an embedded processor. This can be used to build better system for us with the aid of machine learning algorithms.
Though smart devices have made our lives a lot easier and comfortable, the risk of privacy and security still pose a threat. Since internet is the space through which the data is being exchanged and due to the open accessible feature of internet, in many instances, this data can be used for criminal activity. The number of widespread malicious attacks like phishing, denial-of-service (Dos) or distributed denial-of-service (DDoS), Probe Remote to Local (R2L) has increased with the increased use of IoT systems. In order to make the system and data more secure so that it can only be accessible and used by authorized entities, logical controls are being used as software guards. Logical controls can be Intrusion detection, passwords, or access control. Among these logical controls Intrusion Detection System is implemented to fight against IoT attacks, as mentioned before. Intrusion Detection System is designed in a way that it should be able to identify that an attack has taken place in case of a security breach and should be able to respond by sending alerts to the authorized entities. IDS uses two techniques to detect faults and they are Signature based detection which works best at detecting attacks which are known and Anomaly based detection which depends on machine learning algorithms as it works best for attacks that are unknown. The technique we are working on is Anomaly based detection, where the system will train itself so that it can distinguish between stable and unstable or abnormal behavior of the system.
Since we are working with Anomaly based intrusion detection techniques, this re-

quires that the system to identify threats that are known as well as unknown. Anomaly based detection is a supervised learning technique so it needs data set to train its system. We are using choosing to construct a defensive system for both Application Layer, which is host based and Network Later which is network based for a set of chosen machine learning algorithms. Since acquiring data set of labeled data in IoT for intrusion detection is tough so for the application layer data set, which is a replication of Intel Lab Data, we made our own board with 4 sensors to collect data in our houses and CSE Research Lab of BRAC University and at our respective homes for 2 months. Each sensor board contains temperature sensor, humidity sensor, light sensor and voltage between the batteries. Due to condition of the labs being similar most of the data that we collected are similar. As for the network layer data set, we are using UNSW-NB15 data set that contains categorical features. These features are identifying attributes between hosts, attributes representing protocol connections, attributes of TCP/IP, attributes of packet's start/end and round-trip time, general features like protection of the service of protocols and connection features and lastly labeled feature. On these data sets the machine learning algorithms are then implemented.

Support Vector Machine algorithm is a classification algorithm that classifies data into groups that augments the utility for researchers. Using SVM we can split data according to its classification. SVM cannot work well on data that is continuous too, because if the regularization parameters are set high, it shall still form a very thin hyper plane to set the data even when it is continuous. Since we have lots of data and it is arranged in a continuous manner after implementing SVM on the data set, it shows a run-time accuracy of 72 percent in 45s. Similarly, SVM is tested on the network layer data where it classifies by creating a hyper plane to separate attacking and non-attacking classes. Since the computational power required on the actual data set was high, we ran it on a pre-processed data set and where it concluded with an accuracy of 93.8 percent but took 505s to finish.

Logistic Regression was also used and it works by using multi-variable analysis as it predicts a single outcome from multiple variable features. Our application layer data set has 4 features (Temperature, Humidity, Light Intensity and Voltage) and has two classes. The model is trained by using the logistic function which sets a threshold value in probability that finally determines the two classes. This algorithm shows an accuracy of 90.5 percent in 0.27s. Logistic regression is implemented on the network layer and input features that were trained was IP, protocols, port numbers etc. After the algorithm was tested the result column predicts whether the packet is an attacking packet or not and shows 94.7 percent accuracy in just 1.32s.

Another easy to implement algorithm was tested on both application and network layer is kNN. K-Nearest Neighbour is a supervised learning algorithm which is useful in predicting and classifying and depends on labelled data to work. It uses the labeled data to learn a function to give an appropriate output. It works well for all parameters and has low computational time. It assumes similar things in close proximity belong to similar class. Determining the value of k is a difficult job and plays a vital role in the accuracy of the output. Though higher value means less variance and exclusion of outlier value but the downside is it ignore smaller patterns where useful data might reside. For our case optimal value of k was found to be 2 for application layer data and for network layer data we used GridSearchCV to find out the best suitable value of k and it was 16. For application layer the accuracy

came to be 97.7 percent in 0.3s and for network layer it was 93.3 in 20.2 percent.
Decision tree algorithm is another supervised machine learning algorithm where it divides data into smaller parts until all parts fall under similar class. For application layer the best attribute was humidity and it was selected as root of the tree. The tree is made with the root having the highest priority and priority decreases as we go down. This priority is based on the information gain which is dependent on entropy. The result of decision tree for application layer is 94.97 percent with a run-time of 0.28s. As for network layer the first column is where the splitting began with the attribute X [0] and this was the root of the tree and progressed as explained before by splitting until it reached the class Normal or Abnormal. The accuracy of decision tree in the network layer came out to be 89.6 percent in 1.86s.
The last algorithm that is used on both network and application layer is Random Forest. It is a powerful yet simple supervised ML algorithm that can applied for both classifying and regression problems. Unlike decision tree where a single tree is made from the entire data set, this algorithm makes multiple trees. The algorithm samples out data randomly and generates a decision tree for each sample. A vote is performed among the decisions and the best or highest vote is the result. For our application layer data set we split the data into 80 percent train and 20 percent test data from the entire sample. Then running the algorithm the accuracy was 99.2 percent in 3.8s. Just like the application layer it works exactly in the same way for network layer and since network layer data set is quite large random forest is perfect for handling this sort of data set. More the number of trees higher the accuracy. It can handle not only large data set but also missing values and maintains its accuracy. Application of Random Forest in our network layer data set brought an accuracy of 98.2 percent in 23.5s.
Overall, this research paper describes the analysis of five different machine learning algorithms on the collected sensor data and existing UNSW NB15 network data. In our research, we have implemented machine learning algorithms to detect anomalies. There were previous works on intrusion detection using machine learning algorithms. However, most of the works were on the network data [18]. We have done our research on both network and application layer data. Moreover, the previous works on the application layer data in the paper titled "A Model for Anomalies Detection in Internet of Things (IoT) Using Inverse Weight Clustering and Decision Tree" [21] used only a single machine learning algorithm whereas we are using five machine learning algorithms and comparing the accuracy. Furthermore, paper [30] worked on application layer data. However, the data was collected from Kaggle and it was synthetic data set. Our data is non synthetic and made from real life setup. Another important addition to our research is the analysis of the time taken by each algorithm to execute itself.
For the application layer data set and running environment, it can be concluded that Random Forest is the best algorithm in terms of accuracy and Logistic Regression is the fastest algorithm. Moreover, SVM's run time and accuracy is very poor in terms of anomaly detection. Hence, it can be excluded while detecting intrusions. Furthermore, for the network layer, logistic regression has a very decent accuracy with least execution time.
Although we have achieved very good accuracy with the five algorithms, there are other IoT data sets with more features and different complexities where these algorithms might not detect anomalies with great precision. On those data sets, other

machine learning algorithms or intrusion detection techniques might work better. Lastly, the anomaly detection is done on the end devices. We can carry out this detection on the cloud or server in the near future.

Bibliography

[1] Wigmore, *Internet of things (iot)*. [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_things.

[2] J. Morgan, *A simple explanation of 'the internet of things'*. [Online]. Available: https://www.forbes.com/sites/jacobmorgan/2014/05/13/simple-explanation-internet-things-that-anyone-can-understand/#feda0831d091.

[3] B. Johnson, *How the internet of things works*. [Online]. Available: https://computer.howstuffworks.com/internet-of-things.htm.

[4] R. van der Meulen, *Gartner says 8.4 billion connected things will be in use in 2017, up 31 percent from 2016*. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016.

[5] Ranger, *What is the iot? everything you need to know about the internet of things right now*. [Online]. Available: https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/.

[6] J. J and Dr.B.MUTHUKUMAR, "Intrusion detection system (ids): Anomaly detection using outlier detection approach", *International Conference on Intelligent Computing, Communication and Convergence (ICCC2015)*, vol. 48, pp. 338–346, 2015.

[7] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review", *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994–12 000, 2009.

[8] L. Portnoy, "Intrusion detection with unlabeled data using clustering", *Columbia University*, 2010.

[9] S. S. Tanpure, J. Jagtap, and G. D. Patel, "Intrusion detection system in data mining using hybrid approach", *International Journal of Computer Applications*, pp. 18–21, 2016, ISSN: 0975 – 8887.

[10] P. Louridas and C. Ebert, "Machine learning", *IEEE Software*, vol. 33, no. 5, pp. 110–115, 2016.

[11] E. Alpaydın, "Introduction to machine learning", *MIT Press*, p. 9, 2010.

[12] M. Abomhara and G. M. Køien, "Cyber security and the internet of things:vulnerabilities, threats, intrudersand attacks", *Journal of Cyber Security*, vol. 4, pp. 65–88, 2015.

[13] P. Middleton, J. Tully, and P. Kjeldsen, *The internet of things, worldwide, 2013*. [Online]. Available: https://www.gartner.com/en/documents/2625419/forecast-the-internet-of-things-worldwide-2013.

[14] D. Alexander, A. Finch, D. Sutton, A. Taylor, and A. Taylor, *Information Security Management Principles (2nd Eds.)* Swindon, United Kingdom: BCS, The Chartered Institute for IT, 2013.

[15] A. Jain, B. Verma, and J. L. Rana, "Anomaly intrusion detection techniques: A brief review", *International Journal of Scientific & Engineering Research*, vol. 5, no. 7, pp. 1372–1383, 2014.

[16] D. K. Prabha and S. S. Sree, "A survey on ips methods and techniques", *International Journal of Computer Science Issues*, vol. 13, no. 2, pp. 38–43, 2016.

[17] *Internet of things - number of connected devices worldwide 2015-2025.* [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[18] V. Timcenko and S. Gajin, "Machine learning based network anomaly detection for iot environments", Mar. 2018.

[19] R. Fu, K. Zheng, D. Zhang, and Y. Yang, "An intrusion detection scheme based on anomaly mining in internet of things", pp. 315–320, 2011.

[20] S. Mukherjee and D. Sharma, "Intrusion detection using naive bayes classifier with feature reduction", *Procedia Technology*, vol. 4, pp. 119–128, 2012.

[21] A. Alghuried, "A model for anomalies detection in internet of things (iot) using inverse weight clustering and decision tree", *Semantic Scholar*, 2017. DOI: 10.21427/D7WK7S.

[22] S. Madden, *Intel lab data.* [Online]. Available: http://db.csail.mit.edu/labdata/labdata.html.

[23] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set", *Information Security Journal: A Global Perspective*, vol. 25, no. 1–3, pp. 18–31, 2016.

[24] V. Golmah, "An efficient hybrid intrusion detection system based on c5.0 and svm", *International Journal of Database Theory and Application*, vol. 7, no. 2, pp. 59–70, 2014.

[25] S. A. Hajare, "Detection of network attacks using big data analysis", *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 4, no. 5, pp. 86–88, 2016.

[26] N. Moustafa and J. Slay, "Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)", *2015 Military Communications and Information Systems Conference (MilCIS)*, pp. 1–6, 2015.

[27] T. Evgeniou and M. Pontil, "Workshop on support vector machines: Theory and applications", *Support Vector Machines: Theory and Applications*, p. 1, 2001.

[28] A. Zisserman, *The svm classifier.* [Online]. Available: http://www.robots.ox.ac.uk/~az/lectures/ml/lect2.pdf.

[29] H.-A. Park, "An introduction to logistic regression: From basic concepts to interpretation with particular attention to nursing domain", *Journal of Korean Academy of Nursing*, vol. 43, no. 2, p. 1, 2013.

[30] M. Hasan, M. M. Islam, M. I. I. Zarif, and M. M. A. Hashem, "Attack and anomaly detection in iot sensors in iot sites using machine learning approaches", *Internet of Things*, vol. 7, pp. 1–14, 2019.