

SCHOOL OF GRADUATE STUDIES

DESIGN HYBRID BESAD INTRUSION DETECTION SYSTEM USING MACHINE LEARNING ALGORITHM AND SAFE MACHINE LEARNING

WOLKITE, ETHIOPIA

MASTERS THESIS

SOLOMON NEGASA JARA

APRIL,2024

WOLKITE, ETHIOPIA

Wolkite University

School Of Graduate Studies

Design Hybrid Based Intrusion Detection System Using Machine Learning Algorithm and Safe Machine Learning

Wolkite, Ethiopia

A Thesis Submitted to the School of Graduate Studies in Partial Fulfillment of The Requirements for The Degree of Master of Computer Science and Engineering

Solomon Negasa Jara

Major Advisor: Negalign Wake Hundera (Ph.D.)

April 2024

Wolkite, Ethiopia

WOLKITE UNIVERSITY
SCHOOL OF GRADUATE STUDIES

"We at this moment certify that we have read and evaluated this thesis titled" **Design Hybrid Based Intrusion Detection System Using Machine Learning Algorithm and Safe Machine Learning**" prepared under our guidance by **Solomon Negasa Jara**. We recommend they submit to fulfillment of the requirements for the award of MSc. degree in Computer Science and Engineering.

Negalign Wake (PhD)	_____	_____
Name of Major Advisor	**Signature**	**Date**
_____	_____	_____
Name of CO-Advisor	**Signature**	**Date**

As members of the Board of Examiners of the Master of Science thesis open defense Examination, we have read and evaluated this thesis prepared by Solomon Negasa Jara and examined the candidate. We certify we accepted the thesis for fulfilling the requirements for the award of MSc-degree in Computer Science and Engineering.

Sintayewu Hirpasse(PhD)	_____	_____
Name of External Examiner	Signature	Date
Barmura Yibgeta(Asse.Pro)	_____	_____
Name of internal Examiner	Signature	Date
_____	_____	_____
Name of chairperson	Signature	Date

DECLARATION

In my signature below, I declare and affirm that this thesis is my work. I have followed all ethical principles of scholarship in the preparation, data collection, data analysis, and completion of this thesis. By citing sources, I have acknowledged all scholarly matters mentioned in the thesis. I have cited and referenced all sources used in this document, confirming the acknowledgment of all scholarly matters mentioned in the thesis. I have made every serious effort to avoid any plagiarism in preparing this thesis.

The student submits this thesis in partial fulfillment of the requirement for a degree from the School of Graduate Studies at Wolkite University. The student deposits the thesis in the Wolkite University library, and the library makes it available to borrowers under its rules. I declare with sincerity that I have not presented this thesis to any other institution anywhere for obtaining any academic degree, diploma, or certificate.

Use brief quotations from this thesis without special permission, as long as you make accurate and complete acknowledgment of the source. The head of the school or department or dean of the school of graduate Studies may grant requests or permission for extended quotation from, or reproduction of this thesis in whole or in part when he/she judges that the proposed use of the material is in the interest of scholarship. In all other instances, however, individuals must get permission from the author of the thesis.

Name: **Solomon Negasa Jara**

Date: 4/15/2024

School/Department: Computer Science and Engineering

Signed by:

ACKNOWLEDGMENT

First, and most importantly, I want to express my sincere gratitude to the Almighty God for blessing me with strength, protection, and favor, which were crucial for accomplishing this purpose. Without the help, motivation, and knowledge of multiple people, this difficult thesis would not have been workable. This is my chance to thank and honor Negalign Wake Hundera (Ph.D.) sincerely. for his excellent supervision, advice, and unwavering support during this project. I appreciate your prompt attention to all of my emails, your recommendations, and your excellent supervision.

My friends, especially Hana Moti and Bayu Tolosa words are few enough to express my deepest thanks to you. You were always supporting me and encouraging me with your best wishes.

I also want to thank my family sincerely, whose unwavering support, affection, and prayers helped keep me inspired and dedicated to our aim.

****

Through him all things were made; without him nothing was made that has been made.

****

Glory to you, my God!

TABLE OF CONTENTS

ACKNOWLEDGMENT IV
LIST OF ACRONYMS/ABBREVIATIONS XII
ABSTRACT XIV
1 INTRODUCTION 1
1.1 BACKGROUND TO THE RESEARCH 1
1.1.1 Intrusion Detection Approaches 3
1.2 PROBLEM STATEMENT 5
1.3 RESEARCH QUESTIONS 6
1.4 OBJECTIVE OF THE RESEARCH 7
1.4.1 General Objective 7
1.4.2 Specific Objectives 7
1.5 SCOPE AND LIMITATIONS 7
1.5.1 Scope of the Study 7
1.6 LIMITATIONS OF THE STUDY 8
1.7 SIGNIFICANCE OF THE RESEARCH 8
1.8 BENEFICIARY OF THE RESEARCH 9
1.9 RESEARCH ORGANIZATION 11
2 LITERATURE REVIEW 12
2.1 RELATED WORK 13
2.2 OVERVIEW OF INTRUSION AND IDS 15
2.3 CATEGORIES OF IDS 15
2.3.1 Anomaly-based IDS 16
2.3.2 Signature-Based Detection 18
2.3.3 Hybrid IDS 19
2.3.4 Network-Based IDS (NIDS) 21
2.3.5 Host-Based IDS (HIDS) 22
2.3.6 Centralized IDS 23
2.3.7 Distributed IDS 23
2.4 ARTIFICIAL INTELLIGENCE 24
2.5 MACHINE LEARNING (ML) 24

2.5.1 Unsupervised Learning 25
2.5.2 K-Means Algorithm 26
2.5.3 Supervised Learning 27
2.5.4 Semi-supervised Learning 28
2.6 Some Popular Classification Algorithms 29
2.7 Deployment Scenario for IDSs 36
2.7.1 Before the Firewall 36
2.7.2 Inside the Private Network 37
2.7.3 Deployment on Individual Hosts 38
2.8 Safe Machine Learning Idea 39
2.9 Safe Machine Learning Method 40
2.9.1 Chernoff and Bhattacharyya-based Upper Bound Error 41
2.9.2 Kolmogorov-Smirnov Distance 42
2.9.3 Kuiper Distance 43
2.9.4 Cramer-Von Mises Distance 43
2.9.5 Wasserstein Distance 44
2.10 Research Gap 45
3. METHODOLOGY 46
3.1 Overview 46
3.2 Research Design Approach 46
3.3 System Model 48
3.4 Proposed HIDS Framework 48
3.5 Proposed Safe Machine Learning Framework 51
3.6 Capabilities of Safe ml 53
3.7 Type and Source of Data 54
3.7.1 Types of Data 54
3.8 Methods of Data Analysis: - 54
3.8.1 Descriptive 55
3.9 Source of Data 55
3.10 Sample Design 55
3.11 Data Selection 56
3.12 Dataset Description 57

3.13 . Dependent Variable(Label) 58
3.14 Performance Evaluation Metrics 62
4. EXPERIMENT AND RESULTS 64
4.1 Preprocessing and Feature Selection 65
4.2 Data sampling 67
4.3 Data cleaning 68
4.4 Data Normalization: 68
4.5 Feature Engineering 71
4.5.1 K-Fold Cross Validation and Stratified Splitting 73
4.6 Kernel Principal Component Analysis (KPCA): 73
4.7 Machine Learning Model and Results 74
4.8 Experimentation on Signature Detection 74
4.8.1 KNN model 74
4.8.2 KNN model with PCA algorithms 75
4.8.3 Random Forest Algorithms 75
4.8.4 Random Forest algorithms using PCA RF algorithm 76
4.8.5 Extra Tree Algorithms 77
4.8.6 Extra tree algorithms with PCA algorithms 77
4.8.7 DT algorithms 77
4.8.8 DT algorithms with PCA algorithm 78
4.9 Comparison of Models 78
4.10 Correlation Analysis 79
4.11 Safe ML model 80
4.11.1 The Safe ML Statistical Distance Measures results 80
4.11.2 Transformed dataset into classes 80
4.11.3 The Safe ML Statistical distance measures 81
4.11.4 Concatenated the two classes 0 and Class 1 84
4.11.5 Mean Accuracy to the Statistical Distance Measure Algorithms 85
4.12 The graphs/ plots showing accuracy based on ECDF 86
4.12.1 Anderson-Darling distance with KNN line plot. 86
4.12.2 Mean accuracy of Anderson-Darling distance with KNN. 87
4.12.3 Wasserstein_dist_dist with KNN line plot 88
4.12.4 Mean accuracy of Wasserstein_dist with KNN. 89

4.12.5 Kolmogorov Smirnov_dist with KNN line plot. 90
4.12.6 Mean accuracy Kolmogorov_Smirnov_dist with KNN. 91
4.12.7 Anderson-Darling distance with RF line plot. 92
4.12.8 Mean accuracy Anderson_Darling_dist with RF. 93
4.12.9 Mean Accuracy to the Statistical Distance Measure Algorithms 94
4.12.10 Mean Accuracy Safe Machine learning, PCA and Machine learning 96
4.12.11 Mean Accuracy Safe Machine learning and Machine learning using PCA 97
5.1 DISCUSSION 98
5.2 CONCLUSION 101
5.3 FUTURE WORK AND RECOMMENDATION 102
REFERENCES 103
APPENDIX 107
APPENDIX A: TOTAL DATASET FOR EXPERIMENTS 107
APPENDIX B: ANALYSIS OF LINEAR SEPARABILITY IN ANY FEATURE USING PCA 108
APPENDIX C: DISTRIBUTION OF TARGET CLASS IN TRAINING DATA 109
APPENDIX D: SEABORN PLOTTING FOR ANDERSON DARLING 110
APPENDIX E: THE GRAPHS/ PLOTS SHOWING ECDF 111
APPENDIX F: SAFE ML FOR CLASSIFICATION 112
APPENDIX G: DATA CLEANING AND PREPROCESSING 113
APPENDIX H: FLOW CHART OF SAFE ML CLASSIFICATION 114
APPENDIX I: DIAGRAM ANALYSIS OF THE LABEL(DEPENDENT VARIABLE) 115

LIST OF TABLES

Table 3.1: Description of files containing the CICIDS2017 dataset 56
Table 3.2: Overall Characteristics of the CICIDS2017 Data 57
Table 3.3: Class-wise instance Occurrence of the CICIDs2017 database 58
Table 3.4 : Frequency analysis target variable or label 70
Table 3.5: Group of attributes of the dataset 60
Table 3.6: Variable Entered and Removed form model 61
Table 3.7: Confusion Matrix 62
Table 4.1:After Data Cleaning Cleaned dataset. 68
Table 4.2 : converts string features into numerical features 69
Table 4.3 : Descriptive Dataset 70
Table 4.4 : missing correlations of attribute and their significance 71
Table 4.5: KNN model experiment 74
Table 4.6: Performance Evaluation Metrics of KNN 75
Table 4.7: Performance Evaluation Metrics of KNN using PCA 75
Table 4.8: Random Forest model experiment 75
Table 4.9: Performance Evaluation Metrics of RF 76
Table 4.10: Performance Evaluation Metrics of RF with PCA 76
Table 4.11: Performance Evaluation Metrics Extra Tree 77
Table 4.12: Performance Evaluation Metrics Extra Tree using PCA 77
Table 4.13: Performance Evaluation Metrics DT 77
Table 4.14: Performance Evaluation Metrics DT using PCA 78
Table 4.15: Comparison of Models 78
Table 4.16: correlation between attribute 79
Table 4.17: Principal component Analysis permutation 81
Table 4.18: Start run for class 0 attack class 82
Table 4.19: Start run for class 1 or normal class 83
Table 4.20: Running the Statistical distance measure algorithms 84
Table 4.21: Mean Accuracy of the Statistical Distance Measure Algorithms 85
Table 4.22: Mean Accuracy of the Statistical Distance Measure Algorithms 94
Table 4.23: Compare the ML Accuracy with Safe ML ECDF Accuracy 95

LIST OF FIGURES

Figure 1.1: Safe machine concept[4] 2
Figure 1.2: Safe Machine Learning (Safe ml) and Safe Deep Learning (Safe dl)[5] 2
Figure 2.1: Categories of IDS[8] 16
Figure 2.2: AIDS approach based on machine learning[7] 17
Figure 2.3: hybrid-based IDS[2] 20
Figure 2.4: Network-Based IDS (NIDS)[9] 22
Figure 2.5: K-means process flow[34] 27
Figure 2.6: Supervised Learning[36] 28
Figure 2.7: Classification and Regression 30
Figure 2.8: Support vector machine for linearly separable data[42] 33
Figure 2.9: Random Forest[45] 35
Figure 2.10: IDS Before the Firewall[47] 36
Figure 2.11: Inside the Private Network[47] 37
Figure 2.12: Deployment on Individual Hosts 38
Figure 2.13: Probability density functions of class 1 and class 2[5] 39
Figure 2.14: Probability Density Functions and Cumulative Functions[5] 41
Figure 2.15 PDF-based distance measures[3] 41
Figure 2.16: ECDF-based distance measures 42
Figure 2.17: Kolmogorov-Smirnov distance measure Illustration[5] 42
Figure 2.18: Kuiper Distance[5] 43
Figure 2.19: Cramer-Von Mises Distance[5] 44
Figure 3.1: The framework of the proposed HIDS 50
Figure 3.2: Safe Machine Learning Framework 51
Figure 3.3: Application of Safe ML in Security Intrusion Detection 52
Figure 3.4: Cross-validation and ML Classifiers 53
Figure 4.1 dataset preprocessing 67
Figure 4.2: Anderson-Darling distance with KNN line plot 86
Figure 4.3: Mean Accuracy of Anderson-Darling distance with KNN 87

Figure 4.4: Wasserstein_dist_dist with KNN line plot 88
Figure 4.5: Mean accuracy of Wasserstein_dist with KNN 89
Figure 4.6: Kolmogorov_Smirnov_dist with KNN line plot 90
Figure 4.7: Mean accuracy Kolmogorov_Smirnov_dist with KNN 91
Figure 4.8: Anderson_Darling_dist with RF line plot 92
Figure 4.9: Mean accuracy Anderson_Darling_dist with RF 93

LIST OF ACRONYMS/ABBREVIATIONS

IDS	Intrusion Detection Systems
DoS	Denial of Service
WAN	Wide area networks
WPAN	wireless personal area networks
HIDS	Host Intrusion Detection Systems
WSN	wireless sensor networks
RFID	Radio frequency identification
ML	machine learning
IoT	Internet of Things
AI	Artificial Intelligence
H-IDS	Hybrid Intrusion Detection Systems
NIDS	Network Intrusion Detection Systems
DR	Highest Detection Rate
FAR	False Alarm Rate
SVM	Support Vector Machine
CICIDS	Canadian Institute for cybersecurity
CSV	Comma-Separated Values
DDoS	Distributed Denials of Service
DOS	Denials of Service
DT	Decision Tree
FI	feature Importance
FN	False Negative
FR	False Negative Rate
FP	False Positive
FPR	False Positive Rate

KNN	K-Nearest Neighbor
ML	Machine Learning
NIDS	Network Intrusion Detection System
SVM	Support Vector Machine
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate

Abstract

In the subject of computer network security, network attacks have gained international attention. Hence, this thesis aims to evaluate different machine learning classification algorithms and Safe ml that classify network events in intrusion detection systems using a supervised approach and unsupervised types of machine learning classification methods. We used the following methods to carry out the analysis: KNN, Decision Tree, Random Forest, and Extra tree. for a supervised approach. Here we have presented a hybrid Machine learning approach to detect attacks. In the misuse or signature detection module, we used three different classifiers KNN, Decision Tree, Random Forest, and Extra tree which detect known attacks based on the signature database. The unsupervised detection module handles the unknown attack by employing the k-Means Clustering algorithm.

Applying data normalization during the pre-processing phase. label encoder, which, as many ML techniques cannot support them directly, translates string characteristics into numerical features. During the ML model construction process, we employ a comprehensive feature engineering technique using IG and KPCA to eliminate irrelevant, redundant, and noisy features while retaining the essential ones. Also used SPSS to examine associations and evaluate descriptive statistics like mean and standard deviation.

To show the performance of the proposed evaluation method, we conducted the experiment on the CICIDS2017 dataset. The results of our experiments showed that the classification model integrated with the transformation and feature selection method results in superior accuracy, error rate, and reduced false alarms. The result also shows that the Extra Tree model and RF hold the highest accuracy and reduce the false alarm rate.

By utilizing ECDF-based statistical distance measures, the Second Technique accurately predicted model performance safely. This part comprises experiments performed With Safe Machine learning algorithms using Empirical Cumulative Distribution Function (ECDF) using statistical distance measures including the Kolmogorov-Smirnov, Kuiper, Anderson-Darling, Wasserstein, and mixed Wasserstein-Anderson-Darling measures. Then compare all statistical distance accuracy measures of the accuracy of Safe ml with machine learning algorithms accuracy to find the best high confidence model.

Keywords: Intrusion Detection, Machine learning algorithms, Safe Machine learning, normalization, Distance measure

CHAPTER ONE

1 INTRODUCTION

1.1 Background to the Research

Networked computer systems impact every part of our lives nowadays. It has developed beyond our wildest dreams, posing additional security concerns that cause the creation of a new mechanism to ensure network security. Cybercriminals start cyber-attacks against networked computer systems to compromise security and achieve their goals [1].

The concept of intrusion detection began in the 1980s; during the next twenty years, host-based IDS processing audit logs grew into automated network IDS on wireless systems. The purpose of an Intrusion Detection System (IDS) is to detect unauthorized access. Some of these systems and networks that need access protection include wide area networks (WANs) and clouds, local area networks (LANs), ad hoc networks, wireless local area networks (WLANs), and wireless personal area networks (WPANs). In the 85 WPAN families, the three more common networks include wireless sensor networks (WSNs), mobile phones, and radio frequency identification (RFID). The detection techniques used on these systems and networks broadly categorize the IDS, for example... anomalies, stateful packet examination, or rule-based [1] .

The Intrusion Detection System works on two mechanisms: signature-based detection and anomaly-based detection [2]. Signature-based detection uses a known list of rules or indicators from the system attack database to specify whether the activity is malicious, while anomaly-based detection identifies the attack based on unusual user behavior patterns. The system can detect an attack if users perform unusual actions or activities [2].

When there is no label available, the Safe ml concept, introduced by [3], aimed at keeping track of the classifiers' choices. The accompanying figure shows the flowchart for the Safe ml concept. The training phase and the application phase are the two primary portions of this flowchart.

This story's subject falls under the category of "robustness to distributional shift" issues in AI safety. The Safe ml technique that we will describe is based on the following figure [3].

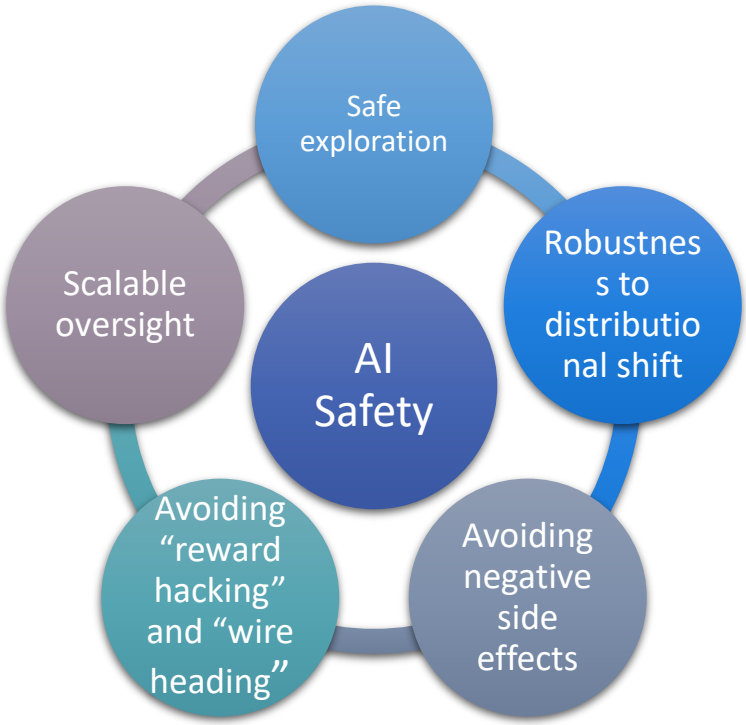


Figure 1.1: Safe machine concept[4]

"Robustness to distributional shift" is one issue in AI safety. Based on the following figure, the SafeML approach we are going to explain includes Safe Machine Learning (SafeML) and Safe Deep Learning (Safe dl).

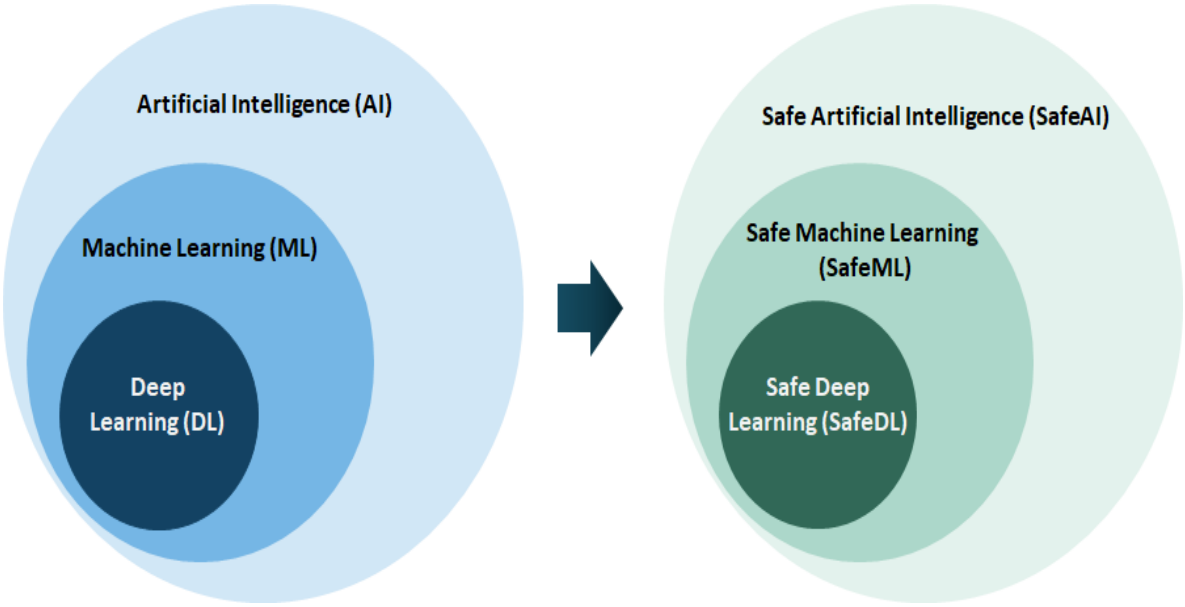


Figure 1.2: Safe Machine Learning (Safe ml) and Safe Deep Learning (Safe dl)[5]

1.1.1 Intrusion Detection Approaches

Researchers have studied intrusion detection for many years and have developed IDS tools [1]. There are several approaches to implementing IDS. Based on its deployment, IDS can act either as a host based or as a network-based IDS. The host-based IDS monitor and analyzes the internals of a computing system, whereas the network-based IDS analyzes network traffic at all layers of the TCP/IP suite and detects intrusions in network traffic flows from both directions. From the perspective of the detection approach applied, it also categorizes into anomaly and signature based.

Signature-based detection approach is an IDS technique in which detection of intrusion is based on the signature of known attacks tuned in the database. In the misuse detection approach, it identifies suspicious data by comparing new instances with the stored signatures or patterns of attacks in the database. It is efficient and comparably low false alarm rate IDS technique but it cannot detect emerging or new attack types [1]

Anomaly-based detection approach detects the behavior of a system deviations from the normal activity as anomalies. The IDS defines a baseline of normal data in network data and monitors new instances for deviations from normal behavior. The IDS compares the new instances with the baseline and notifies any deviation from the baseline as intrusion. Its major advantage is it can detect new attack types but generate high false alarms rates[1].

- **Supervised approach**- this technique needs a training dataset which has labeled instances for normal and anomaly classes. It can detect known attacks only, which are the foremost drawback of this approach.
- **Unsupervised Learning**-When performing unsupervised learning, the machine receives totally unlabeled data. The machine is required to uncover the intrinsic patterns that exist within the data, such as a clustering structure, a low-dimensional manifold, or a sparse tree and graph.Depending on the type of problem, one can apply clustering and dimension reduction techniques in this method.
- **Hybrid approach**- A hybrid intrusion detection method combines supervised and unsupervised network intrusion detection methods. Supervised methods have the advantage of being able to detect known attacks with high accuracy and low false alarms. On the other hand, an unsupervised method can detect novel attacks. Thus, a hybrid intrusion detection method can detect both known and unknown attacks with

high accuracy.

Though the research community has developed many intrusion detection frameworks and systems, researchers still face open challenges in IDS performance and zero-day attack detection. Hence, by taking the advantage of low false alarm of a misuse-based intrusion detection system and the capability of detecting zero-day attacks by an anomaly-based intrusion detection system in this paper, I proposed a hybrid intrusion detection system. I used decision tree (DT), random forest (RF), extra trees (ET), and KNN used as multi-class classifiers for known attack detection for misuse detection and A cluster labeling (CL) k-means used as an unsupervised learner for zero-day attack detection.

In this research, we evaluate a set of selected Machine learning algorithms on the CICIDS data set to overcome the problems in identifying unknown attacks. We measure their performance based on their detection rate and evaluation metrics. There are some major attack categories found on CICIDS benign, DoS (denial of service), and Bot and Infiltration, Brute force. These four attacks have distinct execution dynamics and signatures. However, these four major attacks grouped into one attack category to identify from normal network traffic, which is one drive for this research to investigate if certain detection algorithms are likely to demonstrate superior performance for a given network traffic record.

1.2 Problem Statement

The term "intrusion" refers to harmful behavior that, without the required authorization, threatens a network's or computer system's security from the outside in [1]. It results in the CIA of a computer system or network being lost. For example, an intrusion might undermine a network's Central Intelligence Agency (CIA) by obtaining root-level access, after which it could alter and steal the network's data[1].

With the present development in network technology, and the rise of new networking paradigms like Cloud Computing (CC) and the Internet of Things (IoT), cyberattacks are also developing and changing their strategies at a faster rate than defenders. Security is extremely important [6].

The requirement for protection against intrusive outsiders has grown crucial at this time because of the Internet's rapid expansion and the growing security issues brought on by its widespread use. Therefore, many organizations use intrusion detection systems.

Today, developers build most intrusion detection solutions on the market and freely accessible online around the signature-based method. This method's primary flaw is that it only focuses on signature databases and requires prior knowledge. It also uses rule-based analysis and does not know about unknown attacks and Safe ML.

Since the cloud server's storage center keeps the machine learning data, users will no longer can physically disconnect from one another and secure their data. It is common practice to outsource certain complicated machine learning computations to cloud servers, which raises the possibility of data loss, illegal access, and other security risks [7]. Thus, protecting privacy is becoming essential for machine learning advancement in cloud environments. The growth of safety machine learning and security has led to an expansion in the research of machine learning data privacy preservation [7].

The robustness of IDS to various evasion techniques still needs further investigation. According to [8] improvements in machine learning algorithms are the main means to enhance the detection effect using different feature selection methods and Safe Ml should reduce the number of input variables to those that are believed to be most useful to a model to predict the target variable.

In this study, we assessed a collection of chosen machine learning algorithms with Safe ml on the CICIDS dataset to address the challenges associated with identifying known and

new attacks. Their evaluation metrics and detection rate are used to gauge how well they are doing [9].This effort aims to do that by combining an anomaly-based system with an open-source signature system. We need detection algorithms to improve detection rates and reduce false alarm rates in signature-based systems. In this work, we also use the Safe ml method [10].

Typically, researchers have conducted a great deal of prior research on intrusion detection systems, with less emphasis on Safe Ml, detection rate, false alarm rate, and examination of both known and undiscovered threats. This study aims to close the gaps left by the previously listed issues.

While ML is a powerful tool for enabling data-driven applications, its unfettered use can pose risks to financial stability, privacy, the environment, and in some domains, even life. Poor design, mis-specification of the objective functions, implementation errors typically characterize poor application of ML, choosing the wrong learning process, or using poor or non-comprehensive datasets for training. Thus, we can define safety for ML as a set of actions that prevent any harm to humanity caused by ML failures or misuse. However, there are many perspectives and directions to be defined for ML Safety [3].

1.3 Research Questions

The study addresses the problem by answering the following questions:

- Can we minimize intrusion?
- Can ML algorithms be fully autonomous in detection, safety, and security? Are they safe?
- How to detect intrusion using (SafeML) with Statistical distance measure for signature and anomaly attack detection and Machine Learning algorithm.
- Which detection technique is the best for Hybrid-based Intrusion Detection systems to detect unknown attacks and known attacks using ML algorithms and ECDF using statistical distance measures?

1.4 Objective of the Research

1.4.1 General Objective

This research aims to design Intrusion Detection Systems using a machine learning algorithm with Safe ml to detect the network attack with the highest accuracy and to find the best suitable algorithm to detect the intrusions for known and unknown attacks.

1.4.2 Specific Objectives

The specific objectives are:

- To review the literature on Intrusion Detection System machine learning applications and safe machine learning Algorithms.
- Preprocessing dataset for effective noise removal and resampling the dataset for tackling the class imbalance and clustering the dataset to classes for the prediction.
- To develop models by using a different machine learning algorithm and statistical distance measures (SafeML) that can examine unknown attacks and known attacks.
- Designing a hybrid intrusion detection system and estimating the accuracy.
- Test and evaluate the performance and compare the best one.

1.5 Scope and Limitations

1.5.1 Scope of the Study

The goal of the study was to forecast the Intrusion Detection System's capacity to expect network attacks using machine learning methods with safe Ml, using data from open-source data sources that contained CICIDS 2017. Accuracy, precision, recall, and confusion matrices are used to assess the model's performance, and these classes serve as the foundation for training and testing the model.

The primary goal of this thesis is to design a hybrid intrusion detection system with security and safety. To increase detection rates while maintaining safety and security, this solution, which is appropriate for any network assault, focuses on identifying potential problems and looking into known and new threats.

1.6 Limitations of the Study

In this study, we have proposed a hybrid ML model that combines supervised and unsupervised approaches which solve detect known attacks that have already stored signatures or labeled, and new emerging attacks that have not been labeled. This research will enhance the effectiveness and efficiency of the network intrusion detection system by proposing efficient feature selection and classifier techniques.

The lack of organized and adequate historical data for intrusion detection was one of the limiting factors in this study. In addition, the dataset-related data was low compared to the sources at the country level. This was a challenge for addressing research objectives which depended on the dataset of our country. However, the Ethiopia security agency does not have compiled data or a repository on the intrusion. As a result, the researchers faced a limitation in the study as they had to collect the data from online open-source sources.

The other limitations are Like traditional machine-learning methods, the accuracy of the proposed approach highly depends on the quality of annotated data. If there were many documents assigned to the wrong label in the dataset, this would directly affect how well the system can label positive and negative.

1.7 Significance of the Research

Because of the diverse and pervasive nature of technologies, the rise of new networking paradigms and the dynamic shift in digitalization have created new security difficulties that will cause the development of new methodologies to ensure that enterprise networks meet security standards [11]. This means that using only signature-based intrusion detection systems (IDS) as a security tool cannot meet the present enterprise network security requirements.

Researchers have developed a range of machine learning techniques appropriate for cyber security issues using large data. Researchers in this field have developed many studies and used single algorithms (supervised or unsupervised approaches) and semi-supervised approaches to solve the intrusion detection problem. Looking at the literature, we can see

that most IDS studies use KDD and its upgraded version, NSL-KDD, for performance evaluation. Because internet traffic and attack types are constantly changing, this dataset cannot give the precision for evaluation. To evaluate the performance of IDS models can replicate the current traffic behavior, we have undertaken this study using the new CIC-IDS2017 benchmark dataset.

Our hybrid machine learning model, which we have shown in this work, combines supervised and unsupervised methods. It is a promising way to identify newly developing assaults without labels and known attacks with previously stored or tagged signatures. This study will improve network intrusion detection systems' efficacy and efficiency by putting forth appropriate feature selection and classifier strategies.

In summary, as businesses increasingly depend on data for decision-making, our research will assist them in leveraging extensive data to address issues or gain a competitive edge. It will increase professional understanding for those looking to advance their network security field of study.

1.8 Beneficiary of the Research

Enterprises have undergone significant change because of the IT revolution, which has sped up the creation of big data, cloud-based systems, and the Internet of Things (IoT). It has also made companies more dependent on technology to gain a competitive edge. Despite this, security concerns have emerged as one of the most pressing issues facing businesses, given the continuous evolution of attackers.

One should have a tool to guard against such security vulnerabilities because of the IT's constant change and the sophistication of assaults on computer networks. Within intrusion detection systems, machine learning algorithms are one technology used. By using the network intrusion detection system (IDS), IT staff can investigate threats or suspicious activities more thoroughly and take action to block or halt an attack. The IDS monitor network traffic and sends out notifications when it identifies recognized threats or suspicious activities.

To sum up, as businesses increasingly use data to decide, our research will assist them in effectively using large amounts of data to solve problems and gain a competitive edge. It will increase professional understanding for those looking to advance their network security field of study. Is a software application or device that monitors a network for policy violations or malicious activity?

Some IDS respond to detected intrusion upon discovery, known as intrusion prevention systems (IPS). Other IDS require an incident responder or analyst to investigate an issue and then take the steps to remediate the threat. Protecting your data from cybercriminals with the right IDS is important–choose the type that's right for you. Let's inspect. In summary, this research will help technology and similar businesses by promoting the understanding of how big data can solve problems and gain a competitive edge. This is relevant, as businesses are increasingly adopting data-driven decision-making. Besides this, it will add knowledge for professionals who want to advance their area of study in the network security domain.

Today, the increasing Internet traffic has resulted in increased concern about network security across businesses as attacks also dynamically increased. As a result, the existing security mechanisms of the Internet may not efficiently secure the network. Hence, it is important to find solutions to easily monitor the network, to address them using a combination of various methods. An IDS has become an alternative solution for monitoring computer systems or computer networks for malicious activities or events. In due course, IDS applications are expected to detect intrusions that come from inbound and outbound of the network.

1.9 Research Organization

We have organized the research:

Chapter Two: emphasizes the literature on various detection techniques used in the subject of intrusion detection. It covers the classification of intrusion detection systems, their types, the components of IDS in various architectures, safe machine learning, and ECDF for estimation accuracy. Last, talks cover popular open-source IDS that is based on signatures. IDS based on anomalies.

Chapter Three: talks about methods that apply to this research and are worth discussing. This chapter describes most of our research method using an attack detection approach. This chapter prepares the dataset, preprocesses it, and provides a design for the suggested task.

Chapter Four: In this chapter, the suggested algorithms are used to implement the suggested system. This presentation includes an evaluation of the work using experimental analysis and a comparison to other works.

Chapter Five: Here are the conclusions and next steps for the work.

CHAPTER TWO

2 LITERATURE REVIEW

This chapter discusses network threats, generic machine learning ideas, an overview of intrusion detection systems, popular methodologies, and their advantages and disadvantages, the fundamental component of IDS, and a discussion of a few classification algorithms. Then, we cover the security and safety of machine learning techniques. Lastly, we cover certain metrics for parameter optimization and evaluation of security and safety classification algorithms.

Our review of these extensive works of literature aimed to understand the problem with them and identify a solution to our work. A literature survey in a research report represents the study done to assist in the research's completion. A literature survey also describes a survey of the previously existing material on a topic of the report.

The literature survey depends or focuses on the following ideas:

- We now have and accept theories.
- Recent research, arranged chronologically within the topic of study.
- Obstacles facing the research and any existing projects, if any.

Literature surveys summarize or summarize the current research on a topic. There must be a logical flow to the written structure. It must show how the concepts developed in the study have developed. If a literature review is being completed for a class assignment or submitted for publication in a journal, its length will mostly depend on that aspect of the report. It may examine a few studies on a certain subject or provide an extensive analysis of the important work that researchers have done in the field up to that point.

2.1 Related Work

Researchers have previously released some studies on machine learning methods used in network intrusion detection systems. Each research has different topic concerns, such as regression, feature engineering, feature selection, data reduction, and classification method optimization.

The most common challenges in a network dataset are the complexity and various types of features in that dataset. Researchers have implemented various methods to deal with these challenges by selecting features in the dataset.

Researcher [2] multi-tiered hybrid intrusion detection system (MTH-IDS) model that can detect various types of known and zero-day cyber-attacks on both intra-vehicle and external-vehicular networks for modern vehicles. The proposed MTH-IDS comprises two traditional ML stages (data pre-processing and feature engineering) and four main tiers of learners using multiple machine learning algorithms. By performing data pre-processing and feature engineering, we can significantly improve the quality of the input data for more accurate model learning. The proposed system can effectively detect various types of known attacks with accuracies of 99.99% On the other anomaly intrusion detection, experiments can need further improvement. But is it fully autonomous to protect against intrusion?

[12]The rapid growth of artificial intelligence applications in various domains and particularly in autonomous vehicle software raises concerns in different perspectives, such as AI safety, AI responsibility, AI explain ability and interpretability, human-in-the-loop AI, and AI trustworthiness. This paper addressed distributional shift and its implications for the safety of machine learning or deep learning classification tasks in autonomous vehicle software. The paper proposed SafeML by extending SafeML to improve its capabilities for the human-in-the-loop procedure and ECDF-based statistical distance measures, and applies them to image-based classification algorithms in a model-agnostic way. SafeML II improves the ECDF-based statistical distance measure functions using

bootstrap-based p-value calculation. We believe the proposed Safe ml II approach is generic, so it can integrate with traditional safety assurance methods to enable them to provide assurance for ML/AI models and increase confidence in the provided assurance.

[13]The proposed algorithm is currently only tackling the safety evaluation problem of the machine-learning-based classification. However, we believe that expanding it to address clustering, dimension reduction, or any problem that machine learning algorithms can evaluate through statistical differences and can remain robust to a certain distributional shift or variation in the dataset distribution is easily achievable. This may limit the effectiveness of the discussed distance measures. That being said, we can then use the proposed measures as additional confirmation of the robustness, contributing to certification arguments.

As [14] autonomous vehicles and connected vehicles are vulnerable to various cyber-attacks, IDS is one of the efficient solutions to detect the launched network attacks and secure the vehicle networks. IDS is one of the efficient solutions for detecting the launched network attacks and securing the vehicle networks. In this paper, IDS is based on tree-based machine learning algorithms to detect threats both on the CAN bus and on external networks. We introduced the SMOTE over-sampling method and tree-based averaging feature selection approaches to reduce the impact of class imbalance and computational cost.

[15]According to this review, most of the published studies have used KDD99 and proved as the most used dataset in IDS and machine learning areas. Besides KDD99, many researchers in this area of study also used different datasets. Researchers conducted another study to investigate statistical analysis on the KDD99 dataset, which is one of the most used datasets. They found two important deficiencies that highly affect the performance of the evaluated IDS model and resulted in a poor evaluation of anomaly detection approaches. The dataset contains a huge number of redundant records, with approximately 78% of the records duplicated in the train set and 75% duplicated in the test set.

2.2 Overview of Intrusion and IDS

The threat of infiltration, denial of service assaults, and other misuses of computer and network resources is constant in today's world of ever-increasing internet connectedness [7]. Several network security solutions, including firewalls, encryption, antivirus software, and others, are available to address these issues. Another method for securing networks or computer systems is IDS. To identify patterns of intrusion, intrusion detection systems (IDSs) monitor what goes on within a network or computer system [8].

2.3 Categories of IDS

As Figure 2-1 illustrates, we can divide IDSs into several groups according to distinct standards [8]. According to Figure 2-1, the intrusion detection approach allows us to categorize IDSs into different groups based on distinct standards. These groups include anomaly-based IDS and signature-based IDS. While distributed IDS distributes operations among system hosts, centralized IDS performs all operations at a single location. We can further divide IDS into passive and active IDS.

In centralized IDS, a central location performs all operations, while in distributed IDS, hosts of the system distribute operations among themselves. Based on the response after detecting attacks, one can also classify IDS into passive and active IDS. Passive IDS only raises an alarm and does not take any corrective actions after the detection of attacks, whereas active IDS raises an alarm and can take corrective actions after the detection of the attack. The next sections present a detailed discussion of anomaly-based, signature-based, network-based, and host-based IDSs.

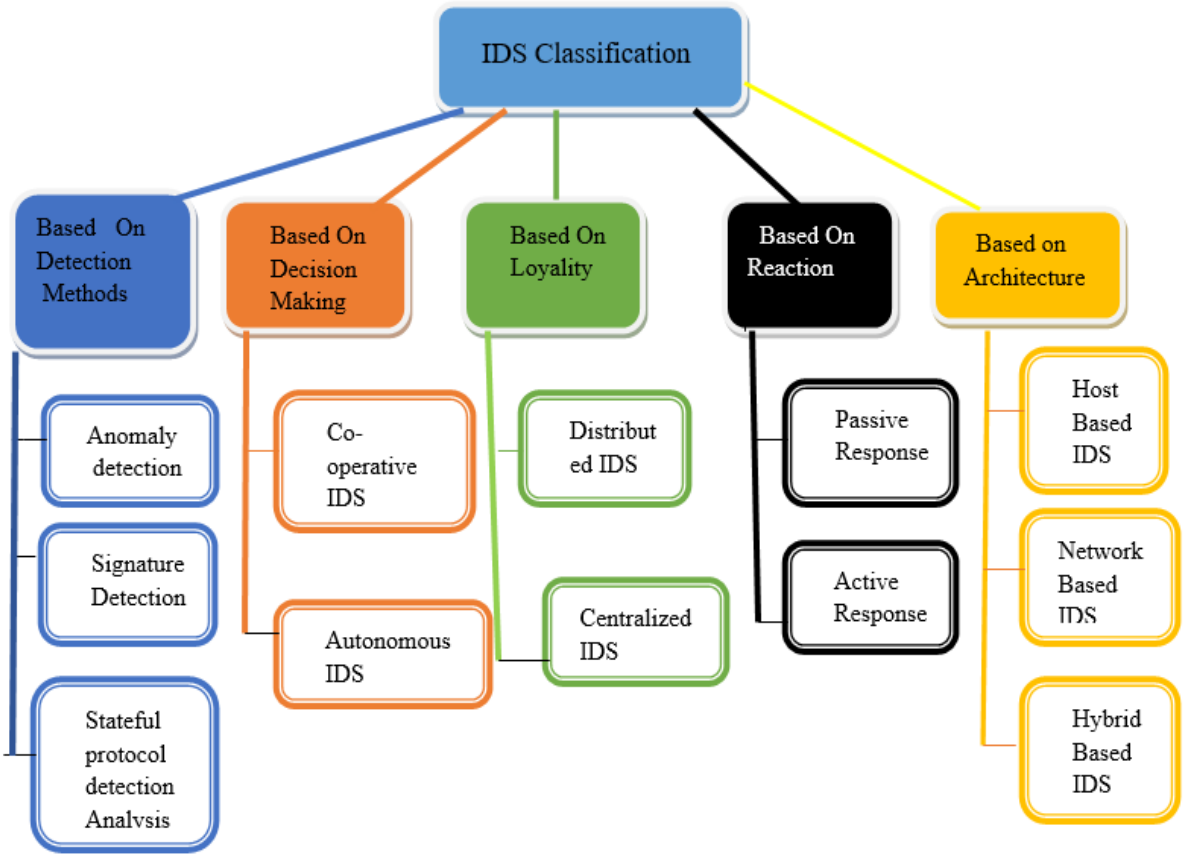


Figure 2.1: Categories of IDS[8]

2.3.1 Anomaly-based IDS

To get access to or compromise a computer system, anomalies try to breach a network or a computer system through the use of malicious behavior [7]. To ensure that the CIA has access to computer system resources, such events need to be identified early on. Misconfiguration of a network or host, overloading of networking infrastructure, malfunctioning of network devices, and malicious activities carried out by attackers or intruders are only a few of the many causes of anomalies [8].

Thus, identifying network attacks is a critical first step in stopping these hazardous occurrences. One technique for intrusion detection systems (IDS) that helps us identify and categorize network anomalies is anomaly-based IDS. Using profiles that list the

resources and services of each allowed user or a group that typically accesses the network, an anomaly detection technique creates a model. Then the system can keep an eye out for questionable behavior by comparing the detected behavior with the constructed model of normal profiles. The system sounds an alert when it detects a departure from the norm. The anomaly-based detection system's ability to identify fresh attacks and high detection rate are its major advantages.

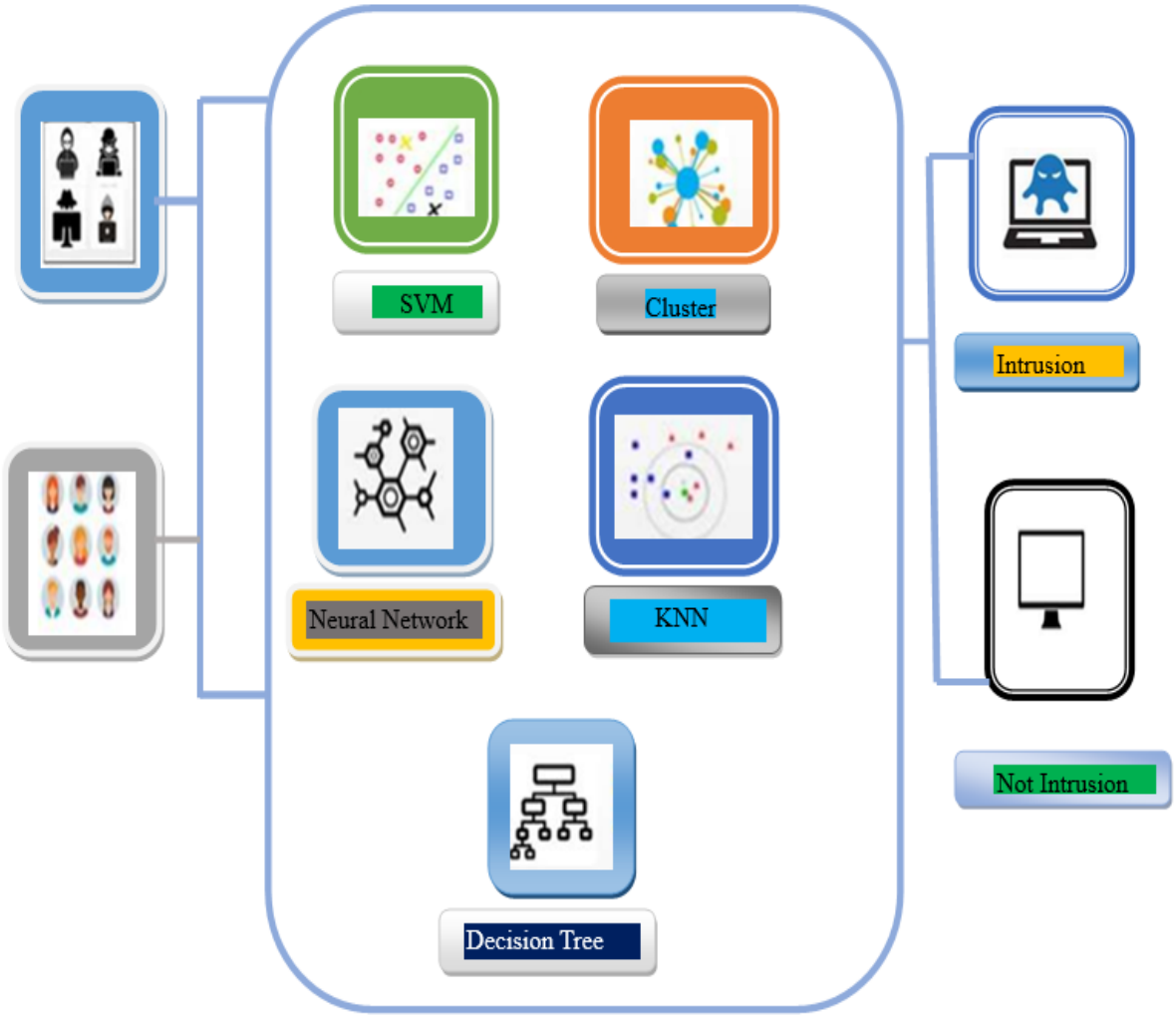


Figure 2.2: AIDS approach based on machine learning[7]

2.3.2 Signature-Based Detection

Thus, one of the most crucial steps in stopping these hazardous occurrences recognizes network attacks. One kind of intrusion detection that helps us identify and categorize network anomalies is anomaly-based intrusion detection. A method of anomaly detection makes Signature detection systems also referred to as abuse detection trigger warnings based on distinctive signatures of known assaults, as opposed to anomaly detection systems, which do so in response to departures from typical network activity [7]. In this context, a signature refers to a collection of features, such as IP addresses, TCP flags, port numbers, network logos, and other network events. This method compares recent activity with the signatures of known attacks kept in a database to find instances of attacks and raises an alarm if it detects any malicious activity. The method of comparing recent activity with the signatures of known attacks kept in a database is more efficient in detecting attacks.

The model's specific design for identifying known assaults results in it rarely producing false alarms. Unfortunately, this method has a low detection rate since it cannot identify unidentified assaults that are not included in the model [7]. An additional issue with this strategy is that additional attacks require constant system updates. Even if there are well-designed signatures, signature-based IDS can't detect unknown attacks because of the lack of an adequate signature database. Organizations that are more concerned with monitoring known attacks from intruders trying to access internal hosts from the Internet can use this approach. A few researches such as [7],utilized a hybrid intrusion detection (the combination of anomaly and signature detection).

Signature-based detection is the simplest form of detection because it just compares the traffic with the signature database. If it finds a match, then it generates an alert. If it does not find a match, then the traffic flows with no problem.

In signature-based detection, detection is based on comparing the traffic with the known signatures for attacks. They can only detect known threats and, hence, are not efficient at

detecting unknown threats. To detect an attack, the signature matching has otherwise, even if the attack has a minor variation from the known threat signature, then the system cannot detect it. For example, in the above example, instead of "I love you" if the subject is "love you, ", the system may not detect the threat. Hence, it is very easy for the attackers to compromise and breach the trusted network [7].

2.3.3 **Hybrid IDS**

IDSs that rely on signatures and anomaly-based IDSs are the two primary categories of TDSs. Signature-based intrusion detection systems use trained machine learning models trained on labeled datasets to identify known attack patterns. They frequently aren't able to identify novel assault patterns that weren't previously recorded in the databases [7].

However, anomaly-based intrusion detection systems (IDSs) can differentiate unknown attack data from normal data through the use of unsupervised learning algorithms. These algorithms operate under the premise that new attack data will statistically more closely resemble known attack data than normal data, but they frequently generate a high number of false alarms [7] .

The detection method is an important aspect of all intrusion detection systems. Signature-based IDS is based on prior threat detection and the creation of accurate signatures. The major advantage of this method is the high accuracy of known attacks. The IDS are, however, unable to detect 0-days and polymorphic threats [16]. 'Misuse Detection' is another name for signature-based detection. Anomaly-based depends on identifying patterns and comparing them to normal traffic patterns. This method requires training the system before deploying it. The accuracy of such a system against 0-days and polymorphic threats is better when compared to signature-based IDS. However, the false positive rate is often high [17].

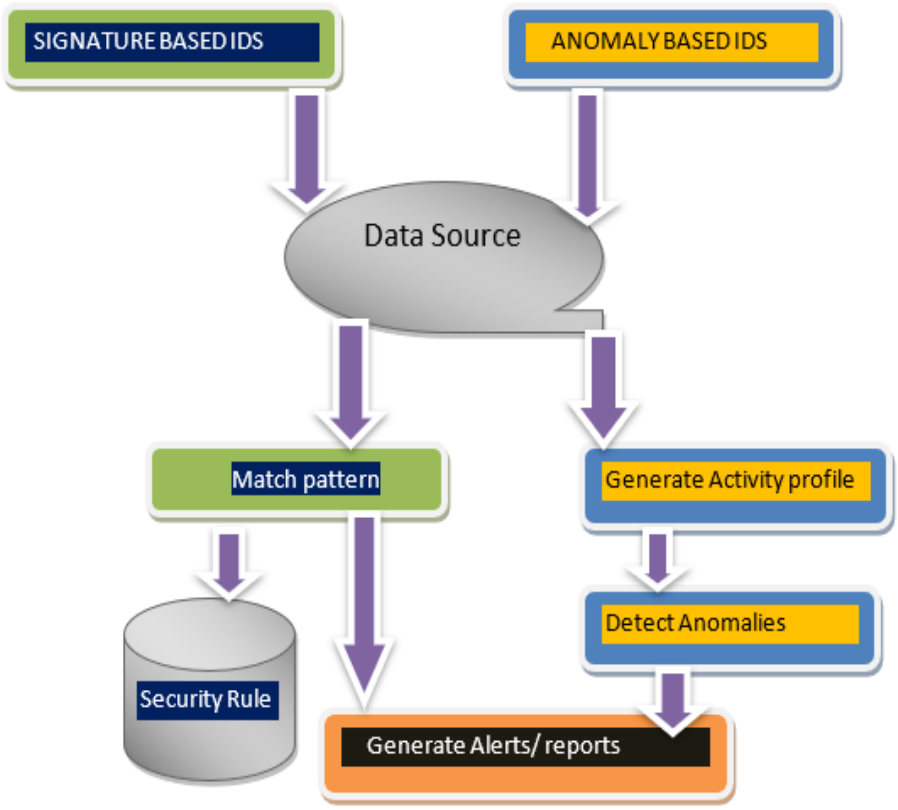


Figure 2.3: hybrid-based IDS[2]

The detection method is an important aspect of all intrusion detection systems. Signature-based IDS is based on prior threat detection and the creation of accurate signatures. The major advantage of this method is the high accuracy of known attacks. The IDS are, however, unable to detect 0-days and polymorphic threats [18]. 'Misuse detection' is another term for signature-based detection. Anomaly-based depends on identifying patterns and comparing them to normal traffic patterns. This method requires training the system before deploying it. The accuracy of such a system against 0-days and polymorphic threats is better when compared to signature-based IDS. However, the false positive rate is often high [18].

2.3.4 Network-Based IDS (NIDS)

A network-based IDS (NIDS) is a software application that monitors and detects network traffic for malicious activity attempted to break into or compromise a network [8]. NIDSs strategically position themselves throughout the system to monitor traffic to and from all devices on the network. It analyzes every LAN passing traffic and looks for identifying attacks in the traffic that is transferred between LANs. The strategically positioned NIDSs throughout the system analyze every LAN passing traffic and actively search for identifying attacks in the traffic transferred between LANs, promptly alerting the administrator upon detecting an assault. For example, as seen in Figure 2-4 [18], the network has installed an IDS at three LANs. Every LAN has an intrusion detection system (IDS) installed, which examines traffic that travels via the network (LAN) to several hosts to identify any unwanted access to the LANs. When the packets detect suspicious patterns, the intrusion detection systems (IDSs) examine both incoming and outgoing traffic. If the intrusion detection systems (IDSs) detect malicious behavior, they raise an alarm to alert network administrators, who can then take action [19].

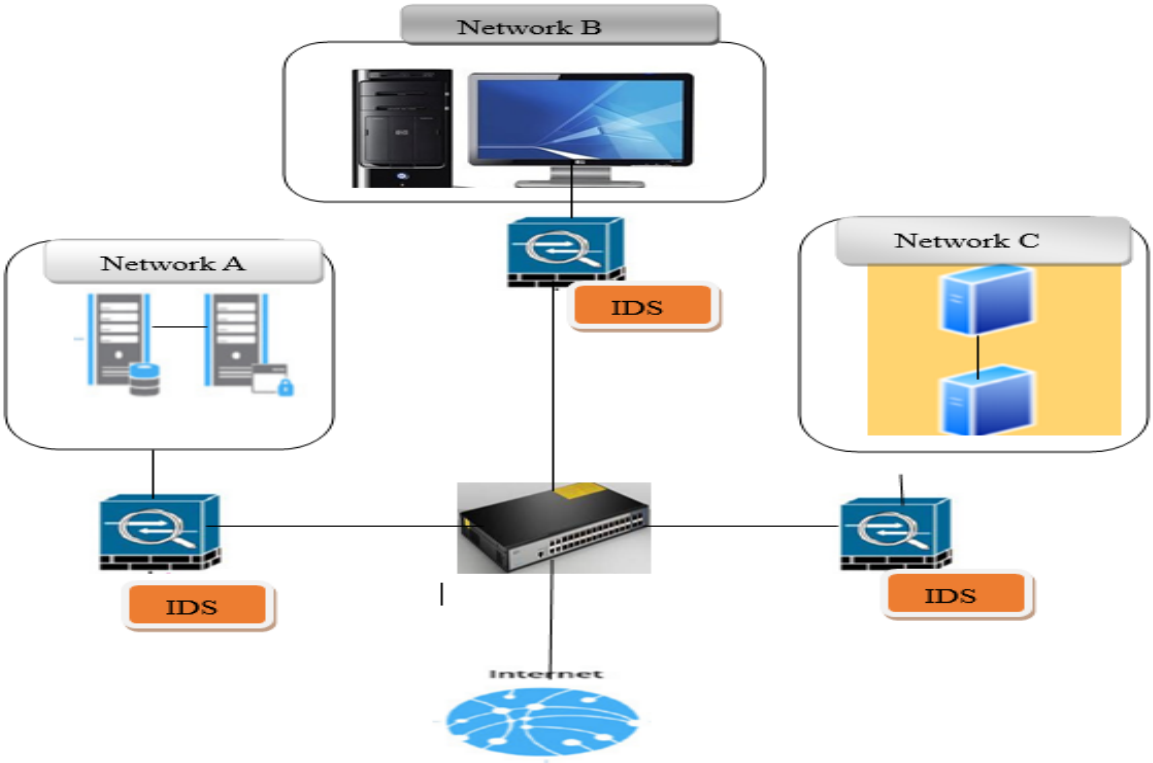


Figure 2.4: Network-Based IDS (NIDS)[9]

2.3.5 Host-Based IDS (HIDS)

A computer, network, or other device that runs its own IDS is called a host-based IDS (HIDS). It primarily monitors and examines a system's internal components to see whether the harmful activity has compromised them. The HIDS has two options when it finds unusual activity: it can either raise an alert or record the activity in the logs. In HIDS, we install a trained model or agent on each host (which could be a web server, computer, firewall, switch, or router) to identify intrusions by analyzing system calls, application logs, file-system modifications like password files, and other host activities. This is in contrast to NIDS, which monitors and detects attacks flowing over the network. Host-based IDS can detect internal attacks as well [20]. Organizations deploy HIDS only on sensitive or critical hosts because it is expensive to place it on every host in the network. In summary, HIDS only provides data relating to the host on which organizations install it, not the network. The drawback of these IDS is that installing the software on several

hosts on the network takes time and can be more expensive. By combining the advantages of both NIDS and HIDS, it is possible to outperform them individually by using their combined strengths. To identify attacks, the NIDS gathers and analyzes network traffic, while the HIDS examines the packets addressed to each host in a hybrid intrusion detection system. While this integration brings benefits, it poses challenges to get different systems to function together and analyze data collected from various systems [21].

2.3.6 Centralized IDS

The centralized IDS manages all IDS operations directly from a central interface with strong processing capabilities and ample storage and memory. All network activity must be able to be monitored by this central console for the deployed IDS agent to access real-time network activity data and instantly analyze the network activity to detect intrusions.[22] .

2.3.7 Distributed IDS

Multiple IDS connected to a network and communicating with a central server to facilitate network monitoring make up distributed intrusion detection systems or DIDS. It can compile data from several sources to identify network threats like distributed denial of service (DDoS) and doorknob assaults. The central analysis server, communication component, and IDS agent are some parts that make up these IDS [23]. Software known as an agent informs the central analysis server about attacks. Deploying multiple agents throughout a network provides the incident analysis team with greater visibility in the network compared to relying on a single IDS system. The various DIDS components connect through messages sent in the communication component to get a system-wide perspective. The primary element responsible for carrying out the system's primary function is the central analysis server. This provides a helpful Web interface to view the current attack status of a network and enables interactive querying of attack data for analysis. It enables analysts to run pre-programmed queries to find attack vectors, like

attack aggregation and statistics collection, using a web interface to identify trends and carry out event analysis. This IDS's capacity to identify attack patterns throughout a whole business network, with geographical portions divided by time zones or even continents, is one of its benefits.

2.4 Artificial Intelligence

The 1950s saw the emergence of artificial intelligence (AI), which is still being researched today. A few pioneers in the still-developing field of computer science wondered if it would be possible to make machines think. AI is a broad field that incorporates many more non-learning techniques besides machine learning and deep learning. [24].

2.5 Machine Learning (ML)

That computers can learn without being programmed to carry out specific tasks gave rise to the field of machine learning (ML). Maybe in the 1990s, scientists and researchers drew attention to it as a subfield of artificial intelligence (AI) that outperforms fixed rule-based models, which need a lot of human labor and time [25]. In traditional computing, a program receives input data and generates output. However, in ML, the ML algorithm uses input data and output data to generate 15 functions or programs that can predict the output based on the learning from the input/output dataset fed to the ML algorithm [26]. Instead of being explicitly programmed, an ML system undergoes training and learns from historical data to optimize computations and provide accurate results.

Nowadays, people employ machine learning algorithms for classification, regression, clustering, or dimensionality reduction tasks of large datasets with high-dimension. As a result, detecting network intrusions comes under ML as it involves the classification of data into normal and abnormal behavior. Network traffic data is increasing; thus, ML helps to recognize the complex patterns in the large datasets by using a learning mechanism to decide or predictions when new data instances are coming. We can classify ML algorithms based on the outcome of the algorithm and the input fed during training as supervised,

unsupervised, or semi-supervised learning [27].Researchers have developed several kinds of intrusion detection systems (IDS) to monitor and analyze network traffic for discovering anomalies in the network. Most techniques rely on anomalies to detect departures from an underlying pattern of typical traffic behavior. The literature claims that the three major methods used by the current ML-based intrusion detection systems are as follows [28]:

- Unsupervised learning
- Supervised learning
- Hybrid learning

2.5.1 **Unsupervised Learning**

Unsupervised ML techniques facilitate the analysis of raw datasets, helping to generate analytic insights from unlabeled data. It has many applications, such as feature learning, data clustering, dimensionality reduction, and outlier detection [29]. Clustering algorithms is one of the unsupervised learning techniques that significantly advances the state-of-the art in unsupervised ML techniques. In this study, we apply a Clustering algorithm to detect anomalies or unknown attacks in computer networks [30]. Anomalous behavior is dynamic. New kinds of anomalous behavior might appear supervised learning needs labeled training data to find them. However, the clustering algorithm is a technique that can identify unknown attacks with no prior knowledge of them and naturally group data based on patterns that are like one another. [30]. The cluster takes an unlabeled dataset as input.
Algorithm and the method attempt to seek intrusion instances inside the data. Unsupervised learning is an interesting technique as it uses unlabeled datasets where it can unconstraint us from the need for labeled data [31].

2.5.2 **K-Means Algorithm**

In machine learning, the most crucial unsupervised learning method for identifying patterns in a set of unlabeled data is clustering. Initially, we divide the training cases into k groups using Euclidean distance similarity and the k-means clustering technique [32]. The distribution of n observations into K clusters, with each observation belonging to the closest cluster, is the general outcome of K-means clustering. In a dataset, n is the number of observations or feature values, and K is the number of clusters that the user inputs. The cluster mean is used to determine if an observation belongs to a cluster. There are several uses for K-means clustering in the fields of intrusion detection and network security. A density zone of normal or anomalous instances in each cluster. When constructing the machine learning model using the clustered dataset as input, the classifier appends the normal or attack label to the original dataset in order to enable the identification of new anomalies [33]. The below Figure 2-5 shows the working process flow of the K-means background.

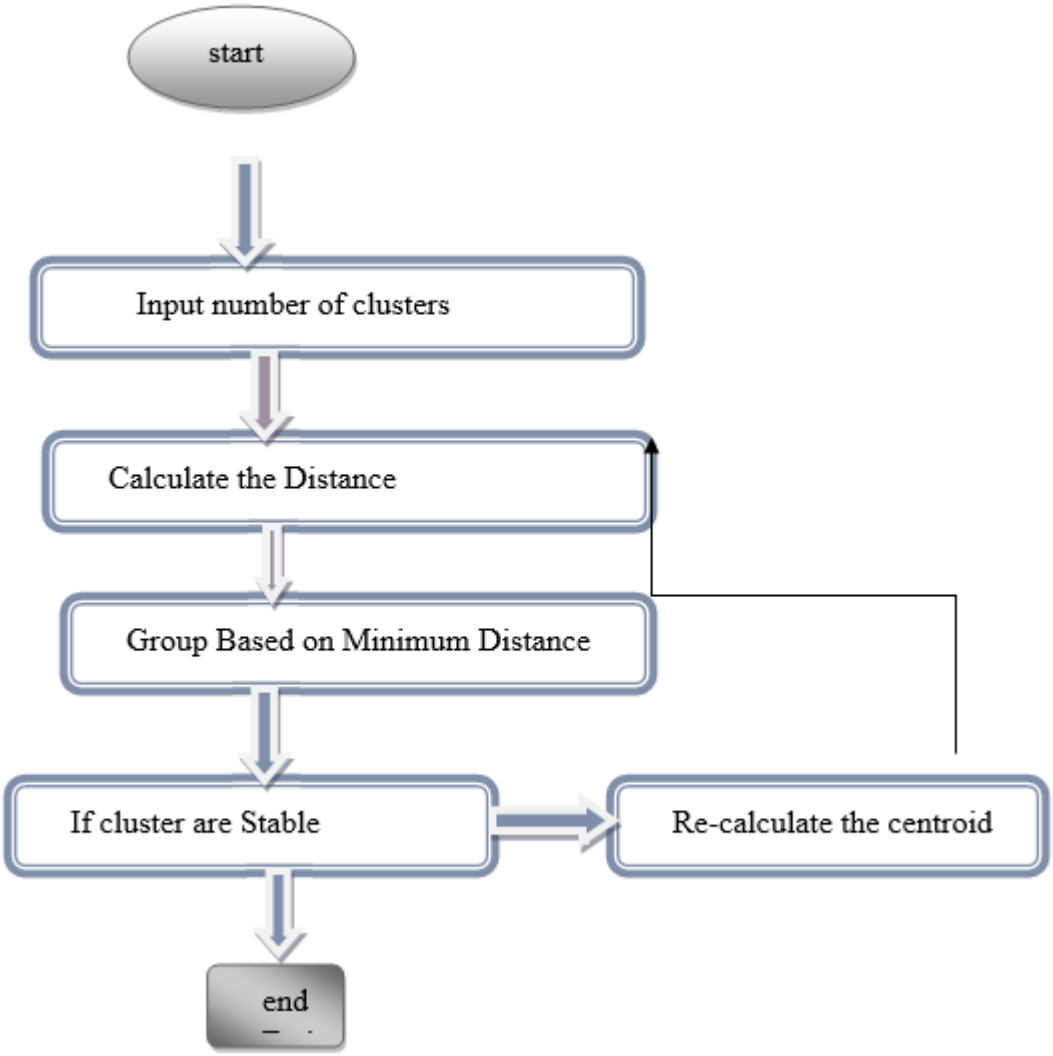


Figure 2.5: K-means process flow[34]

2.5.3 Supervised Learning

The supervised learning dataset includes examples of normal and anomalous classes. The inputs for the supervised learning machine learning approach are a set of predictive variables or features (X) and a set of labels (Y), where each entry in X must have a label assigned by a member of set Y. In this way, the set Y acts as the instructor for the ML models, guiding them during training. This model is used to analyze freshly created data instances to classify them in a class. This kind of ML frequently solves classification and regression problems. Some of the most used supervised learning algorithms are decision

tree (DT), neural network (NN), and support vector machine (SVM) [35]. The researchers chose DT and NN as supervised learning techniques in this study.

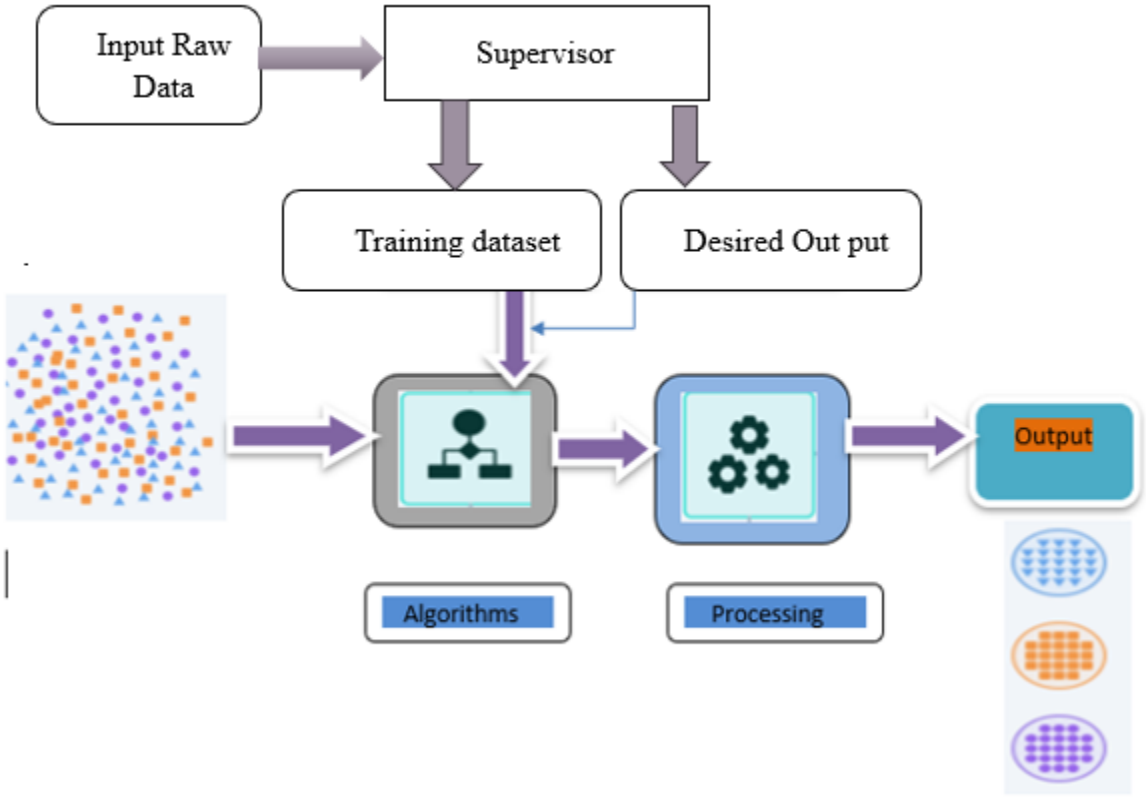


Figure 2.6: Supervised Learning[36]

2.5.4 Semi-supervised Learning

Researchers use a semi-supervised learning method, which combines supervised and unsupervised learning. When they identify a smaller number of labeled data for a particular application [37]. It generates a function mapping from inputs of both labeled data and unlabeled data. The goal of this learning mechanism is to classify some of the unlabeled data using the labeled set of information. The quantity of unlabeled data should be higher than the number of labeled data.

2.6 Some Popular Classification Algorithms

This section presents some of the common ML classification algorithms with their strengths and weaknesses.

Logistic Regression (LR): The logistic regression algorithm considers a linear model that measures the relationship between the target variable and other variables by estimating the probability. It uses a logistic function as shown below [37]**.** Statisticians initially created it to explain the characteristics of ecological population expansion. To predict an output value (y), LR uses 17 weights, or coefficient values, to integrate input values (x) linearly. A value between 0 and 1 results from LR. When the results are closer to 1, it is more obvious that the input variable fits into the category; when the results are closer to 0, it is more apparent that the input variable does not. This approach is appropriate because the model is simple to interpret mathematically.

$$\mathbf{Y} = \frac{e^{Bo+B1xX}}{1+e^{Bo+B1xX}|}$$

Where Y is the predicted output, B0 is the bias or intercept term, and B1 is the coefficient for the single input value (χ).

Making predictions with an LR model is as simple as plugging in numbers into the LR equation and calculating the result. However, one challenge with this algorithm is that we cannot understand the predictions as a linear combination of the inputs.

The logistic regression hypothesis limits the cost function to a value between 0 and 1, making linear functions unsuitable for this task. Various fields such as finance, marketing, healthcare, and social sciences find applications for logistic regression, where it models and predicts binary outcome [37].

Logistic regression is helpful for binary classification problems. It provides probabilities for outcomes, which can be a measure of confidence in predictions. The model allows for interpretation.

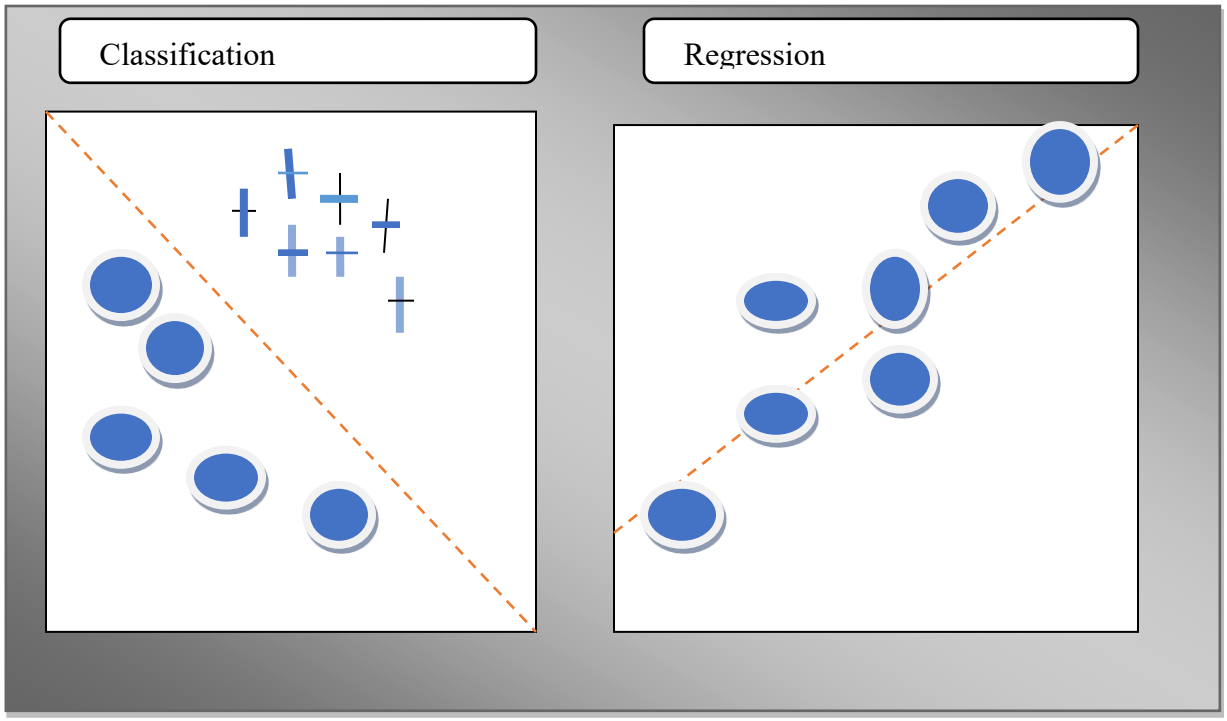


Figure 2.7: Classification and Regression

Decision Trees (DT): A decision tree (DT) is a supervised machine learning method that may create a learning model that learns the decision rule using training data to predict a target variable's class or value [38]. This approach can resolve regression and classification issues. DTs organize themselves into a hierarchical tree by dividing the training data into several groups to identify homogenous subgroups [39]. Knowing which attribute to be placed at the root and each level is the major challenge in constructing a DT. Entropy and information gain are statistical measures that are used to construct a tree in the DT. Entropy deals with calculating the homogeneity of information. To get the total entropy of a dataset, we first split the dataset based on unique attributes. Then, we calculate the entropy for each branch using Eq. [40] and add the entropy for each branch proportionally.

$$E(S) = \sum_{i=1}^{c} - Pi \log_2 Pi$$

Where E(S) is the entropy of the dataset, and Pi is the Probability of randomly picking an element of class i.

Information gain (IG) deals with calculating the expected reduction in entropy. To calculate IG for a split, we use Eq. to subtract the weighted entropies of each branch from the original entropy (entropy before split), which is 18. We consider the attribute with the maximum IG for splitting.

$$Info(A) = E(S)-E(X)$$

Where Info(A) represents the information gain of a specific attribute 'A', E(S) represents the entropy of the dataset given in this equation, and E(X) represents the entropy of a specific attribute in the dataset.

After calculating the IG for each attribute, we rank the attributes in their order and place the attribute with the largest IG as the decision node in the DT. Although IG is a good measure for deciding attribute relevance, it has a bias problem when it applies to attributes with many values [38]. One solution to this problem is using the gain ratio method. Gain ratio (GR) reduces the bias on multivalued attributes by considering the intrinsic information of a split when choosing an attribute using Eq below. The GR takes the number and size of branches into account when choosing an attribute. Intrinsic information is the entropy of the distribution of instances into branches (i.e., how much information do we need to tell which branch an instance belongs to).

$$\text{Gain ratio (Attribute)} = \frac{\text{Gain (Attribute)}}{\text{Intrinsic info (Attribute}}$$

DT algorithms are suitable for multiclass classification. But the problem with DT algorithms is overfitting [39].

Naïve Bayes. An effective and straightforward predictive modeling approach is Naïve Bayes. People regard Naïve Bayes as a probabilistic classifier because it is based on the Bayes' probability theorem. Using the training data as a starting point, the model can compute two different probabilities: (i) the probability of each class and (ii) the conditional probability for each class given each x value. After calculating the probability, the model

can forecast fresh data by applying the Bayes' theorem, as shown.

$$\mathbf{P(Ci/O) = \frac{P(O/Ci)\ P(Ci)}{P(O)}}$$

where P (Ci | O) is the posterior probability of being a class label "Ci" given the attribute value" O", P (O | Ci) is the probability of the attribute "O"

when the class label is "Ci", P (Ci)is the prior probability of the class label "Ci", and P(O) is the prior probability of the attribute value "O".

Naïve Bayes' classifier is preferable when the dataset has a small size and non-correlated or independent attributes. A dataset with many attributes would take a long time to calculate the posterior probabilities, P (Ci | O).

Support Vector Machine (SVM): SVM works by constructing hyperplanes for linearly separable data in multidimensional space using a kernel function [41]. The hyperplanes divide into various groups the sample data. By using this kernel function, the SVM algorithm rearranges objects for the best possible separation. A mapping that is nonlinear from inputs x to feature space f(x) is provided by a kernel function. A straight line with the extra space can divide the mapped item during the mapping process from the input space to the feature space. In high-dimensional space, vectors represent all instances and target classes. The Support Vector Machine (SVM) determines which two points from the two classes are closest to each other and supports the optimal separating line or hyperplane, as illustrated in Figure [42].

Support Vector Machine (SVM) is a powerful machine learning algorithm used for linear or nonlinear classification, regression, and even outlier detection tasks. SVMs enable the performance of a variety of tasks, including text classification, image classification, spam detection, handwriting identification, gene expression analysis, face detection, and anomaly detection. SVMs are adaptable and efficient in a variety of applications because they can manage high-dimensional data and nonlinear relationships' algorithms are very effective as we try to find the maximum separating hyperplane between the different

classes available in the target feature.

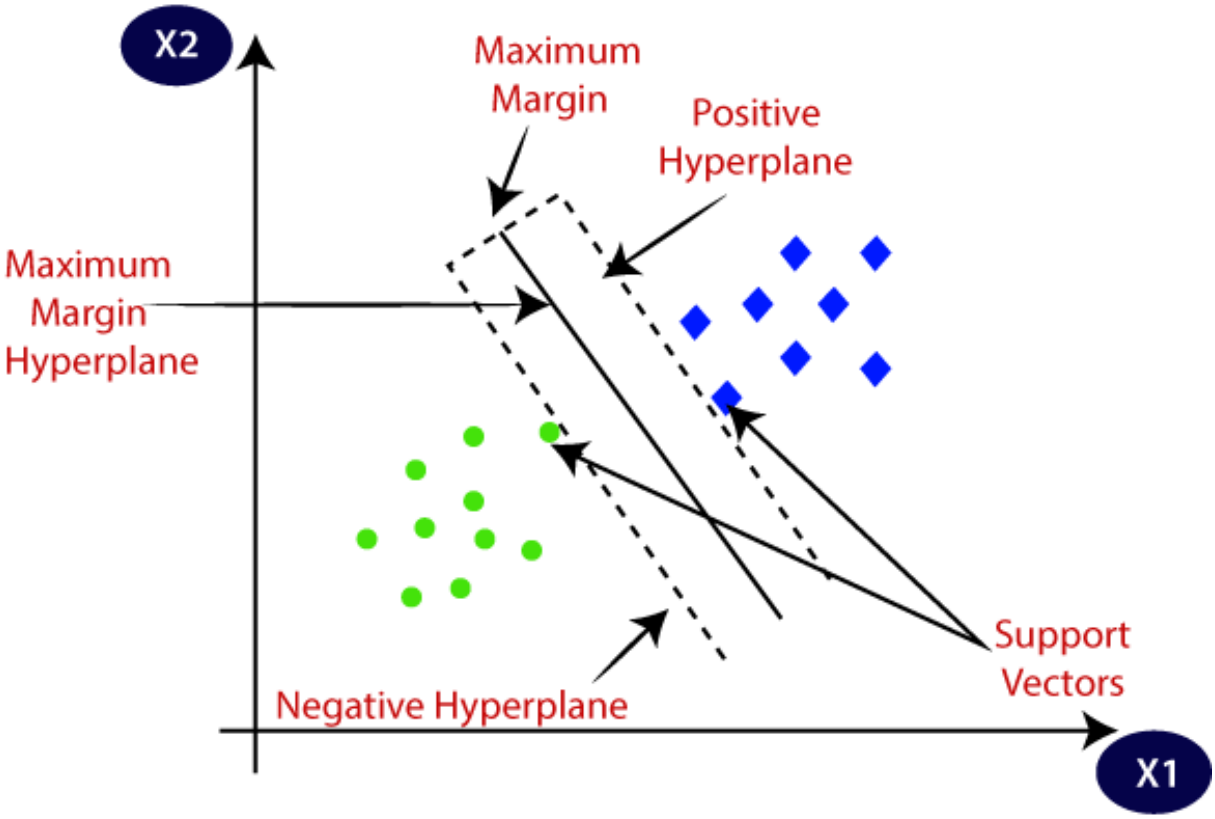


Figure 2.8: Support vector machine for linearly separable data[42]

Even though researchers initially developed SVM to classify linearly separable data into two classes, modern SVM algorithms can now classify data with over two classes and can handle non-linear separation. A multiple-binary support vector machine is used to classify datasets in multi-class issues. SVM handles classification and regression tasks with lightning-fast training speed.

Random Forest: An ensemble classifier called Random Forest fits many DTs to a dataset before combining the predictions made by each tree. Ensemble methods are ways to generate more powerful models by combining several machine learning models. Using a random forest algorithm, the voting mechanism is used to estimate which output class will receive the most votes from each DT [43]. The random forest algorithm allows us to perform classification and regression. The RF algorithm starts by randomly selecting a subset of

training data and generates a single DT with the sub-sample. We repeat the random selection of examples many times, generating a single DT with each sub-sample. These DTs finally form a forest. Figure 2-10 [44] shows RF on a sample of the IDS dataset where we select random samples from the original dataset and construct trees based on these sub-samples. After constructing the forest, individual trees make a classification decision or prediction, and then the final predicted class of observation is determined by the majority vote. One of the major advantages of RF is that it reduces the risk of overfitting.

RF is simpler to construct and train than DTs, and it frequently produces excellent results with a lower risk of overfitting. This results from the voting system that is used to decide and the random selection process that applies to subsets of variables. [44]. RF is appropriate for high-dimensional data modeling because it can handle missing values and can handle continuous, categorical, and binary data.

The Random Forest Algorithm's widespread popularity stems from its user-friendly nature and adaptability, enabling it to tackle both classification and regression problems effectively. The algorithm's strength lies in its ability to handle complex datasets and mitigate overfitting, making it a valuable tool for various predictive tasks in machine learning [45] .

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as with regression and categorical variables, as with classification. It performs better for classification and regression tasks. In this tutorial, we will understand the working of random forest and implement random forest on a classification task[45].

Random forest adds additional randomness to the model while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that results in a better model.

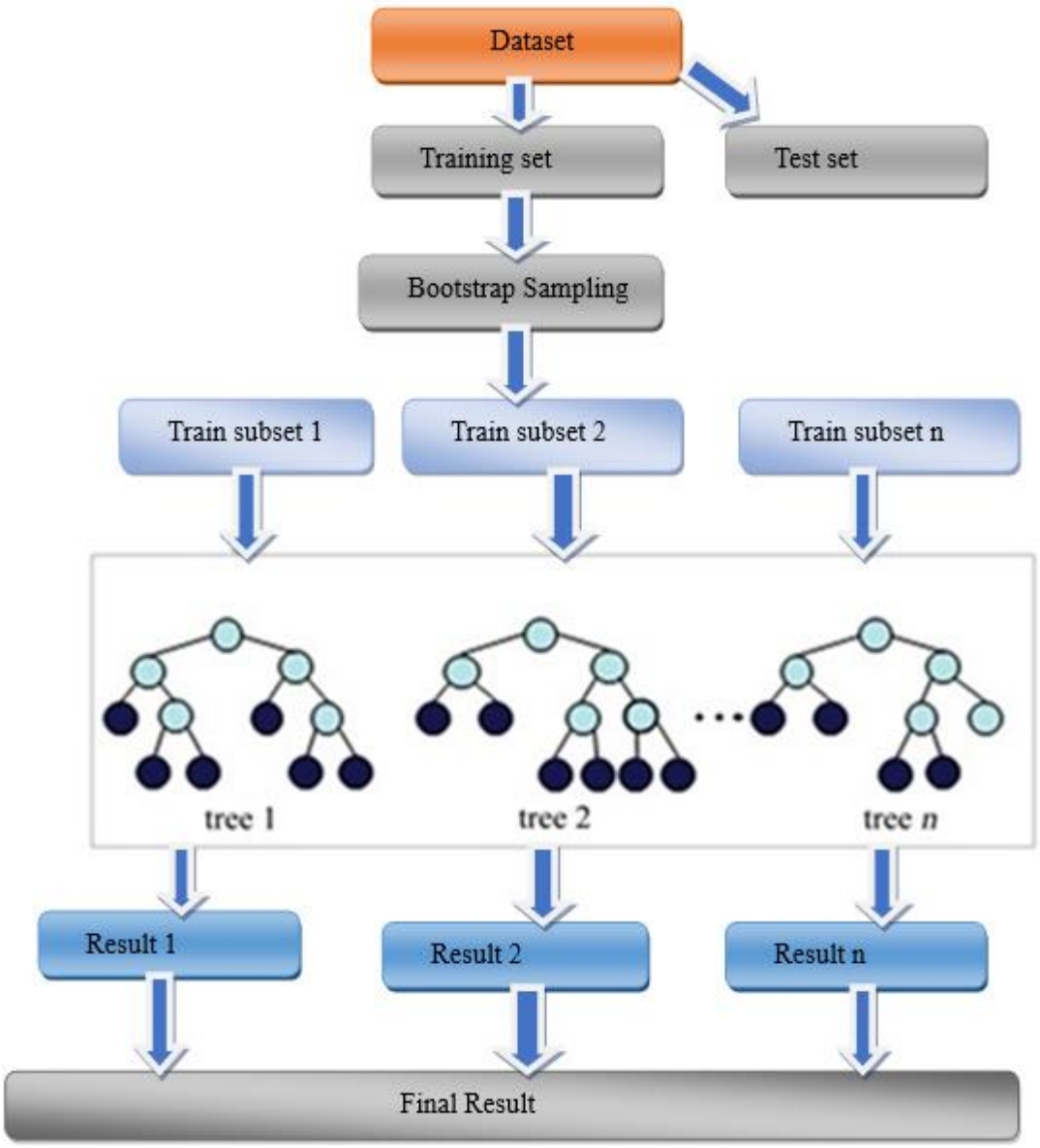


Figure 2.9: Random Forest[45]

The ensemble scheme of RF makes it enough to overcome the problems of overfitting and hence there is no need to prune the trees. Besides, its high prediction accuracy, efficiency, and interpretability make it preferable for various types of datasets [46].

2.7 Deployment Scenario for IDSs

There are several ways to deploy sensors, either network-based or host-based. The sections below show two ways of deploying IDSs in network-based architecture and a single way to host-based.

2.7.1 Before the Firewall

At this stage, the NIDS can monitor all noteworthy network activities, including attacks that might later backfire. To perform real-time analysis and handle high volumes of traffic, we should place NIDS on a faster system. We must set it correctly to lower the number of false alarms. The figure illustrates how to set up IDS before the firewall. [47]

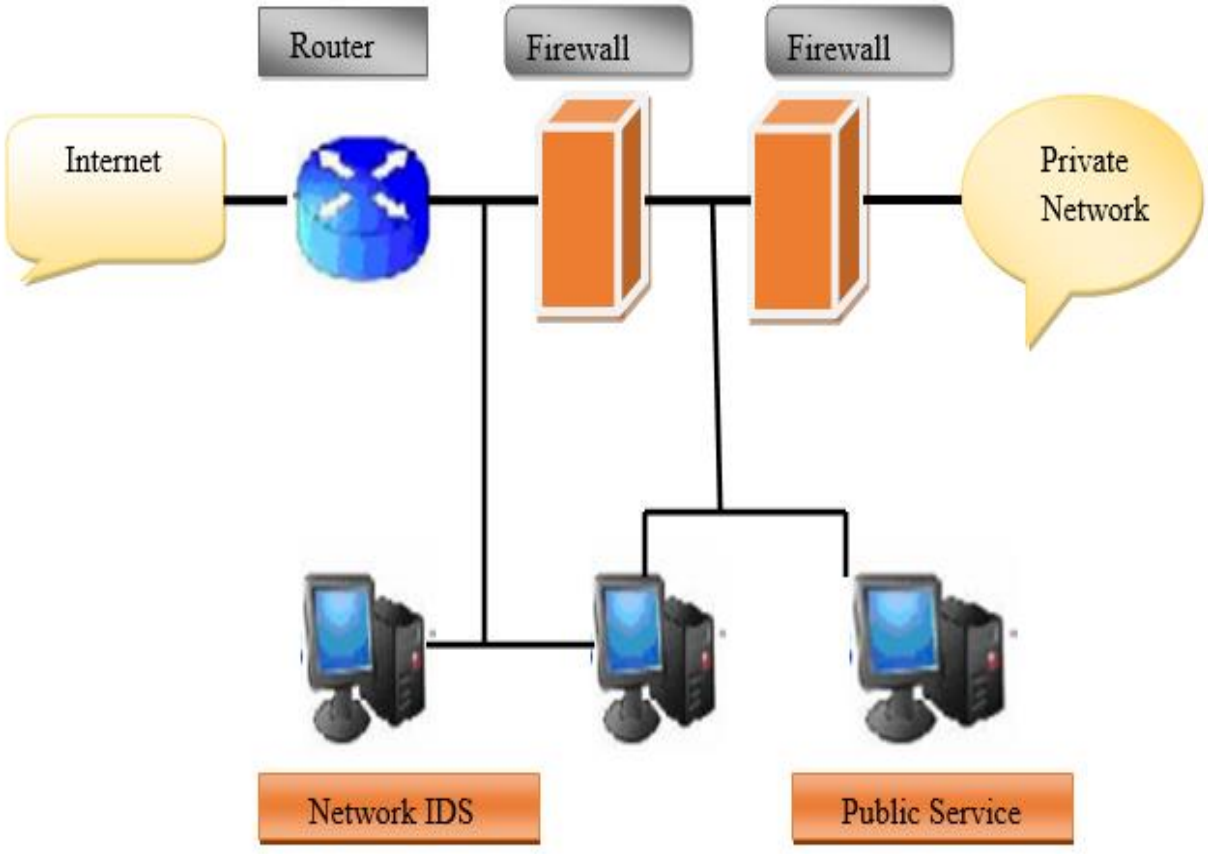


Figure 2.10: IDS Before the Firewall[47]

2.7.2 **Inside the Private Network**

The corporate network, as shown in Figure [48], is another possibility for stationing NIDS. This results in lower application demands because there are fewer potential attacks here than in the scenarios that came before. IDS raises a few false alarms in this situation. Because of the limited visibility within the corporate network, we will not detect failed attacks as we did in earlier instances.

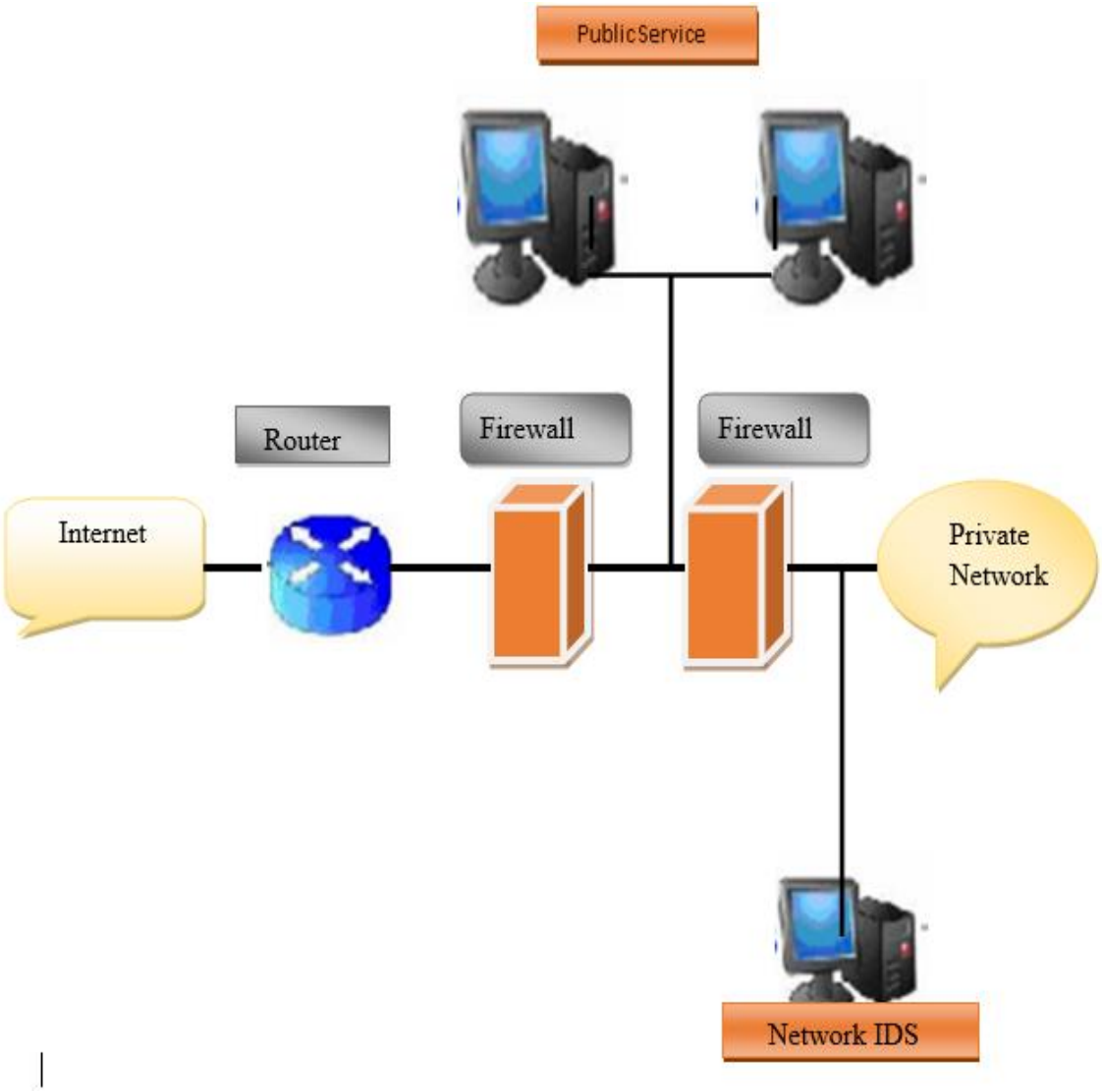


Figure 2.11: Inside the Private Network[47]

2.7.3 Deployment on Individual Hosts

Hosts will have host-based intrusion detection systems installed on them, which collect data from operating system audit trials or the hosts' system logs. It verifies the integrity of the system file besides monitoring all communication coming into and going out of a single computer. Figure 2-16 shows sample host-based IDS.

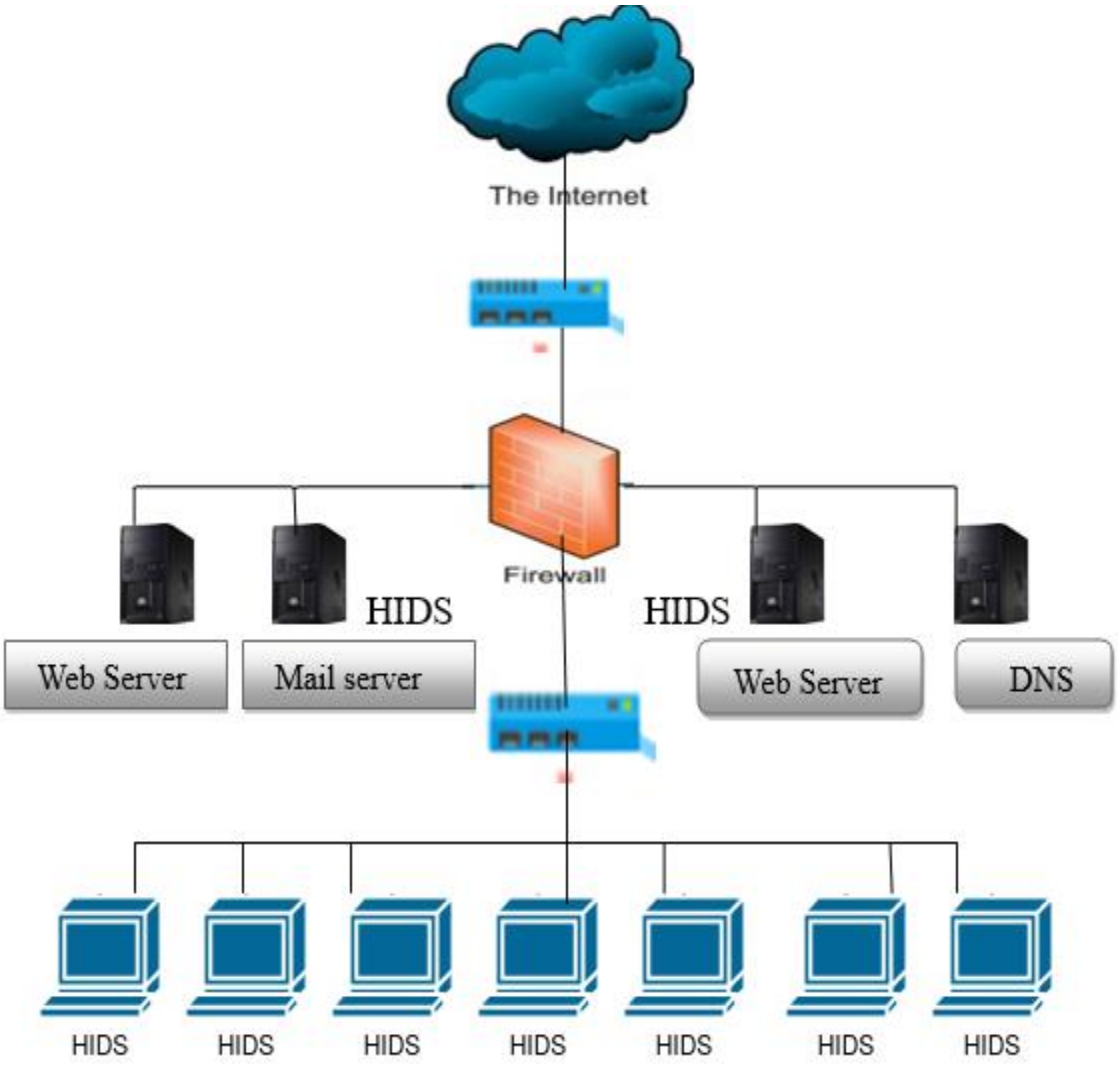


Figure 2.12: Deployment on Individual Hosts

2.8 Safe Machine Learning Idea

The purpose of Safe ml's concept was to monitor the classifiers' judgments somehow in the absence of a label [4].

Classification machine learning methods are commonly used to classify input samples into predefined categories. For example, determining whether a sample falls within predetermined ranges or categories can discover abnormalities. One-dimensional data can use a threshold or line as a basic model for a classifier. Take into consideration a hypothetical measurement t (such as time, temperature, etc.) and the classifier D that is specified as (1) and illustrated in Figure 2-15. Note that Figure displays the actual classes of the input.

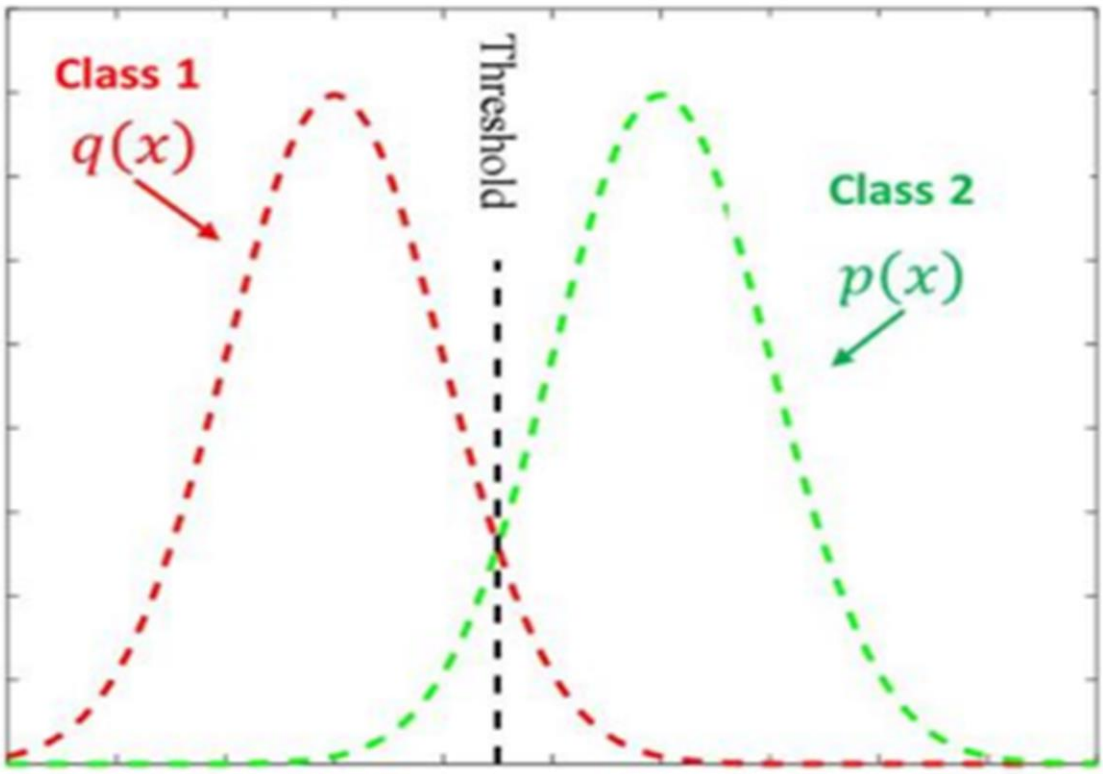


Figure 2.13: Probability density functions of class 1 and class 2[5]

2.9 Safe Machine Learning Method

Let me first clarify that even though the study's focus is on machine learning classifiers. There are two primary phases in this flowchart: the application phase and the training phase [3].

A) During the "training" phase, developers train the ML algorithm offline using a reliable dataset. After finishing the training, we use user-defined KPIs to assess the classifier's performance. In the second stage, we calculate and save the statistical parameters and PDF of each class for later comparison [48] .

B) The second, or "application," phase is an online process wherein the system receives real-time, unlabeled data. For instance, think about the machine vision system in an autonomous car. A system like this has trained to recognize impediments, among other things, so that the car may steer clear of them. One of the many functions of such a system is its ability to recognize impediments, which enables the car to steer clear of them. Unlabeled incoming data is a crucial issue to be aware of throughout the application phase. As a result, there is no guarantee that the classifier will function with the same level of accuracy as it did during training.

Gathering input samples allows for the estimation of each class's PDF and statistical parameters . The system relies on having enough samples to determine the statistical difference reliably, so it may be necessary to accumulate a buffer of samples before proceeding. Using the changed Chernoff error bound, the system compares the statistical difference between each class in both the training phase and the application phase [13].

If the statistical difference is tiny, one can rely on the accuracy and results of the classifier. In the previously given example, the driverless car would carry on operating in this scenario. Rather, the classifier accuracy and results are no longer regarded as genuine if the statistical difference is larger (because the gap between the training and the observed data is too significant). In this situation, the system ought to either alert a human operator or take a different course of action [48].

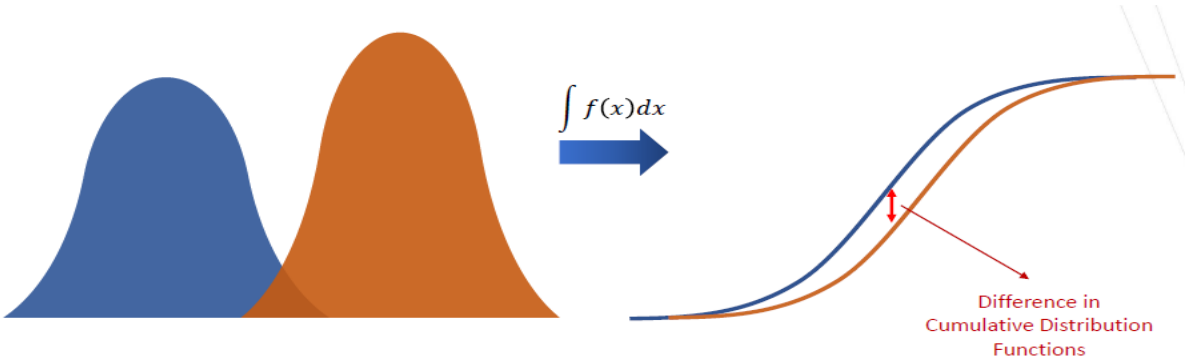


Figure 2.14: Probability Density Functions and Cumulative Functions[5]

The following figure shows four well-known PDF-based distance measure methods.



Figure 2.15 PDF-based distance measures[3]

2.9.1 Chernoff and Bhattacharyya-based Upper Bound Error

The following Python example shows the upper bound error probability estimation using the Chernoff approach. If this code considers " = 0.5," it will use the Bhattacharyya upper limit error estimation [4].

[3] have showed a correlation between the chance of error and the distance between the cumulative distribution functions (also known as CDFs). The following is a list of some of the

more well-known CDF-based distance measures:



Figure 2.16: ECDF-based distance measures

Sometimes it can be easier to use the empirical cumulative distribution function (ECDF) of features. The following Python examples provide ECDF-based distance measures [13].

2.9.2 Kolmogorov-Smirnov Distance

Smirnov-Kolmogorov Distance Let's say we have a dataset that has one feature and two classes. The accompanying figure displays the ECDF of the feature for classes 1 (blue) and 2 (red). We can use Kolmogorov-Smirnov to determine the greatest distance we can measure between two ECDFs [5].

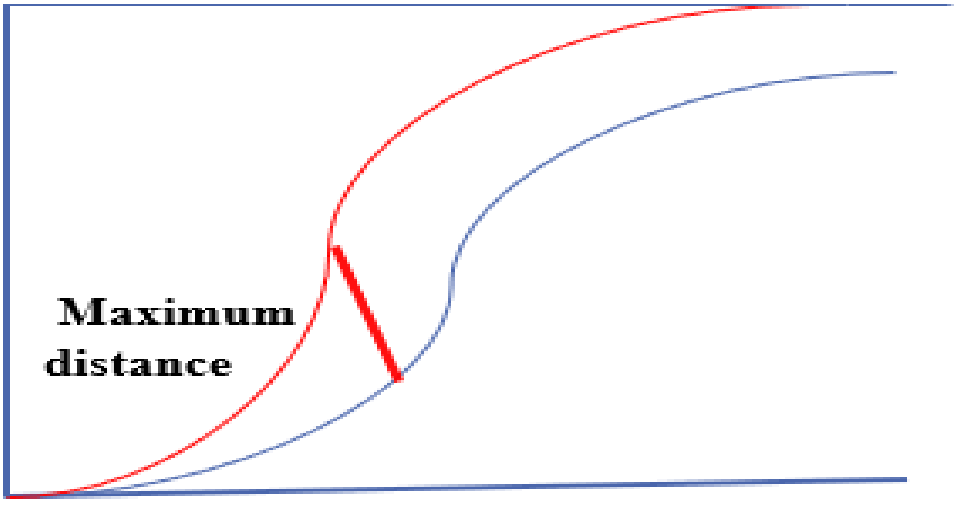


Figure 2.17: Kolmogorov-Smirnov distance measure Illustration[5]

2.9.3 **Kuiper Distance**

The Kuiper distance has a similar functionality to the Kolmogorov-Smirnov distance. However, this method considers two maximum distances, as shown below; a) when blue ECDF has a larger value than red ECDF and b) when red ECDF has a larger value than blue ECDF. You can get the Kuiper distance by adding the two maximum values [13].

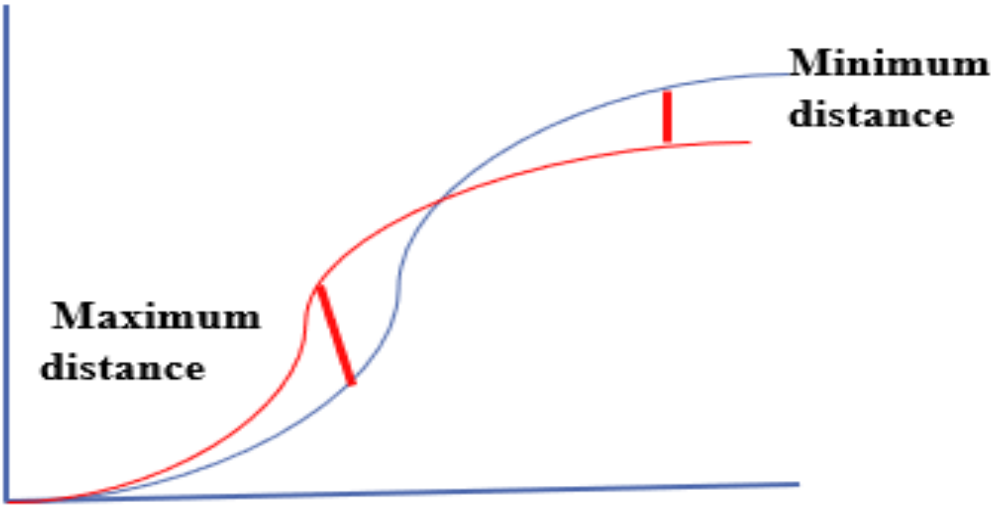


Figure 2.18: Kuiper Distance[5]

2.9.4 **Cramer-Von Mises Distance**

Many small steps form ECDF. Consider the absolute difference of two steps in the same interval as height. If we calculate the summation of all calculated height values for all steps, then we have the Cramer-Von Mises Distance.

Cramér-von Mises's test is an empirical distribution function omnibus test for the composite hypothesis of normality. It uses the summed squared differences between observed and expected cumulative proportions as the test statistic. Statistic definition. The test statistic uses a quadratic distance211 between Fn and F0 [13]:

$$W2n: = n\int(Fn\ (x) - F0(x))2dF0(x)$$

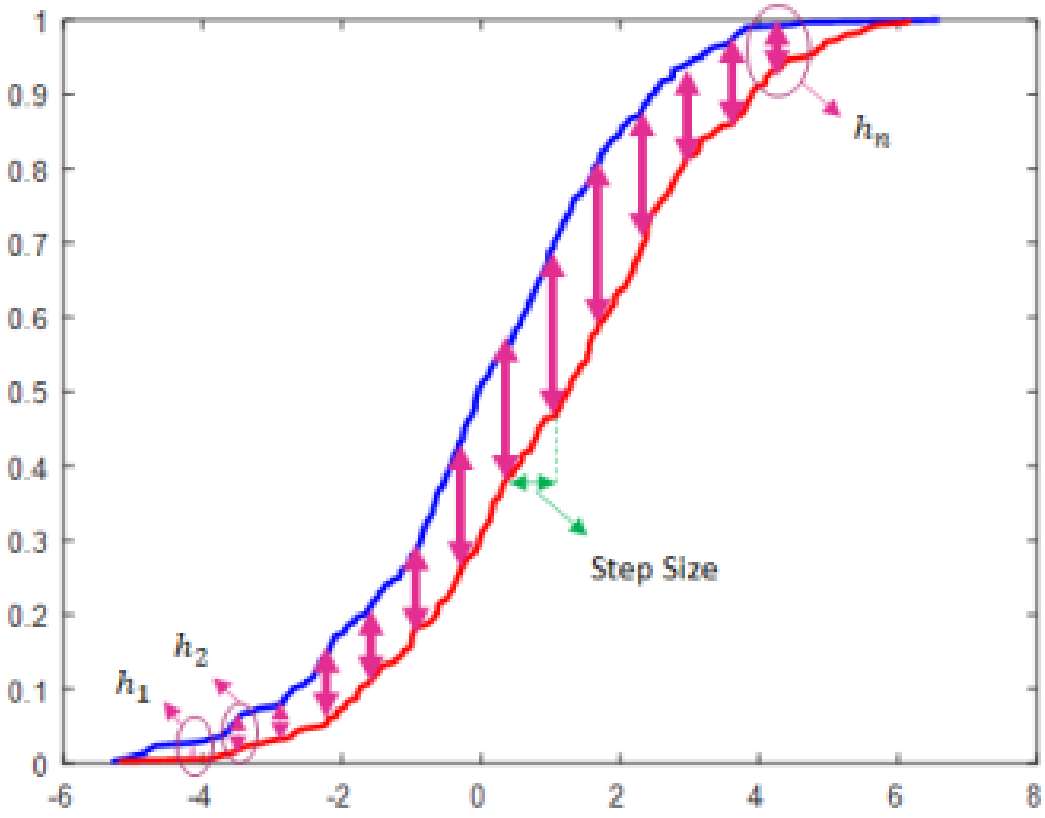


Figure 2.19: Cramer-Von Mises Distance[5]

2.9.5 **Wasserstein Distance**

There are many uses for the Wasserstein distance. In generative adversarial neural networks (GANs), for instance, researchers have employed it as a loss function. For every step, we consider both the height and breadth values of the Wasserstein distance. When we set the power to unity, this algorithm calculates the area between two ECDFs. The Wasserstein distance is equal to the Earth Mover distance when the power factor is 1[3].

2.10 Research Gap

The research community has developed many intrusion detection frameworks and systems, but IDS performance, safety, security, and zero-day attack detection remain open research issues and challenges. One can consider the safety problem of AI and ML algorithms from different perspectives, and investigating the distributional shift in datasets is one of them. Some researchers are investigating this issue and trying to make the existing ML classifiers more robust. They usually use different approaches, such as adding noise, perturbation, and rotation, to make sure their classifier is robust to those changes.

Safety machine learning is still in the early stage of machine learning research, but the most important one for machine learning. As a result, this study identified a gap and attempted to Solve it by doing this research. Most intrusion detection systems on the market today, as well as those that are freely accessible online, use a signature-based method. The fundamental issue with this strategy is that it only focuses on signature databases and requires prior information; it is unaware of unknown attacks and relies solely on rule-based analysis. A specified collection of previously constructed rules is the foundation of rule-based analysis. The template has to be updated often to stay current. This means that even if the sequence of events deviates little from the predefined profile, the system is rigid and cannot identify an attack. One day, the rule-based system might break if the invader is clever.

Many earlier studies on intrusion detection systems have focused less on complexity, false alarm rates, detection rates, and the examination of both known and unknown threats. To close the gap left by the aforementioned issues is the goal of this research project.

The SafeML team is still in the early stage of developing and aims to extend it to deal with time-series data, prediction and regression algorithms, and domain adaptation. It is also possible to use SafeML as an Explainable AI.

CHAPTER THREE

3. METHODOLOGY

3.1 Overview

This study uses design science as its method of inquiry and presents a design for its suggested intrusion detection system. The study details various parts of the suggested IDS, along with their importance and construction methods. It comprises the research strategy, the type and sources of data, the tools and procedures, algorithms, data analysis and display, and assessment metrics [49].

To conclude, we employed research methodologies to gather, arrange, and examine the data about the study goal. Research methodologies positively affect research objectives, convince the public with convincing conclusions, and make suggestions .

The IDS based on ML components, DS employing safe machine learning and security, including data pre-processing, the IDS structural model, detection methods, and evaluation metrics, are all summarized in this chapter. We use IDS, SPSS, and Python to create an IDS based on machine learning. We convert the data from the CIC-IDS2017 dataset into vectors by performing entirely numeric conversion. The technique referred to as data pre-processing [50] involves using IDS, SPSS, and Python to convert the data of the CIC-IDS2017 dataset into entirely numeric vectors.

3.2 Research Design Approach

In this study, the researchers used both qualitative and quantitative research approaches because incorporating both approaches would increase the overall strength of the study. Researchers use a mixed approach or a combination of quantitative and qualitative research approaches when they use more than one data collection technique with an associated analysis technique [51].

Using a machine model and a design science approach, we detect an unknown assault and a known attack in this study. The literature review changed the design of the research project. Surveying the review literature is a crucial first step to selecting a particular topic for in-depth research and gaining a thorough understanding of the study field. This allows the domain of challenges associated with a topic and its comprehension level to be revealed. We can see the significance of prior research on intrusion detection systems in this literature. To identify and point in the proper direction for resolving the issues that were discovered, we considered existing works related to this scientific endeavor [49].

Research design offers a sensible framework for collecting and analyzing research data. The investigator employed the Descriptive Research Design kind of research design. This research design framework served as a roadmap for gathering, analyzing, preparing, and presenting data. It is the type of study design that is most universal.

For analysis attacks in this work, we choose an ECDF-based statistical distance measure, a machine model, and SPSS. Setting attainable goals is the second step in resolving the problem and concentrating on the extent of the study. The final stage involves collecting information from various datasets and searching for trends that can distinguish known and unknown attacks from routine data. In the train-test split step for machine learning techniques, we divide the preprocessed data into training and testing sets. In the validation split phase, we divide the training set into the actual training and validation sets. During the stage of training with default parameters, the machine learning algorithm uses the training data to train and creates a trained model. Then, we evaluate the performance of the trained model on the separated preprocessed testing data.

Then, using a test dataset using IDS measures such as accuracy, precision, recall, and F1-score, we evaluated the model. We detected the problem during the literature review and subsequently presented a remedy. Since its creation, scholars have used the CICIDS2017 dataset to analyze and develop new models and algorithms [52]. The dataset for CI-CIDS2017, according to the author [52], included data on five days' worth of the Canadian

Institute of Cybersecurity's regular and attack traffic.

3.3 System Model

Making data-driven decisions and using various machine learning models to tackle problems has become a primary concern. Developers popularly used three basic types of approaches for the development of ML models. First, we should use just one machine learning algorithm for preprocessing. The second approach uses both supervised and unsupervised learning algorithms in a hybrid fashion [49]. This strategy combines the efforts of the two algorithms to enhance the performance of the model in a particular task. The third method, known as Safe machine learning, involves using two main phases, including the training phase and the application phase, to create an ML model, either supervised or unsupervised.

This work develops a hybrid machine-learning model, which is a blend of supervised and unsupervised algorithms. The K-means algorithm classifies the input features into two groups: attack and normal. The input feature that has been clustered will be used by the classification algorithm [53].

The Python programming language is used to implement the ML models by these system models, completing transforming low-level data into high-level knowledge. The process of developing models is experimental and iterative, allowing for the selection of the models that most closely match the situation.

3.4 Proposed HIDS Framework

Despite the research community developing many intrusion detection frameworks and systems, IDS performance and zero-day attack detection remain open research issues and challenges. Hence, by taking advantage of the low false alarms of misuse-based intrusion detection systems and the capability of detecting zero-day attacks by anomaly-based intrusion detection systems, this paper proposed a hybrid intrusion detection system. used KNN, decision tree (DT), random forest (RF), extra trees (ET), and used as multi-class classifiers

for known attack detection for misuse detection and A cluster labeling (CL) k-means used as an unsupervised learner for zero-day attack detection.

The trained classifier in Safe ml should be able to differentiate between the road and other objects. A significant and crucial problem during the application stage is that the data lacks a label. Therefore, we cannot guarantee that the classifier will function as accurately as it did during the training phase. The application phase will employ the classifier's untrusted labels and collect each class' statistical parameters and probability density function. We compare the statistical differences between each class during the training and application phases using a changed Chernoff Error bound.

If the statistical difference is very low, we can trust the classifier results and accuracy (In this example the autonomous car continues its operation), but if the statistical difference is low, the system can ask for more data and re-evaluate to ensure the distance. In case of are more statistical differences, the classifier results and accuracy are no longer valid, and the system should use an alternative approach or notify the human agent.

Safe machine learning to address both safety and security within a single concept of protection applicable during the operation of ML systems is active monitoring of the behavior and the operational context of the data-driven system based on distance measures of the Empirical Cumulative Distribution Function (ECDF).

In this paper, we propose a hybrid intrusion detection system that detects both known and unknown cyber-attacks using Safe Machine learning and machine learning algorithms on networks with optimal performance. Fig.3-1 shows the architecture of the proposed system, which comprises five main stages: data pre-processing, feature engineering, a signature-based IDS, an anomaly-based IDS, and including some well-known ECDF-based distance measure algorithms with their simple Python implementation.

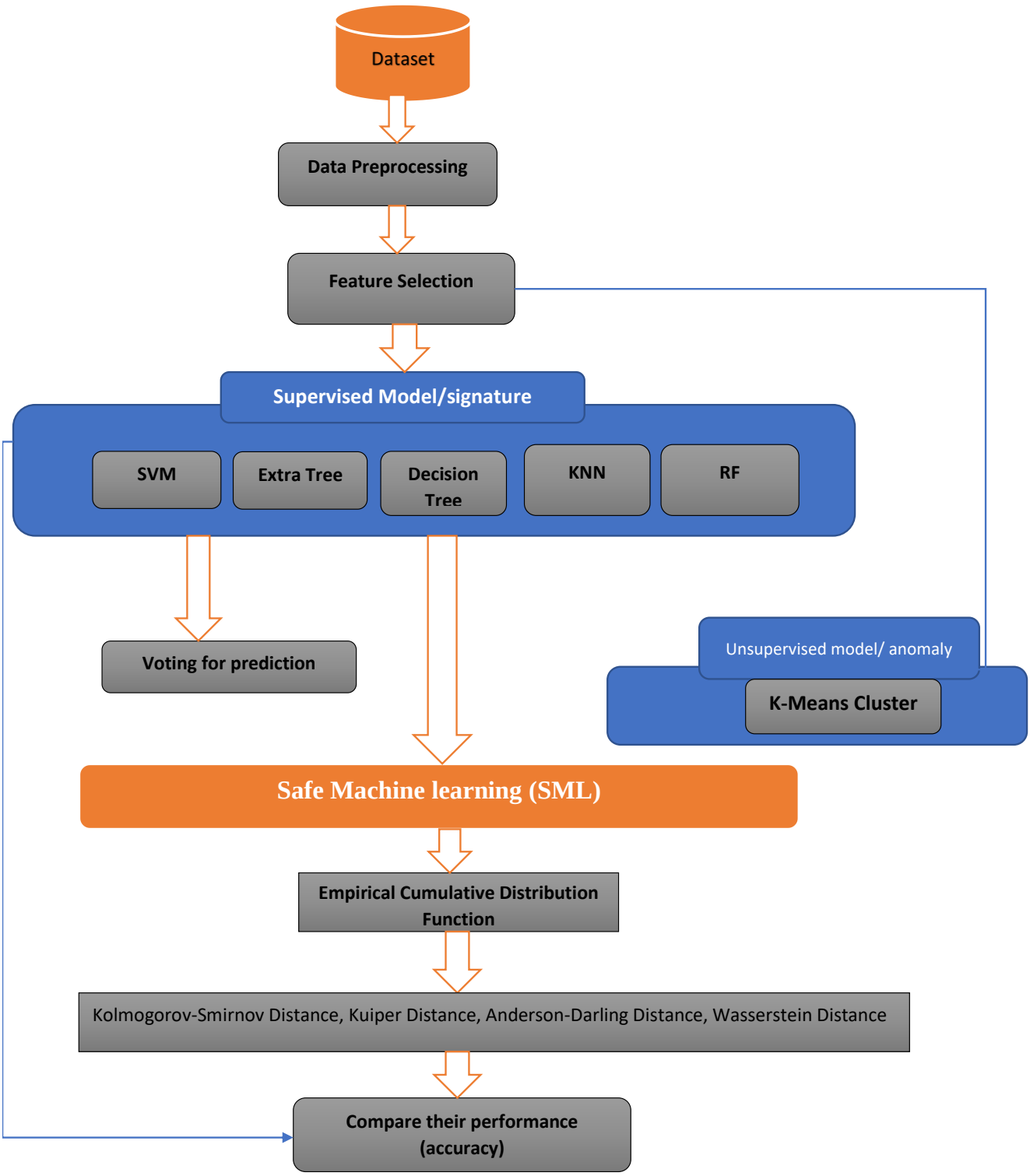


Figure 3.1: The framework of the proposed HIDS

3.5 Proposed Safe Machine Learning Framework

This case study applies the proposed method to the CICIDS2017 dataset, which was originally produced by [52] at the Canadian Institute for Cyber Security (CICS) as an aide to the development and research of anomaly-based intrusion detection techniques for Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs) [9]. The labeled dataset includes both benign (Monday) and malicious (Tuesday, Wednesday, Thursday, Friday)

Safe Machine Learning Framework

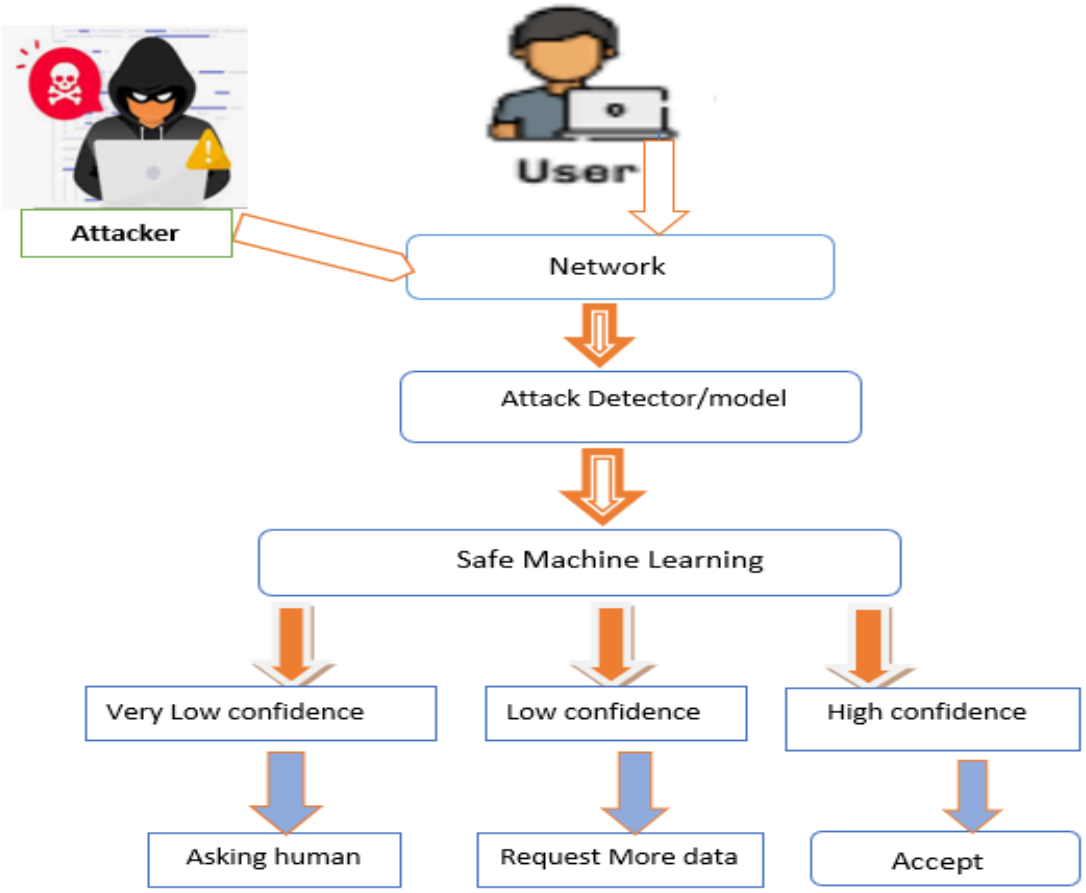


Figure 3.2: Safe Machine Learning Framework

There are countless applications of Machine Learning and Deep Learning for early detection or diagnosis of various kinds of attacks. Can these AI tools be fully autonomous in detection or diagnosis? Are they safe? What is our definition of their safety? Many believe that SafeML or similar approaches can answer those questions. The following figure illustrates the application of SafeML in medical diagnosis [13].

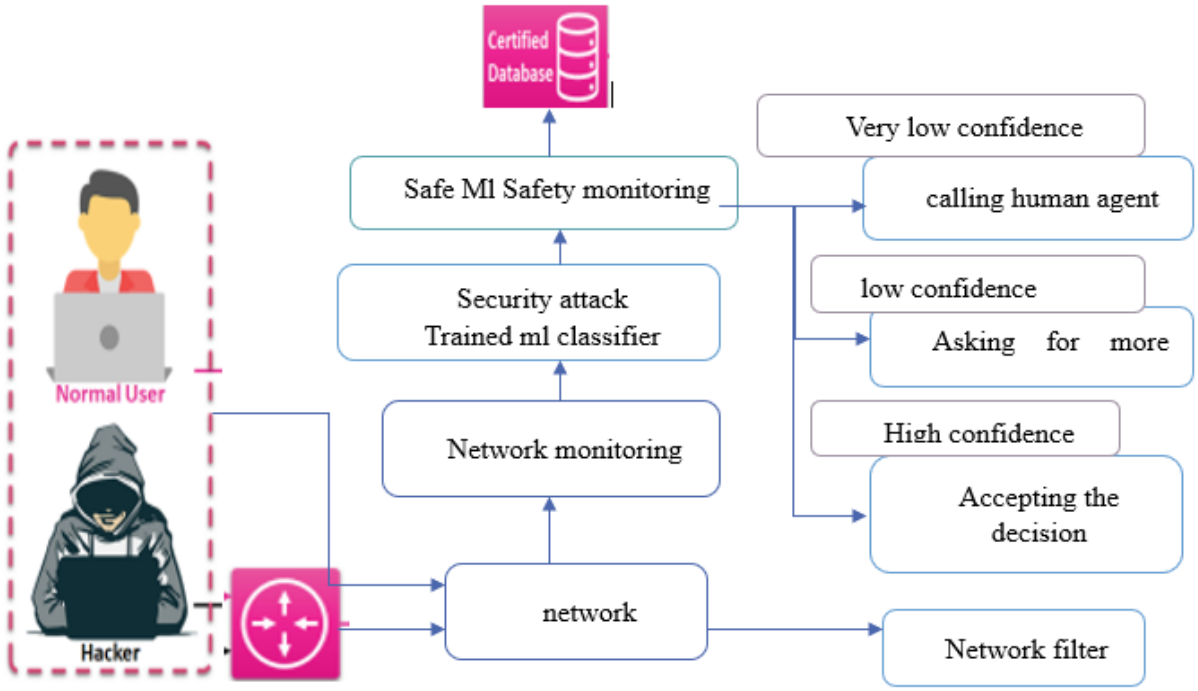


Figure 3.3: Application of Safe ML in Security Intrusion Detection

Poor design, incorrect goal function specification, incorrect implementation, selecting the incorrect learning process, or using subpar or incomplete datasets for training are all common indicators of poor machine learning applications. Therefore, a collection of measures to stop any harm that ML mistakes or misuses could do to humanity can be safe for ML. For ML Safety, many viewpoints and paths need to be established. In actuality, have tackled several research issues related to the certification of machine learning systems that are in use.

Cross-validation and ML Classifiers

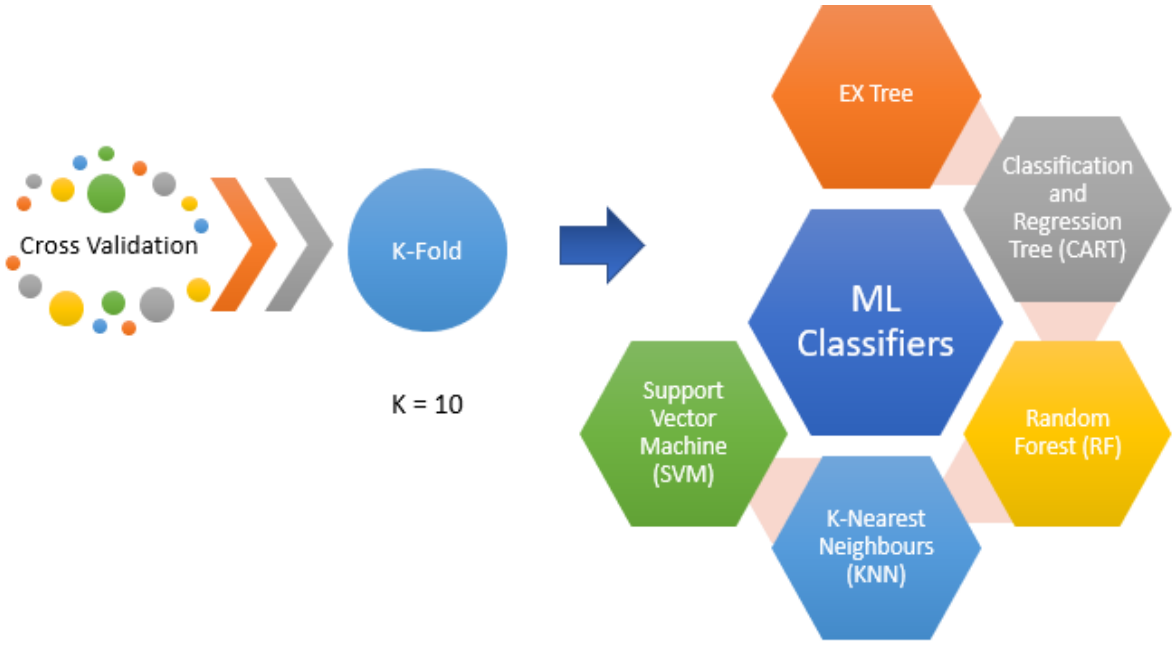


Figure 3.4: Cross-validation and ML Classifiers

3.6 Capabilities of Safe ml

By modifying the existing statistical distance and error bound measures, the proposed method enables estimation of the accuracy bound of the trained ML algorithm in the field with no label on the incoming data. The researchers propose a novel human-in-loop monitoring procedure to certify the ML algorithm during operation. The procedure has three levels of operation:

- nominal operation allowed for assured ML accuracy based on the distance estimation,
- buffering data samples to generate an estimation
- low estimated accuracy, leading to external intervention by automated/human controller being needed

The proposed approach is simple to implement and can support a variety of distributions (i.e. exponential and normal distribution families). Researchers can use the outcome of the

proposed approach as an input for runtime safety analysis in adaptive systems.

3.7 Type and Source of Data

3.7.1 Types of Data

Brute Force, DoS, Web Attack, Infiltration, and DDoS are some attacks that were put into practice. They put into practice one occurrence per day, then gathered and presently keep them in Comma-Separated Values (CSV) files. Cyclometer, a network traffic flow generator created at the University of New Brunswick, creates it from packet capture (PCAP) data. Most of the properties are of the numerical type, but the timestamp also creates nominal and date kinds. You can access the dataset used in this research [54]. It is a reliable dataset that contains benign and recent common attack network flows, which meet real-world criteria. It solves the limitations like some of the existing datasets suffer from a lack of traffic diversity and volumes, some of them do not cover the variety of attacks, while others anonymized packet information and payload which cannot reflect the current trends, or they lack feature set and metadata. This new dataset is publicly available for computer network security and related research works [55].

3.8 Methods of Data Analysis: -

In the analysis of data, we used descriptive statistics such as averages, and percentages as tables and graphs, and we also used metrics including accuracy, recall (or detection rate), FNR (or missed alarm rate), and FPR (or false alarm rate). We used descriptive statistics and machine learning models in the analysis of the collected data. To determine the influence of independent variables on the dependent variables, we qualitatively analyzed the responses with a focus on a dataset of key informants.

3.8.1 Descriptive

This method of data analysis refers to the use of ratio percentage, means, and metrics including accuracy, recall (or detection rate), FNR (or missed alarm rate), and FPR (or false alarm rate) and standard deviation in the process of assessing the attack and its detection and also factor affecting efficiency in the study.

3.8.2 ML models

The ML models that were used in this research are based on the dataset variables dependent and explanatory that are considered. Methods use multiple Machine learning algorithms by using Safe ML algorithms and SPSS.

3.9 Source of Data

To undertake this research and address the major objectives of the study, we will collect open-source data.

3.10 Sample Design

When designing the research study's sample, we considered the necessity of concluding the CIC-IDS2017 sample to accomplish the research purpose and provide answers to the research questions. Table 3-2 shows that the dataset includes traffic data for five days, that includes assault information.

Therefore, to design such a typical intrusion detection, we should merge the traffic data of the day to form a single dataset that IDS can use. This is exactly what we followed to merge these files. By merging the files presented in Table 3.2, we found the whole shape of a dataset that contains 80289 instances and 78 features containing 15 class labels (1 normal + 14 attack labels). Further, examining the instances of the combined files, there are 19 missing values in the column and Duplicate rows–24135. By removing such missing instances, we found a combined dataset of CI-CIDS2017 having 56135 rows × 78 columns instances.

3.11 Data Selection

The CIC-IDS2017 dataset includes the most recent and benign prevalent assaults (PCAPs), which closely mimic real-world data. It also includes the results of the network traffic analysis performed by CIC Flow Meter, which classified flows according to time stamps, source and destination IP addresses, source and destination ports, protocols, and common attack types [54]. Differing from earlier benchmark datasets, the CIC-IDS2017 dataset stands out because of its 78 attributes that encompass both quantitative and qualitative aspects. The dataset classifies all records as either normal or attack. The complete dataset record label belongs to some intrusion classes, one benign class, and other main intrusion classes for intrusion analysis.

Table 3.1: Description of files containing the CICIDS2017 dataset

Name of Files	Day Activity	Attacks Found
Monday - Working Hours.pcap_ISCX.csv	Monday	Benign (Normal human activities)
Tuesday Working Hours.pcap_ISCX.csv	Tuesday	Benign, FTP-Patator, SSH-Patator
Wednesday WorkingHours.pcap_ISCX.csv	Wednesday	Benign, DoS Goldeneye, DoS Hulk, DoS Slow HTTP test, DoS slow loris, Heart bleed
Thursday-Working Hours-Morning-Web Attacks. Cap_ ISCX.csv	Thursday	Benign, Web Attack–Brute Force, Web Attack–SQL Injection, Web Attack–XSS
Thursday-Working Hours-Afternoon-Infiltration. Cap_ ISCX.csv	Thursday	Benign, Infiltration
Friday Working Hours Morning.pcap_ISCX.csv	Friday	Benign, Bot
Friday working hours afternoon PortScan.pcap_ISCX.csv.	Friday	Benign, Port Scan

3.12 Dataset Description

CIC-IDS was aimed to generate diverse and comprehensive benchmark dataset for intrusion detection based on the creation of user profiles which contain abstract representations of events and behaviors seen on the network.

The implemented attacks include Brute Force , Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. A total of 2843553 instances were collected and are currently stored in Comma Separated Values (CSV) files. The total dataset distribution of CIC-IDS.

Table 3.2: Overall Characteristics of the CICIDS2017 Data

Database Name	**CICIDS2017**
Dataset Type	Multi-class
Year of release	2017
Total number of distinct instances	80289
Number of features	78
Number of distinct classes	15

Table 3.3: Class-wise instance Occurrence of the CICIDs2017 database

Class Label	**Number of Instances**
BENIGN	42951
DoS Hulk	2320
Port Scan	20793
DDoS	7747
DoS Slow loris	2347
Bot	2111
Infiltration	36
Web Attack – Brute Force	1580
Web Attack – XSS	404

3.13 . Dependent Variable(Label)

3.13.1 **Port Scan**

4 One popular method used by hackers to find weak spots or open doors in a network is a port scan. Cybercriminals can use a port scan attack to identify open ports and determine whether they are receiving or sending data. It can disclose whether an organization uses firewalls or other active security equipment.

3.13.2 Infiltration

Malicious software that aims to infiltrate and/or harm a user's computer is called an infiltration. Cybercriminals gain access to a system by taking advantage of flaws in hardware, software, or user error, such as weak passwords, social engineering techniques, and phishing scams.

3.13.3 Denial of Service (DoS):

Denial of Service is a kind of attack that aims to shut down a computer system or a network and make it unavailable to its intended users by flooding the target computer to overload it. DoS assaults often cost the victim a lot of time and money.

3.13.4 Benign:

Benign is a realistic background traffic-generated B-Profile system which is proposed by [54].The B-Profile profiles the abstract behavior of human interactions and generates naturalistic benign background traffic. By applying a diverse range of network protocols with different topologies, the B-Profile generates the abstract properties of human and attack behavior.

3.13.5 Brute Force Attack

One of the most common attacks that may find hidden pages and content in a web application and crack passwords is this one. It is a hit-and-try attack carried out with the use of software tools, where the victim ultimately prevails. It is becoming more and more common to attack websites using SQL Injection, which allows a hacker to write a series of SQL queries they may then use to force a database to respond with information.

3.13.6 Distributed Denial of Service (DDoS):

DDoS increases the attack's potency by allowing the victim to receive exponentially more requests. The difficulties in locating the actual source of the attack also make attribution more challenging.

3.13.7 Botnet:

A large group of Internet-connected devices that the owner of the botnet uses for a variety of functions. It gives the attacker access to the device and its connection and can send spam and steal data.

Table 3.4: Group of attributes of the dataset

Group of Attributes	Attributes
Network identifiers	Flow ID; Source IP; Source Port; Destination IP; Destination Port; Protocol; Timestamp.
Flow descriptors	Total Fwd Packets; Total Bwd Packets; Total Length of Fwd Packets; Total Length of Bwd Packets; Fwd Packet Length Max; Fwd Packet Length Max; Fwd Packet Length Min; Fwd Packet Length Mean; Fwd Packet Length Std; Bwd Packet Length Max; Bwd Packet Length Min; Bwd Packet Length Mean; Bwd Packet Length Std; Flow Bytes S; Flow Packets S; Min Packet Length; Max Packet Length; Packet Length Mean; Packet Length Std; Packet Length Variance; Down Up Ratio; Avg Fwd Segment Size; Avg Bwd Segment Size; Fwd Avg Bytes Bulk; Fwd Avg Packets Bulk; Fwd Avg Bulk Rate; Bwd Avg Bytes Bulk; Bwd Avg Packets Bulk; Bwd Avg Bulk Rate; Init Win bytes forward; Init Win bytes backward; act data pkt fwd; min seg size forward; Label
Inter-arrival times	Flow Duration; Flow IAT Mean; Flow IAT std; Flow IAT Max; Flow IAT Min; Fwd IAT Total; Fwd IAT Mean; Fwd IAT Std; Fwd IAT Max; Fwd IAT Min; Bwd IAT Total; Bwd IAT Mean; Bwd IAT Std; Bwd IAT Max; Bwd IAT Min.
Flag features	Fwd PSH flags; Bwd PSH flags; Fwd URG flags; Bwd URG flags; FIN Flag Count; SYN Flag Count; RST Flag Count; PSH Flag Count; ACK Flag Count; URG Flag Count; CWE Flag Count; ECE Flag Count.
Sub-flow descriptors	Subflow Fwd Packets; Subflow Fwd Bytes; Subflow Bwd Packets; Subflow Bwd Bytes.
Header descriptors	Fwd Header Length; Bwd Header Length; Average Packet Size; Fwd Header Length 1
Flow timers	Active Mean; Active Std; Active max; Active min; Idle Mean; Idle std; Idle max; Idle min.

Table 3.5: Variable Entered and Removed form model

Variables Entered/Removed

Model	Variables Entered	Variables Removed	Method
1	idle_min, avg_fwd_segment_size, active_std, subflow_bwd_bytes, urg_flag_count, init_win_bytes_backward, idle_std, fin_flag_count, act_data_pkt_fwd, syn_flag_count, active_min, down/up_ratio, avg_bwd_segment_size, psh_flag_count, min_seg_size_forward, init_win_bytes_forward, ack_flag_count, average_packet_size, active_max, subflow_fwd_bytes, active_mean, idle_max, fwd_header_length.1, idle_mean[b]	rst_flag_count, cwe_flag_count, ece_flag_count, fwd_avg_bytes/bulk, fwd_avg_packets/bulk, fwd_avg_bulk_rate, bwd_avg_bytes/bulk, bwd_avg_packets/bulk, bwd_avg_bulk_rate. They will be deleted from the analysis.	Enter

a. Dependent Variable: label

b. Tolerance = .000 limits reached.

3.14 Performance Evaluation Metrics

Evaluation Metrics There are many methods to measure the performance of ML module prediction. The most common performance measures in IDS Machine learning are precision, recall, and F1 score. In this metrics for evaluating IDS, the higher the value represents that the model performs better. But sometimes, the precision and recall are contradictory, so the evaluation can only consider the balance according to the task. Thus, the F1 score becomes an important indicator, the higher the score, the better performance the IDS has.

The confusion matrix is a table that describes the classification results. Each column of the matrix represents the instance in the prediction class, and each row represents the instance in the actual class. We can summarize the results in the following four basic situations:

- True Positive (TP): Normal records are correctly discriminated by the model.
- False Negative (FN): The system incorrectly identifies malicious attacks as normal records.
- False Positive (FP) occurs when the system incorrectly classifies normal records as an anomaly.
- True Negative (TN): The mode1 successfully identified malicious attacks

Table 3.6: Confusion Matrix

Confusion Matrix		Prediction	
		Normal	Attack
Actual	Normal	True Negative (TN)	False Positive (FP)
	Attack	False Negative (FN)	True Positive (TP)

From these cases of confusion matrix, classification indicators such as [8]: Accuracy, Precision, Recall (Sensitivity), F1 score, and AUC (Area under the Receiver Operating. Accuracy is a suitable metric for balanced datasets. In real network environments; however, normal samples are far more abundant than abnormal samples; thus, accuracy may not be a suitable metric. We calculate the accuracy of the proposed system.

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

Precision defines the ratio of true positive samples to predict positive samples.

$$\text{Precision} = \frac{TP}{TP + FP}$$

We define recall as the ratio of true positive samples to total positive samples, also known as the detection rate**.** The detection rate reflects the model's ability to recognize attacks, which is an important metric

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1-score: defines the harmonic average of the precision and the recall.

$$\text{F1-score} = 2 * \frac{\text{Recall} * \text{Precision}}{\text{Recall precision}}$$

The false positive rate (FPR) defines the ratio of false positive samples to predict positive samples. In attack detection, the FPR is also called the false alarm rate.

The purpose of IDS is the recognition of attacks. In attack detection, the frequently used metrics include accuracy, recall (or detection rate), FNR (or missed alarm rate), and FPR (or false alarm rate)

CHAPTER FOUR

4. EXPERIMENT AND RESULTS

We will create the proposed model by training the designed and constructed system using performance measures covered in this chapter, and we will survey the results.

After all the experiments mentioned above, the system is ready for training and testing. In this section, there are several experiments and discussions to analyze the system performance from different perspectives. The first experiment is testing the effect of training dataset size on performing the system.

We do this by training the system with different percentages of the training dataset, starting from twenty percent up to a hundred percent with a twenty percent increment, and then testing the performance of the trained system with the test dataset. The second experiment is on the effect of attribute selection on performing the system. We select the attribute set that resulted in the best performance as the most discriminant attribute set of the proposed system. Then the experiment 10-fold cross-validation for better system training and evaluation.

We prepare the cross-validation data using the attribute set that resulted in the best performance from the previous experiment. Finally, we train the system using the training dataset, test its performance, and then implement safe machine learning for the safety and security of machine learning.

4.1 Preprocessing and Feature Selection

Starting with preprocessing the dataset, the experimental technique entails choosing the most discriminative features for the proposed system through a feature selection experiment. Following this, a preliminary experiment tests some system parameters

A. Data pre-processing

- Data Sampling by K-means Clustering
- Reduce Class-Imbalance by Over-sampling
- Data Normalization

B. Feature Engineering

- Feature Selection by Information Gain
- Fast Correlation Based Filter (FCBF)
- Kernel Principal Component Analysis (KPCA)

The first step was replacing the missing values of attributes with zero. The next step involved converting all non-numeric values of attributes to their numeric equivalent using the specified method. Finally, we normalized the numeric value of each attribute to be between zero and one.

We perform preprocessing on both the training and testing datasets. After the preprocessing experiment, out of the total features, 19 of them have a value of NaN because of attributes minimum and maximum value is equal. So, the normalization formula gives a divide by zero that results in the NaN value. Since machine learning algorithms only compute numeric values, we removed all attributes with NaN values from the dataset before moving to the next step. The resulting dataset of the preprocessing experiment features, which are numerically valued and normalized between zero and one, is called a version one dataset. Now that the dataset is ready to use for the Dataset version means nothing, but it is the way of identifying different datasets resulting from preprocessing and attribute selection procedures.

After data pre-processing, we can produce a dataset of excellent quality and high representativeness. Yet, generating an ideal feature list through suitable feature engineering

can also enhance the quality of datasets for more precise and effective model learning. When building an ML model, a thorough feature engineering technique using IG and KPCA is used to weed out unimportant, redundant, and noisy features while keeping the crucial ones.

The information gain (IG) method is a popular feature selection (FS) method that is employed to choose significant features. To gauge how much information a feature can contribute to the desired variable, one can employ the IG method or measure changes in entropy. Because of its low computing complexity of O(n), IG is the system of choice in the suggested one because it can quickly determine the relevance score for each feature.

Looking at the original data types

Self-explanatory. Just shows the data types of each feature/ column. Looking at the original data types, Self-explanatory. Just shows the data types of each feature/ column.

Number of data points in class: BENIGN is 42951 (53.495%)
Number of data points in class: Port Scan is 20793 (25.898%)
Number of data points in class: DDoS is 7747 (9.649%)
Number of data points in class: DoS slow Loris is 2347 (2.923%)
Number of data points in class: DoS Hulk is 2320 (2.89%)
Number of data points in class: Bot is 2111 (2.629%)
Number of data points in class: Web Attack Brute Force is 1580 (1.968%)
Number of data points in class: Web Attack XSS is 404 (0.503%)
Number of data points in class: Infiltration is 36 (0.045%)

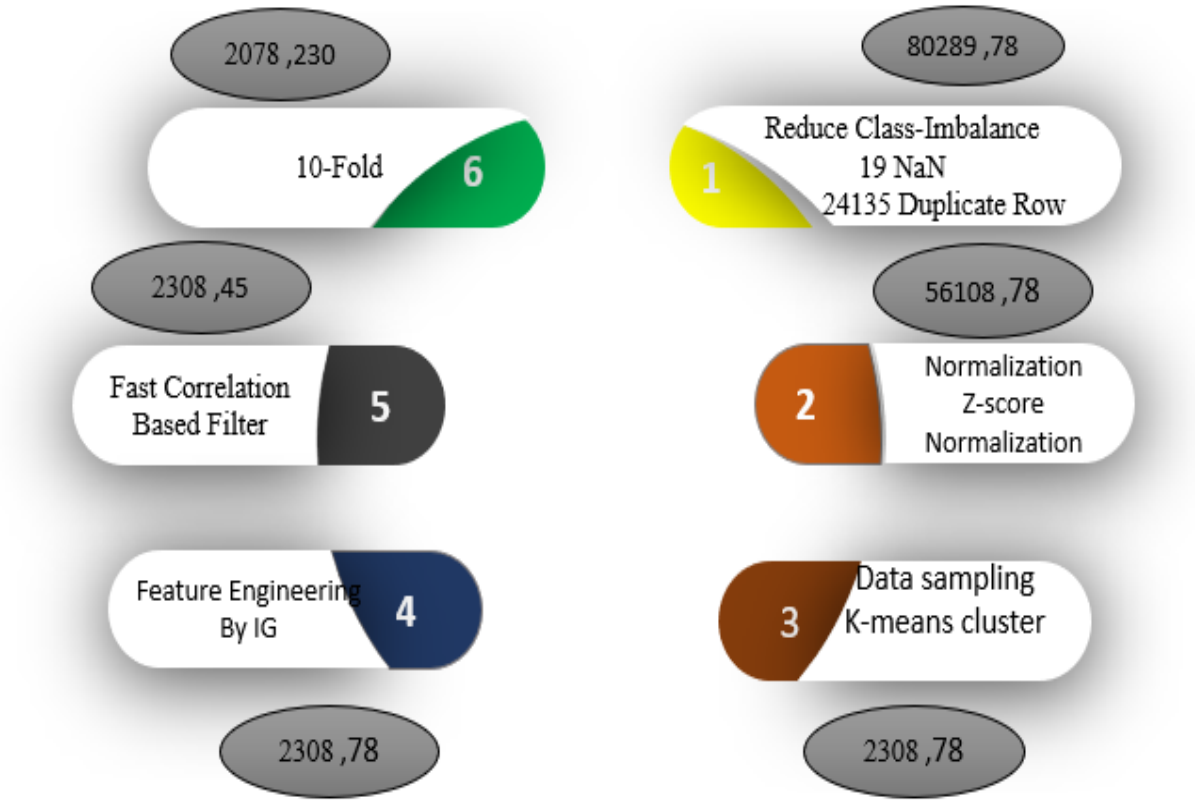


Figure 4.1 dataset preprocessing

4.2 Data sampling

The sample is a small-sized subset for model learning to use k-means cluster sampling. Use k-means to cluster the data samples and select a proportion of data from each cluster.

- Label encoder = Label Encoder()
- Using k-means cluster into clusters=100 then fit and divide into
  - ✓ Major group and
  - ✓ minor group
- Then drop the minor using df_cleaned_minor['klabel'].value_counts()
- Then def typical Sampling (group):
- apply(typical Sampling)

4.3 Data cleaning

4.3.1 After Data Cleaning Cleaned dataset

Table 4.1:After Data Cleaning Cleaned dataset.

	flow_duration	total_fwd_packets	total_backward_packets	total_length_of_fwd_packets	total_length_of_bwd_packets	fwd_packet_length_max	fwd_packet_l
0	4	2	0	12	0	6	
1	1	2	0	12	0	6	
2	3	2	0	12	0	6	
3	1	2	0	12	0	6	
4	609	7	4	484	414	233	
...	...	...	...	...	...	...	
56103	4	1	1	6	6	6	
56104	186	2	2	132	388	66	
56105	81	1	1	0	0	0	
56106	32056	2	2	88	200	44	
56107	31462	2	2	72	328	36	

56108 rows × 78 columns

4.4 Data Normalization:

We use the k-means algorithms to produce a representative and balanced dataset, and then we finish various further data pre-processing procedures for the next steps. To accept the inputs of ML algorithms, the network traffic statistics are first encoded using a label encoder, which converts string features into numerical features because many ML methods cannot support them directly [56]. After that, we normalize the network datasets using the Z-score algorithm because the ranges of the features gathered from network traffic data are frequently very varied and ML models often perform better on normalized datasets. An unbalanced dataset with widely disparate feature scales could produce a biased ML model that primarily emphasizes large-scale features. You can standardize the features using the Z-score approach, making their means and standard deviations equal to 0. We represent each normalized feature value, xn, by applying the Z-score algorithm.

$$X_n = \frac{X-\mu}{\alpha}$$

Where x is the original feature value, μ and σ are the mean and standard deviation of the feature values, respectively. Converts string features into numerical features.

Table 4.2 : converts string features into numerical features

Class Label	**Number of Instances**
0	40869
2	7455
4	2267
3	1872
7	1470
1	1437
8	404
6	299
5	35

Table 4.3 : Descriptive Dataset

	N	Minimum	Maximum	Mean	Std. Deviation
flow_duration	2308	-.4953	2.8215	-.326805	.5629671
total_fwd_packets	2308	-.0139	3.6369	-.002504	.1403386
total_backward_packets	2308	-.0130	2.8175	-.004104	.1087277
total_length_of_fwd_packets	2308	-.0448	122.2147	.316701	4.4595894
total_length_of_bwd_packets	2308	-.0110	.0632	-.010565	.0034436
fwd_packet_length_max	2308	-.2678	32.2314	.279905	2.9431379
fwd_packet_length_min	2308	-.4626	2.3655	-.360722	.3099160
fwd_packet_length_mean	2308	-.3199	36.4268	.355944	3.4687009
fwd_packet_length_std	2308	-.2929	34.9066	.297258	2.6316953
bwd_packet_length_max	2308	-.4870	4.5462	-.365080	.4973591
bwd_packet_length_min	2308	-.5468	5.0945	-.416054	.4411132
bwd_packet_length_mean	2308	-.5420	3.5546	-.403173	.5100384
bwd_packet_length_std	2308	-.4382	5.1614	-.326312	.4858931
flow_bytes/s	2308	-.0548	.9214	-.041120	.0637828
flow_packets/s	2308	-.1801	13.0601	-.064773	.7217903
flow_iat_mean	2308	-.3512	7.8587	-.208557	.4740711
flow_iat_std	2308	-.3973	6.8518	-.249193	.4935093
flow_iat_max	2308	-.4264	4.2567	-.282062	.5139041
flow_iat_min	2308	-.0478	1.2299	-.043716	.0499309
fwd_iat_total	2308	-.4886	2.8332	-.322812	.5594059
fwd_iat_mean	2308	-.2791	8.9088	-.179164	.4739909
fwd_iat_std	2308	-.3842	5.0088	-.229720	.4834194
fwd_iat_max	2308	-.4203	4.2577	-.278622	.5108415
fwd_iat_min	2308	-.1489	9.3788	-.110404	.4670130
bwd_iat_total	2308	-.3897	3.3466	-.260728	.5771048
bwd_iat_mean	2308	-.2398	8.3184	-.179992	.4719530
bwd_iat_std	2308	-.2563	7.9912	-.187339	.4705406
bwd_iat_max	2308	-.3107	5.7031	-.215405	.5373324
bwd_iat_min	2308	-.1729	9.0031	-.128669	.4728044
fwd_psh_flags	2308	-.2055	4.8655	-.124229	.6370376

Table 4.4 : Frequency analysis target variable or label

	Frequency	Percent	Valid Percent	Cumulative Percent
0	323	14	14	14
1	1437	62.3	62.3	76.3
2	64	2.8	2.8	79
3	13	0.6	0.6	79.6
4	17	0.7	0.7	80.3
5	35	1.5	1.5	81.8
6	2	0.1	0.1	81.9
7	13	0.6	0.6	82.5
8	404	17.5	17.5	100
Total	2308	100	100	

4.5 Feature Engineering

After data pre-processing, it is possible to generate a dataset of excellent quality and high representativeness. Yet, generating an ideal feature list through suitable feature engineering can also enhance the quality of datasets for more precise and effective model learning. When building an ML model, a thorough feature engineering technique using IG and KPCA is used to weed out unimportant, redundant, and noisy features while keeping the crucial ones [57].

Feature Selection by Information Gain: The information gain (IG) method is a popular feature selection (FS) method that is employed to choose significant features. To gauge how much information a feature can contribute to the desired variable, researchers can employ IG, or changes in entropy [57]. Because of its low computing complexity of O(n), IG is the system of choice in the suggested one because it can quickly determine the relevance score for each feature. We can choose the most pertinent aspects of the work by evaluating the relevance of each attribute. Assuming that T is the target variable, we represent the IG value got by employing the feature X for each feature represented by a random variable X.

$$IG(T/X) = H(T) - H(T/X)$$

Table 4.5 : missing correlations of attribute and their significance

Warnings
For models with a dependent variable label, the following variables are constants or have missing correlations:rst_flag_count, cwe_flag_count, ece_flag_count, fwd_avg_bytes/bulk, fwd_avg_packets/bulk, fwd_avg_bulk_rate, bwd_avg_bytes/bulk, bwd_avg_packets/bulk, bwd_avg_bulk_rate. We will delete them from the analysis.

Correlations of attribute and their significance

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	-9350.951	590.610		-15.833	.000
	fin_flag_count	.217	.059	.035	3.664	.000
	syn_flag_count	.130	.048	.030	2.712	.007
	psh_flag_count	.476	.056	.153	8.551	.000
	ack_flag_count	.686	.071	.200	9.609	.000
	urg_flag_count	-.082	.044	-.032	-1.889	.059
	down/up_ratio	-.585	.053	-.149	-10.955	.000
	average_packet_size	.975	.230	.244	4.234	.000
	avg_fwd_segment_size	-.129	.036	-.161	-3.567	.000
	avg_bwd_segment_size	-.787	.239	-.144	-3.300	.001
	fwd_header_length.1	128234.663	23177.591	1.363	5.533	.000
	subflow_fwd_bytes	.105	.054	.168	1.961	.050
	subflow_bwd_bytes	9.274	15.609	.011	.594	.552
	init_win_bytes_forward	.322	.068	.089	4.719	.000
	init_win_bytes_backward	1.496	.037	.632	40.466	.000
	act_data_pkt_fwd	-28.051	4.540	-1.469	-6.178	.000
	min_seg_size_forward	570390.092	40566.634	.211	14.061	.000
	active_mean	1.085	.288	.517	3.762	.000
	active_std	-.705	.226	-.256	-3.122	.002
	active_max	.109	.171	.045	.639	.523
	active_min	-.835	.260	-.385	-3.215	.001
	idle_mean	-16.648	2.839	-2.978	-5.864	.000
	idle_std	3.006	.646	.574	4.653	.000
	idle_max	1.846	.752	.339	2.456	.014
	idle_min	14.770	2.792	2.587	5.290	.000

a. Dependent Variable: label

The expression H (T|X) represents the conditional entropy of T over X, with H (T) representing the entropy of the target variable T. The IG value of T over X, IG (T|X), can therefore express the feature relevance of a feature X or the correlation between X and the goal T. If IG (T|X) > IG (T|Y), we consider feature X to be more crucial to the target T than feature Y [57].

4.5.1 K-Fold Cross Validation and Stratified Splitting

K-fold cross-validation is a superior technique to validate the performance of our model. It evaluates the model using different chunks of the data set as the validation set. We divide our data set into K-folds. K represents the number of folds into which you want to split your data. If we use 5-folds, we divide the data set into five sections.

X-train length	2078
y-train length	2078
X-test length	230
y-test length	230

4.6 Kernel Principal Component Analysis (KPCA):

The feature extraction algorithm PCA converts a set of correlated features into a smaller subset of uncorrelated features, known as principal components [58].

KPCA is a better variant of PCA that employs the kernel method to learn a non-linear function or decision boundary to lessen the dimensionality of non-linear data. We choose KPCA because it can adapt to non-linear data and reduce computing complexity, the risk of over-fitting, and distracting noise [58].

4.7 Machine Learning Model and Results

This Section covers the experiments performed during the entire thesis. We broadly divide the experiments into two parts. The first part comprises the experiments done on signature detection. The second part comprises experiments performed on the conjuncture of anomaly and signature detection. Here, we used the CICIDS2017 dataset, which clearly defines the total number of attacks and normal data. For this signature detection experiment, we have selected 56135 data from CICIDS2017 which is randomly selected and comprises 42951 normal data and 13184 attacks that are helpful to calculate TP, TN, FP, and FP which is classified by Machine learning model from the testing dataset.

4.8 Experimentation on Signature Detection

Thus, one of the most crucial steps in stopping these hazardous occurrences recognizes network attacks.

4.8.1 KNN model

Table 4.6: KNN model experiment

Attack	Benign	DDos	Dos Slowlori	Dos Hulk	Bot	Port Scan	Web Attack brure force	Web Attack XSS	Infiltration
Benign	40861	0	0	0	0	0	0	0	0
DDos	0	7455	0	0	0	0	0	0	0
Dos Slowloris	0	0	2267	0	0	0	0	0	0
Dos Hulk	0	0	0	1872		0	0	0	0
Bot	0	0	0	0	1437		0	0	0
Port Scan	0	0	0	0	0	299		0	0
Web.A Brute	0	0	0	0	0	0	1470	0	0
Web.A XSS	0	0	0	0	0	0	0	404	0
Infiltration	0	0	0	0	0	0	0	0	35

```
accuracy                           0.99       230
   macro avg       0.86      0.83      0.84       230
weighted avg       0.98      0.99      0.98       230
```

Table 4.7: Performance Evaluation Metrics of KNN

Attack	Recall	Precision	F1-score	Support
Benign	1	0.97	0.98	32
DDos	1	1	0.92	6
Dos Slowloris	1	1	1	2
Dos Hulk	1	1	0.67	2
Bot	1	1	1	144
web. Attack Brute Force	0	0	0	1
Web. Attack XSS	0.97	0.97	0.97	40
Infiltration	0.67	1	80	3

4.8.2 KNN model with PCA algorithms

Table 4.8: Performance Evaluation Metrics of KNN using PCA

Class	Recall	Precision	f1-score	Support
Class 0(attack)	1	0.99	1	198
Class 1(normal)	0.97	1	0.98	32

```
   accuracy                           1.00       230
   macro avg       1.00      0.98     0.99       230
weighted avg       1.00      1.00     1.00       230
```

4.8.3 Random Forest Algorithms

Table 4.9: Random Forest model experiment

Attack	Benign	DDoS	Dos Slow Loris	Dos Hulk	Bot	Port Scan	Web. Attack Brute Force	Web Attack XSS	Infiltration
Benign	40861	0	0	0	0	0	0	0	0
DDos	0	7455	0	0	0	0	0	0	0
Dos Slow loris	0	0	2267	0	0	0	0	0	0
Dos Hulk	0	0	0	1872		0	0	0	0
Bot	0	0	0	0	1437		0	0	0
Port Scan	0	0	0	0	0	299		0	0
Web.Attack Brute Force	0	0	0	0	0	0	1470	0	0
Web.Atack XSS	0	0	0	0	0	0	0	404	0
Infiltration	0	0	0	0	0	0	0	0	35

```
Accuracy of RF: 0.9826086956521739

Precision of RF: 0.9831183309444179

Recall of RF: 0.9826086956521739

F1-score of RF: 0.982693366072562
```

Table 4.10: Performance Evaluation Metrics of RF

	recall	**Precision**	**f1-score**	**Support**
Benign	1	0.91	0.95	31
DDos	1	1	0.92	6
Dos slow Loris	1	1	1	2
Dos Hulk	0.5	1	0.67	2
Bot	1	1	1	144
Port Scan	1	1	1	1
web. Attack Brute Force	0	0	0	1
Web. Attack XSS	0.95	1	1	0
Infiltration	1	1	0.97	40

4.8.4 Random Forest algorithms using PCA RF algorithm

Table 4.11: Performance Evaluation Metrics of RF with PCA

	Recall	**Precision**	**f1-score**	**Support**
Class 0(attack)	1	0.99	1	198
Class 1(normal)	0.97	1	0.98	32

RF

```
accuracy                           1.00       230
 macro avg       1.00      0.98     0.99       230
weighted avg     1.00      1.00     1.00       230
```

4.8.5 Extra Tree Algorithms

Table 4.12: Performance Evaluation Metrics Extra Tree

Attack	**Recall**	**Precision**	**f1-score**	**Support**
Benign	1	0.94	0.97	31
DDos	1	1	1	6
Dos slow Loris	1	1	1	1
Dos Hulk	1	1	1	2
Bot	1	1	1	144
web. Attack Brute Force	0	0	0	2
Web. Attack XSS	0.95	0.97	0.96	40
Infiltration	1	1	1	3

4.8.6 Extra tree algorithms with PCA algorithms

Table 4.13: Performance Evaluation Metrics Extra Tree using PCA

Attack	recall	Precision	f1-score	Support
Class 0(attack)	1	1	1	198
Class 1(normal)	1	1	1	32

4.8.7 DT algorithms

Table 4.14: Performance Evaluation Metrics DT

Attack	**Recall**	**Precision**	**f1-score**	**Support**
Benign	0.97	0.94	0.95	32
DDos	1	1	1	6
Dos Slow Loris	1	1	1	1
Dos Hulk	1	1	1	2
Bot	1	1	1	144
web. Attack Brute Force	0	0	0	2
Web. Attack XSS	0.97	0.97	0.97	40
Infiltration	1	1	1	3

4.8.8 DT algorithms with PCA algorithm

Table 4.15: Performance Evaluation Metrics DT using PCA

Attack	Recall	Precision	f1-score	Support
Class 0 (attack)	1	0.98	0.99	198
Class 1 (normal)	0.84	1	0.92	32

4.9 Comparison of Models

Comparing different classification ML models and selecting the best one for predicting network intrusions is one goal of this study. In this section, we present a final comparison of the models discussed by scenarios in the previous section. In this research, we conducted experiments by applying PCA to the ML algorithm and the ECDF-based approaches. After preprocessing the training dataset using the ML model, we trained the models over the training dataset. We evaluated the performance of all models using the same testing dataset through 10-fold cross-validation. We present the summary of experimented results for the models built in this study in the Figure below.

Table 4.16: Comparison of Models

ML Model	Accuracy
KNN	99
KNN using PCA	100
RF	98.26
RF using PCA	100
EX	98.26
EX using PCA	100
DT	98
DT using PCA	98

4.10 Correlation Analysis

Pearson's correlation between the classes of the dataset and the statistical ECDF-based distances. As seen, all the distances have more correlation with the classes. We can explain these correlations because of the similarity in their formulation. We evaluated the p-values for the above correlations to be zero, validating the correlation hypotheses using SPSS.

Table 4.17: correlation between attribute

Correlations

		flow_duration	total_fwd_packets	total_backward_packets	total_length_of_fwd_packets	total_length_of_bwd_packets	label
flow_duration	Correlation	1	.354**	.346**	.307**	.339**	.116**
	Sig. (2-tailed)		.000	.000	.000	.000	.000
	N	2308	2308	2308	2308	2308	2308
total_fwd_packets	Correlation	.354**	1	.999**	.980**	.159**	.067**
	Sig. (2-tailed)	.000		.000	.000	.000	.001
	N	2308	2308	2308	2308	2308	2308
total_backward_packets	Correlation	.346**	.999**	1	.982**	.140**	.058**
	Sig. (2-tailed)	.000	.000		.000	.000	.005
	N	2308	2308	2308	2308	2308	2308
total_length_of_fwd_packets	Correlation	.307**	.980**	.982**	1	.105**	.050*
	Sig. (2-tailed)	.000	.000	.000		.000	.016
	N	2308	2308	2308	2308	2308	2308
total_length_of_bwd_packets	Correlation	.339**	.159**	.140**	.105**	1	.105**
	Sig. (2-tailed)	.000	.000	.000	.000		.000
	N	2308	2308	2308	2308	2308	2308
label	Correlation	.116**	.067**	.058**	.050*	.105**	1
	Sig. (2-tailed)	.000	.001	.005	.016	.000	
	N	2308	2308	2308	2308	2308	2308

**. Correlation is significant at the 0.01 level (2-tailed).

*. Correlation is significant at the 0.05 level (2-tailed).

4.11 Safe ML model

4.11.1 The Safe ML Statistical Distance Measures results

The local Python modules are used to run the statistical measure (univariate version) procedures. Because the Python methods differ slightly from the current ones, we must pass in one feature at a time. Subsequently, we use the complete dataset to compute several ECDF-based statistical distance measures after submitting and returning all features. The part below on the graphing of results [3] shows this.

4.11.2 Transformed dataset into classes

The dataset shown below is the PCA-transformed data, with the label/ class feature attached (which has been Label Binarized, into numbers from class names- so the model can understand). To simulate the Safe ml idea, several permutations have to be run. The data split for each permutation will differ, leading to distinct data and varied accuracies in the model. We hope that the statistical difference values would change in a highly correlated, positive manner when the accuracies vary between permutations in the simulation.

Unsupervised learning models in the anomaly-based IDS are more sensitive to features than supervised learning models are.To use kernel principal component analysis (KPCA) for the anomaly-based IDS, the IG-FCBF method is first implemented.

The feature extraction algorithm PCA converts a set of correlated features into a smaller subset of uncorrelated features known as principal components. KPCA is a better variant of PCA that employs the kernel method to learn a non-linear function or decision boundary to lessen the dimensionality of non-linear data. We chose KPCA because it can adapt to non-linear data, reduce computing complexity, reduce the risk of over-fitting, and distracting noise.

PCA works to minimize the dataset's dimensions and search space while attempting to preserve as much information as workable, for example. By identifying the most significant patterns, information, and variation of each dimension and attribute, it may effortlessly reduce the dimensionality by more than half while retaining 99% of the original data. Grouping the most

significant information into "principal components" accomplishes this achievement.Formally, speaking, PCA divides a multivariate dataset into a series of progressively orthogonal components that together account for the greatest percentage of the variance.

Table 4.18: Principal component Analysis permutation

	Principal component 1	Principal component 2	Principal component 3	Principal component 4	Principal component 5	Principal component 6	Principal component 7	label
0	-1.128076	0.341347	-0.176903	0.842777	-0.004074	-1.365439	-0.008075	1
1	-1.088854	0.307496	-0.171442	0.544572	0.002426	-1.597326	0.110531	1
2	-1.161454	0.379465	-0.198678	0.328216	0.117590	-1.233214	0.168816	1
3	-1.126598	0.342312	-0.157708	0.516380	-0.029482	-1.546992	0.110972	1
4	-1.100742	0.339347	-0.135494	0.328539	-0.003569	-1.224257	0.242473	1
...	...	...	...	...	...	...	...	...
2303	-1.313957	0.483647	-0.369534	-0.318104	0.395187	-0.507249	-0.186735	0
2304	-1.313954	0.483655	-0.369599	-0.318079	0.395299	-0.507297	-0.186635	0
2305	-1.313966	0.483634	-0.369744	-0.318046	0.395603	-0.507392	-0.186986	0
2306	-1.313979	0.483615	-0.370092	-0.317948	0.396282	-0.507628	-0.187350	0
2307	-1.313986	0.483603	-0.370225	-0.317913	0.396551	-0.507717	-0.187565	0

2308 rows × 8 columns

The principal component analysis divides the dataset into 7 classes

4.11.3 The Safe ML Statistical distance measures

After executing the machine learning models and determining their correctness, among other valuable performance indicators, researchers can conduct an experiment using safe ml statistical distance measures. We will conduct the genuine run later. This trial run serves as a guide for users to configure the SciKit-Learn and Python code, ensuring that they can easily replicate the outcomes for upcoming projects or case studies [3].

ECDF-based statistical distance measures can predict model performance. The final graphing plotting will show these relationships, allowing for easy identification of the correlation.

This file aims to have a Python code case study for the same dataset and verify the results to see if they match. If it matches, then we can confidently merge this case study into the main branch of the Safe ml repository [5].

In safety-critical environments, predicting a model's accuracy based on the incoming data- before the model can even use it to predict the next 'action' to take- is a great way to spot potential issues before they occur. Then, these potential errors- based on a threshold.

They divided PCA into two classes: class 0 for attack and class (1)0ne for benign.

Table 4.19: Start run for class 0 attack class

	Accuracy	Anderson_Darling_dist	CVM_dist	DTS_dist	Kolmogorov_Smirnov_dist	Kuiper_dist	Wasserstein_dist
0	0.939130	2.913058	21.105885	0.063318	0.177873	0.234554	0.257513
1	0.969565	3.287223	22.815395	0.092323	0.201203	0.278394	0.339909
2	0.930435	2.514147	17.782840	0.072155	0.158023	0.257339	0.282240
3	0.930435	2.181160	14.746914	0.073206	0.130513	0.217259	0.272701
4	0.969565	3.048262	22.332479	0.086191	0.185556	0.246761	0.298991
5	0.969565	2.512626	17.142256	0.071715	0.151859	0.226880	0.262314
6	0.947826	4.009618	29.563682	0.073344	0.240378	0.322244	0.288391
7	0.969565	2.711167	19.028005	0.072690	0.150166	0.215976	0.239572
8	0.965217	3.425525	24.774322	0.074308	0.191501	0.229871	0.285735
9	0.969565	2.417680	16.958005	0.080513	0.145840	0.243940	0.257298
10	0.956522	2.715147	18.849665	0.192406	0.156416	0.234624	0.407893
11	0.960870	2.631988	17.925851	0.084961	0.165988	0.252016	0.296555
12	0.969565	2.057575	14.269759	0.078268	0.148633	0.236301	0.245531
13	0.947826	3.709057	28.091315	0.070472	0.215034	0.266145	0.289294
14	0.965217	2.652076	18.261872	0.081025	0.151598	0.224821	0.286456

We calculate the accuracy for each feature in class o(attack).

Now, running the code for each permutation requires Stratified splitting (KFold), model training and predicting classes via the chosen classifier, and finally, the iteration over each class/ label to calculate the statistical distance. Then, once complete, the results table can plot the results from all the permutations**.**

Table 4.20: Start run for class 1 or normal class

	Accuracy	Anderson_Darling_dist	CVM_dist	DTS_dist	Kolmogorov_Smirnov_dist	Kuiper_dist	Wasserstein_dist
0	0.939130	2.708008	46.336052	0.091520	0.071931	0.100229	0.366445
1	0.969565	2.699368	46.629857	0.057576	0.059386	0.090012	0.200112
2	0.930435	2.290173	40.416341	0.062565	0.062990	0.092851	0.227962
3	0.930435	2.617224	44.521118	0.074875	0.063199	0.084217	0.305151
4	0.969565	2.043269	35.083979	0.087965	0.049054	0.079964	0.292402
5	0.969565	2.927072	51.947730	0.084236	0.062923	0.093776	0.280044
6	0.947826	4.018073	72.596437	0.074802	0.100089	0.124891	0.326709
7	0.969565	2.532175	44.932429	0.067602	0.062887	0.087196	0.266089
8	0.965217	3.027104	53.829371	0.063935	0.073631	0.105950	0.237127
9	0.969565	2.073136	35.890068	0.068463	0.050997	0.073043	0.287369

We run the accuracy for each feature for class 1 (norma1 class).

4.11.4 Concatenated the two classes 0 and Class 1

Table 4.20 : Running the Statistical distance measure algorithms

	Accuracy	Anderson_Darling_dist	CVM_dist	DTS_dist	Kolmogorov_Smirnov_dist	Kuiper_dist	Wasserstein_dist	dataset
0	0.956522	1.912905	12.439519	0.080439	0.113893	0.176632	0.255403	class0
1	0.943478	3.335916	23.923510	0.169128	0.190184	0.248325	0.523895	class0
2	0.947826	3.263113	22.330695	0.106573	0.193054	0.270925	0.360368	class0
3	0.960870	2.663356	18.990614	0.094387	0.174806	0.259126	0.315581	class0
4	0.947826	3.316877	23.615304	0.200566	0.194097	0.256259	0.553583	class0
...	...	...	...	...	...	...	...	...
5	0.939130	2.826390	50.114048	0.092562	0.074050	0.094283	0.354635	class1
6	0.952174	1.733631	30.747682	0.053404	0.045374	0.072085	0.174550	class1
7	0.956522	4.606544	83.059208	0.073237	0.093954	0.114256	0.333683	class1
8	0.956522	2.587506	47.001351	0.061413	0.066622	0.095212	0.230982	class1
9	0.939130	3.560261	62.698101	0.062443	0.069834	0.099192	0.266718	class1

110 rows × 8 columns

We run accuracy for each feature for class o(attack) and class 1(normal). Since there are only 2 classes, we only show 2 result data frames. Each row represents a permutation. Each dataset represents a single class**.**

4.11.5 Mean Accuracy to the Statistical Distance Measure Algorithms

Table 4.21: Mean Accuracy of the Statistical Distance Measure Algorithms

ML Algorithms	Kolmogorov Smirnov_dist	Anderson Darling dist	DTS_dist	Wasserstein	Kuiper_dist
KNN	95.36	95.36	95.36	95.36	95.36
RF	95.41	95.41	95.41	95.41	95.41
DT	95.57	95.57	95.57	95.57	95.57
EX	95.08	95.08	95.08	95.08	95.08

When we compare the machine learning accuracy with safe machine learning statistical distance measure, there is a difference between them. Based on their accuracy, IDS performance is high confidence and no more difference between them. Decision Tree is the highest confidence when we compare between them. Because the Accuracy between the ECDF and ML models is no longer different when we compare them to another model.

4.12 The graphs/ plots showing accuracy based on ECDF

The graphs plot a model's varying accuracy against an ECDF-based statistical distance measure. Each row from the 'concatenated' Pandas Data frame results table represents 1 permutation. Each permutation uses a different dataset split to train, test, and predict classes/ labels.

4.12.1 Anderson-Darling distance with KNN line plot.

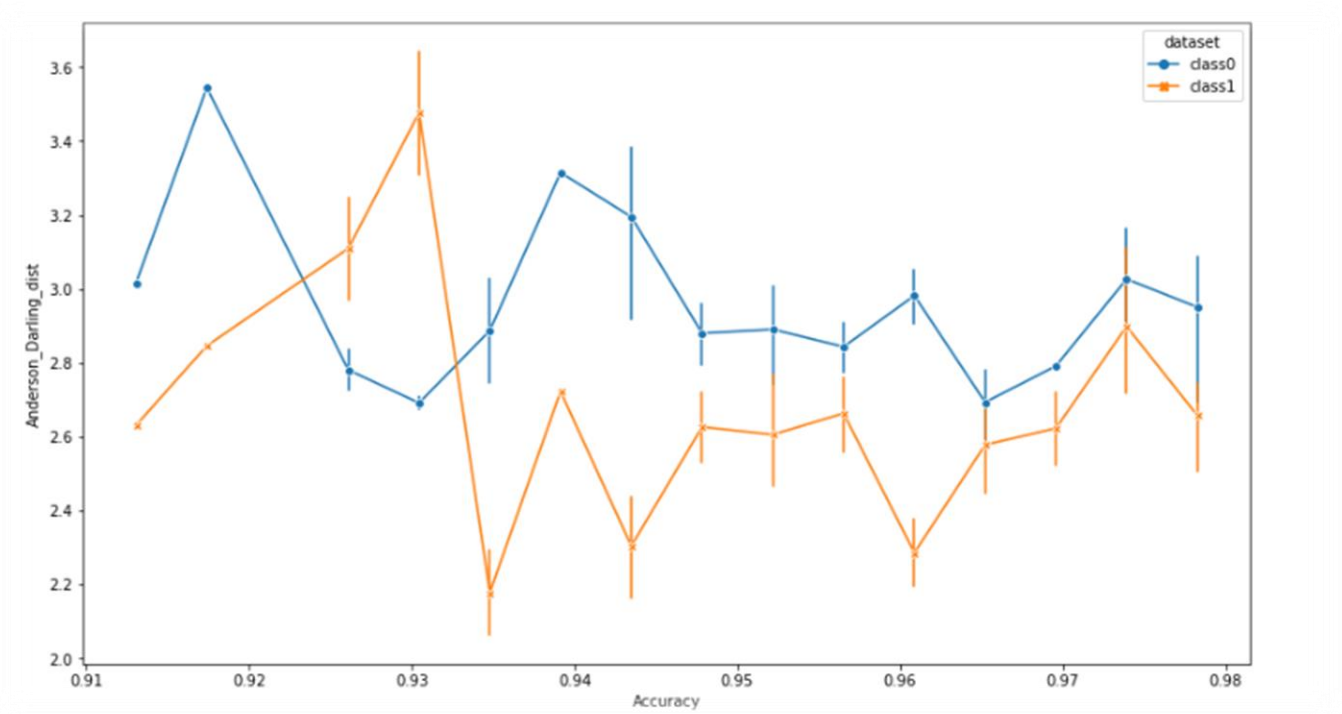


Figure 4.2: Anderson-Darling distance with KNN line plot

Based on Figure, it represents the absolute difference between accuracy estimations of each feature measure for class 0 and class 1 using the Anderson-Darling distance with KNN line pilot.

- Class 0 minimum accuracy 91.
- The maximum accuracy for Class 0 is 98 %.
- The minimum accuracy for Class 1 is 91%.
- The maximum accuracy for Class 1 is 98%.

  Then Anderson Darling's mean accuracy is =95.36%

4.12.2 Mean accuracy of Anderson-Darling distance with KNN.

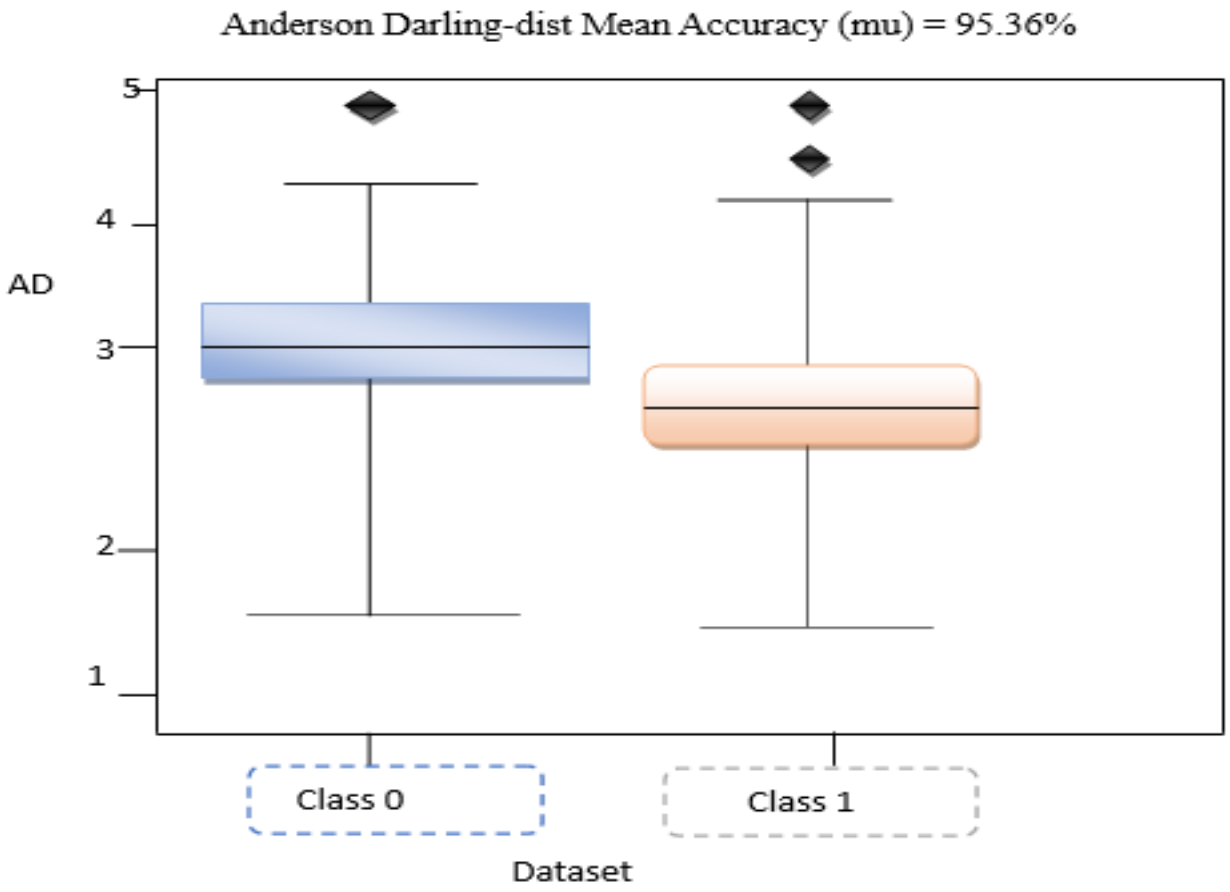


Figure 4.3: Mean Accuracy of Anderson-Darling distance with KNN

Use a box plot to display the variance in the Anderson-Darling distance between each class in the dataset.

The Wasserstein distance with KNN plots shows we can predict accuracy by calculating the ECDF-based statistical distance values based on the data. This is a novel/ new idea in AI/ ML safety and is the basis of the Safe ml idea.

4.12.3 Wasserstein_dist_dist with KNN line plot.

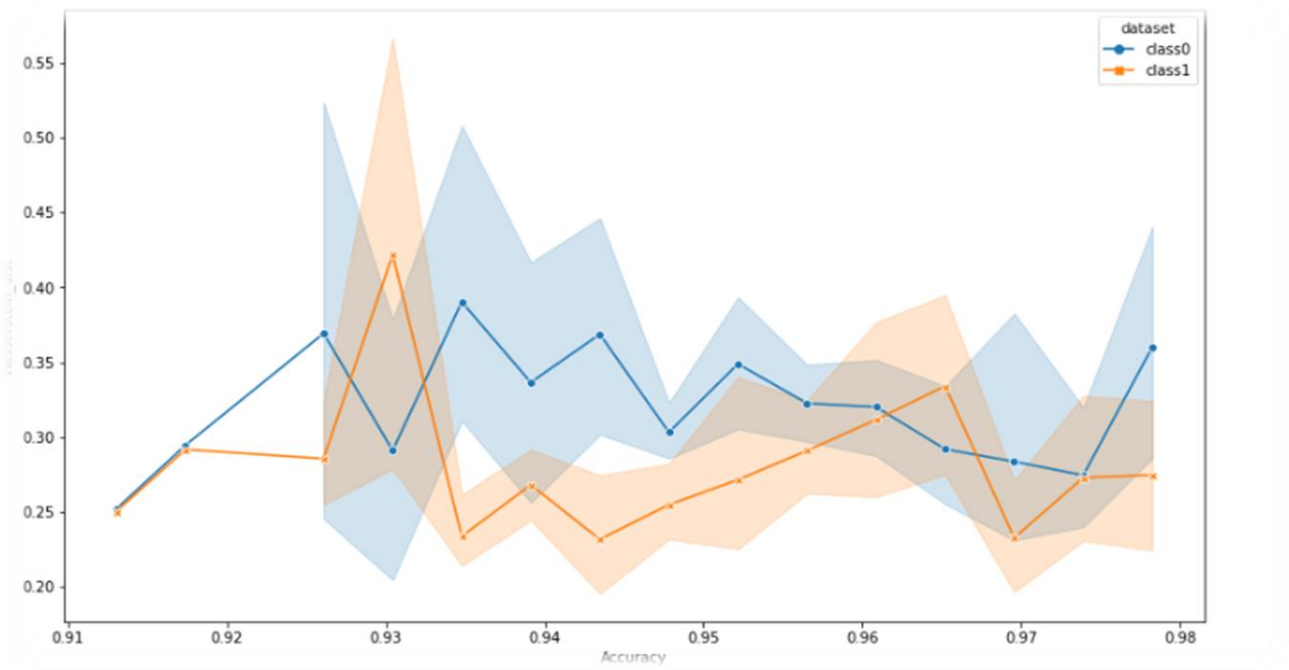


Figure 4.4: Wasserstein_dist_dist with KNN line plot

Based on Figure represents the (absolute) difference between accuracy estimations of each feature measure for class 0 and class 1 using Wasserstein_dist_dist with KNN

- The minimum accuracy for Class 0 is 91.
- The maximum accuracy of Class 0 is 98.
- The minimum accuracy for Class 1 is 91.
- The maximum accuracy for Class 1 is 98.

  Then Wasserstein_dist_dist with KNN Mean Accuracy =95.36%

4.12.4 Mean accuracy of Wasserstein_dist with KNN.

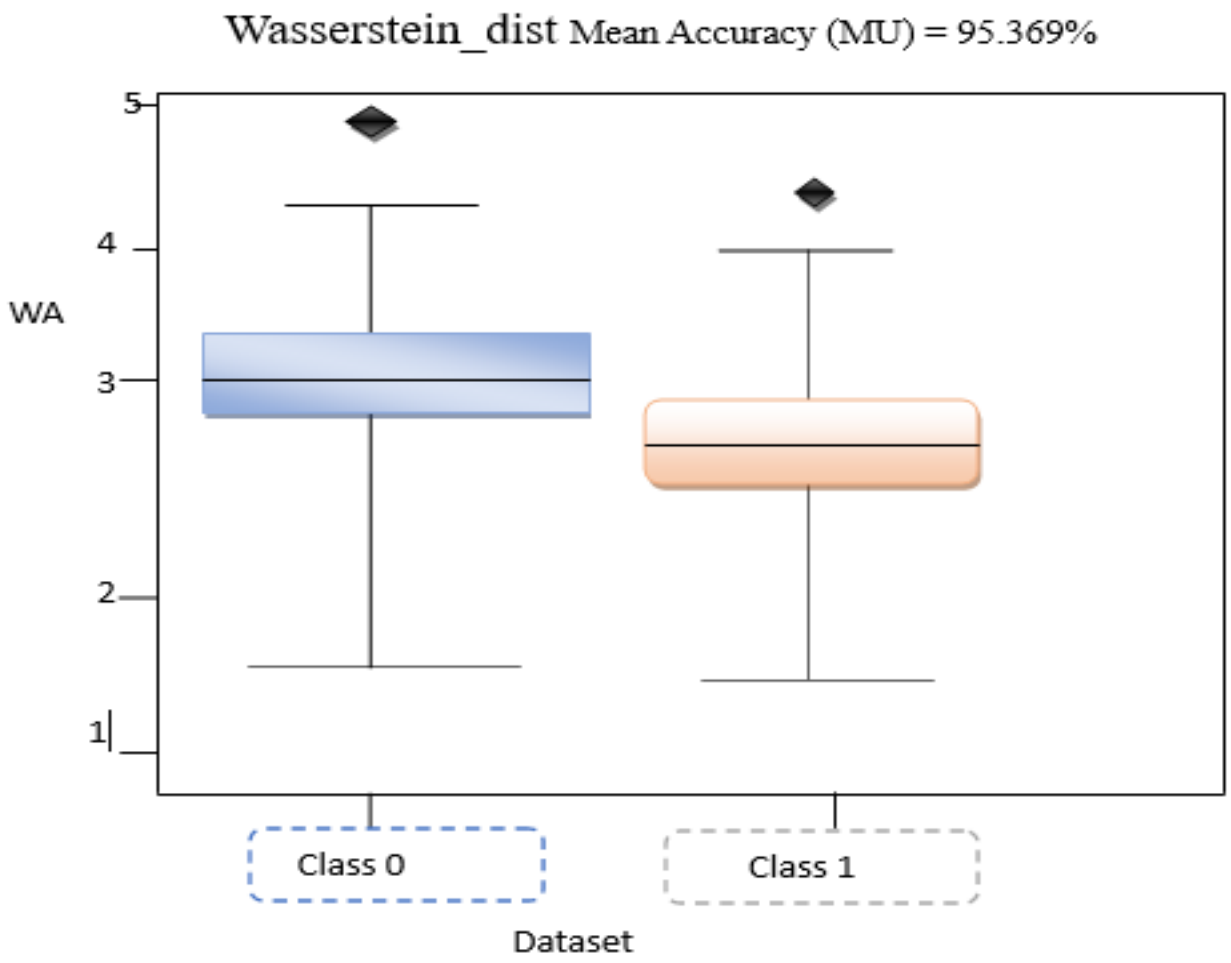


Figure 4.5: Mean accuracy of Wasserstein_dist with KNN

Mean accuracy can be predicted based on the ECDF-based statistical distance values calculated, based on the data. This is a novel/ new idea in the field of AI/ ML safety and is the basis of the SafeML idea for machine learning. Compare the Mean Accuracy of Wasserstein distance with machine learning classifier accuracy.

Essentially, these plots show that accuracy can be predicted based on the ECDF-based statistical distance values calculated, based on the data. This is a novel/ new idea in the field of AI/ ML safety and is the basis of the SafeML idea.

4.12.5 Kolmogorov Smirnov_dist with KNN line plot.

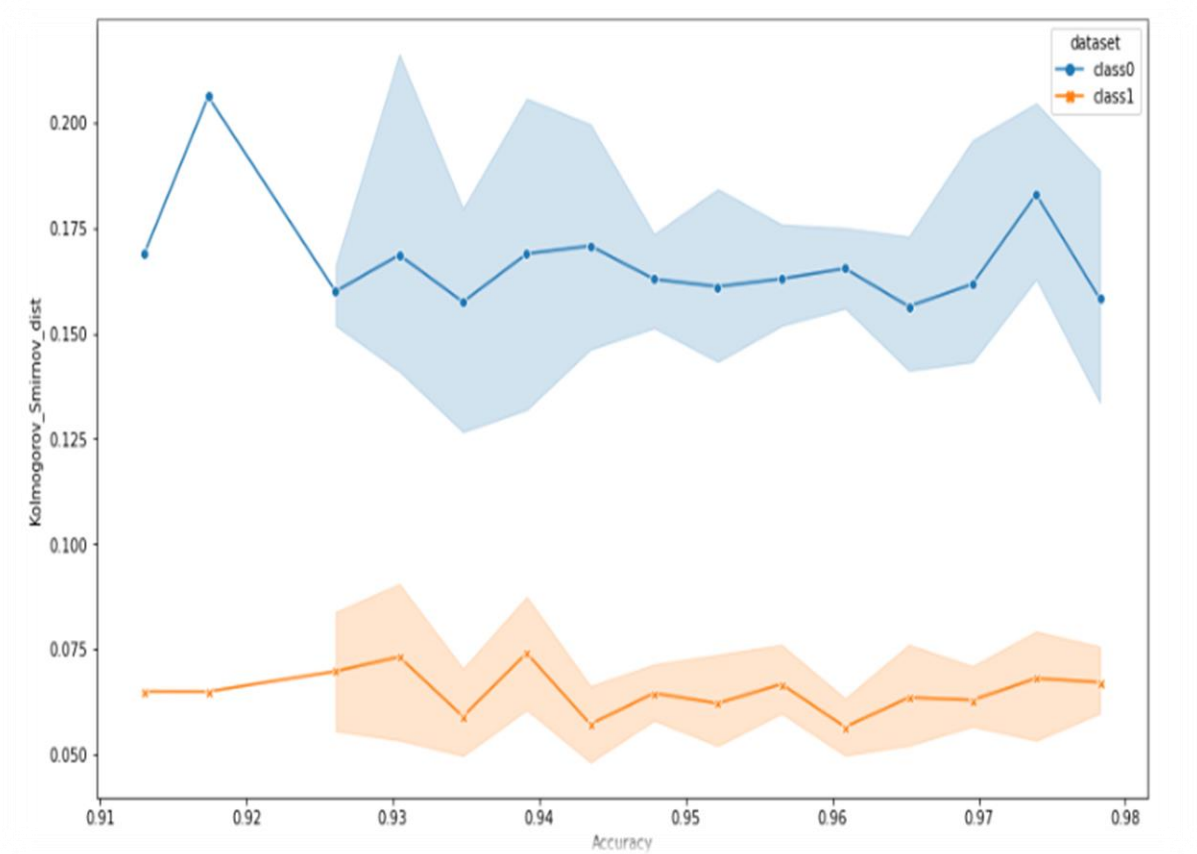


Figure 4.6: Kolmogorov_Smirnov_dist with KNN line plot

Based on Figure represents the (absolute) accuracy estimations of each feature measure for class 0 and class 1 using Wasserstein_dist_dist with KNN

- Class 0 minimum accuracy 91.
- Class 0 maximum accuracy 98.
- Class 1 minimum accuracy 91.
- Class 1 maximum accuracy 98.

Then Kolmogorov_Smirnov_dist with KNN Mean Accuracy =95.36%

4.12.6 Mean accuracy Kolmogorov_Smirnov_dist with KNN.

Essentially, these plots show we can predict accuracy by calculating ECDF-based statistical distance values from the data. This is a novel/ new idea in AI/ ML safety and is the basis of the Safe ml idea.

Kolmogorov Smirnov-dist with KNN Mean Accuracy =95.369%

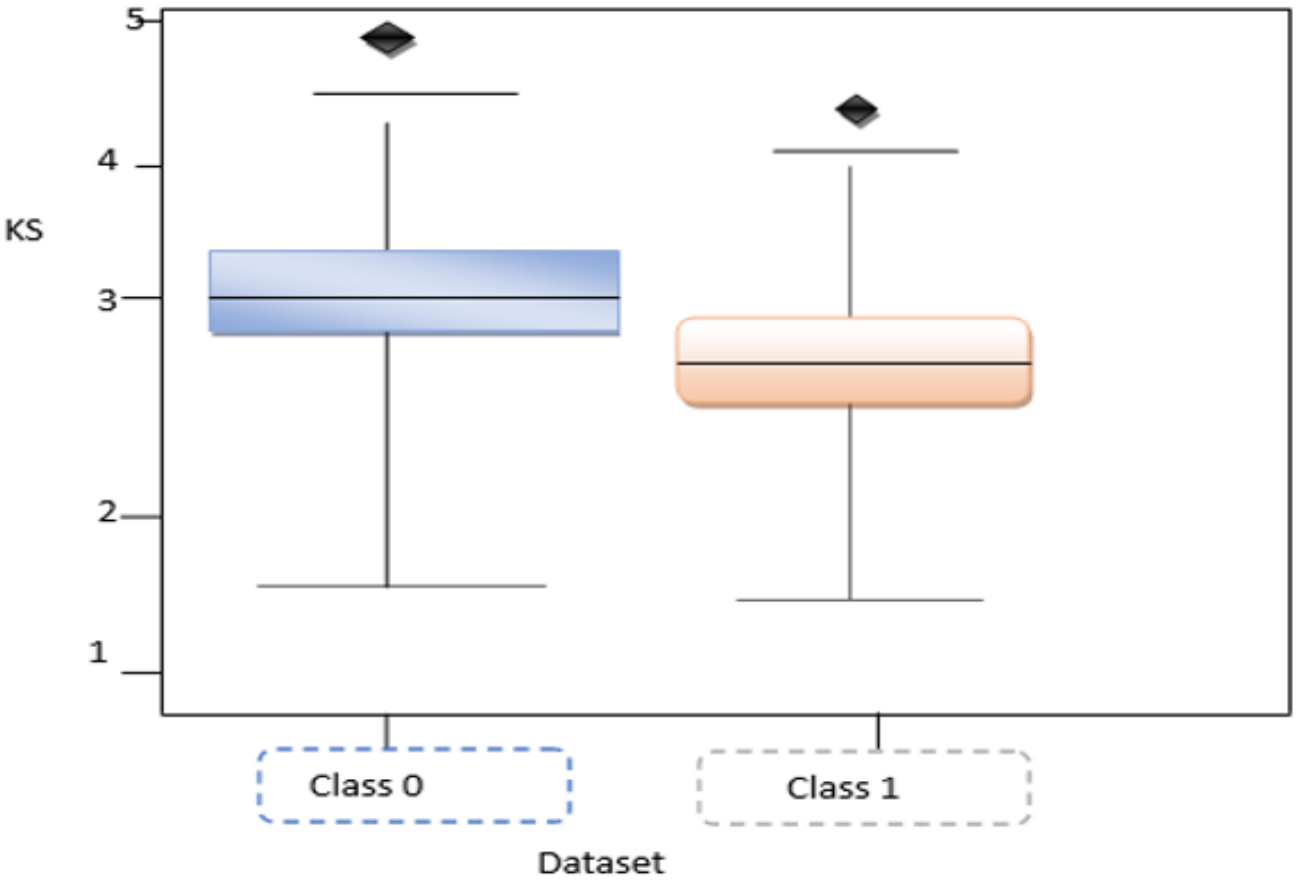


Figure 4.7: Mean accuracy Kolmogorov_Smirnov_dist with KNN

4.12.7 Anderson-Darling distance with RF line plot.

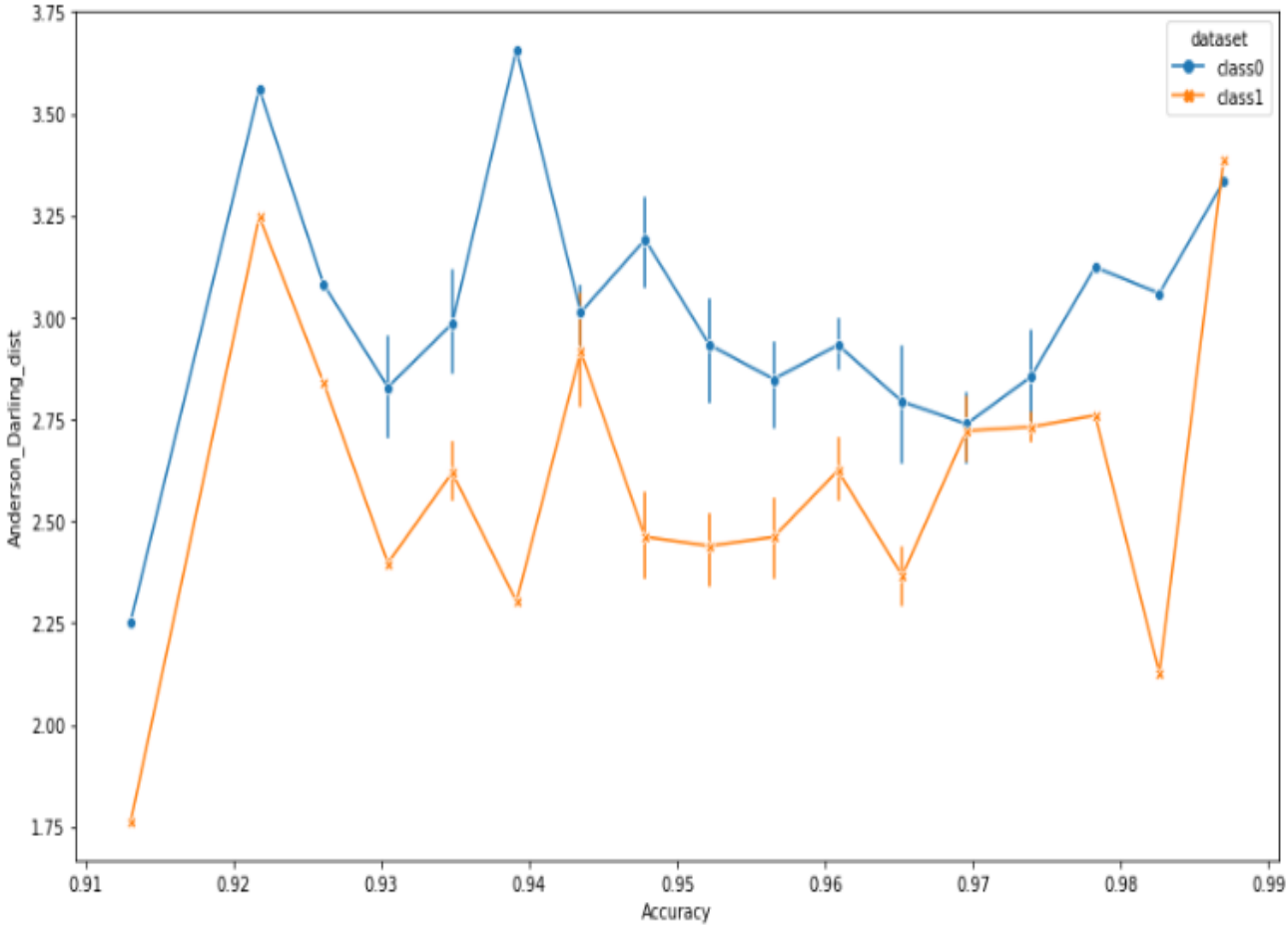


Figure 4.8: Anderson_Darling_dist with RF line plot

Based on the graph shown in Figure 4-7, it represents the (absolute) accuracy estimations of each feature measure for class 0 and class 1 using Anderson-Darling distance with RF.

- The minimum accuracy for Class 0 is 91 %.
- The maximum accuracy of Class 0 is 99 %.
- The minimum accuracy for Class 1 is 91 %.
- The maximum accuracy for Class 1 is 99 %.

Then Anderson_Darling_dist with RF Mean Accuracy =95.41%

4.12.8 Mean accuracy Anderson_Darling_dist with RF.

Mean accuracy Anderson_Darling_dist with RF=95.41%

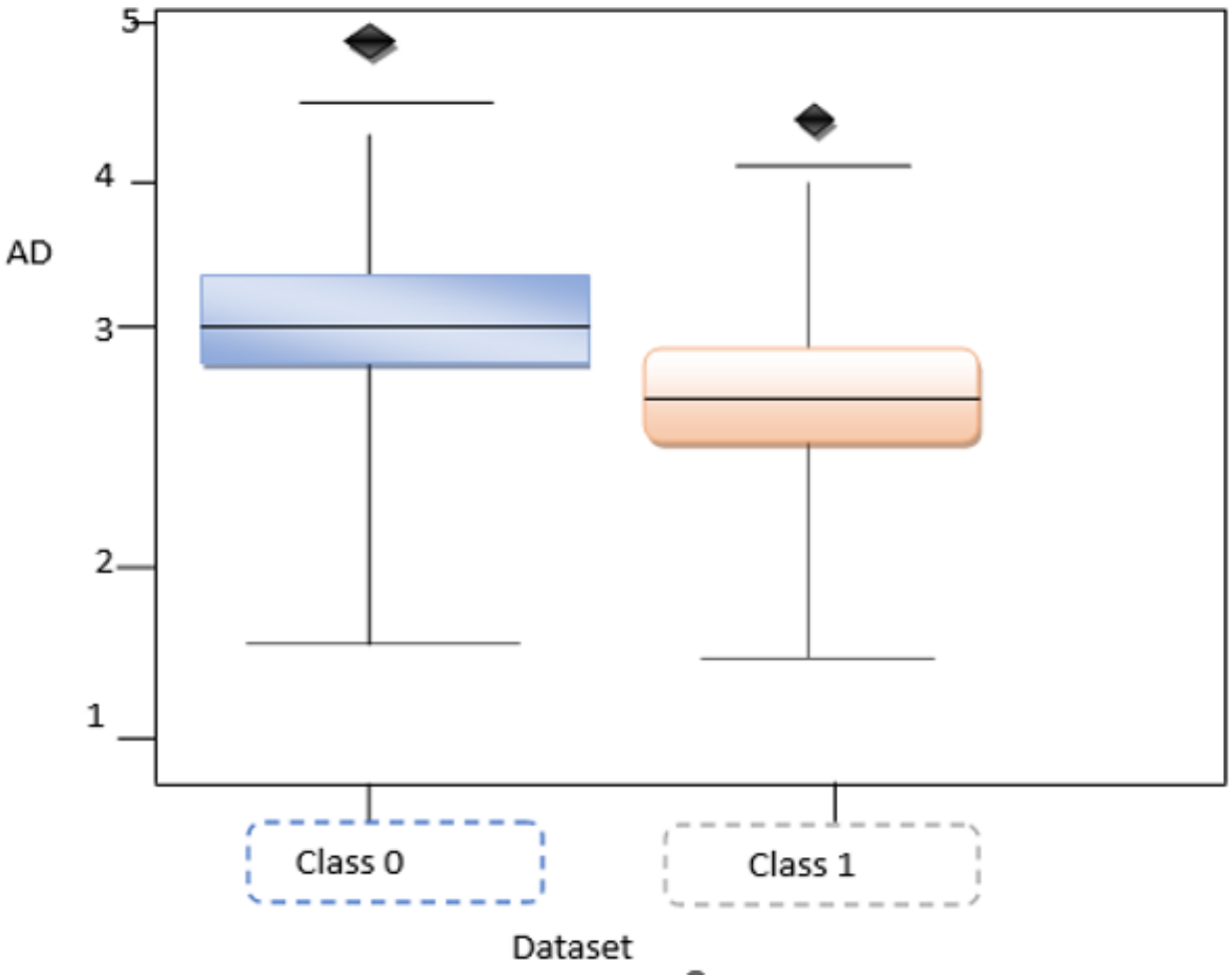


Figure 4.9: Mean accuracy Anderson_Darling_dist with RF

4.12.9 Mean Accuracy to the Statistical Distance Measure Algorithms

Table 4.22: Mean Accuracy of the Statistical Distance Measure Algorithms

ML Algorithms	Kolmogorov Smirnov_dist	Anderson Darling dist	DTS_dist	Wasserstein	Kuiper_dist
KNN	95.36	95.36	95.36	95.36	95.36
RF	95.41	95.41	95.41	95.41	95.41
DT	95.57	95.57	95.57	95.57	95.57
EX	95.08	95.08	95.08	95.08	95.08
SVM	95.45	95.45	95.45	95.45	95.45

When we compare the machine learning accuracy with safe machine learning statistical distance measure, there is a difference between them. Based on their accuracy, IDS performance is high confidence no more difference between them. Decision Tree is the highest confidence when we compare between them.

Table 4.23: Compare the ML Accuracy with Safe ML ECDF Accuracy

ML Algorithm	Kolmogoro Smirnov	Anderson Darling dist	DTS_dis	Wasserst dist	Kuiper dist	ML algorithms Accuracy
KNN	95.36	95.36	95.36	95.36	95.36	99
RF	95.41	95.41	95.41	95.41	95.41	98.26
DT	95.57	95.57	95.57	95.57	95.57	98
EX	95.08	95.08	95.08	95.08	95.08	98.26

When we compare the machine learning accuracy with safe machine learning statistical distance measure, there is a difference between them. Based on their accuracy, IDS performance is high confidence and no more difference between them.

Table 4.24: Compare PCA Accuracy and ECDF Accuracy

ML Algorithm	Kolmogorov Smirnov	Anderson Darling dist	DTS_dist	Wasserst dist	Kuiper_dist	Using PCA Accuracy
KNN	95.36	95.36	95.36	95.36	95.36	100
RF	95.41	95.41	95.41	95.41	95.41	100
DT	95.57	95.57	95.57	95.57	95.57	98
EX	95.08	95.08	95.08	95.08	95.08	100

When we compare **PCA** Accuracy with safe ml ECDF measure RF, the high confidence

4.12.10 Mean Accuracy Safe Machine learning, PCA and Machine learning

Table 4.25:compare the mean Accuracy of safe ml and ml algorithms

ML Algorithm	Safe Ml ECDF-based Distance Measure(accuracy)	Machine learning Algorithm-based Accuracy	Mean Accuracy
KNN	95.36	99	97.18
RF	95.41	98.26	96.83
DT	95.57	98	96.78
EX	95.08	98.26	96.67

When we compare the **mean** Accuracy of safe ml and ml algorithms, KNN the high confidence or high detection rate.

4.12.11 Mean Accuracy Safe Machine learning and Machine learning using PCA

Table 4.26: Compare the mean Accuracy of safe ml and ml algorithms

ML Algorithm	Safe Ml ECDF-based Distance Measure(accuracy)	Machine learning Algorithm-based using PCA Accuracy	Mean Accuracy
KNN	95.36	100	97.68
RF	95.41	100	97.70
DT	95.57	98	96.78
EX	95.08	100	97.54

When we compare the **mean** Accuracy of Safe Ml and ML using PCA, RF the high confidence or high detection rate.When we compare **all RF** , the high confidence or high detection rate

CHAPTER FIVE

5. CONCLUSION AND DISCUSSION

5.1 Discussion

This Section covers the experiments performed during the entire thesis. We broadly divide the experiments into two parts based on ML algorithms and safe machine learning algorithms using Empirical Cumulative Distribution Function (ECDF). The first part comprises the experiments done on signature detection and anomaly. The second part comprises experiments performed With Safe Machine learning algorithms using Empirical Cumulative Distribution Function (ECDF) using statistical distance measures including the Kolmogorov-Smirnov, Kuiper, Anderson-Darling, Wasserstein, and mixed Wasserstein-Anderson-Darling measures. This experiment also involves machine learning tools and others like SPSS. The evaluation becomes easy using this tool. Here, we used a CICIDS2017 dataset that clearly defines the total number of attacks and normal data. For this experiment, we have selected 56135 data of CICIDS2017, which is randomly selected and it comprises 42951 normal data and 13184 attacks. That is helpful to calculate TP, TN, FP, and FP, which are classified by ML from the testing dataset.

The rapid growth of artificial intelligence applications in various domains, particularly in autonomous intrusion detection, raises concerns from different perspectives, such as AI safety, AI responsibility, AI explain ability and interpretability, human-in-the-loop AI, and AI trustworthiness. This paper addressed distributional shift and its implications for the safety of machine learning classification tasks: intrusion detection. The paper proposed SafeML by extending SafeML to improve its capabilities for the human-in-the-loop procedure and ECDF-based statistical distance measures and applies them to intrusion detection algorithms for attack detection. I also used SPSS for descriptive analysis, such as mean, correlation analysis, standard deviation, etc. In this research, we have proposed some mechanisms to increase the performance of IDS detection using a Machine learning model with safe machine learning and security. The problem in IDS is the considerable number of data and features that cause the irrelevant features

and outlier data in the dataset. To handle that problem, we introduce a hybrid machine learning mechanism, which combines the feature selection process representing supervised learning with the data reduction process as the unsupervised learning method addition to safety and security using ECDF-based Statistical Distance Measures for Building Trustworthy and Explainable Machine Learning (ML) Classifiers.

In this research, we extensively studied benign, Dos, Dos, DDoS, brute force, infiltration, web attacks, Web Attack XSS, Bot intrusion detection systems, signature and anomaly-based IDS, as well as supervised and unsupervised modes of classification. We also conducted experiments to measure the capability of the proposed method using data classifiers such as KNN SVM, RF, ET, Decision Tree, and k-means Clustering. Based on the experimental results, we find that reducing the number of features by selecting only the relevant ones can improve the system's performance. Besides, separating important features leads to better results, specifically in the feature selection method. The K-means method can detect the local outlier data. After selecting 78 features from all the subsets, the application of feature selection, which reduced the number of features, resulted in the best performance. The result is almost not equal to the performance given by the set of Cleaned features and the time taken to build the model on the reduced features is less than the time taken by the set of Cleaned features.

We made a comparison of KNN, RF, ET, Decision Tree, and k-Means Clustering approaches of intrusion detection systems. The results of our experiments showed that classification RF results in high confidence, error rate, and reduced false alarm rate.

The expansion of ML applications to safety-critical domains is a major research question. We have identified and evaluated sets of statistical distance measures that can provide estimated upper error bounds in classification tasks based on the training and observed data distance. We have proposed how to use this approach as part of safety and security-critical systems, enabling active monitoring and enhancing their robustness. The study is still in its early stages, but we believe the results offer a promising starting point.

All attacks can't be of the same type, even when the signature-based classifiers yield accurate results for known attacks. To detect the unknown attacks, we therefore integrated the unsupervised module. We improved our findings by employing a changed k-means cluster technique.Notwithstanding, the unsupervised method's accuracy did not compete with signature-based classifier sets.

A number of experiments have done on the implementation of the proposed system. The first Experiment is on the efect of the dataset size in the performance of the proposed system and the result proves that the classication accuracy of the system increases in proportion with the size of the training dataset and the best classication accuracy is achieved is 100% when using 100% of the training dataset.

However, it could recognize a sizable percentage of the previously unidentified data types as attacks and legitimate traffic. We integrated the advantages and disadvantages of both approaches into a classifier that can apply to any situation and any type of traffic. There are a few areas where we could improve our proposed final module.

5.2 Conclusion

This study uses machine learning approaches to construct several hybrid machine learning models with safe machine learning. It integrates feature selection, supervised learning, unsupervised learning techniques, and distance measure ECDF like Kolmogorov-Smirnov, Kuiper, Anderson-Darling, Wasserstein, and mixed Wasserstein-Anderson-Darling measures. The hybrid machine learning model aims to identify newly emerging assaults without labels or those that have not yet been identified. In this study, we assess the performance of the suggested models, which are derived from the Canada Institute of Cyber Security at the University of New Brunswick, using the CIC-IDS2017 dataset.

Using the ET, KNN, DT, and RF algorithms for supervised learning, the K-Means unsupervised learning algorithm, PCA, and feature selection approaches, the models use the most recent research trends in the field, as described in the literature review section. Using the CIC-IDS2017 dataset, we implemented the models in the Python programming language.

We made a comparison of KNN, RF, ET, Decision Tree, PCA, and k-Means Clustering approaches of intrusion detection systems. According to our experiment, the machine-learning model use PCA is high Accuracy or KNN, RF, ET are 100 accuracies and according to safe machine learning ECDF method distance measure method RF is 95.41 and mean accuracyof RF is 97.70 the most relative when we compare to other than RF is high confidence, error rate, and reduced false alarm rate. To protect an organization's network from assaults, network security uses a variety of methods and tools. The primary method of resolving network security issues and reducing the influence of intruders on the uninterrupted operation of an organization is the use of intrusion detection systems.

This more intelligent security method will monitor and promptly notify issues while keeping a vigilant watch on the network. The hybrid intrusion detection system, which can analyze both known and new threats, is the major topic of our thesis. In several organizations, this lessens the issues brought on by hacker-caused network attacks.

The contribution of this paper to addressing both safety and high confidence within a single

concept of protection applicable during the operation of ML systems is active monitoring of the behavior and the operational context of the data-driven system based on distance measures of the Empirical Cumulative Distribution Function (ECDF).

5.3 Future Work and Recommendation

This study of safe machine learning is still in the early stages, but we believe the results offer a promising starting point. In the future, machine learning techniques for building trustworthy and explainable machine learning (ML) classifiers should incorporate safety and security features, as well as tools to customize the IDS and handle imbalanced data in many classes.

Even though our effort has yielded some encouraging results, more system improvement is required to detect more known and undiscovered assaults. As depicted in the experiment, this safe ML and lack of safety affect the overall performance of the system. So in the future thesis of machine-based thesis work, added safe machine is the best for getting excellent performance or prediction.

The proposed approach is now just solving the safety evaluation problem of the machine-learning-based classification. However, we believe it may be simply expanded for clustering, dimension reduction or any problem that can be evaluated using statistical difference.Machine learning methods can withstand shifts or variations in dataset distributions. This may impair the effectiveness of the discussed distance measures. The recommended measures can provide extra validation of robustness and contribute to certification.

REFERENCES

1. Sugi SSS. Investigation of Machine Learning Techniques in Intrusion Detection System for IoT Network. 2020;1164–7.
2. Yang L, Moubayed A, Shami A. MTH-IDS : A Multi-Tiered Hybrid Intrusion Detection System for Internet of Vehicles. :1–17.
3. Aslansefat K, Sorokos I, Whiting D, Tavakoli Kolagari R, Papadopoulos Y. SafeML: Safety Monitoring of Machine Learning Classifiers Through Statistical Difference Measures. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 2020;12297 LNCS:197–211.
4. Aslansefat K. How to Make Your Classifier Safe. Towar Data Sci [Internet] 2020;(June). Available from: https://towardsdatascience.com/how-to-make-your-classifier-safe-46d55f39f1ad
5. Aslansefat K, Sorokos I, Whiting D, Kolagari RT, Papadopoulos Y. SafeML : An Approach for Safety Monitoring of Machine Learning Classifiers through Statistical Difference Measure. 2018;
6. Lee I. Internet of Things (IoT) cybersecurity: Literature review and iot cyber risk management. Futur Internet 2020;12(9).
7. Li J, Kuang X, Lin S, Ma X, Tang Y. Privacy preservation for machine learning training and classification based on homomorphic encryption schemes. Inf Sci (Ny) 2020;526:166–79.
8. Liu H, Lang B. Machine learning and deep learning methods for intrusion detection systems: A survey. Appl Sci 2019;9(20).
9. Chua TH, Salam I. Evaluation of Machine Learning Algorithms in Network-Based Intrusion Detection Using Progressive Dataset. Symmetry (Basel) 2023;15(6).
10. Jose S, Malathi D, Reddy B, Jayaseeli D. A Survey on Anomaly Based Host Intrusion Detection System. J Phys Conf Ser 2018;1000(1).
11. Brunetti F, Matt DT, Bonfanti A, De Longhi A, Pedrini G, Orzes G. Digital transformation challenges: strategies emerging from a multi-stakeholder approach. TQM J 2020;32(4):697–724.
12. Aslansefat K, Kabir S, Abdullatif A, Vasudevan V, Papadopoulos Y. Toward Improving Confidence in Autonomous Vehicle Software: A Study on Traffic Sign Recognition Systems. Computer (Long Beach Calif) 2021;54(8):66–76.
13. Aslansefat K. Safety Evaluation of Machine Learning Classifiers through Statistical Difference Measures (Implementation in R). 2020;
14. Yang L, Moubayed A, Hamieh I, Shami A. Tree-based Intelligent Intrusion Detection System in Internet of Vehicles. (Ml).
15. Atilla O, Hamit E. A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015. PeerJ 2016;(April):0–21.
16. Jeevaraj D, Karthik B, Sriram M, Vijayaragavan SP, Gokulakrishnan D. Intrusion detection in WSN using Supervised Machine Learning Techniques. Int J Intell Syst Appl

Eng 2024;12(9s):483–90.
17. Charbuty B, Abdulazeez A. Classification Based on Decision Tree Algorithm for Machine Learning. J Appl Sci Technol Trends 2021;2(01):20–8.
18. Dumoulin J, Canevet O, Villamizar M, Nunes H, Khaled OA, Mugellini E, et al. UNICITY: A depth maps database for people detection in security airlocks. Proc AVSS 2018 - 2018 15th IEEE Int Conf Adv Video Signal-Based Surveill 2018;
19. Yeo LH, Che X, Lakkaraju S. Understanding Modern Intrusion Detection Systems: A Survey. 2017;Available from: http://arxiv.org/abs/1708.07174
20. Bukac V, Tucek P, Deutsch M. Advances and challenges in standalone host-based intrusion detection systems. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics) 2012;7449 LNCS:105–17.
21. Sun X, Zhang P, Liu JK, Yu J, Xie W. Private Machine Learning Classification Based on Fully Homomorphic Encryption. IEEE Trans Emerg Top Comput 2020;8(2):352–64.
22. Kumar RL, Khan F, Kadry S, Rho S. A Survey on blockchain for industrial Internet of Things: Blockchain for Internet of Things. Alexandria Eng J 2022;61(8):6001–22.
23. Aloffi BO, Alzain MA, Masud M, Jhanjhi NZ, Al-amri J. Real Time Network Security Intrusions and Risk Management : A Survey Detector Monitored System - Information Source. 2021;12(10):3101–9.
24. Khuat TT, Kedziora DJ, Gabrys B. The Roles and Modes of Human Interactions with Automated Machine Learning Systems. 2022;Available from: https://arxiv.org/abs/2205.04139v1
25. GRESSE VON WANGENHEIM C, ALVES N da C, RAUBER MF, HAUCK JCR, YETER IH. A Proposal for Performance-based Assessment of the Learning of Machine Learning Concepts and Practices in K-12. Informatics Educ 2021;00(00).
26. Kuleto V, Ilić M, Dumangiu M, Ranković M, Martins OMD, Păun D, et al. Exploring opportunities and challenges of artificial intelligence and machine learning in higher education institutions. Sustain 2021;13(18):1–16.
27. Massarelli C, Campanale C, Uricchio VF. A handy open-source application based on computer vision and machine learning algorithms to count and classify microplastics. Water (Switzerland) 2021;13(15).
28. Rajasekaran K. Classification and Importance of Intrusion Detection System. 2020;(June):6–10. Available from: http://sites.google.com/site/ijcsis/
29. Sarker IH. Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective. SN Comput Sci 2021;2(3).
30. Sarker IH. Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. SN Comput Sci [Internet] 2021;2(6):1–20. Available from: https://doi.org/10.1007/s42979-021-00815-1
31. Sharma P, Jain S, Gupta S, Chamola V. Role of machine learning and deep learning in securing 5G-driven industrial IoT applications. Ad Hoc Networks 2021;123:1–38.
32. Maseer ZK, Yusof R, Bahaman N, Mostafa SA, Foozy CFM. Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset.

IEEE Access 2021;9:22351–70.
33. Jang-Jaccard J, Nepal S. A survey of emerging threats in cybersecurity. J Comput Syst Sci http://dx.doi.org/10.1016/j.jcss.2014.02.005
34. Trends E. An Outlier Detection Using Clustering Algorithms and Its Techniques. 2022;8(2):639–42.
35. Burkart N, Huber MF. A survey on the explainability of supervised machine learning. J Artif Intell Res 2021;70:245–317.
36. Sun B, Zhang Y, Zhou Q, Zhang X. Effectiveness of Semi-Supervised Learning and Multi-Source Data in Detailed Urban Landuse Mapping with a Few Labeled Samples. Remote Sens 2022;14(3).
37. Kost S, Rheinbach O, Schaeben H. Using logistic regression model selection towards interpretable machine learning in mineral prospectivity modeling. Geochemistry 2021;81(4).
38. Divakaran S. Data Science: Principles and Concepts in Modeling Decision Trees. 2022;55–74.
39. Anelli VW, Di Noia T, Di Sciascio E, Ferrara A, Mancino ACM. Sparse feature factorization for recommender systems with knowledge graphs. Association for Computing Machinery; 2021.
40. Liu J, Xiao K, Luo L, Li Y, Chen L. An intrusion detection system integrating network-level intrusion detection and host-level intrusion detection. Proc - 2020 IEEE 20th Int Conf Softw Qual Reliab Secur QRS 2020 2020;122–9.
41. Rizwan A, Iqbal N, Ahmad R, Kim DH. Wr-svm model based on the margin radius approach for solving the minimum enclosing ball problem in support vector machine classification. Appl Sci 2021;11(10).
42. Gaye B, Zhang D, Wulamu A. Improvement of Support Vector Machine Algorithm in Big Data Background. Math Probl Eng 2021;2021.
43. Pan Y, Zhao X, Xu Z, Li J, Li Y, Liu L. Research on Abnormal Behavior Recognition of Buses Based on Improved Support Vector Machine. J Adv Transp 2021;2021.
44. Pellegrino E, Jacques C, Beaufils N, Nanni I, Carlioz A, Metellus P, et al. Machine learning random forest for predicting oncosomatic variant NGS analysis. Sci Rep [Internet] 2021;11(1):1–14. Available from: https://doi.org/10.1038/s41598-021-01253-
45. Anaissi A, Kennedy PJ, Goyal M, Catchpoole DR. A balanced iterative random forest for gene selection from microarray data. BMC Bioinformatics 2013;14(1).
46. López B, Torrent-Fontbona F, Viñas R, Fernández-Real JM. Single Nucleotide Polymorphism relevance learning with Random Forests for Type 2 diabetes risk prediction. Artif Intell Med 2018;85:43–9.
47. Teixeira D, Malta S, Pinto P. A Vote-Based Architecture to Generate Classified Datasets and Improve Performance of Intrusion Detection Systems Based on Supervised Learning. Futur Internet 2022;14(3).
48. Whiting D. S ml: s m m l c s d m. 2018;1–17.
49. Goundar S. Chapter 3 - Research Methodology and Research Method. Cloud computing.

2012;(Marchhttps://www.researchgate.net/publication/333015026_Chapter_3_-_Research_Methodology_and_Research_Method/comments
50. Azam Z, Islam MM, Huda MN. Comparative Analysis of Intrusion Detection Systems and Machine Learning-Based Model Analysis Through Decision Tree. IEEE Access 2023;11:80348–91.
51. Daniel E. The Usefulness of Qualitative and Quantitative Approaches and Methods in Researching Problem-Solving Ability in Science Education Curriculum. J Educ Pract 2016;7(15):91–100.
52. Panigrahi R, Borah S. A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. Int J Eng Technol 2018;7(3.24 Special Issue 24):479–82.
53. Ahmad T. A hybrid machine learning method for increasing the performance of network intrusion detection systems. 8(2021):1–29.
54. Pelletier Z, Abualkibash M. Evaluating the CIC IDS-2017 Dataset Using Machine Learning Methods and Creating Multiple Predictive Models in the Statistical Computing Language R. 2020;5(2):187–91.
55. Abdulraheem MH, Ibraheem NB. A DETAILED ANALYSIS OF NEW INTRUSION DETECTION. 2019;97(17).
56. ElSayed MS, Le-Khac NA, Albahar MA, Jurcut A. A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique. J Netw Comput Appl 2021;191(July).
57. Azhagusundari B, Thanamani AS. Feature Selection based on Information Gain. Int J Innov Technol Explor Eng 2013;2(2):18–21.
58. Kherif F, Latypova A. Principal component analysis. Mach Learn Methods Appl to Brain Disord 2019;209–25.

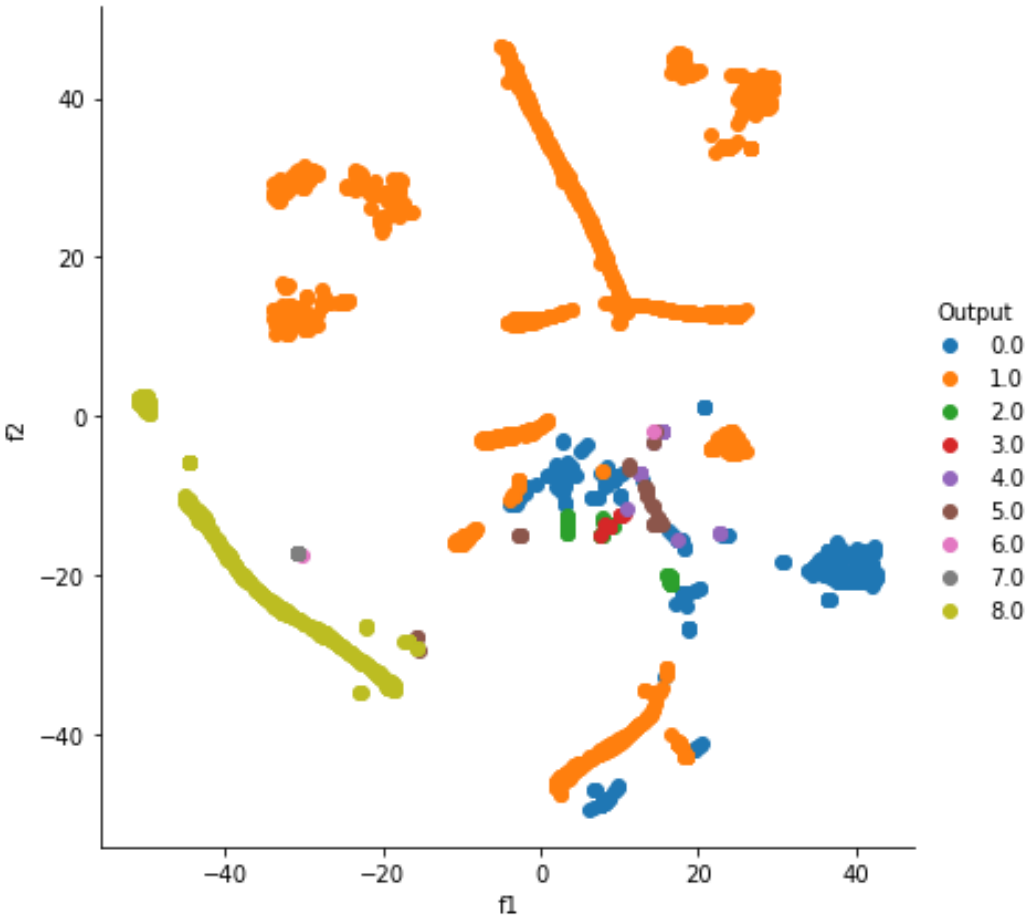
APPENDIX

Appendix A: Total Dataset for Experiments

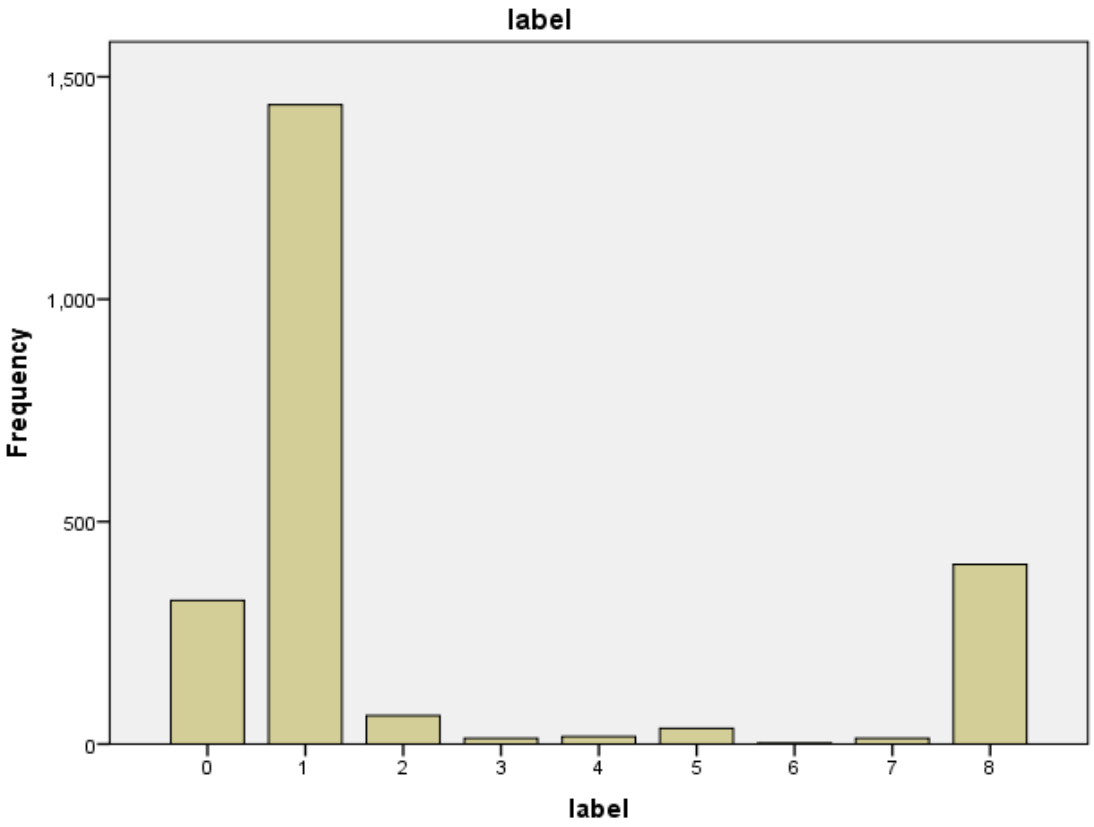
	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	Fwd Packet Length Std	Bwd Packet Length Max	.
0	4	2	0	12	0	6	6	6.0	0.0	0	.
1	1	2	0	12	0	6	6	6.0	0.0	0	.
2	1	2	0	12	0	6	6	6.0	0.0	0	.
3	1	2	0	12	0	6	6	6.0	0.0	0	.
4	3	2	0	12	0	6	6	6.0	0.0	0	.
...	...	...	...	...	...	...	...	...	...	...	.
80284	81	1	1	0	0	0	0	0.0	0.0	0	.
80285	32056	2	2	88	200	44	44	44.0	0.0	100	.
80286	52	1	1	0	0	0	0	0.0	0.0	0	.
80287	31462	2	2	72	328	36	36	36.0	0.0	164	.
80288	3	2	0	12	0	6	6	6.0	0.0	0	.

80289 rows × 78 columns

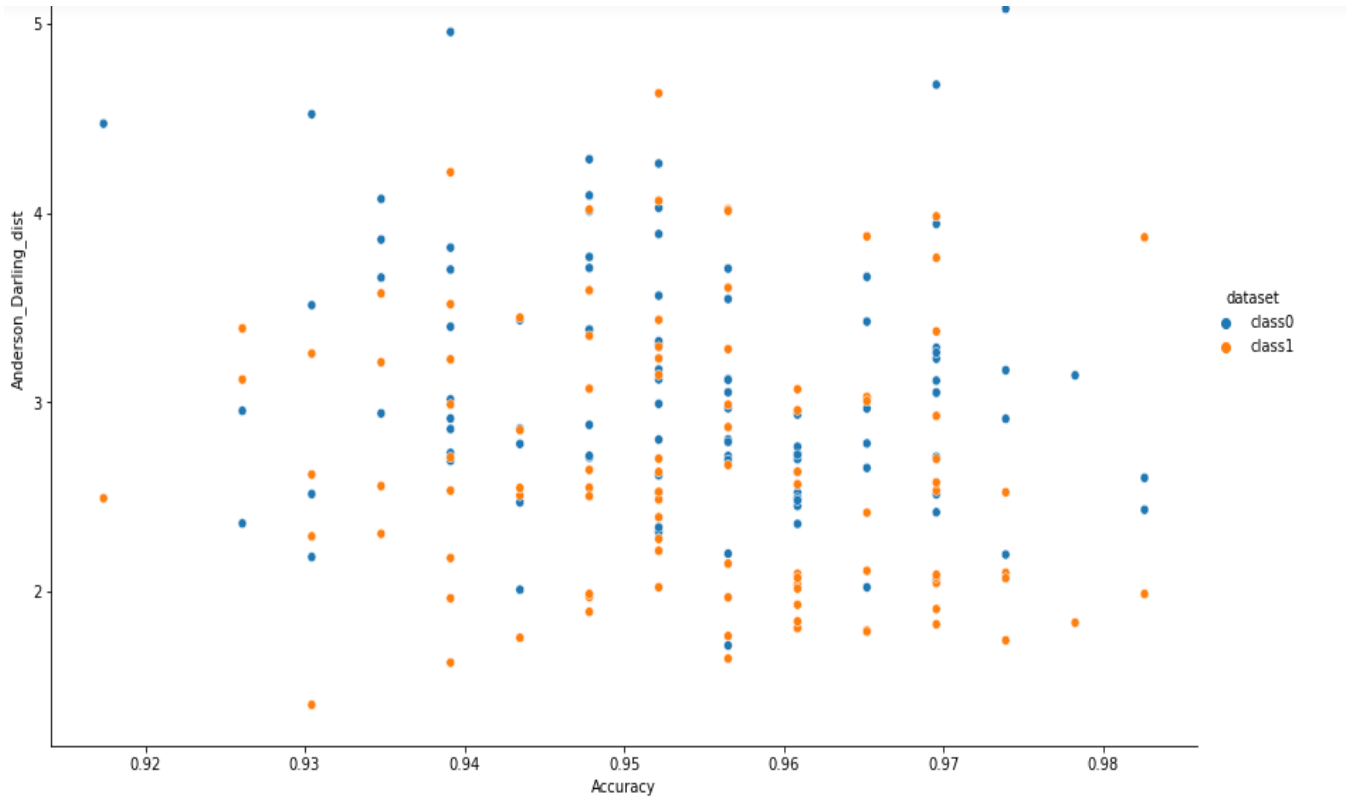
Appendix B: Analysis of linear separability in any feature using PCA



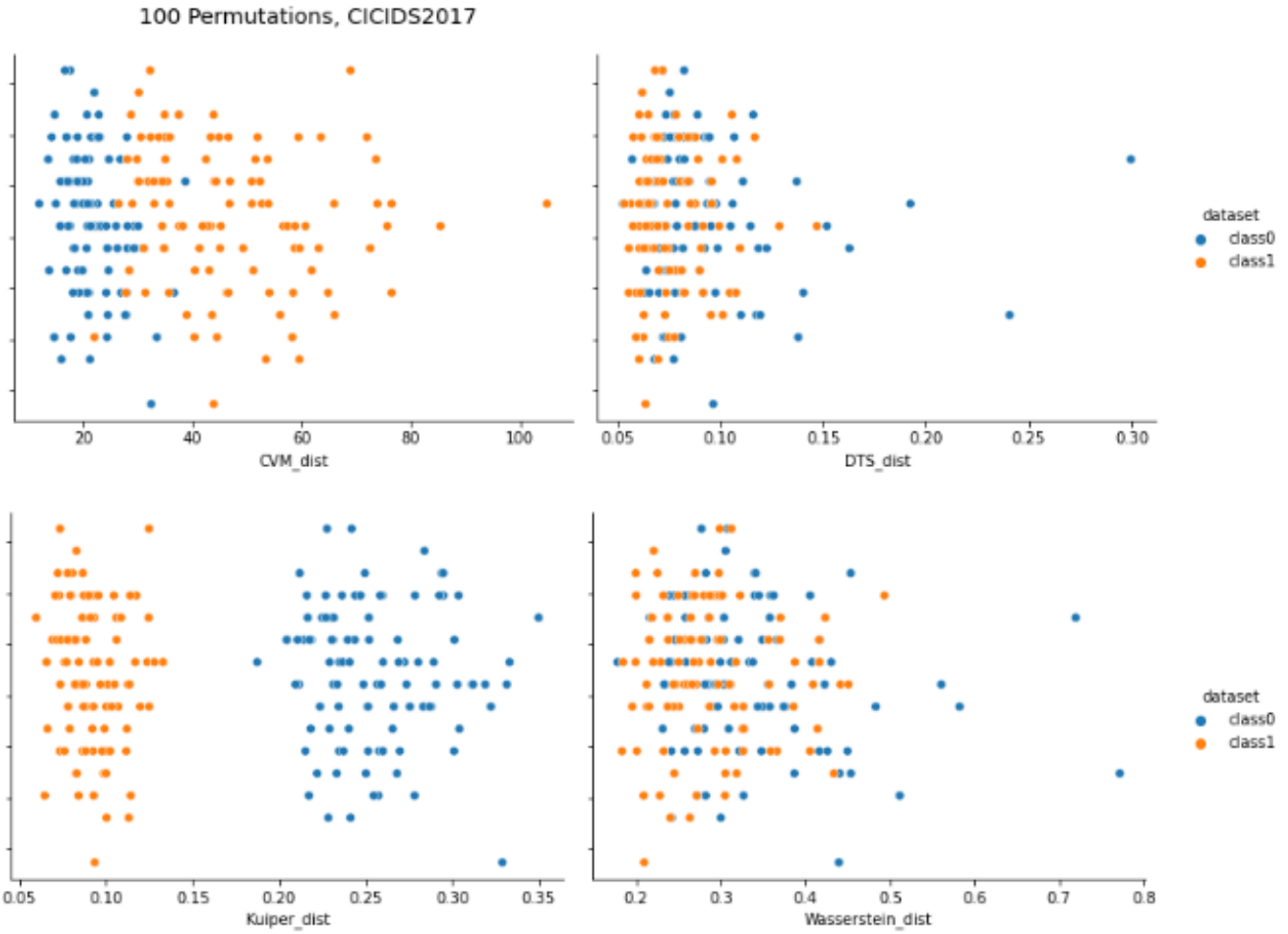
Appendix C: Distribution of target class in training data



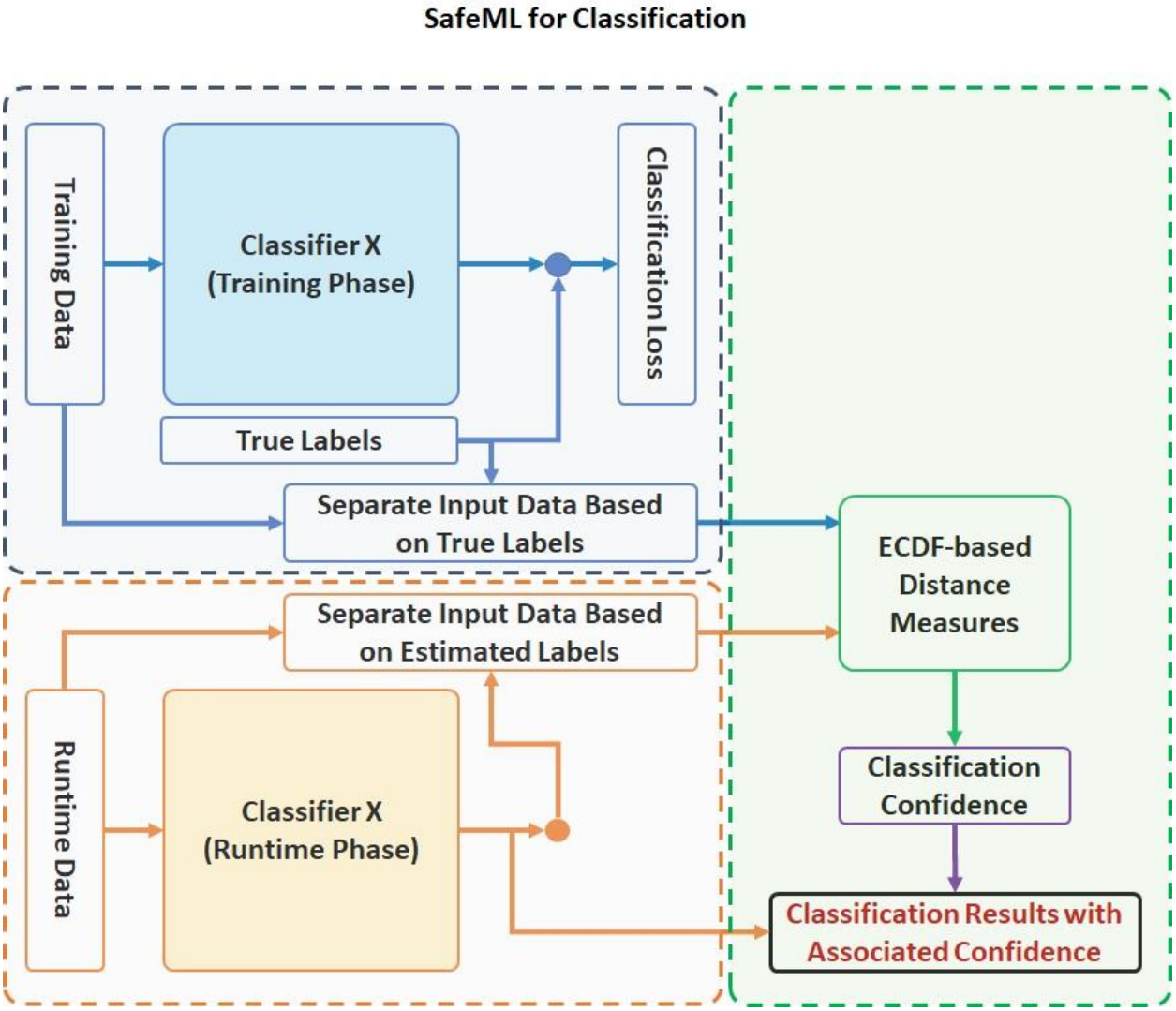
Appendix D: Seaborn plotting for Anderson Darling



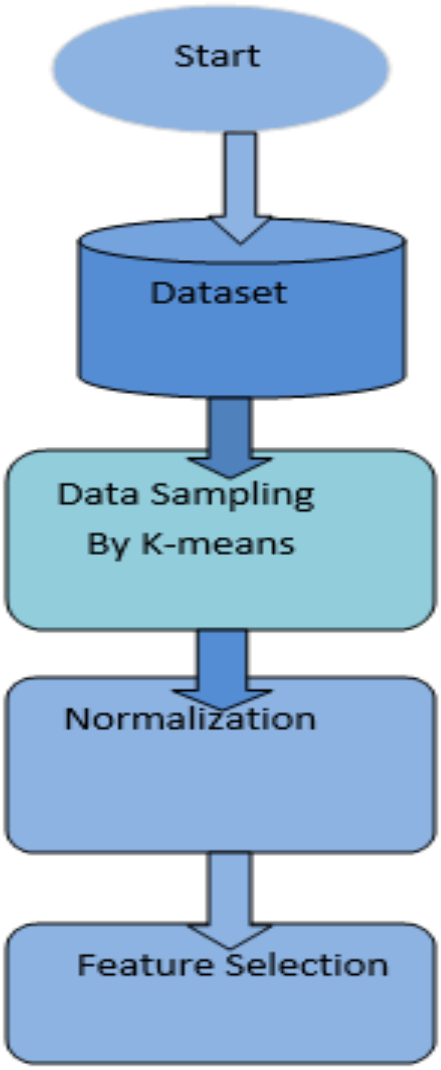
Appendix E: the graphs/ plots showing ECDF



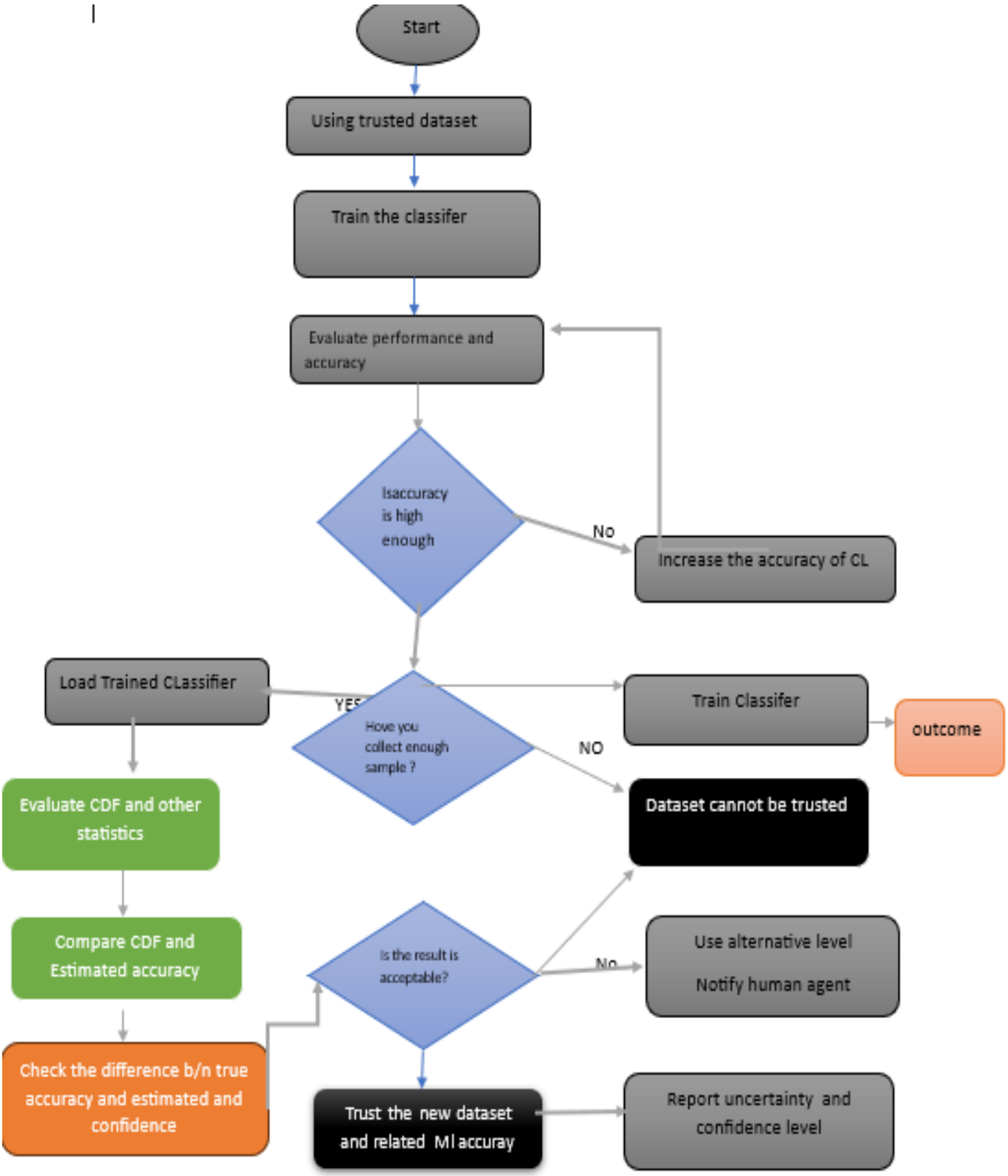
Appendix F: Safe ML for Classification



Appendix *G*: Data Cleaning and Preprocessing



Appendix H: Flow Chart of Safe ML classification



Appendix I: diagram analysis of the label(dependent Variable)

