

# Demystifying Machine Learning Models for IOT Attack Detection with Explainable AI

by

Rabeya Khatun Muna  
21341056

Homaira Tasnim Maliha  
18101455

Mahedi Hasan  
17101544

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  
September 2021

© 2021. 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:**




---

Rabeya Khatun Muna  
21341056



---

Homaira Tasnim Maliha  
18101455



---

Mahedi Hasan  
17101544

# Approval

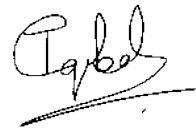
The thesis/project titled “Demystifying Machine Learning Models for IOT Attack Detection with Explainable AI” submitted by

1. Rabeya Khatun Muna (21341056)
2. Homaira Tasnim Maliha (18101455)
3. Mahedi Hasan (17101544)

Of Summer, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on September 12, 2021.

## Examining Committee:

Supervisor:  
(Member)



---

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

Co-Supervisor:  
(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

Internet of things (IoT) dramatically is changing our lives with its newly invented devices and applications which leads to various emerging cybersecurity challenges or threats. The rapid growth of IoT arouses security and privacy issues that need more attention to ensure the safety of human personal data, saving from serious damages. Over the year, several techniques have been conducted to establish IoT attack detection model so that it can detect attacks efficiently. Unfortunately, it is difficult to identify a good model that can detect both binary and multi-type attacks with accurately. The prediction result of models provides very little knowledge to the users or experts how the model classify attacks for detection which can not be understood through a simple output. Thus, it is getting necessary to understand the reasons behind the prediction to make people trust on the model by providing the insight view of the model.

In the paper, we have introduced Explainable Artificial Intelligent on our proposed model for making the model faithful enough and human understandable, by explaining the strategy of the detection model for predicting the attacks and representing the features or properties influence of respective prediction. For this, we have establish an IoT attack detection model by using Xg-boost classifier on a dataset, name, IoT Intrusion Dataset[11], that supports both binary and multi-class classification to classify the attacks for detection. We have also used Explainable AI tools, named, Shap, Lime, and ELI5 to validate the performance of the model through analyzing the property of the established model by representing each feature's contribution and action of the model, for each prediction to give a clear idea how efficient the model is, for detecting the IoT attacks.

**Keywords:** Xg-boost, Explainable AI, XAI, Lime, Shap, ELI5, IoT attack, SMOTE, PCA

## **Dedication (Optional)**

We want to dedicate our work to our parents, without whom we could never have come so far in life.

## **Acknowledgement**

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, to our supervisor Dr. Muhammad Iqbal Hossain and our co-supervisor Dr. Md. Golam Rabiul Alam for their kind support and advice in our work. He helped us whenever we needed help.

And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

# Table of Contents

Declaration	i
Approval	ii
Abstract	iii
Dedication	iv
Acknowledgment	v
Table of Contents	vi
List of Figures	viii
List of Tables	ix
Nomenclature	x
<b>1 Introduction</b>	<b>1</b>
1.1 Problem Statement . . . . .	1
1.2 Aims and Objectives . . . . .	2
1.3 Thesis Outline . . . . .	2
<b>2 Related Work</b>	<b>3</b>
2.1 Literature Review . . . . .	3
2.2 Machine Learning Models . . . . .	5
2.2.1 Principal Component Analysis . . . . .	5
2.2.2 XG-BOOST . . . . .	5
2.3 Explainable Artificial Intelligence (XAI) Tools . . . . .	6
2.3.1 ELI5 . . . . .	6
2.3.2 LIME . . . . .	7
2.3.3 SHAP . . . . .	7
<b>3 Proposed Model</b>	<b>9</b>
3.1 Work Plan . . . . .	9
3.2 Dataset Collection and Description . . . . .	10
3.3 Data Preprocessing . . . . .	13
3.4 Feature Selection . . . . .	13
3.4.1 Mutual Information Based Feature Analysis . . . . .	13

3.4.2	Applying Principal Component Analysis(PCA) for Feature Selection . . . . .	15
3.5	Oversampling . . . . .	17
3.5.1	Applying SMOTE for Oversampling . . . . .	17
3.6	Training and Testing . . . . .	17
3.7	Classification of Models . . . . .	18
3.7.1	Applying XGBoost for Classification . . . . .	18
3.7.2	Applying Cross-Validation . . . . .	19
3.8	Implementation XAI Tools On Model . . . . .	20
3.8.1	Applying LIME To IoT Attack Detection Model . . . . .	20
3.8.2	Applying SHAP To IoT Attack Detection Model . . . . .	21
3.8.3	Applying ELI5 To IoT Attack Detection Model . . . . .	22
<b>4</b>	<b>Performance Evaluation</b>	<b>23</b>
4.1	XG-Boost Output Analysis . . . . .	23
4.1.1	Label feature . . . . .	24
4.1.2	Category feature . . . . .	25
4.1.3	Sub-Category feature . . . . .	27
4.2	XAI-Tools result analysis . . . . .	29
4.2.1	Sub-category . . . . .	29
4.2.2	Category . . . . .	34
4.2.3	Label . . . . .	38
<b>5</b>	<b>Conclusion</b>	<b>42</b>
	<b>Bibliography</b>	<b>44</b>



# List of Figures

3.1	Proposed Model . . . . .	9
3.2	Classes of Target Features. . . . .	11
3.3	Correlation Matrix of IoTID20 Dataset. . . . .	12
3.4	LABEL - feature importance by Mutual Information Classification Algorithm. . . . .	14
3.5	Category feature importance by Mutual Information Classification Algorithm.. . . . .	14
3.6	Sub-category feature importance by Mutual Information Classification Algorithm. . . . .	15
3.7	Iot dataset no of components vs variance . . . . .	16
3.8	14 selected feature by using PCA. . . . .	17
3.9	Label Features. . . . .	18
4.1	LABEL - confusion matrix. . . . .	25
4.2	CAT - confusion matrix. . . . .	26
4.3	sub cat - confusion matrix. . . . .	28
4.4	Weight of each features for sub-category using ELI5. . . . .	29
4.5	Feature contribution analysis of Sub-category by ELI5 for the classification using 399520 test data input . . . . .	30
4.6	The output of Sub-category for IoT attack detection using Lime with 399520 Test data input. . . . .	31
4.7	SHAP output for Sub-category using 399520 Test input data. . . . .	33
4.8	SHAP Summary Plot for subcategory. . . . .	34
4.9	Cat ELI5 weight. . . . .	35
4.10	CAT - eli5 output. . . . .	35
4.11	CAT - lime output. . . . .	36
4.12	CAT SHAP OUTPUT. . . . .	37
4.13	CAT - summery plot. . . . .	38
4.14	LABEL ELI5 weight. . . . .	39
4.15	LABEL ELI5 output. . . . .	39
4.16	LABEL LIME output. . . . .	40
4.17	LABEL SHAP output. . . . .	40
4.18	LABEL - summery plot. . . . .	41

# List of Tables

3.1	Attack types and Instances of Binary, Category and Sub-category in IoTID20 Dataset . . . . .	11
3.2	PCA Table with feature names. . . . .	16
3.3	Grid values and Optimal parameters Of XGBOOST Algorithm. . . . .	19
4.1	LABEL Classification report of the proposed method. . . . .	24
4.2	The Data Representation of Confusion Matrix for Label target Column. . . . .	25
4.3	CAT - Classification report of the proposed method. . . . .	26
4.4	The Data Representation of Confusion Matrix for Category Target Column. . . . .	27
4.5	Sub-Cat - Classification report of the proposed method. . . . .	27
4.6	The Data Representation of Confusion Matrix for Sub-Category Target Column. . . . .	28

# Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

*AI* Artificial Intelligence

*CAT* Category

*CV* Cross Validation

*DSF* DoS-Synflooding

*ELI5* Explain Like I'm 5

*IoT* Internet of Things

*LIME* Local Interpretable Model-Agnostic Explainable

*MAF* Mirai-AckFlooding

*MAS* MITM ARP Spoofing

*MHB* Mirai-Hostbruteforceg

*MHF* Mirai-HTTP Flooding

*ML* Machine Learning

*MLP* Multi-Layer Perceptron

*MUF* Mirai-UDP Flooding

*PCA* Principal Component Analysis

*SHAP* Shapley Additive Explanations

*SHP* Scan Hostport

*SMOTE* Synthetic Minority Over-sampling Technique

*SPS* Scan Port OS

*SubCat* Sub Category

*XAI* Explainable Artificial Intelligence

*XGB* eXtreme Gradient Boosting

# Chapter 1

## Introduction

### 1.1 Problem Statement

The term Internet of Things(IoT) refer to the group of application, machine or devices that has the ability to become interconnected with each other, that constantly communicate, store and exchange data without any human interference while connecting with the internet. It is an interrelated system with computing devices, digital machines, objects, or humans over a network without requiring human-to-human or human-to-computer interaction, which plays an insignificant role in the way of life. According to the information of the report [9], around 75 billion of devices will be invented within 2025 as IoT technology is required in all sphere of life and numerous field such as digital home, industry, city and many more which enhance the productivity in economic sectors and the quality of life. As the Internet is getting faster, cheaper, easily integrated to other devices or systems, and available to all, its usability and application on devices are drastically increasing to every corner of the world which is becoming one of the biggest concern now. With the rapid growth of the IoT devices, there is a great thread to be attacked that can lead to the device vulnerability and violation of personal data that causes drastic loss and massive security issue. In recent decades, attacking on IoT devices on the rise, which causes security threats and vulnerability that lead to global concern to save human's personal data and IoT devices from being corrupted. Several steps have been taken to detect attacks.However, it needs to be efficient enough so that people can rely on it. For this, Explainable Artificial Intelligence (XAI), invented by Lent et.al[10], is a great method to interpret the model for clarifying the behaviour of each feature and their participation to give a correct output. By observing the output only ,it is difficult to understand how does the model came to such decision for given inputs and what are participation tendency of each input domain to have a specific output of the model.

Explainable AI is a machine learning technology that has the ability to explain a prediction accurately at the individual level. Adding Explainable AI at each steps of model not only give internal view and activities of the model, but also helps to improve the design of the model through explaining the outcome of machine learning models for the following reasons:

(a) Explainable AI shows the reports consist of false positive and false negative that helps understanding the decision path of each edge and improving robustness of the system by highlighting antagonistic input feature or domain that have impact on

changing the prediction.

(b) Explainable result ensures that the data driven decision is correct from bias by auditing the data used for training the machine learning (ML) models like detection[10].

## 1.2 Aims and Objectives

In this paper, we have proposed an effective machine learning model, that not only have the ability to classify any kind of attacks but also gives a novel explanation of individual attacks for each classification of IoT attack detection model with a faithful manner for evaluating the model and making the user trust on it by representing the individual prediction of each attack. Our research aim is not only to build an IoT attack detection model but also conducting a security analysis using XAI methods like Shap, Lime and Eli5 to ensure that an underlying truthful consequence exists in the model reasoning. The objectives of this research are:

- We proposed a model that supports both binary and multi-class classification for detection.
- We proposed to implement three different tools of XAI to exhibit the underlying procedure of IoT attack for prediction, Like-
  - (a) Eli5: Explains their top influenced feature for the final prediction by showing weights through debugging the machine learning classification model,
  - (b) Lime : selects number of best features to describe the complex output and
  - (c) Shap: contributes to show feature importance with feature effects on the prediction model through relevant plots, to make the model easily understandable for any non-experts.
- We tested our proposed model's consistency, correctness and assurance property using XAI methods to achieve the goal for detecting attacks accurately.

## 1.3 Thesis Outline

The rest of the paper is structured as follows where each section holds a context related to our paper. In **Chapter 2**, the background information of our researcher is presented, that is consist of related research work of machine learning model, Explainable AI methods and literature review of related papers and in **Chapter 3** the essential steps for establishing the proposed is described sequentially. We also described the implementation of XAI method in **Chapter 4**. We evaluate the output of XAI methods in **chapter 5** and give the final results and decision of model in **chapter 6**. Finally, we concluded in the last chapter of the paper.

# Chapter 2

## Related Work

### 2.1 Literature Review

Supervised learning is a subcategory of machine learning and artificial intelligence which is defined by its use of labelled datasets to train algorithms to classify data or predict outcomes accurately. So basically, Supervised learning is a way of teaching or training a machine using labelled data. First of some data is being tagged with the correct answer. From that point onward, the machine is given another arrangement of examples(data) so the supervised learning algorithm examines the training dataset and delivers the right result from labelled data. Supervised learning is classified into two categories of algorithms: Regression and Classification. In this paper [5], we observed that the use of actual IoT network traffic with specific network layer attacks have been implemented which is in contrast to other works creating supervised learning models, with generic datasets. Moreover, the result of the proposed model shows that when the model operates with unknown data taken from the same network topology, it achieves up to 100% accuracy and when it operates in an unknown topology, it achieves 81% accuracy [5]. We can use Supervised learning as it permits gathering data and produces data output from past experiences. Also, Supervised machine learning assists with settling different kinds of genuine calculation issues. For classification and detection of the IoT attacks, we can use Supervised learning with XGBOOST classifier on IoT Intrusion Dataset [14] to establish a model.

In this research paper[10], the proposed model is used by attackers who have no knowledge about the underlying architecture of the data model. They use the model for performing attacks on the black-box settings. For evaluating the qualitative attack, this method is used by most of the searchers. The future work will focus on three areas by using the proposed method, which is the study of defence mechanisms against the proposed attack, the Extendable proposed method to compromise the privacy and confidentiality properties, and Examining the security robustness of other XAI with different neural network architectures.

In this paper [13], XAI is examined as a procedure that can be utilized in the investigation and analysis of healthcare information by AI-based frameworks and a proposed approach giving the point of achieving accountability, transparency, result tracing and model improvement in the area of medical care. These explanations

can be dissected with the assistance of a clinician's information. This examination will empower approval of expectations made by the AI model by clinicians to empower transparency. In the event that if the predictions are right, at that point explanations alongside clinical information can be utilized to create significant experiences and proposals. In the event that predictions are incorrect, at that point, the differences among explanations and clinician's information can be utilized to follow factors for incorrect expectations and empower improvement in the deployed AI model.

In the research paper [10], Explainable Artificial Intelligence (XAI) methods are used to help users better understand the internal workings of ML models. The analysts fundamentally centered around XAI techniques for covering different security properties and danger models applicable to the online protection area including recognition of various kinds of IoT assaults. They used a dataset to train the model where if the attackers infuse the deficiencies in the dataset, it will give some wrong segregation and attack recognition which causes system failure. In the paper [7], the author mentions one of the methods of explainable AI that is XSP P-LAIN. The method gives in-detail information on the property model and data, how data is collected, and when the model trains the data, and if there is any noise added or not. Only that, the XT-PLAIN method is used at the time of deployment and training. In our research, we can use the XAI method for getting in-detail information about the dataset and model which will give an explanation, of which part of the noise is added. In the end, to investigate the data model and data set for proper classification of IoT attack detection, we can use the Explainable AI method for data reliability for proper training.

Another research paper [6], proposes an Explainable Artificial Intelligence system for handling big data for faster decisions and responses. With the increasing number of connected devices with the network, the attacking techniques are also increasing, making the system vulnerable. In this paper, six algorithms have been mentioned for predicting malicious records and proper classifying of attack patterns which are Artificial Neural Network (ANN), Support Vector Machine (SVM), Random Forest (RF), Extra Trees (ET), Gradient Boosting (GB) and Naive Bayes (NB) which we can use for our own research to learn the complex patterns of attack, mapping the input vectors of the dataset in high dimensional feature space, for the supervised classification task, to increase the performance of the detections and minimizing error which will increase the faster learning for prediction.

To understand the predictions of the machine learning model, we need the process of interpretation of that machine learning model which helps us to debug, test reliability and the casualty of the features and help us to make it learn bias so that it can predict being fair [3]. LIME, SHAP, and ELI5 are the libraries of interpretation, which has been used in solar PV power generation as XAI tools to an application of a random forest forecasting model so that the predictions of the model can be understood and explained as well as add to embracing XAI tools for smart framework applications [11]. In this article, the public dataset from GEFCOM 2014 has been used. As per the outcomes, XAI tools can give point by point data to interpret the model features and results just as the progress of the model's outcomes through

explainability and transparency. This study has given a couple of pointers on the best way to pick an XAI tool - LIME, SHAP and ELI5. Each XAI tool has its own qualities and impediments, in terms of the computing cost, explanation, features, weights, execution time, and so on [11]. Moreover, A system dependent on Shapley Additive explanations (SHAP) that gives local and global explanation to any IDS was proposed in [15] and the outcomes show that the translation results, produced by the system, are reliable with the attributes of the particular attacks, and the outcomes are natural. However, SHAP is as yet not possible to work in real-time progressively in spite of the fact that it has a quick calculation for interpreting AI models.

## 2.2 Machine Learning Models

### 2.2.1 Principal Component Analysis

Principal Component Analysis(PCA) is a useful algorithm that reduces the number of variables of a dataset by using liner algebra to transform the dataset into a compressed form while maintaining as much information as possible . According to the paper[2], PCA is not feature selection algorithm, rather, it is a transform technique that is used to reduce correlation between the features and has the ability of dimension reduction which makes the algorithm potential enough to select important features. Janecek and Gansterer(2008) has specified some attributes of PCA in the research paper[1] and they are, a)The covariance is 0 for each of two of features, b)features are sorted descending order with respect to their variance, c)the features take hold the maximum amount of variance of the data and the following attributes consecutively captures residual data as much as possible. Song et al. - 2010 tried to use feature extraction technique of PCA for feature selection through analysis in the research paper[2] by following some procedures which is summarized in our paper, using a mathematical equation,

$$y = \sum_{j=1}^n b_j x_j \quad (2.1)$$

the feature extraction result, in equation1, is used corresponding to  $x$ , which is an eigenvector of covariance matrix of PCA, of an arbitrary sample vector  $b$  where  $b = [b_1, b_2, \dots, b_n]^T$ ,  $x = [x_1, x_2, \dots, x_n]^T$  and  $n$  is the dimensionality of sample vectors. For the feature extraction results of the  $j$ th feature element of samples, the absolute value  $x_j$  evaluates the contribution where the small absolute value  $x_j$  is, the contribution of the  $j$ th feature component is less of samples and removing the small value  $b_k x_k$  (suppose,  $k$ th is the small enough) from the equation1 will not make any change on the result of feature extraction, from which, it is understandable that  $k$ th feature component of samples is not important enough and is removable. In the way, PCA can select important features for classification.

### 2.2.2 XG-BOOST

According to the paper [8], Xg-boost is known as eXtreme gradient boosting which is an implementation of gradient boosted decision trees designed for speed and per-



formance. Xg-boost is an algorithm used for supervised learning both regression and classification which supports parallel processing, whose goal is to optimize a cost function by composing the given equation(2),

$$\psi(\delta) = \sum_{i=1}^n d(a_i, \hat{a}_i) + \sum_{j=1}^j \beta(f_j) \quad (2.2)$$

The equation 2 represents  $d$  as a loss function which consist of predicted value  $a_i$  and the number of case  $n$  in the training set. There is a tree denotes as a  $f_j$  from the ensemble trees where  $j$  represents a number of generated tree. There is an another terms in the equation 2, that is a regularization term ( $\beta$ ) which is defined as,

$$\beta(f_j) = \varphi Z + \frac{1}{2} \left[ \xi \sum_{l=1}^Z |h_l| + \mathcal{L} \sum_{l=1}^Z h_l^2 \right] \quad (2.3)$$

In the equation 3,  $\varphi$  is denoted as minimum split loss reduction and  $\mathcal{L}$ , for regularization term in the weight  $h$ , associated to each leaf where T is represented as number of leaf. By the mentioned equation, XGBOOST algorithm gives classification results which explained in short, retrieve from the paper[4]. For more detail understanding of the algorithm with mathematical process, derivation of each equation of the process and implementation, the paper [4] can be viewed where the explanation of XGBOOST algorithm for classification is given.

## 2.3 Explainable Artificial Intelligence (XAI) Tools

### 2.3.1 ELI5

ELI5 represents the expression, "Explain Like I'm 5." As the name indicates, it means the ELI5 helps to troubleshoot AI classifiers and explain their top prediction through a straightforward, easy and great visualized way in Python so that explanation can be understood easily. It is a Python package that uses the interpreting-random-forest feature weights approach. Moreover, it permits to visualise and troubleshoot different ML models, for example, Scikit-learn, XGBoost, LightGBM, CatBoost, Sklearn-crfsuite, and Keras. Eli5 supports tree-based and other parametric linear models by showing weights for each feature to explain how effective the features are in contributing to get the final prediction decision on all parts of the tree. ELI5 provides an independent implementation of this algorithm for XGBoost and most scikit-learn tree entity which is definitely on the path towards model-agnostic interpretation but not purely model-agnostic like LIME [12]. The ELI5 prediction can be portrayed as the amount of the feature contribution + the mean given by the highest locale that covers the whole training set. We can use ELI5 in our model by showing weights for each feature which portrays the effectiveness of every features' contribution to the final prediction decision covering every tree including individual data-point prediction in an easily understandable way.

### 2.3.2 LIME

Lime, which stands for LOCAL INTERPRETABLE MODEL-AGNOSTIC EXPLANATIONS, is an XAI tool whose aim is to make humans understand the whole system by interpreting the underlying classification model for prediction. As an ideal model explainer, Lime does not only have Model Agnostic property, but also it provides local model interpretability, which means it can be used to explain any complex model locally by modifying a single data sample by twisting the feature values and observing the following impact on the output which make the model easier to understand globally. Lime explains the model in the following manner, retrieved from the paper[11] to make it understandable to the user,

$$E(x) = \text{Arg } \bar{a}_{p \in P} R(f, p, \gamma_x) + \xi(p) \quad (2.4)$$

In the equation(3),  $P$  is defined as a set of interpretable models and  $p$  acts over a domain  $\{0, 1\}^{d'}$  which is denoted as absent or present for interpretable components, which works to measure the complexity  $\xi(p)$  of the explanation  $p$ , mentioned in the paper[11]. For interpretation, model needs to be simple for which the complexity of the complexity  $\xi(g)$  needs to be as low as possible. In the both paper[11] and paper[3], it is mentioned that we need to minimize the portion  $R(f, p, \gamma_x)$  of the equation(3), as much as possible while having  $\xi(p)$  low enough to make the model understandable by humans where  $f(x)$  denotes the probability that  $x$  belongs to a certain class of the classification and  $\gamma_x$  is used as a proximity measurement which defines which defines the size of the measurement neighborhood around instance  $x$  which ensures both interpretability and local fidelity, retrieved from the paper[11]. In the way, the Lime produce explanation for any model which is summed up in short in equation(3).

### 2.3.3 SHAP

SHAP-Shapley Additive Explanations, as the name indicates, explains the prediction of the features by its additive properties, that is, computing the contribution of each feature to the difference between the actual prediction and the mean prediction. More specifically, SHAP explains the prediction by using some function to calculate the sum of Shapley values for all possible combinations of the inputs which ultimately guarantee a fair distribution of each feature [12]. The SHAP explanation technique figures Shapley esteems from coalitional game theory that reveals to us how to decently disseminate the prediction among the features. SHAP indicates the explanation as:

$$x(c') = \phi_0 + \sum_{j=1}^F \phi_j c'_j \quad (2.5)$$

In the equation(4),  $i$  is denoted as feature and  $\phi_j$  is its attribute,  $c'$  refers to coalition vector where  $c'_j = 1$  means feature is present and  $c'_j = 0$  means, feature is not present. The number of input, features are referred as  $F$  and  $x$  is denoted as the model explanation.

To compute shapley values, SHAP assumes some features values only i.e, features values playing that are present or absent. Efficiency, Symmetry, Dummy and Addi-

tivity properties are satisfied by Shapley values only.

$$x(c') = E_i(\hat{f}(i)) + \sum_{j=1}^F \phi_j \quad (2.6)$$

For local accuracy, SHAP indicates the explanation as equation(5) where  $c'_j = 1$  and  $\phi_0 = E_i(\hat{f}(i))$  as its Shapley efficiency property.

SHAP can be executed in Python to get a visualization tool for every feature with its significance. It works with tree-based models from the Scikit-learn bundle in Python too. SHAP takes the average of the magnitude of the SHAP values from the dataset to compute global feature significance. In this paper [11], SHAP XAI is applied on Solar PV Forecasting where the information about the contribution of every feature remaining visually concise, are provided by SHAP value either positively or negatively. Higher SHAP value of a feature affects higher either negatively or positively on the model output [11]. We can use this SHAP in our research for local interpretability of the models providing a partial dependence plot which will help to understand how the marginal effect of one or two features has on the predicted outcome of the model.

# Chapter 3

## Proposed Model

### 3.1 Work Plan

The purpose of the proposed IoT attack detection model is to identify the types of attacks and classify them based on their own characteristic and unique features or identification so that any IoT device or application can easily detect them to avoid serious vulnerabilities. Missing any one of the stages may lead to failing to format the model properly. After establishing the detection model, Explainable AI is required to evaluate the model. In order to form a successful model, some steps or processes need to be followed sequentially. In figure 3.1, the process of the proposal is exhibited.

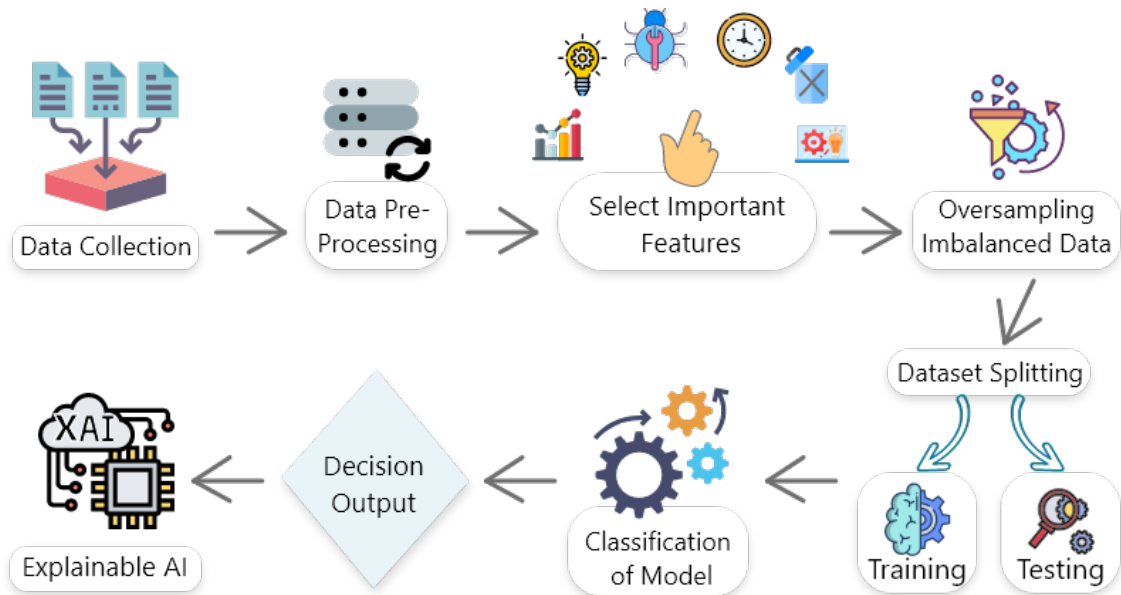


Figure 3.1: Proposed Model

IoT attack detection model needs five major steps for classification and producing prediction and they are-

- Data Collection and Description: This stage is concerned with collecting reliable data that needs to be suitable for our model and less intrusion to get

good accuracy and good prediction. 'IoT Intrusion Dataset 2020' dataset is used which is the collection of all IoT attack-related information compressed in CSV file.

- Data Processing Input Data: Raw data can not be used directly to the model for classification. It needs some preprocessing to fit with the model by replacing missing values, encoding some label features and dropping highly correlated redundant data by applying persona correlation algorithm.
- Data Selecting Important Features: This stage is important to identify those features which are more relevant and participate the most to improve the performance of the model and classification. PCA algorithm is used to select important features.
- Data Oversampling: This stage is required to compensate highly imbalanced dataset as it is one of the data analysis techniques that is used to adjust the class distribution of dataset.
- Training and Testing: After the preprocessing, the data needs to be split into training and testing data to fit with the model to minimize the effects of data discrepancies and understand the characteristics of the model better.
- Classification and prediction: This is the last stage which is done by using XG-BOOST classification algorithm for classifying attacks for detecting the IoT attack and giving the correct decision.
- Explaining with XAI Tools:

The process which is mentioned above is followed sequentially to get a good prediction for categorizing the attack correctly.

## 3.2 Dataset Collection and Description

Ullah and Mahmoud developed and analyzed a dataset, named IoTID20 for detecting malicious activity in the IoT network, that contains more than 52.56 million data which is collected from various types of resources such as smart home devices, IOT victim devices, laptops, smartphones, SKT NUGU (NU 100), EZVIZ Wi-Fi Camera and many more to generate IoTD20 dataset, mentioned in the paper[14]. The shape of the dataset is (625783, 84) which means there are 84 columns, 625783 rows in the dataset. Among the 84 columns, 3 column works as label or target output columns, named: Label, Category and Subcategory, shown in figure 3.2.

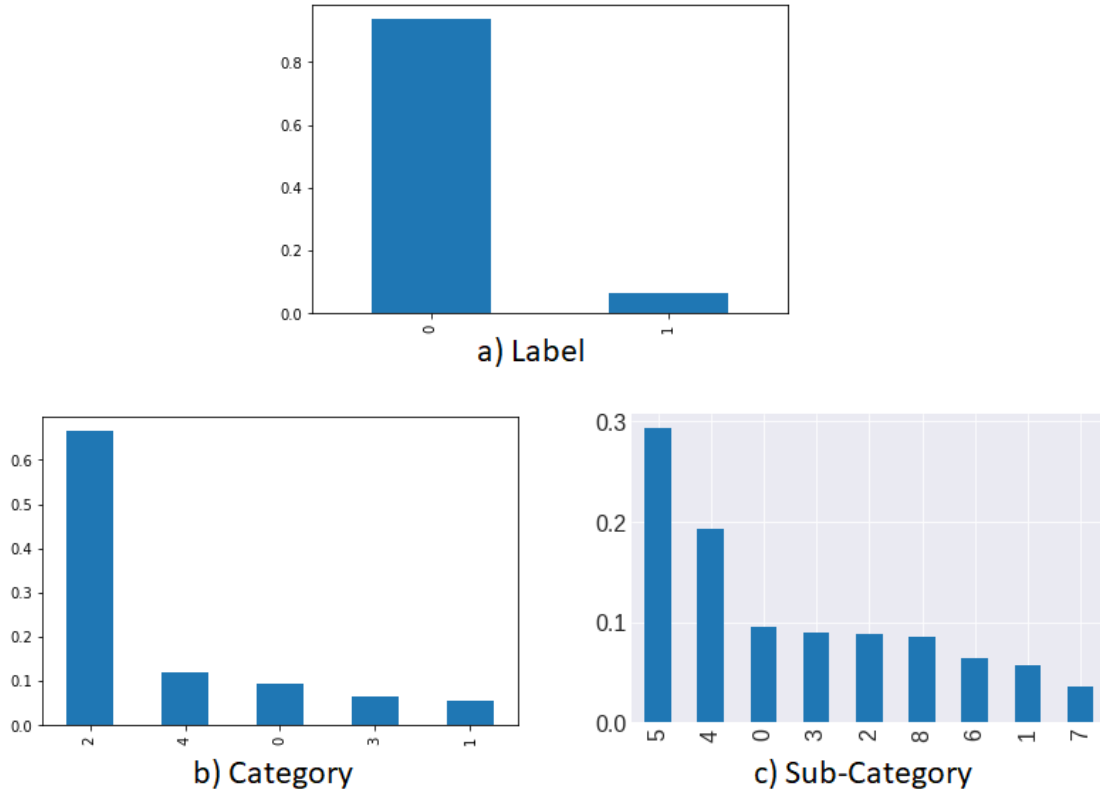


Figure 3.2: Classes of Target Features.

From the figure 3.2 The three target output column or labels consist of binary and multi-type attacks. The type of attack of each target output columns and their instances are represented in Table 3.1.

Label		Category		Sub-Category	
Type	Instance	Type	Instance	Type	Instance
DSF	585710	DSF	59391	DSF	59391
MAS	40073	MAS	35377	MAS	35377
-	-	MAF	415677	MAF	55124
-	-	MHF	40073	MHF	55818
-	-	MHB	75265	MHB	121181
-	-	-	-	MUF	183554
-	-	-	-	Normal	40073
-	-	-	-	SHP	22192
-	-	-	-	SPS	53073

Table 3.1: Attack types and Instances of Binary, Category and Sub-category in IoTID20 Dataset

The correlation of the dataset is also analysed in the section.

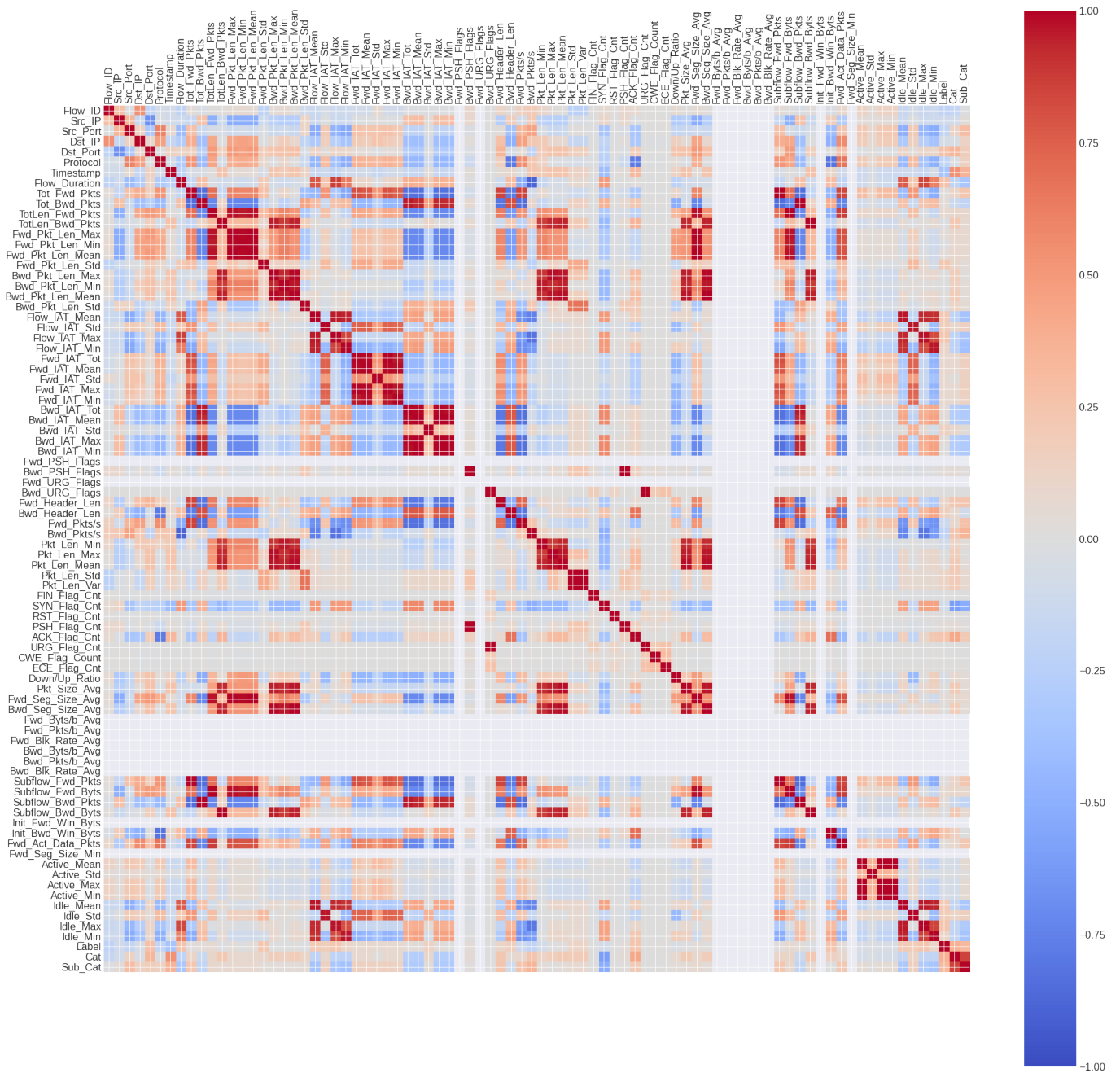


Figure 3.3: Correlation Matrix of IoTID20 Dataset.

The features of the dataset are correlated with each other, shown in fig 3.3. The highly correlated features are removable. Moreover, the dataset contains three types of data: numeric, string or object and categorical. There is also some missing value that needs to be preprocessed in the next stage.

The raw dataset can be downloaded from the given reference[14].

### 3.3 Data Preprocessing

Data processing is a required step in every project or research as data without proper processing can lead to misleading results. The dataset, which is used in our research contains 625783 rows x 86 columns who has more than 52.56million data in the dataset. Using more than 52.56 million data for fitting and evaluating without preprocessing, can be computationally expensive, difficult and time-consuming, that can lead to errors, over-fitting and wrong prediction. For this reason, data processing is required to clean data and make the data suitable for model. To process the dataset, first, we need to convert all object or string data type columns to integer data type columns as model can only take numerical or integer or float values for further processing and training testing. Moreover, target columns of object data type, are encoded using Label encoder. After applying label encoder in target columns, all the label or target features are convert to integer data type. Then, all data, which contains infinite value, is replaced with NAN value and the NAN value or missing data are filled up with row mean value. Then, the whole dataset is split into an independent columns, denoted as  $X$  and dependent columns, denoted as  $Y$  where the shape of independent columns  $X$  are (625783,81). The independent columns determine the outcome of the dependent or target columns, named: Label, Category and Sub-Category. In this way, the dataset[14] is processed for our model by changing the data type to integer, replacing the infinity data to NAN value and filling the missing data with row mean of the dataset so that the data can be used for further process of the model without any errors or interruption.

### 3.4 Feature Selection

Feature selection is the process of input variables which helps to reduce computational time and cost of modeling and improve prediction models.To analysis the features participation in general and make the model fast and effective, two method or algorithm have been used, explained in the section.

#### 3.4.1 Mutual Information Based Feature Analysis

Feature selection is a vital process for any classification model as the result of model highly depends on the features, that participate the most for classification. To increase the performance of the model, removing the unwanted feature is not enough. We need to find the most relevant ones to increase the performance of the model.

For this, `mutual_info_classify` algorithm is used to get the general idea of each features performance for analysis through the figure.



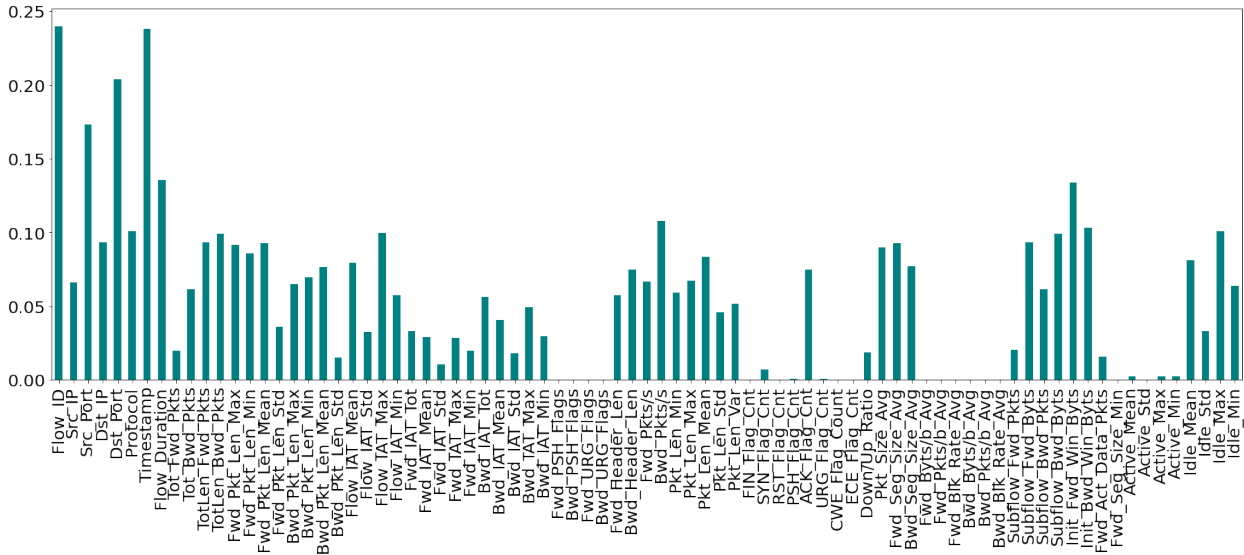


Figure 3.4: LABEL - feature importance by Mutual Information Classification Algorithm.

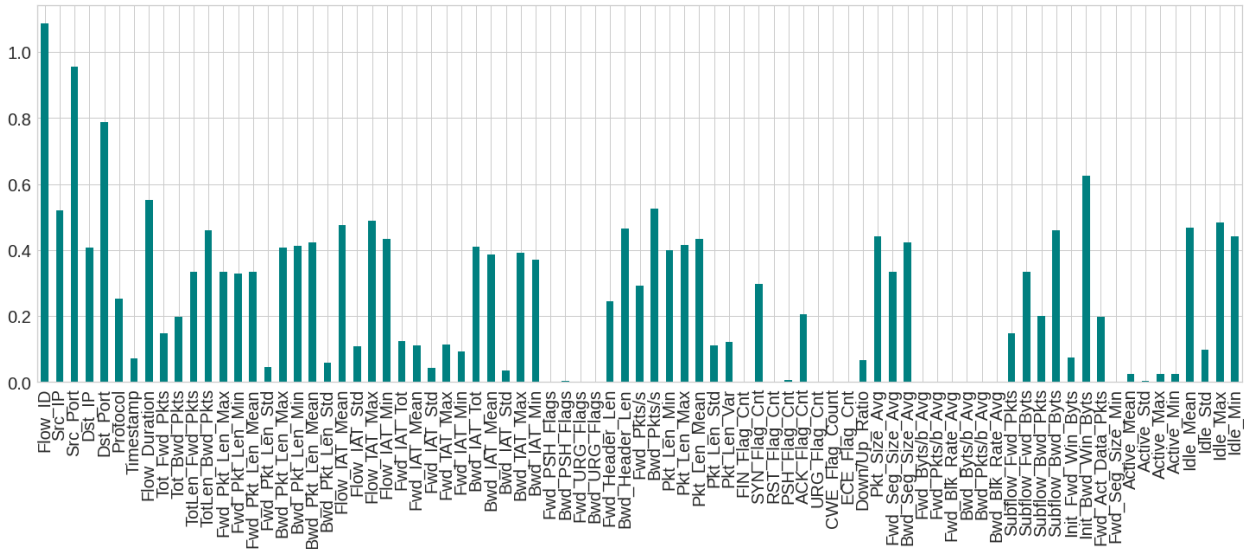


Figure 3.5: Category feature importance by Mutual Information Classification Algorithm..

From Figure 3.4, 3.5 and 3.6, we, not only have found some feature have no effect on the result for all three different target features or columns, but also has gotten a general idea about the contribution of each features which can be a good help to understand that we do not need all 81 independent features or column for classification, rather we need columns who has good influence on our prediction and more relative to our target output.

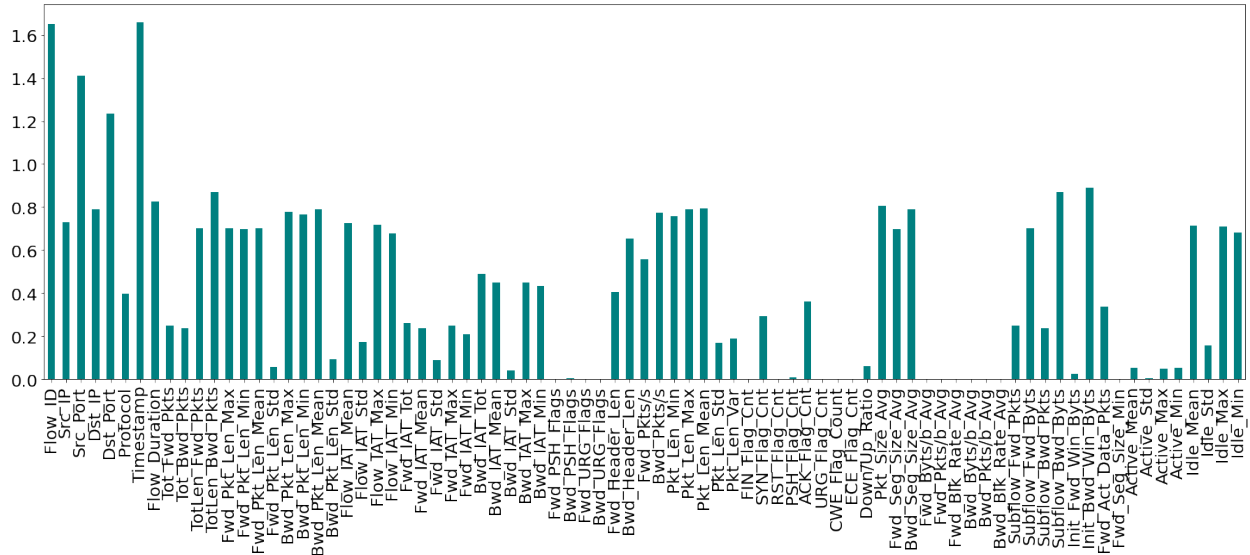


Figure 3.6: Sub-category feature importance by Mutual Information Classification Algorithm.

For this, we have decided to use PCA for feature selection algorithm that not only make the features uncorrelated but also make the algorithm select features in next part.

### 3.4.2 Applying Principal Component Analysis(PCA) for Feature Selection

PCA is a transformation model who has both the feature extraction and dimension reduction characteristics whose goal is to discover the direction of maximum variance in high dimension data and project the data fewer in to a new subspace who has the same or less dimension than the original one. In our proposed model, we have used PCA Algorithm in order to select important feature for reducing the training time by following some steps,

**Step1** we have standardized feature from dataset as PCA is effected by scale.

**Step2** Covariance matrix is calculated using the the samples from dataset to solve all the eigenvalues and eigenvectors where the eignevector, denoted as  $V$  are selected respectively  $V_1, V_2, \dots, V_n$  with respect to the first  $n$  largest eigenvalues.

**Step3**, the feature extraction results in equation1 is used to calculate the contribution of  $j$ th feature component retrieve from the dataset.

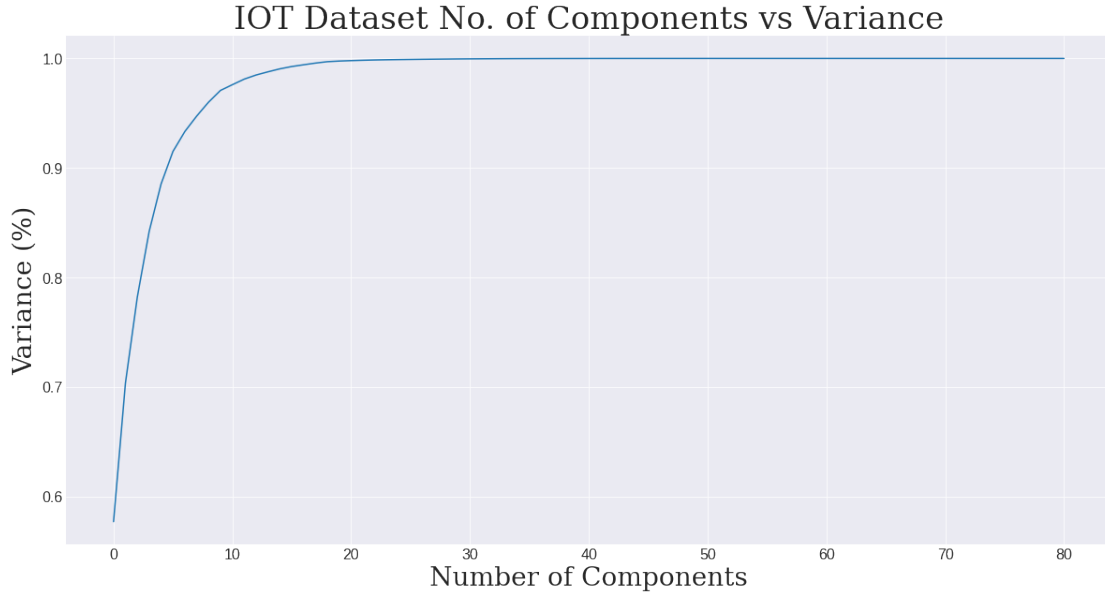


Figure 3.7: Iot dataset no of components vs variance

**Step4**,  $y_j$  is sorted in descending order and stored it into  $PC_j$ . As the dataset has multi-class, the model needs  $n$  dimensional features. We select as much PCA components as possible having within 99% of total variance. From the dataset, 14 feature is selected out of 81, whose results are  $PC_0, PC_1 \dots PC_{13}$ .

PC	Feature	Feature Names
PC0	43>16>18>58>17	<i>Bwd.Pkts/s, Bwd.Pkt.Len.Max, Bwd.Pkt.Len.Mean, Pkt.Size.Avg, Bwd.Pkt.Len.Min</i>
PC1	51>45>46>16>19	RST.Flag.Cnt, Bwd.IAT.Min, Fwd.PSH.Flags, Bwd.Pkt.Len.Max, Bwd.Pkt.Len.Std
PC2	51>4>70>48>0	RST.Flag.Cnt, Dst.Port, Subflow.Bwd.Byts, Pkt.Len.Var, Flow.ID
PC3	45>46>19>12>57	Bwd.IAT.Min, Fwd.PSH.Flags, Bwd.Pkt.Len.Std, Fwd.Pkt.Len.Max, Down/Up.Ratio
PC4	48>16>18>58>17	Pkt.Len.Var, Bwd.Pkt.Len.Max, Bwd.Pkt.Len.Mean, Pkt.Size.Avg, Bwd.Pkt.Len.Min
PC5	0>45>19>46>50	Flow.ID, Bwd.IAT.Min, Bwd.Pkt.Len.Std, Fwd.PSH.Flags, SYN.Flag.Cnt
PC6	19>48>13>2>42	Bwd.Pkt.Len.Std, Pkt.Len.Var, Fwd.Pkt.Len.Min, Src.Port, Fwd.Pkts/s
PC7	35>50>15>1>2	Bwd.IAT.Min, SYN.Flag.Cnt, Fwd.Pkt.Len.Std, Src.IP, Src.Port
PC8	6>50>35>48>4	Timestamp, SYN.Flag.Cnt, Bwd.IAT.Min, Pkt.Len.Var, Dst.Port
PC9	15>6>70>48>12	Fwd.Pkt.Len.Std, Timestamp, Subflow.Bwd.Byts, Pkt.Len.Var, Fwd.Pkt.Len.Max
PC10	1>70>5>0>19	Src.IP, Subflow.Bwd.Byts, Protocol, Flow.ID, Bwd.Pkt.Len.Std
PC11	15>12>51>6>43	Fwd.Pkt.Len.Std, Fwd.Pkt.Len.Max, RST.Flag.Cnt, Timestamp, Bwd.Pkts/s
PC13	1>4>19>48>16	Src.IP, Dst.Port, Bwd.Pkt.Len.Std, Pkt.Len.Var, Bwd.Pkt.Len.Max

Table 3.2: PCA Table with feature names.

**Step5**, We transform the 81 feature to 14 selected feature whose shape is (625783,14).

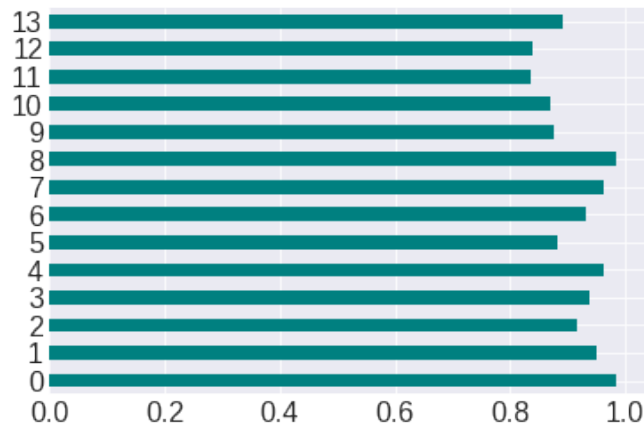


Figure 3.8: 14 selected feature by using PCA.

In this manner, 14 feature is selected for classification by using the data transformation, dimension reduction and feature extraction method of PCA algorithm.

## 3.5 Oversampling

Oversampling is a technique to duplicate samples from the minority class. Using oversampling is necessary as the imbalance class may effect our machine learning model by ignoring the minority class completely which can lead to false outcome or poor performance of the model. From the figure 3.2, it is evident that the dataset, we used in our detection model, is highly imbalanced as the distribution of data in the training dataset across the class is not equal, that can lead to a misleading prediction by thinking that the model is performing good.

### 3.5.1 Applying SMOTE for Oversampling

Synthetic Minority Oversampling ,known as SMOTE is a commonly used oversampling technique where the synthetic samples are generated observing the minority class as imbalance class creates bias that has the tendency to predict majority class. While increasing the number of data for minority class, it provides no new information to the model. Firstly, Smote chooses random data from minority class and set knearest neighbors from data. Then synthetic data is created between the random data and randomly selected k–nearest neighbor and, that procedure keeps repeating until the proposition between minority class and majority class is equal whose shape is (1598070, 14). For this reason, oversampling data is used as it helps our classification model for detecting the attacks of the model through correct prediction by avoiding the biasness towards majority class.

## 3.6 Training and Testing

Splitting Train data and test data are the two important concepts in machine learning for prediction where training data or set is used to train the model and testing

data or set is used to test the trained model. The train and test split are appropriate to use when the dataset is very large and the model needs a good estimated performance. The dataset [14] which is used for my model, consists of 52.58million data where it has 84 columns and 625783 rows which is large enough to split the data into train and test data to get a good performance from the model. In the proposed model, the data is split into two sets where 25 data is used for the test and 75 data is used for the train. In previous stages, we have divided the whole dataset into input vector  $X$  and output vector  $Y$  which is enough to split the data for training and testing. After splitting the dataset into training and testing, train shape of input vector  $x$  is (1198552, 14) and test shape of  $X$  is (399518, 14). On the other hand, for output vector  $Y$ , train shape is (1198552,) and test shape is (399518,). In this way, the data is split into train and test according to the given ration to fit with the model for prediction.

### 3.7 Classification of Models

A classification model refers to a model that tries to predict one or more outcomes by observing certain values from the dataset. In supervised learning, training data is fed into a classification algorithm and compared train data with test data. By the pattern of the data, it will fall into the predetermined category which falls under classification. For detecting the IoT attacks, a good classification model is needed for classifying the attack accurately. In our model, two classification models or algorithms are used to identify which algorithm works best for the model and classy more accurately.

#### 3.7.1 Applying XGBoost for Classification

In our proposed model, the XGBoost algorithm is used for classification to identify the types of attacks. For classification, XGBoost follows certain process, shown in figure 3.9,

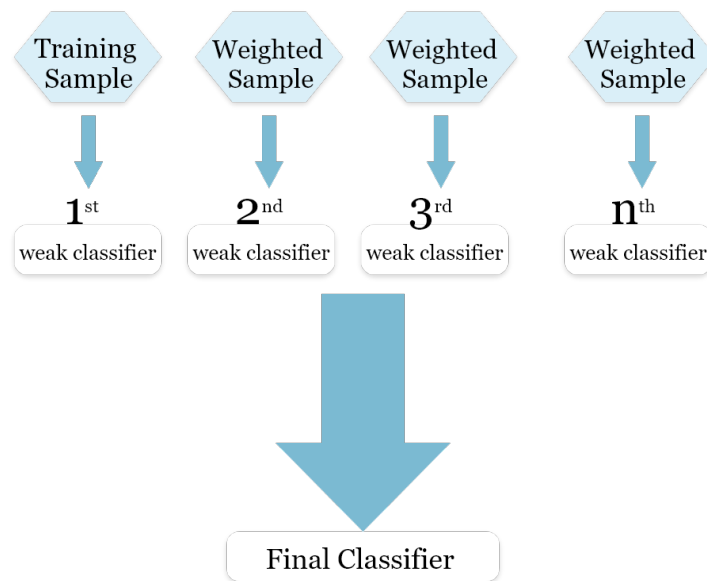


Figure 3.9: Label Features.

In this algorithm, decision trees are created sequentially where weights ( $w$ ) are assigned with independent variables, to fed them into the decision tree. In our model, to predict results, weighted variables which are predicted wrong by the tree, are fed into the second decision tree to achieve a strong and more precise prediction for classifying the attacks.

From the detail explanation of XGBOOST algorithm in researched paper[8], we are enlightened that XGBOOST has certain key parameters to perform and few of them, are selected to maximize the system accuracy by using grid search through testing the combination of all set of values of relevant parameters, shown in the Table 3.3,

Parameter	Definition	Range	Default	Grid Values	Optimal
Parameter	Definition	Range	Default	Grid Values	Optimal
eta	Learning rate	[1, $\infty$ ]	10	0.001, 0.01, 0.1	0.1
max depth	Maximum Depth of a Tree	[0, $\infty$ ]	6	5, 10, 12	12
Subsample	Ration of random samples to be consider for training	[0, 1]	1	0.5, 0.6, 0.7, 0.8	0.8
min child weight	Minimum Observation for a child	[0, $\infty$ ]	1	0.1	1
n.estimators	Number of trees to be used in the forest	[1, $\infty$ ]	10	10, 20, 50, 100	10

Table 3.3: Grid values and Optimal parameters Of XGBOOST Algorithm.

To classify the model using XGBOOST, large number of max depth is chosen which is 12 for the large number of considered features and each feature's values for the tree construction. In the model, the best weighing factor  $\eta$  is taken 0.2 in order to avoid overfitting as  $eta(\eta)$  defines as a slow learning rate for the correction of new trees added to the model. For getting a good accuracy, the n\_estimators is set 10 to specify how many sequential tree we need to create, that attempt to correct the trees in each iteration and colsample\_bytree is set 0.8 for using each feature per tree in percentage. The subsample value is set 0.8 which uses all the training set to construct the output classification model. The three target features use XGBOOST algorithm for classification where Label hold 2 type attack, Category hold 5 type attack and Sub-category hold 9 type attack which is also known as class. for this reason, we have used binary objective for Label feature and multi:softprob objective for multi-class Category and sub-category feature or column. In the way, The model has improved significantly by setting the optimal values in each parameter and using parameter tuning that gives 100% accuracy for Label feature, 100% accuracy for Category feature and 86% accuracy for Sub-Category feature which proves that the model can detect attacks really well using the XGBOOST algorithm as the overall classification accuracy of the detection model is 95% on average.

### 3.7.2 Applying Cross-Validation

We will be using K-Fold Cross-Validation. K-Fold CV is a resampling procedure without substitution which is utilized for hyperparameter tuning to such an extent that the model with the most ideal worth of hyperparameters can be prepared. Then cross validation is implemented on the trained data. At first k-1 folds are created for

trained data and the rest of them is created for testing data. This cycle is performed again and again for K times. The model performance is determined for a specific set of hyper-parameters by taking the mean and standard deviation of all the K models made. The hyperparameters giving the most ideal model is determined. At long last, the model is prepared again on the trained data utilizing the most ideal hyperparameter and the speculation execution is figured by ascertaining model execution on the test dataset.

## 3.8 Implementation XAI Tools On Model

In the paper, an Explainable AI method or technique is used on the IoT attack detection model to make the results or output understandable for humans by showcasing the underlying details of the model. In the model, XG-BOOST is used to train the data from the dataset for classification where the model is trained with 15 estimators tuned with the hyper-parameter of the model to achieve maximum accuracy for three different label features: Label(100), Category(100) and Sub-Category(86) which are the decent results for the model. However, XAI tools, like LIME, SHAP and ELI5, are used for the model interpretation where Lime explains the contribution of each feature on the prediction, SHAP summarizes the combination of feature importance with the feature effects and ELI5 shows the weight of each feature who has the contribution for prediction.

### 3.8.1 Applying LIME To IoT Attack Detection Model

Local Interpretable Model-Agnostic Explanations, in short Lime is used to make an attempt to understand the complex IoT attack detection machine learning model as it has the ability to explain all sort of models for which prediction probability has obtained. As an ideal model explainer, Lime does not only have Model Agnostic property but also it provides local model interpretability which means it can be used to explain our established detection model locally by modifying a single data sample by twisting the feature values and observing the following impact on the output which make the model easier to understand globally. Some techniques are followed for performing Lime Algorithm in our IoT attack detection model to visualize explanation of individual predictions. At first, fake data is created around the observation for building a local linear model, known as perturbed data. Perturbed data is a hyperparameter, where number of samples are created to explain the model which will have 0 or 1 values for each categorical data by picking up the random values based on the occurrence frequency of categorical value in the training dataset, where 1 denotes categorical value is same as the observation to be explained and 0 denotes the reverse characteristic of 1. It gives the decision between 0 and 1 for each attack or category of Label features or categorical columns. Then, the prediction function is used to make prediction on perturbed data to train the local linear model which is denoted as  $\Omega(g)$  in equation(3). Next, the features are decentralized by converting continuous variable to discrete to make the format human understandable. After that, the euclidean distance is identified by comparing between each data point in the perturbed data with the original datapoint that has given an idea how far the point is away from original observation. The euclidean

distance is denoted by  $\mathcal{L}(f, g, \pi_x)$  in equation 3 for pointing out either the distance is smaller or higher with 0 and 1 as it is converted with the value between 0 and 1 which identify either it is closest to the observation or exactly the same as the observation. As Lime uses standard feature selection technique like highest weight, forward selection, lasso path etc to select top  $n$  features from the 14 selected features that have influenced on prediction. At the end, Local linear model is created by Lime with all input data to explain the local behaviour through observation, using the weights from the classification model. In the way, Lime is applied in three label feature, Label, Category and sub-category to interpret it into human understandable format.

### 3.8.2 Applying SHAP To IoT Attack Detection Model

SHAP (Shapley Additive exPlanations) is one of the most advanced methods used in XG-boost for local explanation and consistency in global model structure based on game theory. For the local interpretability, we have applied SHAP in our model. We have used TreeSHAP to determine the impacts of the features with its conditional expectation  $E_{i_n|i_x}(f(i)|i_x)$  where  $i$  is an instance,  $x$  is the feature subset. As SHAP has additive property, the TreeSHAP average the predictions, weighted by node sizes, and the mean of the Shapley value nodes is the prediction for the instance  $i$  given the subset  $x$ . The prediction starts from the baseline which is the average of all predictions for Shapley values. According to these predictions, all features show different trends for all selected points where, the color red with arrow direction (to higher) shows the feature values that cause increased prediction. The color blue with arrow direction (to lower) shows the feature values that cause decreased prediction and the magnitude of the feature's importance showed through their visual size. Moreover, we will be using SHAP feature importance which is going to present the features according to their importance through the size of absolute Shapley values. To do so, first, we need to add the absolute Shapley values  $\phi_j$  per feature. After that, we need to sort the features by decrementing their importance and lastly, plot them. For example, in Sub-Category, Feature-8 is the most important feature. With this Feature importance plot, we will only get to know which feature is more and less important but no other information beyond that. That is why we will be going to use Summary Plot, to gain more information about the relationship between feature values and their effects along with their importance. After that, Summary plot will be used. The points on the summary plot determine the Shapley value for a feature and an instance where X-axis represents SHAP value (impact on model output) and Y-axis as Feature values. The color red to blue determines the value of features as high to low. The feature elements are plotted according to their significance Covering points are jittered in the y-pivot position, so we get to understand the appropriation of the Shapley esteems per feature. In these ways, we will be applying SHAP in three label features which are- Label, Category, and Sub-Category for getting an efficient explanation of the predictions.



### 3.8.3 Applying ELI5 To IoT Attack Detection Model

ELI5-”Explain Like I’m 5” helps to resolve AI classifiers and explain their predictions in an easy way so that it can be human readable. ELI5 uses the out-of-box feature importance computation methods for prediction to display the estimator’s global weights as HTML table, text, etc. We will use an interpretation table that gives us weight associated with each feature as Feature importance without any direction. The color green shows the most important feature and red shows the least one. Which feature is an important feature and how it is impacting (decreasing or increasing) on the model can’t be said due to not having any sign with their weight. That’s why we have used *explain\_prediction* to explain how these features are impacting the model. We have used the *explain\_prediction* method of ELI5 so that we get a clear understanding of the contribution positively or negatively of each feature in predicting the output. We will be taking less value initially, then we will increase the value gradually to see how it works with a bigger set of features as the feature importance does not stay the same within. Then the *show\_prediction* method will be used to understand how the XGBoost classifier arrived at a prediction. `show_prediction` accepts all functions such as, `explain_prediction` arguments and all `format_as_html` keyword arguments, to get explanation and customize formatting in a single call. There are three columns for each attack that shows how different features contribute to the prediction of the dataset and their values respectively. If we sum up the values of individual features and BIAS in each column, the highest one will be class predicted by the model. The score is the sum of the contribution of each feature and the bias. From its table we will get to know the contribution and value of each feature that will represent the highest to lowest predicted features. This how ELI5 has been used to explain the predictions in straight forward way in three label-feature, Label, Category and sub-category.

# Chapter 4

## Performance Evaluation

### 4.1 XG-Boost Output Analysis

In the paper, XGB classification algorithms have been used on IoT attack detection to predict the attack accurately. After that, K-fold CV is implemented on 3 label features – Label, Cat, Sub-Cat so that we can understand the dataset is overfitted or not. Moreover, K-fold CV algorithm is used to test the performance ability of the model for predicting the attacks.

In order to evaluate the performance of the proposed model, IoT Intrusion Dataset 2020 [14] is used by implementing python in Google Colaboratory. After running the model in Google Colaboratory, the results of the classification of data are obtained. In order to verify the performance, we have to understand some factors. They are,

**Classification Metrics** is performance metrics for evaluation and measure the classification model's prediction quality by identifying true positive, true negative, false positive and false negative of prediction.

**Confusion Matrix** is used to evaluate the prediction of classification model by summarizing the number of correct and incorrect prediction through calculation using the following factors,

- True Positive: The model predicted it positive and it is true. The diagonal values of confused matrix is True positive for the corresponding attacks.
- True Negative: The model shows Prediction is negative and it is true which is calculated by the sum of all columns and rows excluding the values of that class's column and row.
- False Positive: The model prediction is positive and it is false. which is calculated by the sum of all values of the corresponding column of particular attack, excluding the True Positive value of that attack from the confusion matrix.
- False Negative: The model shows it negative but actual prediction is positive and, for a particular class, it is calculated by the sum of values in the corresponding row of the attack, excluding the True Positive of that particular attack.

True positive of each class can be identified easily from the confusion matrix figure 4.1,4.2,4.3 of Label, Category and sub-category. On the other hand , for determining the True negative, False negative and False positive of each attacks, the data representation table 4.6, 4.4,4.2 need to be used for Label, Category and Sun-category target columns by following the described methods, mentioned in the confused matrix description section.

**Precision** is the ration of true positive for all the positive prediction of our proposed model.

**Recall** is the ration of true positive of all the positive of our used dataset IoTID20.

**F1 score** identifies the percentage of positive prediction are correct. It is a weighted harmonic mean of recall and precision.

**Support** quantifies the number of correct positive prediction, made by the classification model.

After performing the Xg-boost classification algorithm for the proposed model, some performance metrics and confused metrics has obtained:

#### 4.1.1 Label feature

For Label feature, IoT attack detection models performance measure and insight view is exhibited through the classification metric table and confusion matrix output:

<b>Attack</b>	<b>Precision</b>	<b>Recall</b>	<b>f1-score</b>	<b>support</b>
<b>Anomaly</b>	1.00	1.00	1.00	117149
<b>Normal</b>	1.00	1.00	1.00	117135
-	-	-	-	-
<b>accuracy</b>	-	-	1.00	234284
<b>macro avg</b>	1.00	1.00	1.00	234284
<b>weighted avg</b>	1.00	1.00	1.00	234284

Table 4.1: LABEL Classification report of the proposed method.

From table 4.1 we can see that, for both Anomaly and Normal attack class, how many features were classified in relation to the amount of positive prediction is calculated and the result is averaged as 100%. Then, in relation to the total amount of prediction its calculated in Recall and the result averaged as 100%. As both Precision and Recall gives the same value, f1-score is automatically gives the same value. The value of averaged Support determines the number of correct positive prediction 234284 which is made by the classification model.

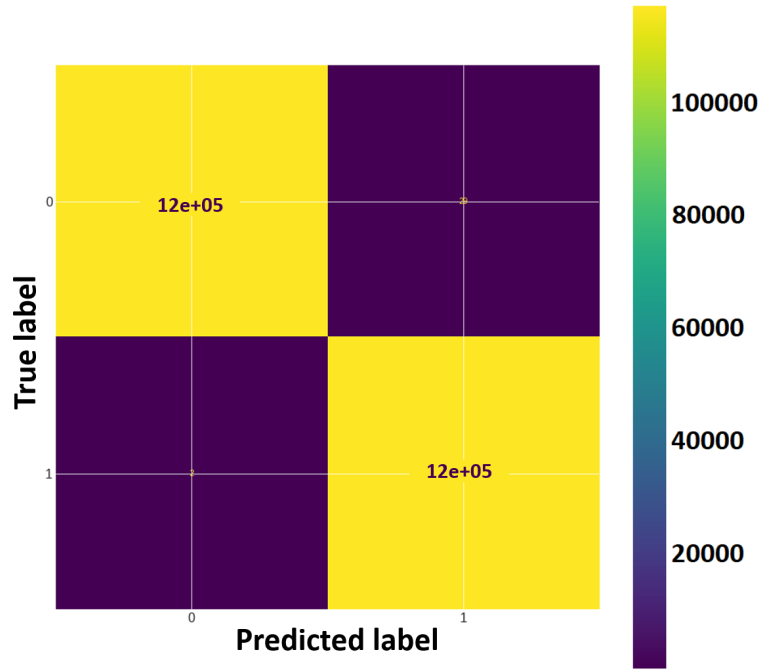


Figure 4.1: LABEL - confusion matrix.

	Attack	Anomaly	Normal
Attack	117120	29	
Anomaly	2	117133	

Table 4.2: The Data Representation of Confusion Matrix for Label target Column.

The accuracy of the model is **100%** for detecting attacks of Label Target columns.  
**After k-Fold CV:** Accuracy of the model is **99.99%** for detecting attacks of Label Target columns.

### 4.1.2 Category feature

For Category feature, IoT attack detection models performance measure and insight view is exhibited through the classification metric table and confusion matrix:

Attack	Precision	Recall	f1-score	support
DSF	1.00	1.00	1.00	104070
MAS	1.00	1.00	1.00	103800
Mirai	1.00	1.00	1.00	104034
Normal	1.00	1.00	1.00	104073
Scan	1.00	1.00	1.00	103617
-	-	-	-	-
accuracy	-	-	1.00	519594
macro avg	1.00	1.00	1.00	519594
weighted avg	1.00	1.00	1.00	519594

Table 4.3: CAT - Classification report of the proposed method.

From table 4.3 we can see that, for all the attack class (DSF, MAS, Mirai, Normal, Scan), how many features were classified in relation to the amount of positive prediction is calculated and the result is averaged as 100%. Then, in relation to the total amount of prediction its calculated in Recall and the result averaged as 100%. As both Precision and Recall gives the same value, f1-score is automatically gives the same value. The value of averaged Support determines the number of correct positive prediction 519594 which is made by the classification model.

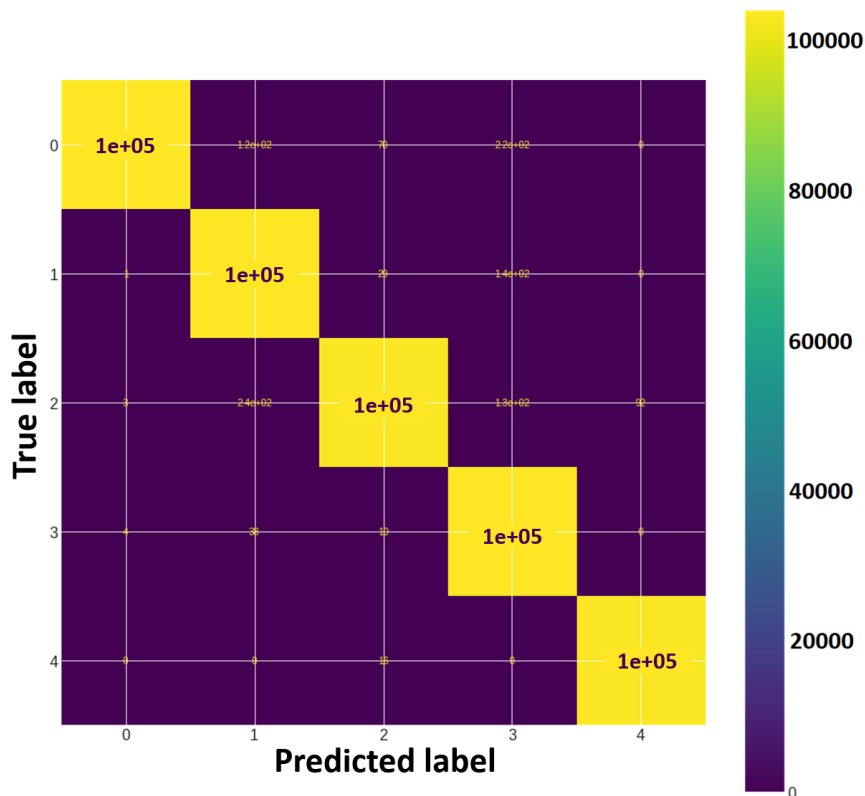


Figure 4.2: CAT - confusion matrix.

Attack	DSF	MAS	Mirai	Normal	Scan
DSF	103664	118	70	218	0
MAS	1	103634	29	136	0
Mirai	3	236	103571	132	92
Normal	4	38	10	104021	0
Scan	0	0	16	0	103601

Table 4.4: The Data Representation of Confusion Matrix for Category Target Column.

The accuracy of the model is **100%** for detecting attacks of Category Target columns. **After k-Fold CV:** Accuracy of the model is **99.7%** for detecting attacks of Label Target columns.

### 4.1.3 Sub-Category feature

For Sub-Category feature, IoT attack detection models performance measure and insight view is exhibited through the classification metric table and confusion matrix:

Attack	Precision	Recall	f1-score	support
DSF	1.00	1.00	1.00	45749
MAS	1.00	1.00	1.00	45823
MAF	0.43	0.72	0.54	39321
MHF	0.44	0.29	0.35	39376
MSB	1.00	1.00	1.00	45708
MUF	1.00	0.70	0.82	45801
Normal	1.00	1.00	1.00	45864
SHP	0.94	0.96	0.95	45878
SPS	0.96	0.94	0.95	45949
-	-	-	-	-
accuracy	-	-	0.86	399469
macro avg	0.86	0.84	0.84	399469
weighted avg	0.88	0.86	0.86	399469

Table 4.5: Sub-Cat - Classification report of the proposed method.

From table 4.5 we can see that, for all the attack class (DSF, MAS, MAF, MHF, MHB, MUF, Normal, SHP, SPS ), how many features were classified in relation to the amount of positive prediction is calculated and the result is averaged as 88%. Then, in relation to the total amount of prediction its calculated in Recall and the result averaged as 86%. F1-score is averaged and gives the value of 86% The value

of averaged Support determines the number of correct positive prediction 399469 which is made by the classification model.

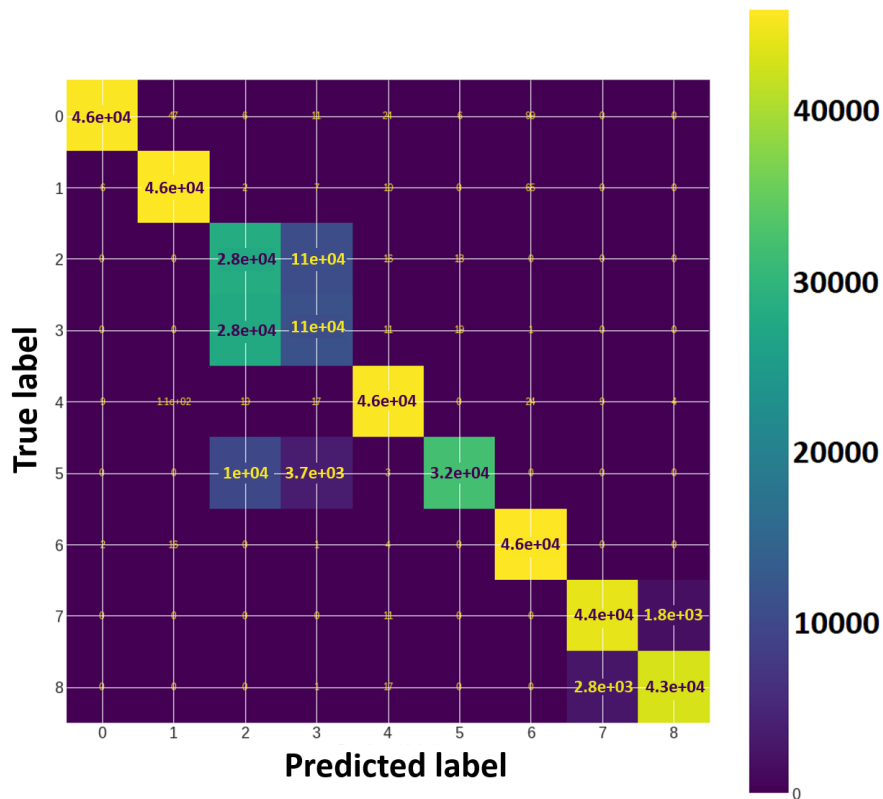


Figure 4.3: sub cat - confusion matrix.

Attack	DSF	MAS	MAF	MHF	MHB	MUF	Normal	SHP	SPS
DOS	45643	39	3	0	38	0	83	0	0
MAS	9	45968	2	4	7	0	42	0	0
MAF	0	0	31063	8185	15	17	0	0	0
MHF	0	0	30315	9028	10	15	3	0	1
MHB	5	94	14	19	45338	1	15	8	5
MUF	0	0	10789	2804	2	32088	0	0	0
Normal	7	18	0	0	3	0	45951	0	0
SHP	0	0	0	0	7	0	0	44183	1563
SPS	0	0	0	0	22	0	0	3082	43013

Table 4.6: The Data Representation of Confusion Matrix for Sub-Category Target Column.

The accuracy of the model is **86%** for detecting attacks of Sub-category Target columns.

**After k-Fold CV:** Accuracy of the model is **85.4%** for detecting attacks of Label Target columns.

## 4.2 XAI-Tools result analysis

Based on the accuracy result only, it is difficult to decide the model's standard. Due to this, We have chosen to implement three different XAI tools on model to observe each output corresponding to their input features for evaluating it's ability to detect attacks.

### 4.2.1 Sub-category

At first Sub-category target output column is chosen for applying Eli5,LIME and SHAP xai tools for experiment to show feature importance and performance in model for evaluation.

#### ELI5

In figure 4.4, Eli5 has shown the average weights of 14 feature of classification machine learning model by using the feature importance computation method which is the average gain of the feature, used in trees.

```
eli5.show_weights(xgb_estimator1)
```

Weight	Feature
0.1660	f4
0.1391	f7
0.1267	f8
0.0853	f2
0.0671	f11
0.0620	f1
0.0560	f13
0.0489	f5
0.0481	f12
0.0474	f0
0.0451	f3
0.0429	f9
0.0362	f6
0.0293	f10

Figure 4.4: Weight of each features for sub-category using ELI5.

This is an interpretation table 4.4 that gives us weight associated with each feature as Feature importance without any direction by using `show_weights`. The table 4.4 shows each features weight that is calculated by overall performance of each feature of the model. The color green shows the most important feature and gradually it becomes lighter color to show the lesser important features. Here, we can see feature number 4 is the most important feature which refers to `Dst_Port`. By using weights only in figure 4.4, it is difficult to understand each feature's contribution for individual class for prediction. For this, we have used `showprediction` library function of ELI5 to show the features contribution for individual class or attack.



y=0 (probability 0.070, score -0.545) top features			y=1 (probability 0.070, score -0.545) top features			y=2 (probability 0.071, score -0.540) top features		
Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value
+0.008	<BIAS>	1.000	+0.081	x3	-0.851	+0.288	x11	0.219
-0.000	x2	-0.310	+0.007	<BIAS>	1.000	-0.000	x3	-0.851
-0.000	x1	0.999	-0.000	x4	0.134	-0.002	x4	0.134
-0.000	x5	0.137	-0.000	x13	0.054	-0.014	x5	0.137
-0.000	x12	0.043	-0.000	x12	0.043	-0.016	x10	0.089
-0.000	x6	-0.033	-0.000	x2	-0.310	-0.034	x12	0.043
-0.001	x11	0.219	-0.001	x6	-0.033	-0.034	x13	0.054
-0.002	x10	0.089	-0.005	x10	0.089	-0.036	x0	1.210
-0.002	x13	0.054	-0.006	x5	0.137	-0.038	x9	0.170
-0.003	x8	0.267	-0.011	x1	0.999	-0.041	<BIAS>	1.000
-0.005	x9	0.170	-0.092	x11	0.219	-0.053	x2	-0.310
-0.007	x7	-0.209	-0.126	x7	-0.209	-0.090	x6	-0.033
-0.106	x0	1.210	-0.148	x9	0.170	-0.095	x7	-0.209
-0.428	x4	0.134	-0.245	x8	0.267	-0.116	x8	0.267
						-0.257	x1	0.999

y=3 (probability 0.070, score -0.544) top features			y=4 (probability 0.077, score -0.453) top features			y=5 (probability 0.070, score -0.545) top features		
Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value
+0.413	x11	0.219	+0.539	x11	0.219	+0.117	x11	0.219
+0.011	x4	0.134	+0.244	x2	-0.310	+0.020	x3	-0.851
-0.000	x3	-0.851	+0.079	x12	0.043	+0.018	<BIAS>	1.000
-0.001	x12	0.043	+0.048	x1	0.999	+0.009	x4	0.134
-0.010	x10	0.089	+0.011	<BIAS>	1.000	-0.000	x12	0.043
-0.035	x2	-0.310	-0.019	x0	1.210	-0.000	x13	0.054
-0.037	x13	0.054	-0.040	x5	0.137	-0.003	x10	0.089
-0.039	x0	1.210	-0.062	x10	0.089	-0.023	x9	0.170
-0.042	<BIAS>	1.000	-0.116	x6	-0.033	-0.032	x0	1.210
-0.073	x5	0.137	-0.118	x3	-0.851	-0.043	x7	-0.209
-0.114	x9	0.170	-0.165	x4	0.134	-0.054	x5	0.137
-0.130	x8	0.267	-0.173	x13	0.054	-0.099	x1	0.999
-0.146	x6	-0.033	-0.204	x7	-0.209	-0.156	x8	0.267
-0.165	x7	-0.209	-0.204	x9	0.170	-0.300	x2	-0.310
-0.177	x1	0.999	-0.274	x8	0.267			

y=6 (probability 0.070, score -0.546) top features			y=7 (probability 0.121, score 0.002) top features			y=8 (probability 0.380, score 1.142) top features		
Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value	Contribution <sup>?</sup>	Feature	Value
+0.007	<BIAS>	1.000	+0.897	x7	-0.209	+0.819	x8	0.267
-0.000	x12	0.043	+0.508	x5	0.137	+0.398	x7	-0.209
-0.000	x13	0.054	+0.135	x9	0.170	+0.282	x10	0.089
-0.000	x3	-0.851	+0.109	x3	-0.851	+0.258	x5	0.137
-0.000	x2	-0.310	+0.094	x2	-0.310	+0.139	x9	0.170
-0.000	x6	-0.033	+0.093	x13	0.054	+0.131	x3	-0.851
-0.001	x5	0.137	+0.079	x6	-0.033	+0.108	x2	-0.310
-0.001	x10	0.089	+0.057	x1	0.999	+0.088	x13	0.054
-0.003	x1	0.999	+0.013	<BIAS>	1.000	+0.011	<BIAS>	1.000
-0.003	x11	0.219	-0.110	x0	1.210	-0.002	x0	1.210
-0.013	x9	0.170	-0.185	x8	0.267	-0.011	x1	0.999
-0.126	x7	-0.209	-0.303	x4	0.134	-0.049	x6	-0.033
-0.407	x8	0.267	-0.364	x10	0.089	-0.088	x12	0.043
			-0.366	x12	0.043	-0.379	x4	0.134
			-0.655	x11	0.219	-0.563	x11	0.219

Figure 4.5: Feature contribution analysis of Sub-category by ELI5 for the classification using 399520 test data input

In figure 4.5, we have used Eli5 show\_prediction to give the clear view of weights and score of contributed individual features for making a prediction of particular class by taking **399520** test data as an input conservatively where score is calculated by the sum of all participated feature. Here, 9 attacks are represented as  $y$  from 0 to 8 respectively in Sub-Category target columns accordingly. The color green shows the most important feature and red shows the least one. The prediction of each attack is the sum of positive features included bias and the score of each class is the sum of positive and negative feature including bias. For instance, in  $y=1$  (MAS-attack), the most important feature is  $x3$  that represents Dst\_IP with highest positive contribution in the prediction and the class 8, which is Scan Port OS attack has the highest probability for test input 399520. Moreover, the features which may good for predicting one class, may not have a good influence for another class. For example in (4.5), class 4, feature  $x11$  works as a top positive influenced feature. On the other hand, same feature works as a negative influenced one for class 7. Overall, we get a idea of positive and negative impact of features for individual attacks using ELI5.

## LIME

Lime shows each features contribution of the prediction values where one color in right side shows positive impact and another color in left shows negative impact of that feature on target and each class gives two decision.

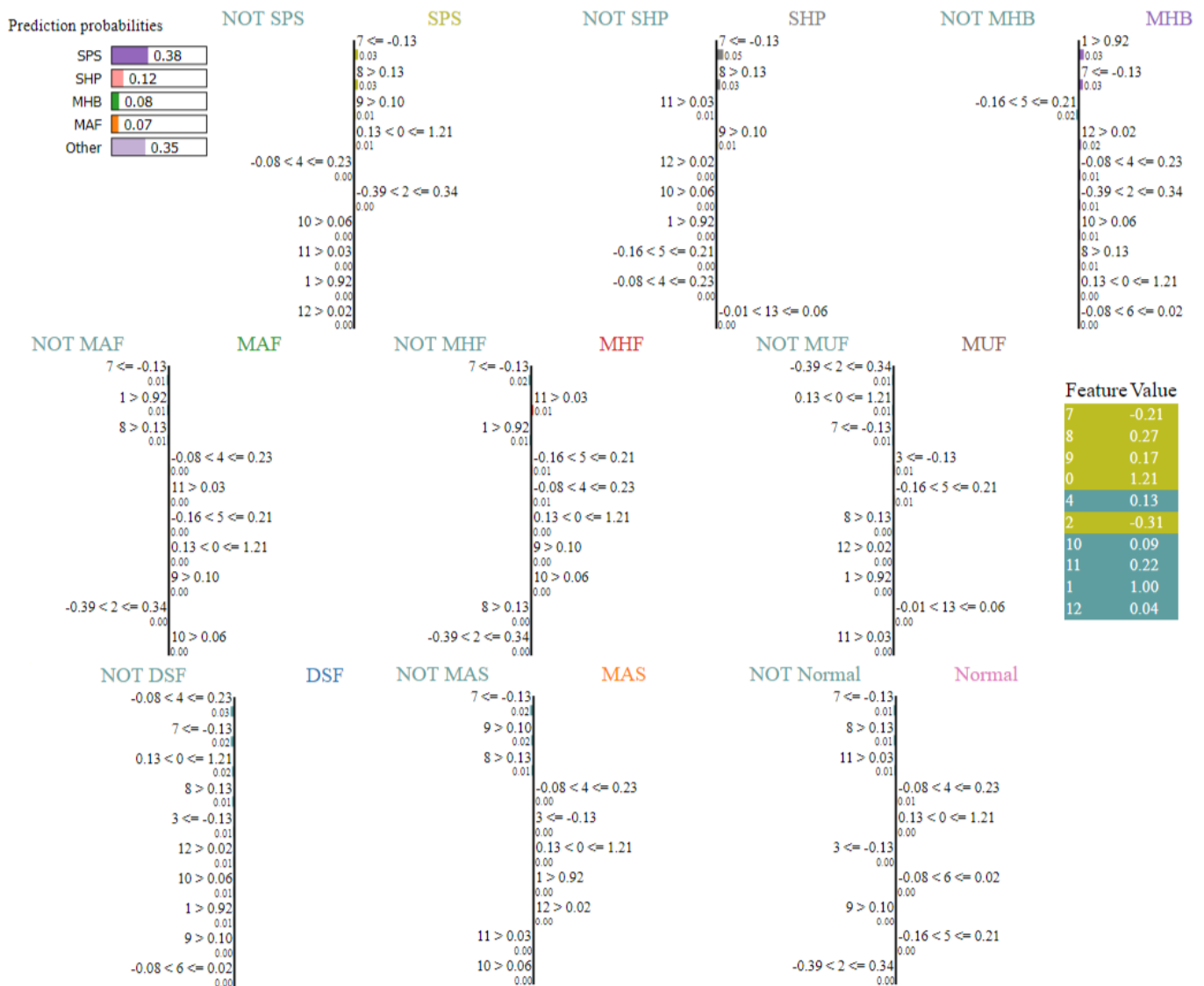


Figure 4.6: The output of Sub-category for IoT attack detection using Lime with 399520 Test data input.

From the figure 4.6, The Lime gives decision using three parts. The left side shows the prediction probability of each class for the model using a fixed input data, the middle part of the output shows two decision of each class where left side shows negative feature contribution and right side shows positive feature contribution and the contribution of each feature for particular class is sorted with high to low, most influenced features to least influenced feature and, at the right most part the feature value table is given which shows the value of each features impact corresponding to the attack of  $y_{test}[399520]$ , which is 8 means SPS.

For example, the attack corresponding the test value 10 feature has influence for predicting the attack SPS where the right side, Green color is for positive contributed features and the left side, Teat color is for negative influenced feature.

Moreover, the top most influenced feature is X7(Flow\_Duration) which is in the positive side of the attack section and all the related feature's value for the class is also given in table, name feature value. In another word, we can also say, the first column represents the prediction probabilities of attacks where SPS is the predicted by the classifiers. The second column shows features' contributions to the probability where feature number 7,8,9 (Flow\_Duration, Tot\_Fwd\_Pkts, Tot\_Bwd\_Pkts) for SPS attack are the most important feature impacting positively and 11,1,12 (TotLen\_Bwd\_Pkts, Src\_IP, Fwd\_Pkt\_Len\_Max)as least important feature. The third column displays the original data values from where we can see features with the positive and negative impact by colors as Green and Teal with their values. The same explanation goes with all attacks sub-category column for Lime.

## SHAP

At the end, Shap is used. In here, figure:4.7 The prediction starts from the baseline which is the average of all predictions. Features with red color influenced positively that increases the prediction value closer to 1. On the other hand, Features with blue color has negative influence and decreases the prediction value towards negative. At the actual prediction of the data instance the features get balanced.

In figure: 4.7 we predicted the 9 attacks of Sub-cat accordingly. For the first attack we can see DSF as -0.03 is the average predicted probability. Here feature number 10 (Tot\_Bwd\_Pkts) is in red that means they are increasing the prediction by impacting positively and according to the visual size, The biggest impact comes from feature 10 (0.4155), while features 4,0,1,9,2 (Dst\_Port, Flow\_ID, Src\_IP, Tot\_Bwd\_Pkts, Src\_Port) are in blue and they are decreasing the prediction by impacting negatively and among them, the visual size of feature 2 (Src\_Port) shows the magnitude of its negative effect (0.1708) as least for this instance.

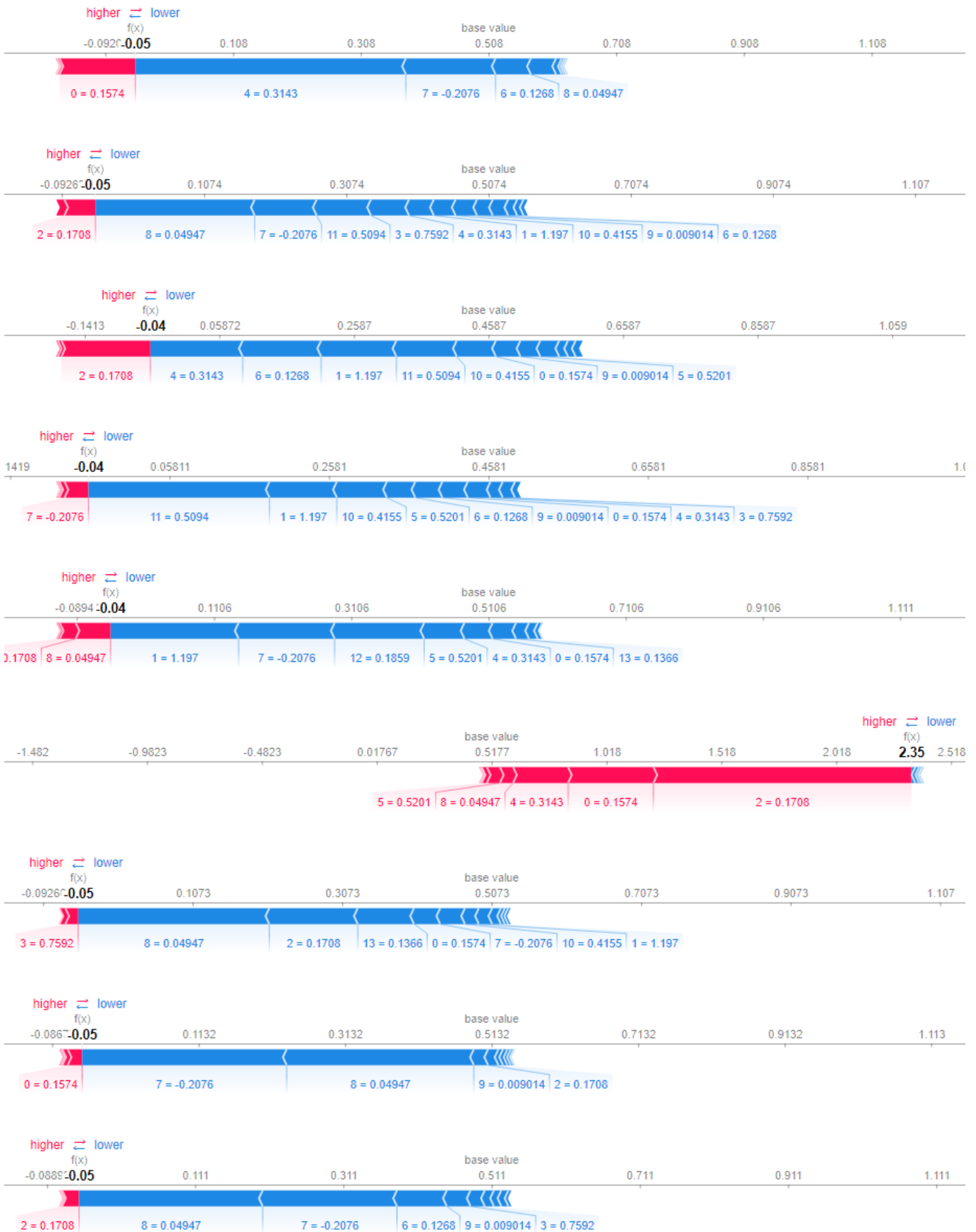


Figure 4.7: SHAP output for Sub-category using 399520 Test input data.

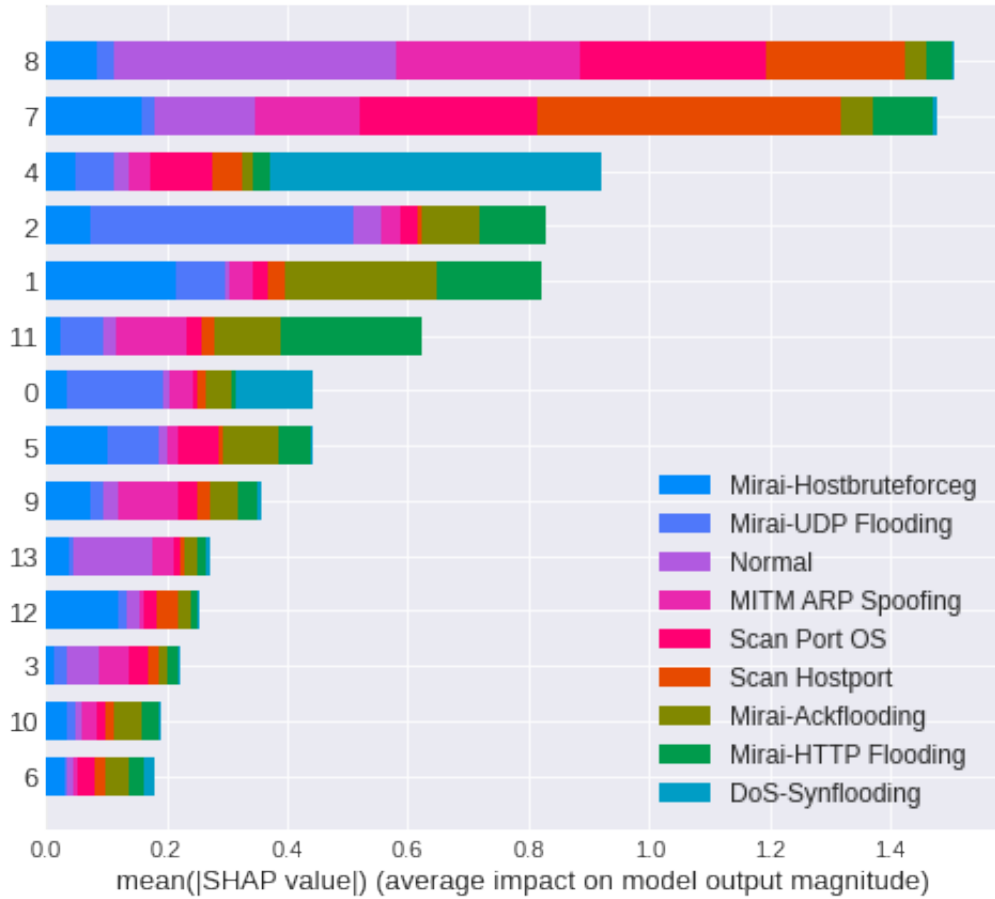


Figure 4.8: SHAP Summary Plot for subcategory.

We used Summary Plot figure:4.8, to get a straight-forward understanding of the visualization of the whole data. In Y-axis, all of the features are plotted where top most feature 8 (Tot\_Fwd\_Pkts) represents as the most contributor to the prediction and least contributor is feature 6 (Timestamp). On the X axis, All the Shap values(attacks) are plotted. The attacks name are given with the color so that we can understand which feature is impacting more or less on which attack. The contribution of the features increases in the plot as the Shap values moves away from 0. It can be observed from the figure:4.8 that, the most contributor feature 8 (Tot\_Fwd\_Pkts) contributes on Normal the most as per the visual magnitude of the attack.

## 4.2.2 Category

At first Category target output column is chosen for applying Eli5,LIME and SHAP xai tools for experiment to show feature importance and performance in model for evaluation.

### ELI5

Here, the fig: 4.9 representing weights associated with each features as feature importance from high to low according to color dark green to low green. from the

Weight	Feature
0.1690	f4
0.1508	f7
0.1494	f8
0.0907	f13
0.0830	f1
0.0754	f2
0.0502	f0
0.0393	f12
0.0386	f6
0.0361	f9
0.0332	f11
0.0331	f3
0.0314	f5
0.0197	f10

Figure 4.9: Cat ELI5 weight.

figure it is observed that feature number 4 (represents Dst\_Port) is the most important feature according to its highest weight (0.1690). On the other hand, the least important feature is 10 (represents TotLen\_Fwd\_Pkts) as its weight is the lowest (0.0197). But here we can't say anything about the characteristics of the important feature, whether it is impacting positively or negatively.

y=0 (probability 0.092, score -0.583) top features			y=1 (probability 0.091, score -0.585) top features			y=2 (probability 0.635, score 1.354) top features		
Contribution?	Feature	Value	Contribution?	Feature	Value	Contribution?	Feature	Value
+0.196	x0	-1.359	-0.000	x0	-1.359	+0.776	x2	-0.527
-0.000	x12	-0.007	-0.000	x2	-0.527	+0.168	x5	-0.155
-0.000	x2	-0.527	-0.000	<BIAS>	1.000	+0.108	x1	-0.319
-0.000	x8	-0.024	-0.001	x6	-0.078	+0.089	x9	-0.120
-0.000	x11	-0.027	-0.001	x5	-0.155	+0.068	x6	-0.078
-0.001	x9	-0.120	-0.003	x13	0.014	+0.065	x8	-0.024
-0.001	<BIAS>	1.000	-0.009	x3	0.020	+0.057	x4	-0.221
-0.002	x3	0.020	-0.016	x10	-0.134	+0.018	x7	0.023
-0.002	x10	-0.134	-0.037	x11	-0.027	+0.002	x13	0.014
-0.003	x13	0.014	-0.060	x9	-0.120	+0.001	x11	-0.027
-0.006	x7	0.023	-0.083	x12	-0.007	+0.001	<BIAS>	1.000
-0.059	x1	-0.319	-0.115	x7	0.023	+0.001	x10	-0.134
-0.310	x6	-0.078	-0.259	x8	-0.024	+0.001	x3	0.020
-0.395	x4	-0.221				-0.000	x12	-0.007

y=3 (probability 0.091, score -0.586) top features			y=4 (probability 0.091, score -0.587) top features		
Contribution?	Feature	Value	Contribution?	Feature	Value
-0.000	x13	0.014	+0.007	x2	-0.527
-0.000	x12	-0.007	+0.000	<BIAS>	1.000
-0.000	x5	-0.155	-0.000	x12	-0.007
-0.000	x6	-0.078	-0.000	x13	0.014
-0.000	x2	-0.527	-0.000	x0	-1.359
-0.001	x10	-0.134	-0.001	x6	-0.078
-0.001	<BIAS>	1.000	-0.005	x9	-0.120
-0.003	x1	-0.319	-0.008	x5	-0.155
-0.004	x11	-0.027	-0.019	x4	-0.221
-0.005	x9	-0.120	-0.038	x1	-0.319
-0.092	x7	0.023	-0.077	x8	-0.024
-0.480	x8	-0.024	-0.445	x7	0.023

Figure 4.10: CAT - eli5 output.

From figure 4.10, we can observe that each features are represented according to their importance and their contribution (positive and negative) on the prediction. Here 5 attacks of Category (DSF, MAS, Mirai, Normal, Scan) attacks are represented as  $Y$  from 0 to 4. Green color shows the important features and the Red ones represents the least. For example, in  $y=0$  (DSF-attack),the most important feature is  $x_0$  that represents Flow\_ID with highest positive contribution (+0.196)in the prediction and fetaure  $x_4$  that is Dst\_Port with least contribution (-0.395). Moreover, feature  $x_{12}$ (Fwd\_Pkt\_Len\_Max) works as a second top positive influenced feature for DSF (class 0). On the other hand, same feature works as a negative influenced one for MAS(class 2).

## LIME

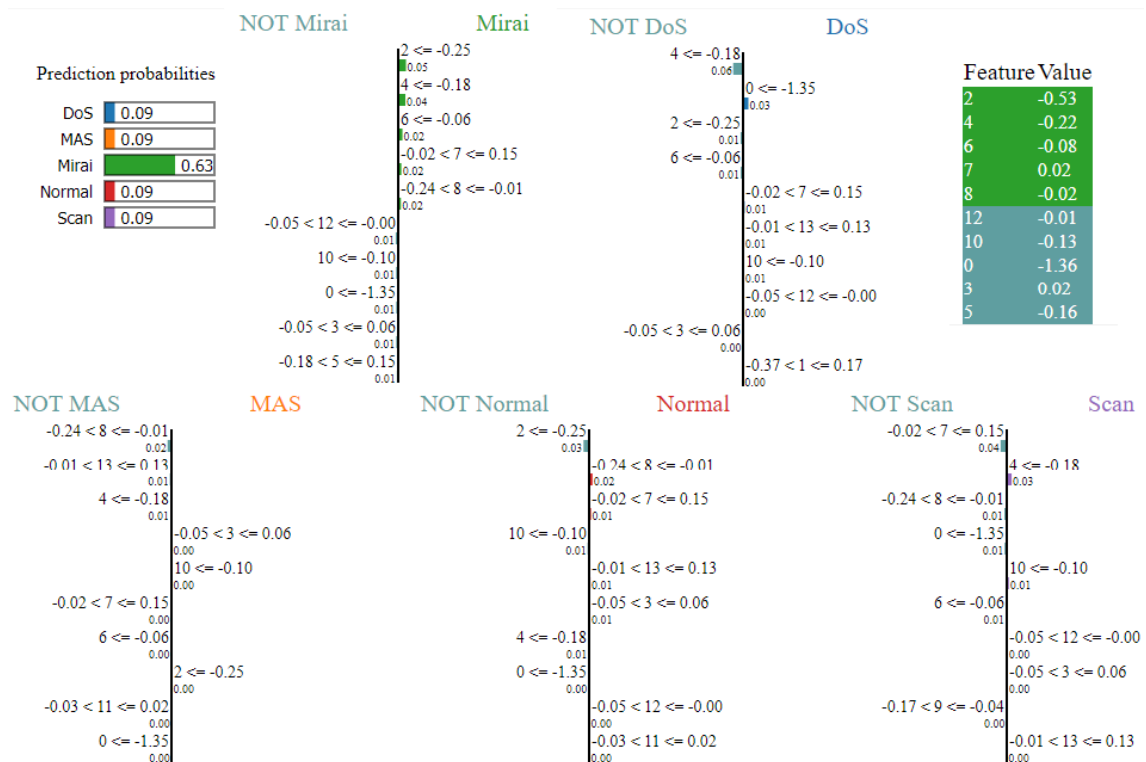


Figure 4.11: CAT - lime output.

From figure: 4.11, the attack Mirai, DoS, MAS, Normal and Scan corresponding the test value 14 feature has influence for predicting the attack where the right side, Green color is for positive contributed features and the left side, Teal color is for negative influenced feature. We can see the top most influenced feature is  $X_2$ (Flow\_Duration) for Mirai, which is in the positive side of the attack section. The first column represents the prediction probabilities of attacks predicted by the classifiers. The second column shows features' contributions to the probability where feature number 2,4,6 (Src\_Port, Dst\_Port, Timestamp) for SPS attack are the most important feature impacting positively and 0,3,5 (Flow\_ID, Dst\_IP, Protocol)as least important feature. The third column displays the original data values from where we can see features with the positive and negative impact by colors as Green and Teal with their values.

# SHAP

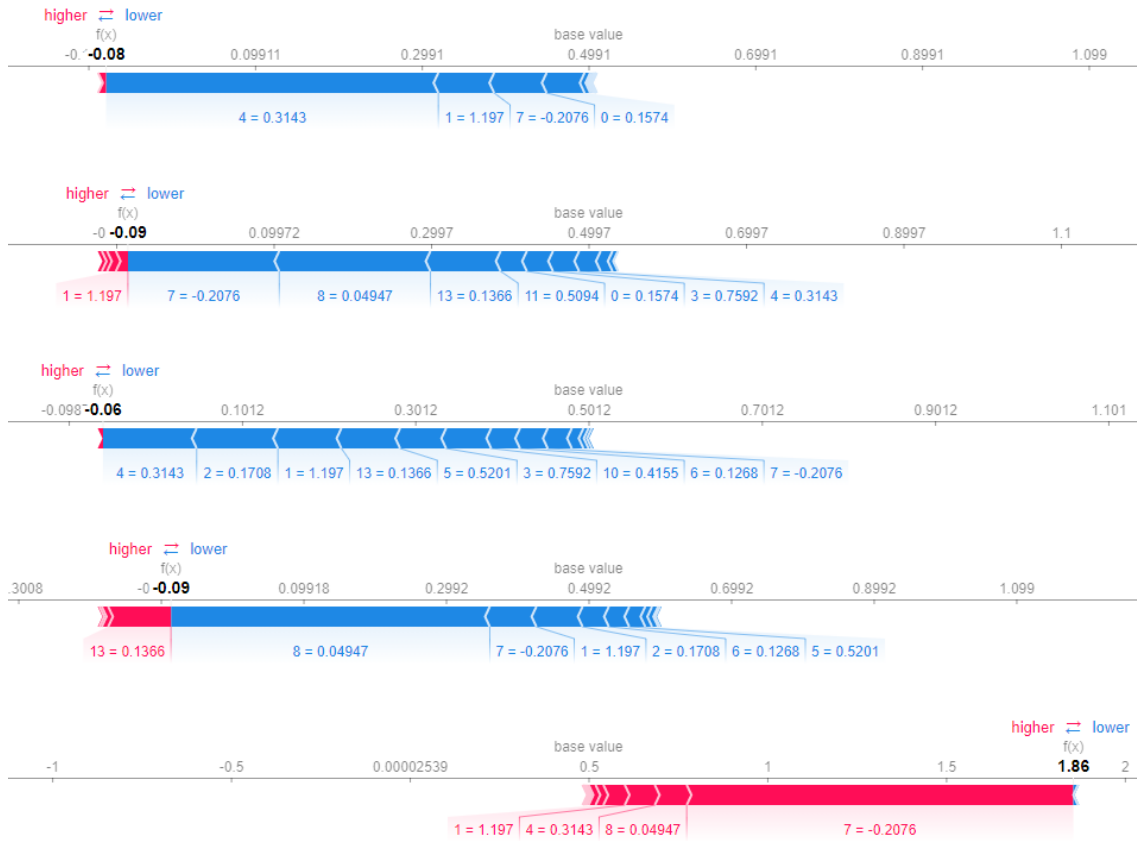


Figure 4.12: CAT SHAP OUTPUT.

From the figure: 4.12, prediction starts from the baseline, average predicted value for DSF is (-0.08), for MAS is (-0.09), Mirai is (-0.06), Normal is (-0.09) and for Scan is (1.86). Feature number 1 (Src\_IP) for MAS is in red that means the feature is impacting positively. According to the visual size, feature 1 shows the magnitude of most impacting positively feature on the prediction. Similarly, the features (7,8,13,11,0,3,4) are in blue, means they are negatively impacting on the prediction and among them 4 (Dst\_Port) is the least impacting feature negatively.



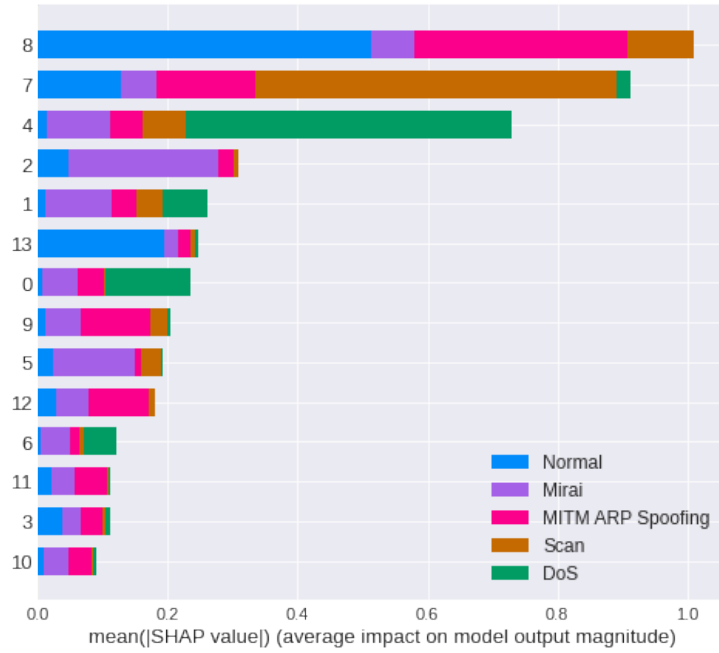


Figure 4.13: CAT - summery plot.

From figure: 4.13, All the features of CAT classes are plotted in  $Y$  axis. All the CAT classes are plotted in  $X$  axis. Beside, the attack names are identified by colors. From the figure, we can observe that, the most contributor feature is 8(Tot\_Fwd\_Pkts) that contributes mostly on Normal class and the least contributor is 10(TotLen\_Fwd\_Pkts) that is close to 0. Here, The contribution of the features increases in the plot as the Shap values moves away from 0.

### 4.2.3 Label

At first Label target output column is chosen for applying Eli5,LIME and SHAP xai tools for experiment to show feature importance and performance in model for evaluation.

#### ELI5

Here, Just like above described for Sub-cat and Cat, the fig: 4.14 representing weights associated with each features as feature importance from high to low according to color dark green to low green. from the figure it is observed that feature number 8 (represents Tot\_Fwd\_Pkts) is the most important feature according to its highest weight (0.3291). On the other hand, the least important feature is 1(Src\_IP) as its weight is the lowest (0.0072).

```
eli5.show_weights(xgb_estimator1)
```

Weight	Feature
0.3291	f8
0.1858	f7
0.1066	f13
0.0751	f6
0.0636	f9
0.0484	f2
0.0387	f11
0.0320	f3
0.0258	f5
0.0237	f12
0.0234	f0
0.0221	f4
0.0183	f10
0.0072	f1

Figure 4.14: LABEL ELI5 weight.

**y=0** (probability **1.000**, score **-20.075**) top features

Contribution?	Feature	Value
+9.135	x7	-0.169
+6.223	x8	0.162
+1.129	x9	0.181
+1.122	x2	-0.298
+0.817	x4	0.153
+0.731	x11	0.201
+0.619	x3	-0.915
+0.575	x6	-0.031
+0.561	x1	0.984
+0.388	x0	1.207
+0.386	<BIAS>	1.000
+0.301	x13	0.078
+0.281	x12	0.066
-0.850	x5	0.361
-1.342	x10	0.101

Figure 4.15: LABEL ELI5 output.

From figure 4.12, we can observe that each features are represented according to their importance and their contribution (positive and negative) on the prediction. Here  $y=0$  is the class of Label attack has the highest probability for test input 234283. Green color shows the important features and the Red ones represents the least. Here,  $x7$  (Flow.Duration) is the most important feature with (+9.135) contribution and the the least important feature is  $x10$  (TotLen.Fwd.Pkts) with (-1,342) contribution.

## LIME

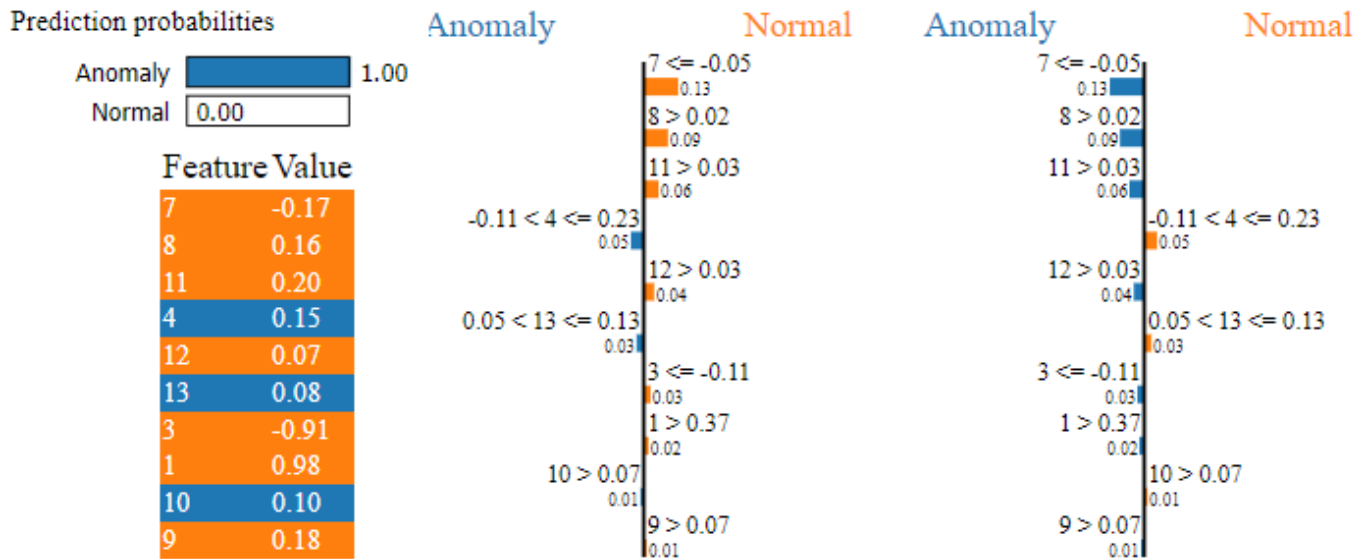


Figure 4.16: LABEL LIME output.

From figure: 4.16, the attack corresponding the test value 14 feature has influence for predicting the attack Normal where the right side, Orange color is for positive contributed features and the left side, Blue color is for negative influenced feature. Moreover, the top most influenced feature is X7(Flow\_Duration) which is in the positive side of the attack section. The first column represents the prediction probabilities of attacks where Normal is predicted by the classifiers. The second column shows features' contributions to the probability where feature number 7,8,11 (Flow\_Duration, Tot\_Fwd\_Pkts, TotLen\_Bwd\_Pkts) for SPS attack are the most important feature impacting positively and 4,13,10 (Dst\_Port, Fwd\_Pkt\_Len\_Min, TotLen\_Fwd\_Pkts)as least important feature. The third column displays the original data values from where we can see features with the positive and negative impact by colors as Green and Teal with their values

## SHAP

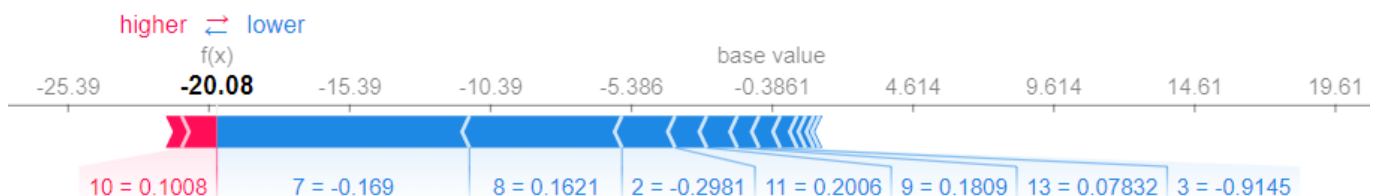


Figure 4.17: LABEL SHAP output.

From the figure: 4.17, prediction starts from the baseline, average predicted value is (-20.08) for Label class attack. Feature number 10(TotLen\_Fwd\_Pkts) is in red that means the feature is impacting positively. According to the visual size, feature 10 shows the magnitude of most impacting positively feature on the prediction.

Similarly, the features (7,8,2,11,9,13,3) are in blue, means they are negatively impacting on the prediction and among them 3 (Dst\_IP) is the least impacting feature negatively.

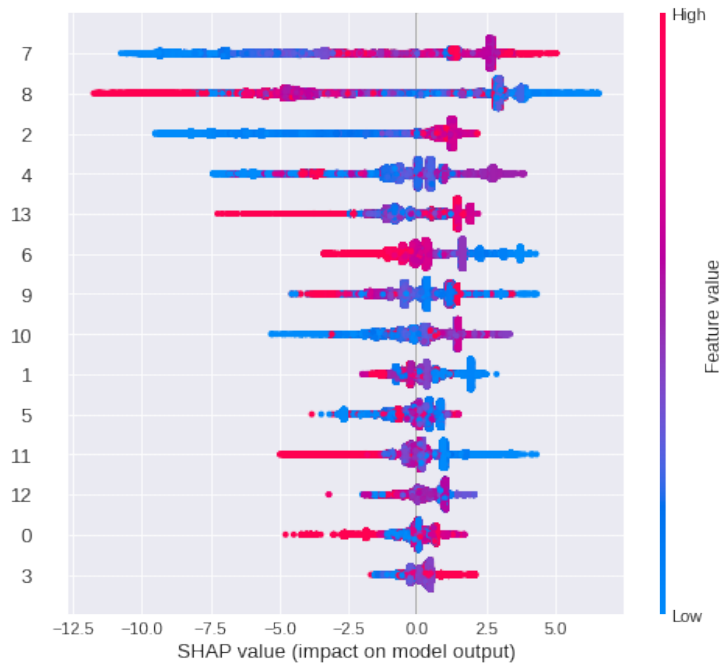


Figure 4.18: LABEL - summery plot.

The points on the summary plot represents the Shapley value for a feature and an instance where X-axis represents SHAP value (impact on model output) and Y-axis as Feature values. The color red to blue determines the value of features as high to low. The feature elements are plotted according to their significance Covering points are jittered in the y-pivot position, so we get to understand the appropriation of the Shapley esteems per feature. In the same way, Explainable AI is implemented on both Label and Category target columns for visualizing the change of features both positive and negative impact with most to least important feature in a sorted way for particular class for taking a particular test data input.

# Chapter 5

## Conclusion

In the Internet of Things (IoT) period, user devices produce a lot of data that can be utilized to improve the user experience of a system. As a huge number of the Internet of Things (IoT) devices are conveyed, the security and privacy issues in IoT excite increasingly more consideration. This is on the grounds that IoT devices are frequently restricted in computing capacity and energy, making them especially powerless against foes. IoT devices are more presented to and sadly harder to be protected from cyber-attacks than computers. As a result, recognizing attacks to protect IoT devices from malicious practices is critical to widening the uses of IoT. Despite the fact that there are many attack identifiers existing, attack methods are progressively changing with the recently evolved methodologies of attackers based on environmental feedback to dodge identification. Which gets challenging for the attack defenders to recognize the new attack by observing its pattern and explain the prediction in human-readable way. For this reason, we have established a model to ensure security for devices or individual's personal data or information from threats or IoT attacks by classifying them with supervised learning using XGBOOST algorithm which we have introduced in the paper in our proposed model which works profoundly to detect IoT attack. Moreover for giving stable, correct and proper explanation of the predictions efficiency and leading a security investigation XAI techniques like Lime, SHAP and Eli5 where, Lime helped in feature selection to explain critical outputs and SHAP presented the feature importance along with their effects on the model and Eli5 helped to explain the top influenced features in a straightforward manner so that any users or specialists can understand the reason behind the attack detection decision output.

# Bibliography

- [1] A. Janecek, W. Gansterer, M. Demel, and G. Ecker, “On the relationship between feature selection and classification accuracy,” *Journal of Machine Learning Research - Proceedings Track*, vol. 4, pp. 90–105, Jan. 2008.
- [2] F. Song, Z. Guo, and D. Mei, “Feature selection using principal component analysis,” in *2010 International Conference on System Science, Engineering Design and Manufacturing Informatization*, vol. 1, 2010, pp. 27–30. DOI: 10.1109/ICSEM.2010.14.
- [3] M. T. Ribeiro, S. Singh, and C. Guestrin, “*why should i trust you?*”: *Explaining the predictions of any classifier*, 2016. arXiv: 1602.04938 [cs.LG].
- [4] I. L. Cherif and A. Kortebi, “On using extreme gradient boosting (xgboost) machine learning algorithm for home network traffic classification,” in *2019 Wireless Days (WD)*, 2019, pp. 1–6. DOI: 10.1109/WD.2019.8734193.
- [5] C. Ioannou and V. Vassiliou, “Classifying security attacks in iot networks using supervised learning,” in *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2019, pp. 652–658. DOI: 10.1109/DCOSS.2019.00118.
- [6] S. R. Islam, W. Eberle, S. K. Ghafoor, A. Siraj, and M. Rogers, “Domain knowledge aided explainable artificial intelligence for intrusion detection and response,” *CoRR*, vol. abs/1911.09853, 2019. arXiv: 1911.09853. [Online]. Available: <http://arxiv.org/abs/1911.09853>.
- [7] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, “Application of deep reinforcement learning to intrusion detection for supervised problems,” *Expert Systems with Applications*, vol. 141, p. 112963, Sep. 2019. DOI: 10.1016/j.eswa.2019.112963.
- [8] B. Bhati, G. Chugh, F. Al-Turjman, and N. Bhati, “An improved ensemble based intrusion detection technique using xgboost,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, Aug. 2020. DOI: 10.1002/ett.4076.
- [9] T. Gu, A. Abhishek, H. Fu, H. Zhang, D. Basu, and P. Mohapatra, “Towards learning-automation iot attack detection through reinforcement learning,” in *2020 IEEE 21st International Symposium on “A World of Wireless, Mobile and Multimedia Networks” (WoWMoM)*, 2020, pp. 88–97. DOI: 10.1109/WoWMoM49955.2020.00029.
- [10] A. Kuppa and N.-A. Le-Khac, “Black box attacks on explainable artificial intelligence(xai) methods in cyber security,” in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–8. DOI: 10.1109/IJCNN48605.2020.9206780.

- [11] M. Kuzlu, U. Cali, V. Sharma, and Ö. Güler, “Gaining insight into solar photovoltaic power generation forecasting utilizing explainable artificial intelligence tools,” *IEEE Access*, vol. 8, pp. 187 814–187 823, 2020. DOI: 10.1109/ACCESS.2020.3031477.
- [12] P. B. OnClick360 and OnClick360, *Interpretable machine learning with limeeli5 shap interpretml*, Feb. 2020. [Online]. Available: <https://www.onclick360.com/interpretable-machine-learning-with-limeeli5-shap-interpret-ml>.
- [13] U. Pawar, D. O’Shea, S. Rea, and R. O’Reilly, “Explainable ai in healthcare,” in *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 2020, pp. 1–2. DOI: 10.1109/CyberSA49311.2020.9139655.
- [14] I. Ullah and Q. Mahmoud, “A scheme for generating a dataset for anomalous activity detection in iot networks,” pp. 508–520, May 2020. DOI: 10.1007/978-3-030-47358-7\_52.
- [15] M. Wang, K. Zheng, Y. Yang, and X. Wang, “An explainable machine learning framework for intrusion detection systems,” *IEEE Access*, vol. 8, pp. 73 127–73 141, 2020. DOI: 10.1109/ACCESS.2020.2988359.