

Intrusion Detection System for Various IoT Attacks on MQTT protocol

by

Farhan
17201109

Mueth Ul Ehsan Ananno
17301142

Tanzil Talat Anonto
17201078

Sazzad Alam Tomal
16301124

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
January 2022

© 2022. Brac University
All rights reserved.

Declaration

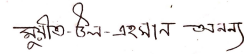
It is hereby declared that

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:



Farhan
17201109



Mueth Ul Ehsan Ananno
17301142



Tanzil Talat Anonto
17201078



Sazzad Alam Tomal
16301124

Approval

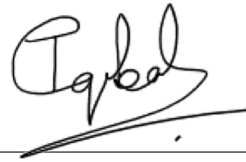
The thesis titled “Intrusion Detection System for Various MQTT and IoT Attacks” submitted by

1. Farhan (17201109)
2. Mueth Ul Ehsan Ananno (17301142)
3. Tanzil Talat Anonto (17201078)
4. Sazzad Alam Tomal (16301124)

Of Fall, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 20, 2022.

Examining Committee:

Supervisor:
(Member)



Dr. Muhammad Iqbal Hossain
Assistant Professor
Department of Computer Science and Engineering
Brac University

Co-supervisor:
(Member)

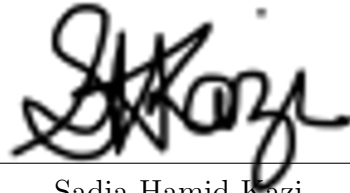


Rafeed Rahman
Lecturer
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)



Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

In recent times the number of IoT devices have increased massively but the security of these devices has not been upgraded. Due to this huge number, achieving high security is a big challenge. The use of a classic centralized design in IoT systems can compromise security. Gadgets in the Internet of Things have sensors that collect information about their surroundings which is required for the correct operation of IoT setups. If not secured, or if stolen or otherwise hacked, this data could have a number of undesirable consequences. The Internet of Things devices use a variety of communication protocols. The MQTT protocol is one of the protocols that has previously been standardized by ISO. Because of its low memory use and low bandwidth requirements, this protocol is popular among IoT developers. Occasionally, IoT devices communicate sensitive data that is only meant for authorized devices. The protocol, on the other hand, merely authenticates the security mechanism, which does not encrypt the data being carried by default which is a major concern in terms of data privacy, authentication, and data integrity. This paper highlights numerous reasons why many IoT systems do not have proper security mechanisms in place. Following that, it also shows and explains how we may easily attack this protocol in a variety of assault scenarios. Finally, after learning about the flaws in We can increase our security by looking into this protocol. awareness, particularly of the MQTT protocol, and then implementation.

Keywords: MQTT; attacks; security; centralized

Acknowledgement

Firstly I would like to thank the almighty for all his blessing and guidance for which we were able to accomplish our research. Secondly, we would like to thank our supervisor, Dr. Muhammad Iqbal Hossain, for his tremendous mentorship and support towards our work. His massive knowledge in this research area helped us throughout our study. We would also like to pass our warmth gratefulness to Rafeed Rahman for his humble guidance and motivation throughout this research. Lastly, this would not be possible without the love and support of our family and friends.

Table of Contents

Declaration	i
Approval	ii
Abstract	iv
Dedication	v
Acknowledgment	v
Table of Contents	vi
List of Figures	viii
List of Tables	ix
Nomenclature	ix
1 Introduction	1
1.1 Research Purpose	2
1.2 Research Objective	2
1.3 Thesis Structure	3
2 Literature Review	4
2.1 MQTT	4
2.2 Related Works	4
3 Methodology	7
3.1 Dataset	8
3.2 Attacks	9
3.3 Data Preprocessing	9
3.4 Training the train data	10
3.5 Evaluate the Performances	11
4 Algorithms	12
4.1 Decision Tree Classifier	12
4.2 Random Forest Classifier	13
4.3 Support Vector Machines (SVM)	14
4.4 Extra tree Classifier	15
4.5 Logistic Regression	16

4.6	Boosting	16
4.6.1	AdaBoost	17
4.6.2	Gradient Boosting	17
4.7	Stacking	18
4.7.1	Linear SVM with Gradient Boosting	18
4.7.2	Logistic Regression with Gradient Boosting	19
5	Result & Analysis	20
5.1	Confusion Matrix	20
5.2	Classification Report	20
5.3	Analysis	21
6	Conclusion	31
	Bibliography	33

List of Figures

3.1	Flow chart of determining the respective algorithm's accuracy	7
3.2	Testbed scenario considered to create the MQTTset dataset	8
4.1	Decision Tree	13
4.2	Random Forest-01	14
4.3	SVM	15
4.4	Boosting	16
4.5	AdaBoosting	17
4.6	Stacking	18
4.7	Linear SVM with Gradient Boosting	19
4.8	Logistic Regression with Gradient Boosting	19
5.1	Confusion Matrix of Decision Tree	22
5.2	ROC curve of Decision Tree	22
5.3	Confusion Matrix of Random Forest	23
5.4	ROC curve of Random Forest	23
5.5	Confusion Matrix of Linear SVM	24
5.6	ROC curve of Linear SVMe	24
5.7	Confusion Matrix of Extra tree Classifier	25
5.8	ROC curve of Extra tree Classifier	25
5.9	Confusion Matrix of Logistic Regression	26
5.10	ROC curve of Logistic Regression	26
5.11	Confusion Matrix of AdaBoost	27
5.12	ROC curve of AdaBoost	27
5.13	Confusion Matrix of Gradient Boost	28
5.14	ROC curve of Gradient Boost	28
5.15	Confusion Matrix of Stacking Linear SVM with gradient boosting	29
5.16	ROC curve of Stacking Linear SVM with gradient boosting	29
5.17	Confusion Matrix of Stacking Logistic Regression with Gradient Boosting	30
5.18	ROC curve of Stacking Logistic Regression with Gradient Boosting	30

List of Tables

2.1	Rejected Data-sets and the reason	6
5.1	Precision and F1-Score of Applied Algorithms	21

Chapter 1

Introduction

MQTT is an OASIS-recognized Internet of Things communications protocol. IBM released MQTT in 1999 as a standardized publish/subscribe Push protocol. MQTT was designed to convey data accurately even when there was a considerable network delay and a low bandwidth network [1]. It is an application layer protocol. The application layer acts as a channel between the network and the end nodes (IoT devices). The application layer is often implemented by the browser on desktops, laptops, and mobile devices. MQTT is appropriate for devices with minimal resources and electricity. Despite restricted network capacity, the protocol is designed to convey tiny code footprints in a reliable and real-time way. MQTT provides architectural benefits over typical computer networking application layer protocols when resource restricted IoT devices must communicate bi-directionally over limited bandwidth. It's a simple publish/subscribe messaging protocol that's ideal for connecting faraway devices with simple coding and network traffic. Device-to-cloud and cloud-to-device communication is possible using MQTT which makes it simple to send messages to large groups of objects. Furthermore, it is lightweight and efficient which is why it can be scaled to connect millions of devices. Publishers publishing messages and users subscribing to topics are frequently referred to as a Publish/Subscribe model in the MQTT protocol.[2] The usual MQTT architecture can be broken down into two parts, Clients and Brokers. The Client, which can be either a Publisher or a Subscriber, always establishes a network connection with the Server which is the broker. [3] The broker is in charge of receiving all messages, screening them, determining who has subscribed to each message, and then sending the message to those who have subscribed. The evolution of the MQTT protocol, and particularly the MQTT brokers that use it, reflects new developments in M2M and IoT systems, and they must adapt to these developments as well.[4] MQTT protocol is best suited for environments such as embedded devices with limited processing capacity or devices connected to an unreliable network.[2] “ It's a binary protocol with a 2-byte fixed header and small message payloads of up to 256 MB ”. [5] Many big corporations, particularly in the automotive, industry 4.0, transportation, and entertainment industries, use MQTT. MQTT does not allow encryption by default, regardless of whether the broker system uses authentication or not. It is also unconcerned about data integrity and port obfuscation. Because of the growing popularity of IoT devices, their security has become a serious problem. There are a variety of assaults that we are aware of, one of which being the MQTT attack. Currently, billions of smart devices are available in the Internet of Things ecosystem,

and while the devices connect with one another, they employ many protocols, each of which differs in terms of popularity, security, efficiency, and other factors. Different types of procedures are used to check MQTT's popularity and how it works, such as comparing it to an IoT reference model, assessing MQTT's basics, comparing it to other protocols, advantages/disadvantages of MQTT, and growth in MQTT research.

1.1 Research Purpose

The Internet of Things (IoT) is a term used to describe the interconnection of extremely diverse networked entities and networks that use a variety of communication patterns (Heer et al., 2011).[6] There is an enormous increase in the number of IoT devices in recent times. The homogeneity of these devices, the ability to auto connect, and the deployment of devices in unsecured networks all add to the security risk. Because most IoT devices are low-power and resource-constrained, MQTT quickly became the industry standard for IoT communications. The majority of IoT devices have CPU and bandwidth limitations. The MQTT connection uses about 86 bytes of data (CONN and Connack). Instead of connecting to a server, client devices and applications publish and subscribe to topics which are managed by a broker. The MQTT protocol's central component, the broker, is responsible for receiving, filtering, and determining where messages should be transmitted. The main purpose of our research is to maximize the security of IoT devices which use MQTT protocol. MQTT has many drawbacks when it comes to security. For instance, Authentication is provided via the MQTT protocol via username and password, with some broker implementations adding additional mechanisms. As a result, MQTT security is dependent on the use case and broker chosen. TLS is used by most brokers to offer security; however, it has a considerable impact on performance, particularly during the handshake.[7] In recent times the usage of IoT devices has immensely increased which has raised various security concerns. Identity theft, data integrity, D2D connectivity, and other aspects of device security are not carefully addressed. Furthermore, the majority of the privacy and the security features they present remain in their infancy. [8] It should be noted that security issues for MQTT and MQTT-SN are the responsibility of the user.

1.2 Research Objective

Our main research objective is to understand the architecture of IoT devices and focus on optimizing the security against attacks like MQTT with the limited computation of IoT devices. Furthermore, we will test various existing machine learning algorithms that are used by other related works, our goal is to test various machine learning techniques to enhance security with minimal resources effectively. The objectives of this research are as follows:

- To intensively understand how exactly the MQTT protocol works.
- To analyze more on the security of MQTT attacks.
- To develop a model against these attacks using machine learning algorithms.

- To develop an efficient model.
- To offer suggestions on upgrading the model.

1.3 Thesis Structure

In this research paper we added 3 parts in chapter 1. Those are introduction, purpose of research, objective of research and thesis structure. The 2nd chapter describes what MQTT is and some works done by others on this topic. Chapter 3 contains our proposed work flow and data set's description including data preprocessing. The name of chapter 4 is Algorithms where we described 9 algorithms because these are the algorithms that we used on our proposed model. Lastly chapter 5 contains the detailed analysis of the result obtained from our proposed model.

Chapter 2

Literature Review

2.1 MQTT

Message Queuing Telemetry Transport or MQTT is a very lightweight protocol which was originally designed for transferring messages, following the publish and subscribe method. As most of the IoT devices are lower powered and have minimum resources, MQTT quickly became standard for IoT messaging. Most IoT devices come with CPU and bandwidth constraints. MQTT connection approximately uses 86 bytes of data (CONN and Connack). The devices connected in the protocol, that are clients and brokers, communicate with each other. The broker handles the data transmission between clients. Instead of connecting with a server, client devices and applications publish and subscribe to topics handled by a broker. Broker is the center part of the MQTT protocol which is responsible for receiving, filtering and determining where the messages need to be sent. A client is anyone who can connect with the broker and can have a m2m (machine to machine) connection. So, every MQTT client can receive and send data from and to the broker if they are designed to do that. But in the original MQTT protocol, there is not much periodization of security. By default, MQTT doesn't support any encryption whether the broker's system uses authentication or not. It also doesn't concern data integrity and port obscurity. Due to the increased usage of IoT devices, the security of these devices has become a major concern. As we know, machine learning comes in different types like supervised and unsupervised. In case of supervised machine learning, we split a particular dataset into two parts. One is training and another one is a test. We use labeled datasets to train an algorithm. Basically, an algorithm experience from the labeled dataset and achieve expertise. Any statistical machine learning framework contains a domain set which is the set of objects we may wish to label, number of features and a label set containing possible number of labels, from all this we get a training set based on whom we will train a specific algorithm which outputs a prediction rule. We apply this prediction rule to the test set and determine the efficiency of that algorithm.

2.2 Related Works

The research paper [9] uses algorithms like NVM, Naive Bayes, Ada Boost etc to detect unusual data flow in an IoT test bed environment. The results from both of these researches were mostly accurate as the datasets were very small or custom

made in a specific IoT environment. The centralized architecture of IoT and IoT devices are a major cause of security vulnerabilities. In a survey conducted [10], they discussed the use of blockchain algorithms to minimize the causes of these vulnerabilities. There are a lot of security vulnerabilities of various IoT and industrial IoT devices. The problems caused by the centralized architecture of IoT and IoT devices can be solved using blockchains. Furthermore, various attacks on IoT devices such as physical, software, network and data attacks are observed because of its vulnerabilities. [3] The architecture of MQTT, various domains where MQTT is used, different brokers of MQTT made it very important in the IoT sector, though current issues in MQTT are alarming if not solved soon.[11] MQTT is also very efficient as it is 6 times faster than HTTP which makes it more acceptable to use in smaller data transferring devices like IoT.[3] Because of its simple architecture it has a lot of drawbacks which still needs to be solved [11]

As now-a-days, billions of smart devices are present in the Internet of things environment and as the devices communicate with each other, they use different kinds of protocol to communicate, each is different from other in terms of popularity, security, efficiency etc. To verify popularity and how MQTT works, different kind of procedure is taken like comparing it with IoT reference model, analyzing fundamentals of MQTT, comparing it with other protocols, advantages/disadvantages of MQTT and growth in MQTT research.[10]

ARTEMIS is another anomaly-based intrusion detection system using machine learning algorithms that creates alerts whenever an intruder is detected. This system uses six Different kind of algorithms that are Autoencoder, Single-Objective Generative Adversarial Active Learning Algorithm, Random Forest Classifier and Isolation.[11] As mentioned how the safety precaution has become a censorious snag with increasing usage of IoT devices, researchers have come up with datasets that are not only legitimate but also attack traffic on MQTT protocol and on cyber-attacks against the MQTT network. Methodologies like Machine Learning Algorithm, Artificial Intelligence Algorithm and naïve bayes algorithm were implemented. All the algorithms used have an accuracy above 85% except naïve bayes algorithm, which approximately has above 60%. [12]

One of the major beneficial facts that IoT has brought is by combining the plenty of Weather reporting devices into one particular place which is perhaps much more user friendly and modern. A survey on this aspect discusses how Arduino Uno Card is used for data collection and cloud server for future redemption. Moreover, throughout the process data security was kept secured through encryption and decryption process where Homomorphic encryption technique was used for this. [13]

Table 2.1: Rejected Data-sets and the reason

Data Set Names	Reasons
N-BaIoT	Only for Wi-Fi network
UNSW-NB15	Not for IoT devices
NIMS	Not for IoT devices
MedBIoT	Not applicable for attacks
KDDCUP99	Not for IoT devices
NLS-KDD	Not for IoT devices
Custom datasets	No raw traffic found

Chapter 3

Methodology

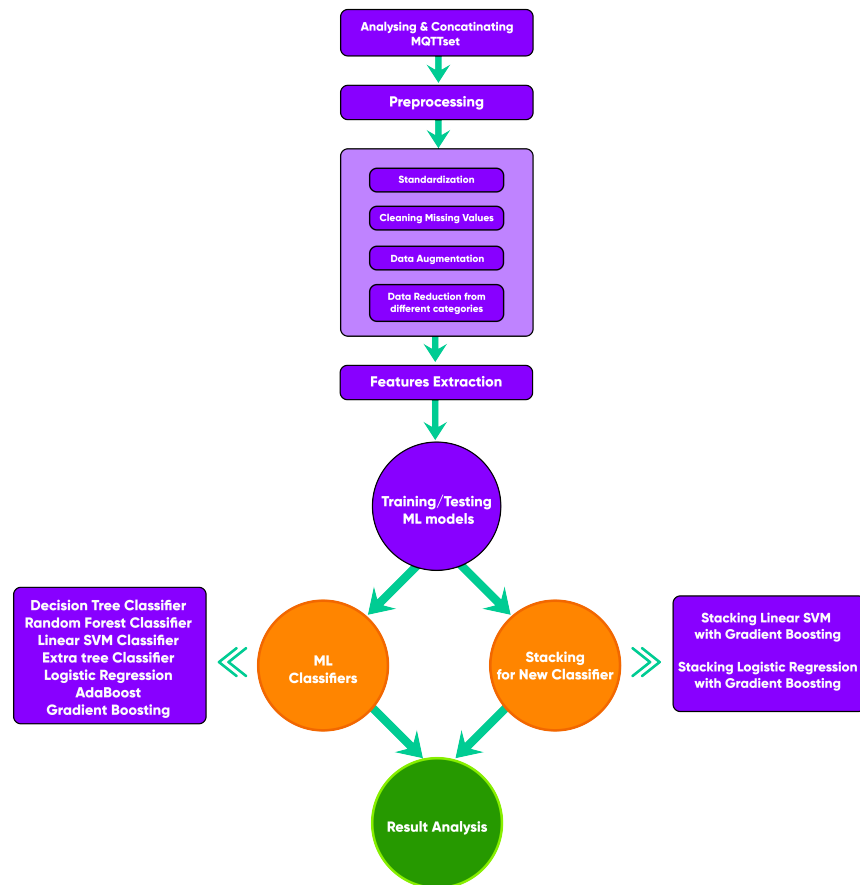


Figure 3.1: Flow chart of determining the respective algorithm's accuracy

For building our proposed system, we have divided our work methodology into 4 parts. It basically comprises implementing Random Forest, SVM, Decision Tree.

Fig. 1 shows a block diagram of the methodology. We have outlined our methodology as shown below:

- Dataset Loading
- Data Preprocessing
- Training the train data of the dataset with Random Forest, SVM, Decision Tree.
- Evaluate the Performances.

3.1 Dataset

We worked on a dataset called MQTTset which aims to reflect data especially on MQTT protocol. The MQTTset was generated using IoT-flock that is can generate network traffic similar to MQTT version 3.1.1. It emulates virtual devices which communicate with mqtt and CoAP in a m2m fashion. In addition to simulating different iot configurations, it can also implement different cyber-threats like publish flood,packet crafting attacks etc. The dataset is created with different mqtt clients representing different kinds of real life scenarios and connected them with a mqtt broker.

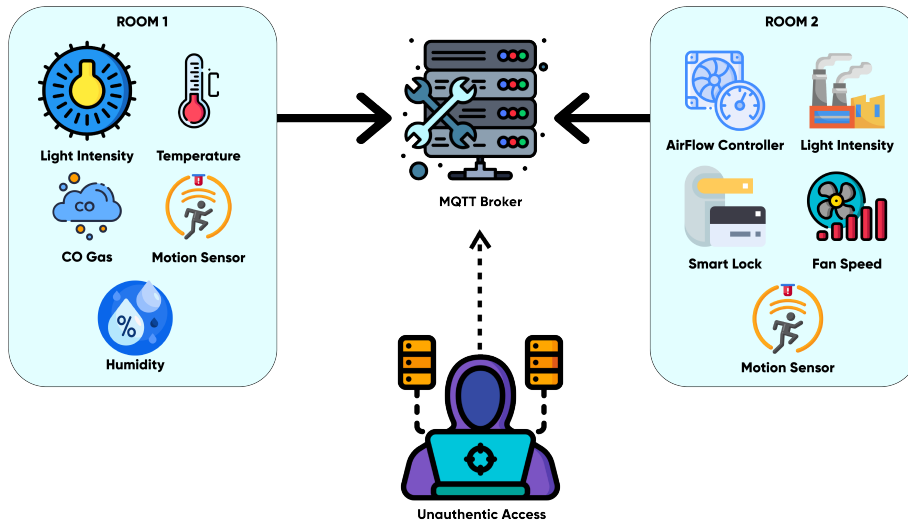


Figure 3.2: Testbed scenario considered to create the MQTTset dataset

In this implementation, sensors communicate with the broker and get data such as temperature, humidity, light intensity, door opening-closing etc. from the respected sensors. In case of emulating different cyber attacks, the malicious device is directly connected to the broker to execute the cyber-attacks. Each sensor retrieves information after some intervals depending on the sensor type. As an example, we can configure the humidity sensor to retrieve information during predefined intervals, on the other hand- door open or closing only retrieves information when an action is

occured. Each sensor has its own data profile and topic where they can publish or retrieve information using subscriber function. The MQTT traffic is developed as a packet capture (PCAP) file.

3.2 Attacks

Denial of Service(DoS): When an intruder tries to disable a device by interfering with its usual protocols then the type of attack is called denial of service. These types of attacks are usually overwhelms a targeted device with requests of services till the traffic becomes unprocessable to the device, causing more users to experience denial-of-service attacks. A DoS attack is defined by the fact that it is launched from a single computer.

SlowITe: Slow DoS against Internet of Things Environments (SlowITe) is an unique distributed denial of service (DDoS) attack against the MQTT protocol. This threat is characterized as a Slow DoS Attack since it targets a network service with minimum bandwidth and resources. MQTT uses TCP and this is why it is considered by SlowITe, and Slow DoS attacks can only target TCP-based protocols. SlowITe's main purpose is to connect to the server as many times as possible to fully leverage the MQTT broker's concurrent connections. The attacker initiates all accessible connections to achieve DoS. (at the application layer). The intruders main aim is to engage the MQTT broker with as little bandwidth as available.

Brute Force Authentication: A brute force attack obtains confidential information like usernames, passwords, and passphrases. Assailants can acquire access to data protected by credentials by continually submitting alternative combinations. Computer's power tries to crack something in this case, the credentials that protect sensitive data. Passwords, encryption keys, API keys, and SSH logins are all common targets for brute force assaults.

The MQTTset consisted of 6 csv files namely legitimate, dos, bruteforce, malformed, slowite, flood. These data were recorded from the testbed and separated by their respective csv files. Initially, we started working with 12,081,179 out of which 11,915,716 were Legitimate, 130,223 were dos, 14,501 were malformed, 9202 were slowite, and 613 were flood type data. All six csv files contain 60 attributes that are described in the table below.

3.3 Data Preprocessing

The first step of data manipulation is to import the dataset. This was done by extraction from the cloud using a drive. As datasets are mostly in Comma separated values(csv), to fetch data values from those we have to go through several steps. We used pandas which is built on NumPy library. Panda is an essential method used for structure and analysis of data. Hence, we imported the NumPy library and used the panda method to access and manipulate our csv file.

Data preprocessing is a method for removing non-essential variables from ML models that do not add to their accuracy. Furthermore, it performs all essential changes on the raw data to assist in the performance of ML models, resulting in greater outcomes and accuracy. After loading the dataset, data preprocessing was applied to it. As it is an important step for preparing the data for applying it before training our models for model accuracy. Intrusion detection architecture can be mainly of two types, host-based IDS and network-based IDS. As our main concern is based on MQTT which is a host-based architecture we aren't concerned about the other layers of the IoT network. As the name suggests, signature-based intrusion detection involves searching traffic for a predefined attack pattern. This predefined attack pattern is our target value based on which our model would be trained. Therefore, we chose the features that are best fit for our main objective of signature-based intrusion detection. These selections were made by eliminating attributes that were not related to MQTT. First step of our preprocessing was concatenating all the csvs into a single mqttdataset.csv. All the raw data sets with 60 attributes each were merged into one with a target categorizing the type of attack. We used a wrapper method for preprocessing which considers the performance of one Machine Learning algorithm as an evaluation criteria while iteratively removing the worst attributes, this is known as backward elimination. In backwards elimination we use all possible features and check the performance and iteratively delete the worst performing features in each iteration. For instance, the performance matrix we used here is mean. Whenever the process comes across a feature that has a mean of more than 0.3 then the feature is removed. Furthermore, all attributes that were found to have a standard deviation of 0 were considered useless and removed. Based on this we selected 19 features out of 60 features. Our next step was detecting and eliminating duplicate and NaN/Null values. This is a very important step of preprocessing which prevents the model from overfitting. First we analysed the number of duplicate data present in the processed data set. According to our observation there were no duplicate values however there were 12204 NaN/Null Values in the attribute "tcp.analysis.initial_rtt". To resolve this by replacing the NaN/Null values in the column by the mean of the column. Noisy data are data that are useless data items or features that causes disruption in the classification. Furthermore, we noticed that there was a huge difference in the number of data between legitimate and other attacks. This was causing the model to overfit and misclassify flood data as legitimate in many scenarios. To resolve this we have implemented data augmentation which also made the model overfit due to high duplication. Finally to minimize this we only reduced legitimate data from 11,915,716 to 399,977.

3.4 Training the train data

After preprocessing the data, we split the dataset into two separate csv files. One is for training the model (mqtt_train) containing all the features and target value denoted by x_train and y_train. The other is for testing the model (mqtt_test) which will enable us to predict the accuracy of our trained model. We split the preprocessed data in such a way that 80% are reserved for training the model and the other 20% data will be used to analyse the accuracy of the model trained. After splitting the model the train set had 532,370 with 19 selected attributes and the

test set contained 133,093 with 19 selected attributes. These train and test datasets were applied on 9 ML algorithms :

- *Decision Tree Classifier*
- *Extra tree Classifier*
- *Linear SVM*
- *Logistic Regression*
- *Random Forest Classifier*
- *Boosting Algorithms*
 1. *AdaBoost*
 2. *Gradient Boosting*
- *Stacking*
 1. *Stacking Linear SVM with Gradient Boosting*
 2. *Stacking Logistic Regression with Gradient Boosting*

3.5 Evaluate the Performances

A model's accuracy can be defined by dividing the total number of classes predicted by the total number of predictions. After training our model and running the above-mentioned classifiers we defined the accuracy of our model based on the deviation from the test.

Chapter 4

Algorithms

Machine learning is a new method used in artificial intelligence (AI) that enables the machine to train and build itself without being explicitly supervised. Computer programs that can handle large data sets themselves usually develop machine learning approaches. Although it usually provides faster, more accurate findings in identifying potential for profit or risk threats, full training may take more time and money.

4.1 Decision Tree Classifier

Machine learning uses a variety of methods, so choosing the right format for the given data and subject matter is a vital factor while building a ML system. The major two arguments for using the Decision Tree are as follows: Decision Trees are designed to show one's thinking skills while making decisions, making them easier to understand. Secondly, due to the tree-like structure of the decision tree, it can be easily interpreted. This algorithm checks the root values with the values of the record feature (real data). It then follows the branch and jumps to the next location depending on the comparison. The algorithm checks the number of attributes and sub-nodes after which it moves on to the next location. It repeats this process until it approaches the leaf blade of the tree. The following algorithm can help you understand the whole process: First, S recommends starting with the root node, which covers the entire database. Using the Selection Rate, select the appropriate database attribute (ASM). We need to divide the S into subsets with strong attribute values. Then creating a node in the decision tree with the best features. After which we create new decision trees in a repetitive manner using subsets of generated data. Continue this process until the nodes can no longer be separated, when the last node is called a leaf node. But in the trees of complex decisions, separation is not so easy so in those cases we use a lot of voting. For the decisions, the model will choose the one which maximizes the information gain. Here where entropy comes in. It is the average level of uncertainty inherent in the variable's possible outcomes. If the value of entropy is very high, it means we are very uncertain about a randomly picked point that we want to classify. Decisions tree 1st calculates the entropy of all the possible splits. Then it subtracts combined entropy of all the child nodes for a particular split from the entropy of the parent node to calculate the information gain. As we want to maximize the information gain, the decision which gives the bigger information gain value will be chosen by

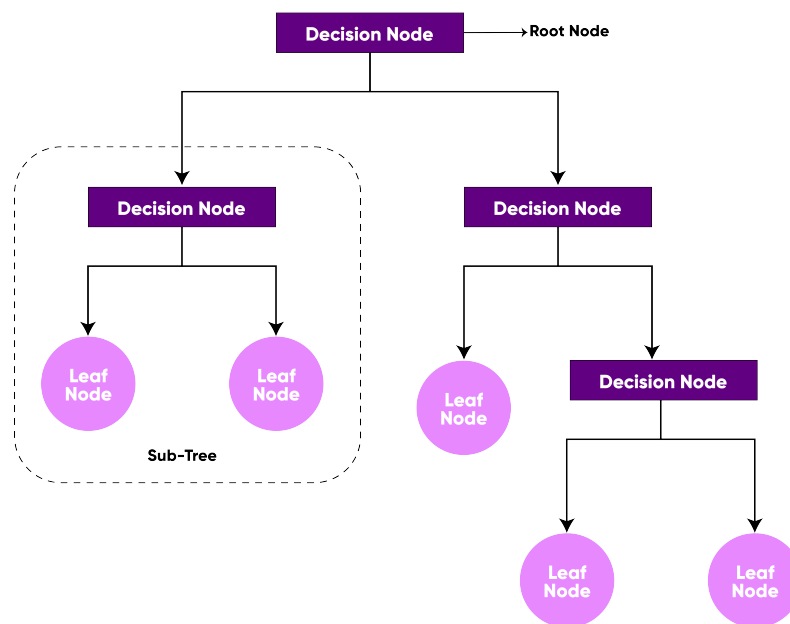


Figure 4.1: Decision Tree

the model. The degree of impurity in a particular property can be determined by an entropy metric. It denotes the unpredictability of data. The benefits of the decision tree are straightforward because we follow a method which one can use when making real choices in life. It can be very helpful in solving decision-making problems. It is essential to consider all possible outcomes to the problem. Compared to other methods, data purification is less required. However, there are some disadvantages, such as the following: Decision tree is as complex as it has several stages. It may have a common error, which can be corrected in the form of a random Forest.

4.2 Random Forest Classifier

Random forest is also another supervised machine learning approach that is one of the most flexible and easy to use algorithms. As Decision tree algorithm is very sensible to training data because based on the feature and target values the shape of the tree changes vastly which results in high variance. So instead of working with one decision tree, it uses multiple trees, hence the name random forest. Random forest algorithm trained with “bagging” method. The main idea of this method is that a number of learning models increases the overall outcome. This method contains two parts, Bootstrapping and Aggregation. First, it chooses random samples of the given dataset and each sample will contain the same number of rows. But any row can appear more than once because it uses random sampling with replacement. This process is called “bootstrapping”. “Bootstrapping” helps the model to be less sensitive to the original data. Although the number of rows is the same, for each sample, it will choose a subset of randomly chosen features from the main dataset. It helps to reduce the correlation between the trees. Then a decision tree will be constructed for each of the samples and will get a prediction result. When we pass

any new datapoint to be classified, although it can be used for both classification and regression, it will go through all the trees one by one and get the results for each tree. It will perform majority votes between all the results and based on that provide the final result. This process is called aggregation.

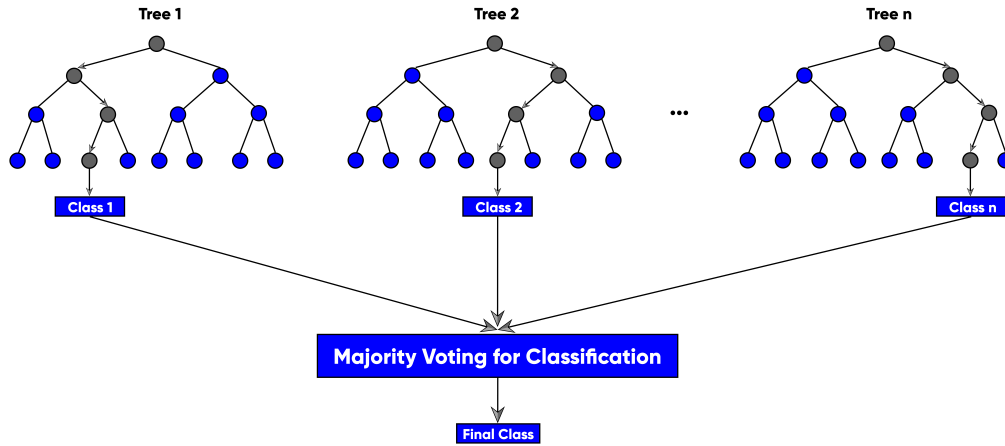


Figure 4.2: Random Forest-01

The benefits of Random Forest Tree is It may be used to solve problems involving classification and regression. Because the output is determined on majority voting or averaging, it eliminates the problem of overfitting. Even though the data set contains null/missing values, it functions well. Each decision tree formed is distinct from the others, demonstrating the parallelization characteristic. Because the average answers from a vast number of trees are used, it is quite stableIt preserves the diversity as all aspects are taken into account during the construction of each decision tree, although this is not always the case. The touch of greatness is not touched. Feature area is reduced as each tree does not inspect all buildings.But the drawbacks are when compared to decision trees, where decisions are taken by following the route of the tree, the random forest is far more complicated. Because of its intricacy, it takes longer to train than other models. Each decision tree must create output for the supplied input data whenever it needs to make a prediction.Random Forest is a very efficient method commonly used in various fields due to its efficient operation. It can manage binary, continuous, or partial data. One of the best features about a random forest is that it can accommodate missing variables, so it is a good decision for anyone who wants to create a model quickly and successfully. The random forest is a fast, easy, flexible, and durable model, but it has some drawbacks.

4.3 Support Vector Machines (SVM)

Support Vector Machine (SVM) algorithm is a dependable classification algorithm that performs fast and admirably with a limited amount of data to analyze. Like random forest, SVM is also a supervised machine learning approach which is used for two-group classification problems similar to our research. SVM uses classification algorithms that outperforms newer algorithms in terms of speed and performance with an exception that there are a small number of samples (in the thousands).

This is the reason why this approach is particularly best suited to text classification issues, in cases where we have access to a few thousand tagged samples which is why we choose to apply this approach. This algorithm classifies two groups using a soft margin classifier which allows misclassifications that makes the observations more accurate. The algorithm's goal is to find a hyperplane in an N-dimensional space (N- the number of selected features) which differs between data points. By allowing certain misclassifications this algorithm follows a bias/variance trade off.

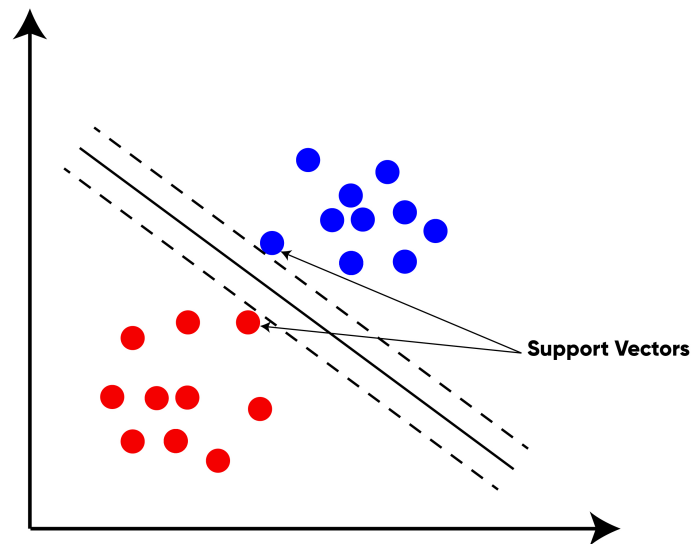


Figure 4.3: SVM

When an observation is made that is not within its soft margin, the algorithm is biased and misclassifies it which in turn decreases the variance in the output. Whenever there is a clear margin differentiating between classes, SVM performs exceptionally better than other algorithms. In high-dimensional spaces, SVM is more effective. Furthermore, SVM requires comparatively less memory. However, it comes with some disadvantages which are also big data sets, the SVM method does not perform well. If a data set contains noisy data, such as overlapping target classes, SVM does not perform well.

4.4 Extra tree Classifier

Extremely Randomized Trees Classifier (Extra Trees Classifier) is a type of integrated learning approach that yields the effect of classification by combining the results of a few related decision trees collected in the "forest". It is similar in concept to the Random Forest Classifier, except for the decision-making trees in the forest. The Extra Trees Forest's Decision Trees are all created from the first sample of training. Then, in each test site, each tree is given a random selection of k features from the element set, in which we must select the best data segmentation according to a specific mathematical condition (usually the Gini Index). decision trees are produced from this random sample of features. The total standard deviation of the

mathematical process used in the element classification decision is calculated electronically for each element during forest construction to select features using the above forest structure. To make a feature selection, each element is sorted by a drop-down program with Gini Values, and the user selects the advanced features of k. The entropy of Extra Classifier Tree is given below :

$$\text{Entropy}(S) = \sum_{i=1}^c -p_i \log_2 (p_i)$$

4.5 Logistic Regression

Logistic Regression is a 'Statistical Learning' methodology that comes under a supervised Machine Learning technique for 'Classification' problems. It has built a strong reputation over the previous two decades, particularly in the financial industry, due to its exceptional capacity to discover defaulters. The logistic sigmoid function converts the result of logistic regression in terms of a probability value. It uses a complex cost function, known as 'Sigmoid function' or 'logistic function' which is within a value between 0 and 1. As a result, line functions fail to be defined as they may have a value greater than 1 or less than 0, which is not systematically possible.

$$0 \leq h_{\theta}(x) \leq 1$$

4.6 Boosting

Boosting is an ensemble modeling strategy that aims to create a strong classifier out of a large number of weak ones. It is accomplished by constructing a model from a sequence of weak models. To begin, a model is created using the training data. The second model is then created, which attempts to remedy the faults in the previous model. This is repeated until whole training data set is accurately predicted.

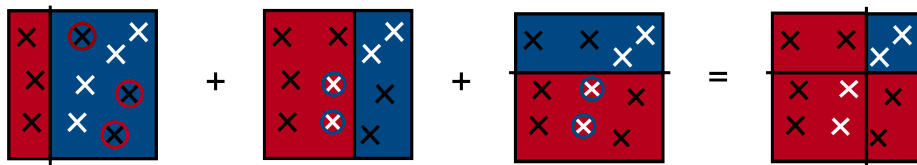


Figure 4.4: Boosting

Boosting has long been a popular method for dealing with binary classification issues. By turning a number of poor learners into strong learners, these methods boost prediction power. Boosting algorithms work on the idea of first building a model on the training dataset, then building a second model to correct the faults in the first model. This process is repeated until the mistakes are reduced and the dataset is accurately predicted. Boosting is considered to be 3 types and we used

two boosting which are :

4.6.1 AdaBoost

AdaBoost is a Machine Learning approach that is employed as part of an ensemble method. Decision trees with one level, or decision trees with only one split, are the most popular algorithm used with AdaBoost. Decision Stumps is another name for these trees.

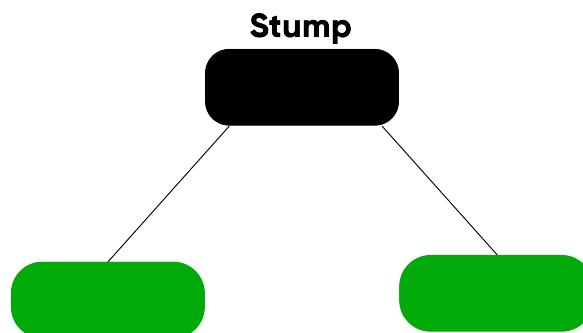


Figure 4.5: AdaBoosting

This algorithm creates a model by assigning equal weights to all of the data points. Then it gives us the wrongly divided points of maximum weight. In the following model, all the points with the highest weight are given extra value. It will continue to train the models until a small error returns.

4.6.2 Gradient Boosting

Gradient boosting is one of the most powerful algorithms in the field of machine learning. Machine learning algorithm errors can be categorized as: bias error and diversity error. Gradient boosting is one of the development methods used to reduce model bias error. The basic dimension of the gradient growth algorithm, unlike the Adaboosting algorithm, cannot be determined. The Bayes Gradient Boost rating has been adjusted, i.e. Stump Decision. We can change the n-rate of the gradient growth algorithm, such as AdaBoost. The default n estimator value of this algorithm is 100 if the n estimator value is not specified. The gradient magnification algorithm can be used not only for continuous prediction but also for class phase variables (as a Classifier). The Mean Square Error (MSE) is a costly function when used as a default, while Log loss is a cost function when used as a category. If we wish to minimize Bias error, we usually use the Gradient Boosting Algorithm. This algorithm is also useful for both retrofitting and editing problems. MSE is a costly operation in retrospect problems, while Log-Loss is a costly operation in the case of segregation.

4.7 Stacking

Stacking is the process of combining multiple categories or models into one model. This can be created in a variety of ways, the most popular being bagging and boosting. Bagging allows for the comparison of many comparable models with great flexibility to minimize variability. An alternative paradigm is stacking, it is used to look at different models of the same problem. The premise is that we may face a learning problem using many types of models, each of which can read part of the problem but not the whole problem area. As a result, we may create a number of different readers and use them to create a centralized prediction, one for each model studied. After which we add a new model that reads the same target to intermediate predictions. The term comes from the fact that the latter model is believed to be stacked on top of each other. As a result, we may improve your overall performance, and we may end up with a higher model than any intermediate model.

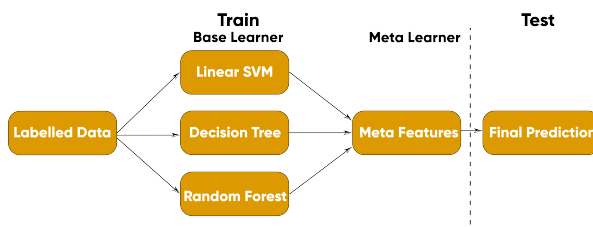


Figure 4.6: Stacking

However, we must remember that, like any machine learning method, it does not provide any guarantees. We used the opposite K validation to split training data into K-folds. In K-1 components, a basic model is included, and Kth component predictions are created. With each piece of training data, we do so. The performance of the basic model in the test set and then calculated by integrating it into the whole set of train data.

4.7.1 Linear SVM with Gradient Boosting

Linear SVM is used when the data is linearly separable, that indicates when a dataset can be categorized into two categories using only a linear line, it is known as linearly separable data, and hence the classifier is known as Linear SVM. On the other hand, gradient boosting is when we want to reduce the Bias error. In our proposed model we implemented stacking by Linear SVM and Gradient boosting algorithm to make a new classifier.

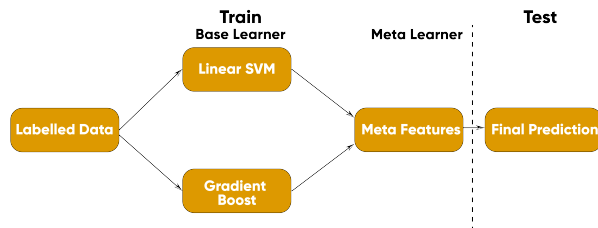


Figure 4.7: Linear SVM with Gradient Boosting

4.7.2 Logistic Regression with Gradient Boosting

Logistic regression is a supervised machine learning classifier that can predict the probability of a target variable. As the nature of our aim or dependent variable is contradictory, therefore we end up with only two classes. In other words, dependent variable is binary, with 1 indicating success and 0 indicating failure. We used this algorithm with Gradient Boosting and implemented stacking to use an ascended classifier.

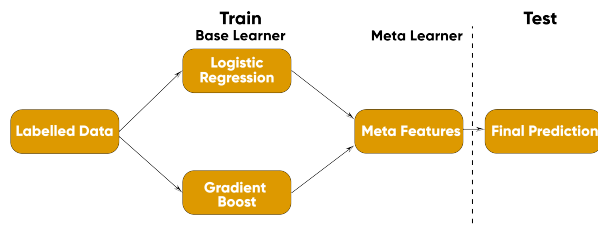


Figure 4.8: Logistic Regression with Gradient Boosting

Chapter 5

Result & Analysis

After running the model using Collab, results of classification of data are obtained. We applied 9 ML algorithms; they are Decision Tree, Random Forest, Linear Support Vector Machines (SVM), Extra Tree Classifier, Logistic Regression, AdaBoost, Gradient Boost, and Stacking. After pre-processing, we used trainset to train the model and used the testset to test the data and evaluate the performance of the model by different performance metrics.

5.1 Confusion Matrix

Another way of evaluating performance of a machine learning algorithm is Confusion Matrix. A confusion matrix presents a table layout of different outcomes like true positive, true negative, false positive and false negative of predictions and results which help us to visualize the nature of a model's outcomes.

(T.P) True Positive means, the algo predicted the value as a positive value and the actual value is also positive. On the other hand in case of (F.P) False Positive, the algo predicted the outcome as Positive but actual value is negative. (T.N) True Negative stands for, the algo predicted Negative and the actual outcome is negative and (F.N) False Negative means the algo predicted negative but the outcome is negative.

5.2 Classification Report

We derived the classification report of used algorithms by using python's sklearn library. Parameters like Precision, Recall, F1 score is generated by the classification report.

- **Precision:** Precision is the ratio of True positives to the total predicted positive sample. The formula for prediction is:

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

- **Recall:** The ability of an algorithm to find all positive samples is known as recall. The formula for the recall is:

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

- **F1 score:** The weighted average of Recall and Precision. The formula for F1 score is:

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision}).$$

- **Support:** The number of samples of the true sample that lie in that class is denoted by the support.

- **Micro average:** The result of averaging the total true positives, false negatives and false positives samples.

- **Macro average:** Unweighted mean of metrics calculated from each label.

- **Weighted average:** Calculates weighted mean of metrics calculated from each label and support .

5.3 Analysis

Table 5.1: Precision and F1-Score of Applied Algorithms

Algorithm	Precision	F1-Score
Decision Tree Classifier	0.97	0.96
Random Forest Classifier	0.99	0.99
Extra Tree Classifier	0.98	0.98
Linear SVM	1.00	1.00
Logistic Regression	0.93	0.91
Ada Boost	0.97	0.98
Gradient Boosting	0.91	0.92
Logistic Regression with Gradient Boosting	0.96	0.93
Linear SVM with Gradient Boosting	0.99	0.99

- Decision tree Classifier** For the Decision Tree, we get an accuracy score of 0.96. Here, the recall value for legitimate is 0.99. Although, the decision tree also showed low recall value for the dos ,it did not detected any positive true value for flood. The other recall values are 0.91 for slowIte, 1.00 for malformed,1.00 for bruteforce.

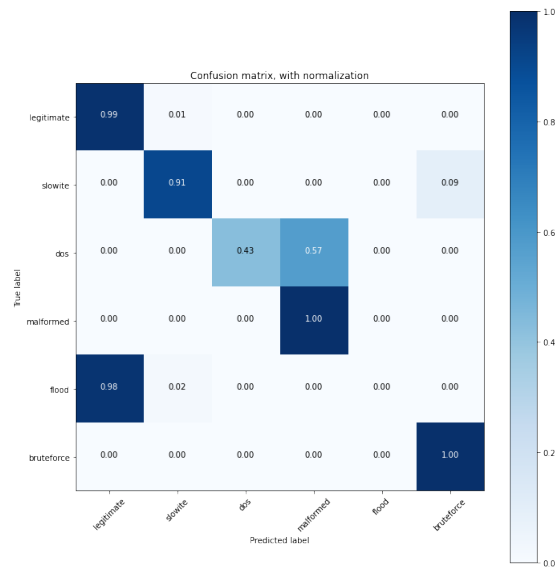


Figure 5.1: Confusion Matrix of Decision Tree

- Decision tree Classifier-ROC** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

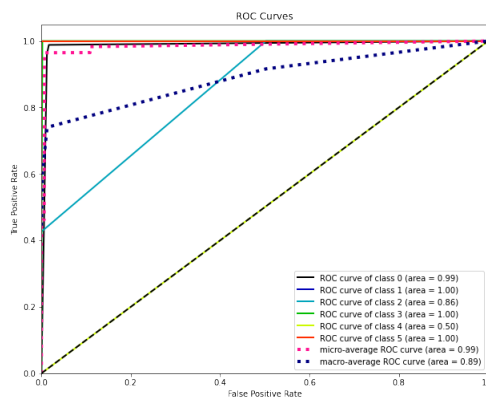


Figure 5.2: ROC curve of Decision Tree

- Random Forest Classifier** For the Random forest, we get an accuracy score of 1.00. Random forest classifier, shows recall value of 1.00 for legitimate, slowlfe, dos, malformed and bruteforce. But it shows very low recall value for flood.

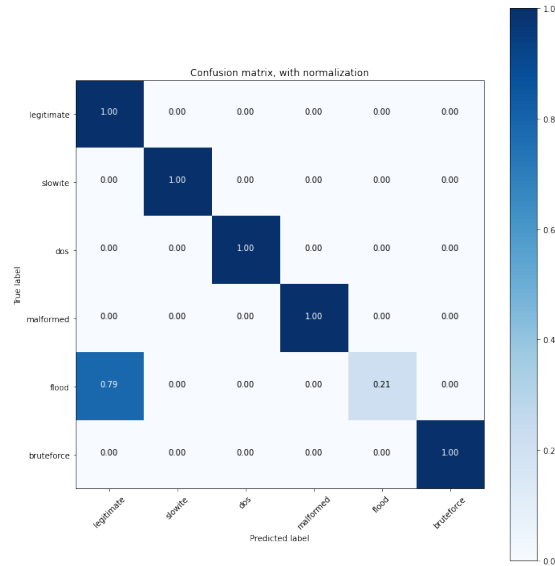


Figure 5.3: Confusion Matrix of Random Forest

- ROC Random Forest Classifier** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowlfe, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

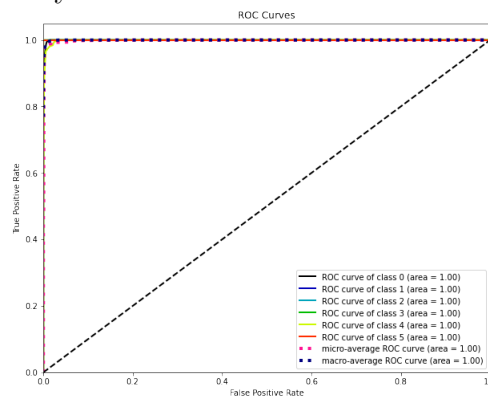


Figure 5.4: ROC curve of Random Forest

● **Linear SVM** For the Linear SVM, we get an accuracy score of . The precisions values are 0.86 for legitimate, 1.00 for slowIte,0.00 for dos, 1.00 for malformed,0.98 for flood and 1.00 for bruteforce. The recall values are 0.98 for legitimate, 1.00 for slowIte,0.00 for dos,1.00 for malformed,0.90 for flood and 1.00 for bruteforce.

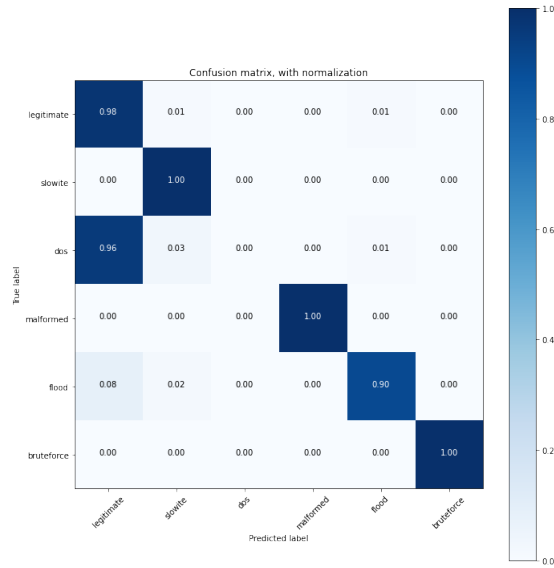


Figure 5.5: Confusion Matrix of Linear SVM

● **ROC of Liner SVM** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

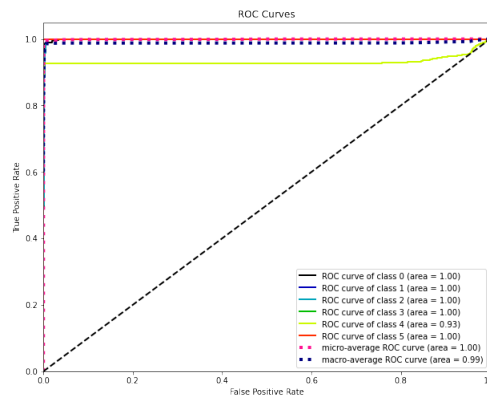


Figure 5.6: ROC curve of Linear SVMe

● **Extra tree Classifier** For the Extra Tree Classifier, we get an accuracy score of 0.98 . For Extra tree classifier, although shows high precision for legitimate values, shows low recall value of 0.49. SlowIte, dos, malformed shows high precision and recall values. On the other hand, flood also shows low recall value of 0.60.

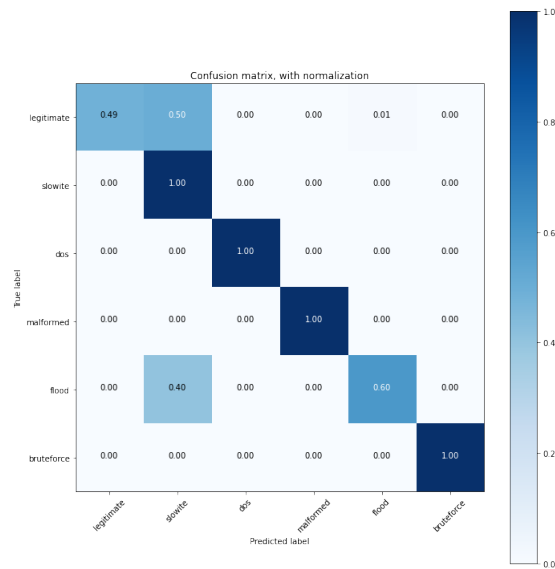


Figure 5.7: Confusion Matrix of Extra tree Classifier

● **ROC Extra tree Classifier** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

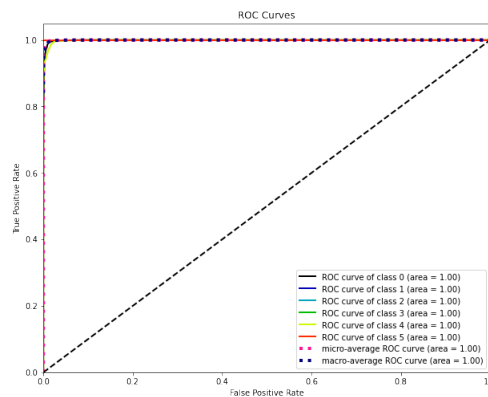


Figure 5.8: ROC curve of Extra tree Classifier

• **Logistic Regression** For the Logistic Regression, we get an accuracy score of 0.91. Logistic regression shows only high precision and recall value for malformed and shows the highest recall value of 1.00 for brute force. Other high precision values are 0.96 for legitimate which have a recall value of only 0.34.

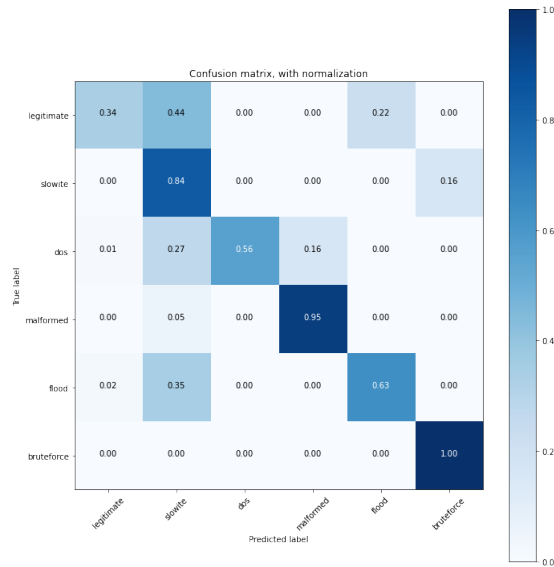


Figure 5.9: Confusion Matrix of Logistic Regression

• **ROC Logistic Regression** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowite, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

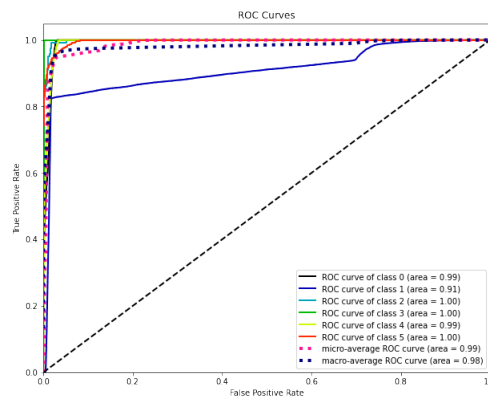


Figure 5.10: ROC curve of Logistic Regression

- AdaBoost** For the Logistic AdaBoost, we get an accuracy score of 0.98. The precisions values are 0.57 for legitimate, 1.00 for slowIte,0.26 for dos, 1.00 for malformed,0.00 for flood and 1.00 for bruteforce. The recall values are 1.00 for legitimate, 1.00 for slowIte,1.00 for dos,1.00 for malformed,0.00 for flood and 1.00 for bruteforce.

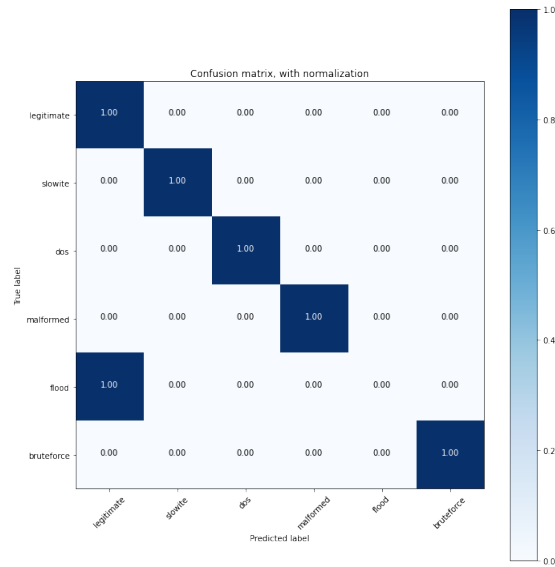


Figure 5.11: Confusion Matrix of AdaBoost

- ROC of AdaBoost** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

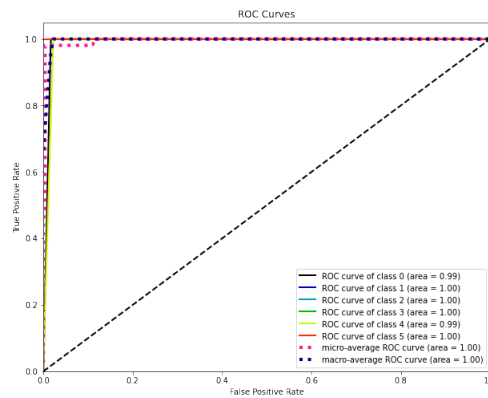


Figure 5.12: ROC curve of AdaBoost

- Gradient Boosting** For Gradient Boosting, we get an accuracy score of 0.92. The precisions values are 0.00 for legitimate, 0.95 for slowIte, 0.94 for dos, 0.96 for malformed, 0.00 for flood and 0.24 for bruteforce. The recall values are 0.00 for legitimate, 0.78 for slowIte, 0.98 for dos, 1.00 for malformed, 0.00 for flood and 1.00 for bruteforce.

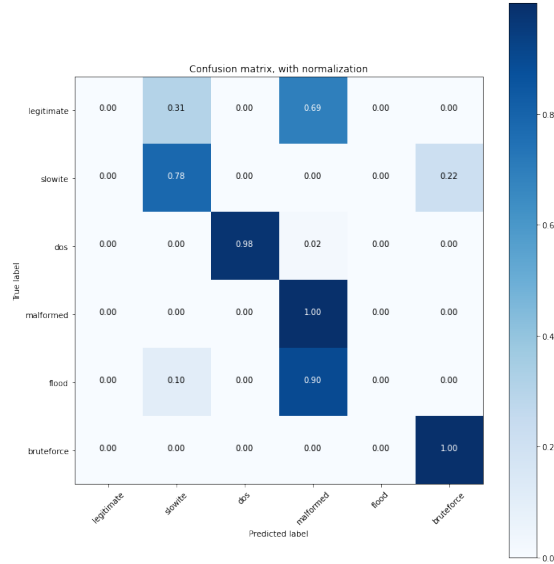


Figure 5.13: Confusion Matrix of Gradient Boost

- ROC of Gradient Boosting** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

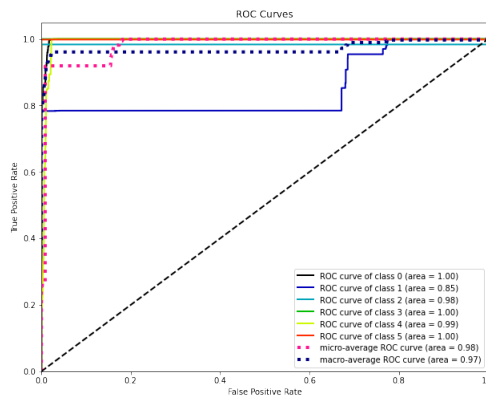


Figure 5.14: ROC curve of Gradient Boost

- Stacking Linear SVM with gradient boosting** For Linear SVM with Gradient Boosting, we get an accuracy score of 0.95. The precisions values are 0.94 for legitimate, 0.99 for slowIte, 0.94 for dos, 1.00 for malformed, 1.00 for flood and 0.24 for bruteforce. The recall values are 0.98 for legitimate, 0.78 for slowIte, 0.98 for dos, 1.00 for malformed, 0.85 for flood and 1.00 for bruteforce.

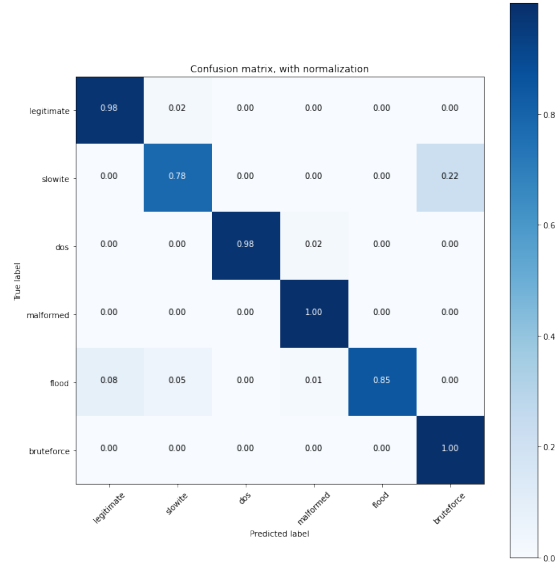


Figure 5.15: Confusion Matrix of Stacking Linear SVM with gradient boosting

- ROC Linear SVM with gradient boosting** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, greenyellow for flood and coral for bruteforce.

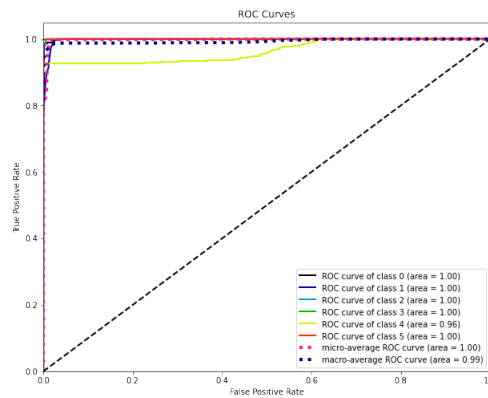


Figure 5.16: ROC curve of Stacking Linear SVM with gradient boosting

- Stacking Logistic Regression with Gradient Boosting** For, Logistic Regression with Gradient Boosting we get an accuracy score of 0.93. The precisions values are 0.97 for legitimate, 0.89 for slowIte,0.87 for dos, 1.00 for malformed,0.73 for flood and 0.24 for bruteforce. The recall values are 0.98 for legitimate, 0.78 for slowIte,0.98 for dos,1.00 for malformed,0.85 for flood and 1.00 for bruteforce.

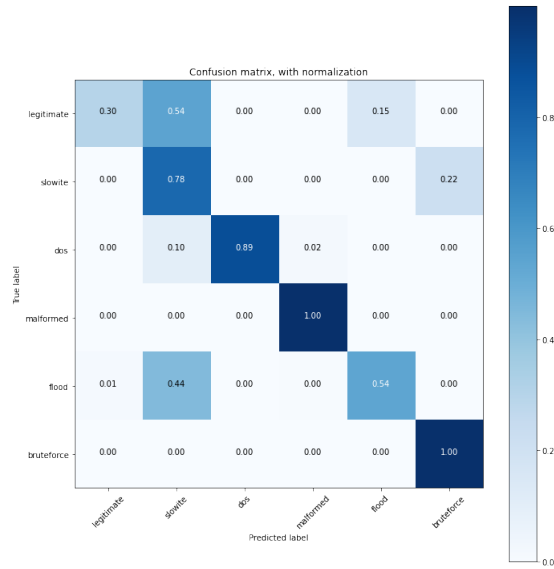


Figure 5.17: Confusion Matrix of Stacking Logistic Regression with Gradient Boosting

- ROC Logistic Regression with gradient boosting** Here, black color represents the roc curve of the legitimate value, The other values are by color, blue for slowIte, Cyan for dos, green for malformed, yellow for flood and coral for bruteforce.

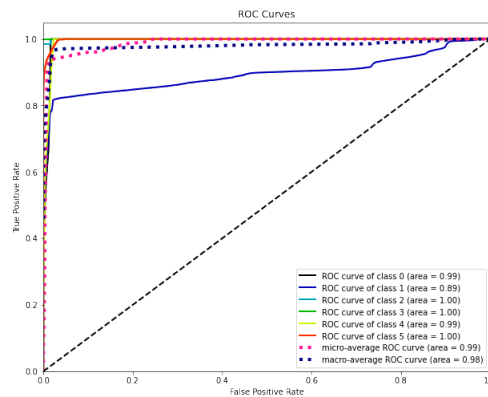


Figure 5.18: ROC curve of Stacking Logistic Regression with Gradient Boosting

Chapter 6

Conclusion

The expansion of the usage of IoT devices has led to a crucial security unsettlement. Although a number of security solutions are being fabricated through researchers to shelter these data in information systems, wistfully these analyses are still inadequate. This manifests why IoT devices are yet a censure issue when it comes to data proceeding. According to our result and analysis stacking Linear SVM with gradient boosting algorithm performed the best in identifying each attack. The development of these ML approaches in an IoT network scenario is vital for the security of MQTT-based IoT devices. A ground of this snag could be the limitation of research being conducted till date. Furthermore, we will be working on developing a real-time intrusion detection system. As we mentioned throughout our research, MQTT is a lightweight and simple protocol and IoT having low-bandwidth and low computation power, our IDS should be deployed in a decentralized ML server in order to reduce the overhead while enhancing the security of MQTT-based IoT devices. As a result, the motive of this research is nothing but to appraise this vacancy of security with the requisite height of protection.

Bibliography

- [1] S. Kraijak and P. Tuwanut, "A survey on internet of things architecture, protocols, possible applications, security, privacy, real-world implementation and future trends," in *2015 IEEE 16th International Conference on Communication Technology (ICCT)*, IEEE, 2015, pp. 26–31.
- [2] V. Lampkin *et al.*, *Building smarter planet solutions with MQTT and IBM websphere MQ telemetry*. IBM Redbooks, Dec. 2012.
- [3] D. Soni and A. Makwana, "A survey on mqtt: A protocol of internet of things (iot)," in *International Conference On Telecommunication, Power Analysis And Computing Techniques (ICTPACT-2017)*, vol. 20, 2017.
- [4] B. Mishra and A. Kertesz, "The use of mqtt in m2m and iot systems: A survey," *IEEE Access*, vol. 8, pp. 201 071–201 086, 2020.
- [5] D. Thangavel *et al.*, "Performance evaluation of mqtt and coap via a common middleware," in *2014 IEEE ninth international conference on intelligent sensors, sensor networks and information processing (ISSNIP)*, IEEE, Apr. 2014, pp. 1–6.
- [6] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the ip-based internet of things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, Dec. 2011.
- [7] A. Kalita, "Internet of things," *Challenges and Research Opportunities*, 2017.
- [8] X. Wang, J. Zhang, E. M. Schooler, and M. Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the iot," in *2014 IEEE International Conference on Communications (ICC)*, IEEE, 2014, pp. 725–730.
- [9] K. S. Kiran, R. K. Devisetty, N. P. Kalyan, K. Mukundini, and R. Karthi, "Building a intrusion detection system for iot environment using machine learning techniques," *Procedia Computer Science*, vol. 171, pp. 2372–2379, 2020.
- [10] J. Sengupta, S. Ruj, and S. D. Bit, "A comprehensive survey on attacks, security issues and blockchain solutions for iot and iiot," *Journal of Network and Computer Applications*, vol. 149, p. 102 481, 2020.
- [11] R. Atmoko, R. Riantini, and M. Hasin, "Iot real time data acquisition using mqtt protocol," in *Journal of Physics: Conference Series*, vol. 853, 2017, p. 012 003.

- [12] E. Ciklabakkal, A. Donmez, M. Erdemir, E. Suren, M. K. Yilmaz, and P. Angin, “Artemis: An intrusion detection system for mqtt attacks in internet of things,” in *2019 38th Symposium on Reliable Distributed Systems (SRDS)*, IEEE, 2019, pp. 369–3692.
- [13] Y. NarasimhaRao, P. S. Chandra, V. Revathi, and N. S. Kumar, “Providing enhanced security in iot based smart weather system,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, no. 1, pp. 9–15, 2020.