“CYBER SECURITY THREATS AGAINST INTERNET OF THINGS(IOT) DEVICES: ANALYSIS AND SECURITY SOLUTIONS”

Submitted By

Shishir Chandra Das

ID: 201-35-589

Department of Software Engineering

Supervised By

Nuruzzaman Faruqui

Assistant Professor

Department of Software Engineering

Daffodil International University

This thesis submitted in fulfillment of the requirement for the degree of Bachelor of Science in Software Engineering

Spring 2024

APPROVAL

This thesis titled on "CYBER SECURITY THREATS AGAINST INTERNET OFTHINGS(IOT) DEVICES: ANALYSIS AND SECURITY SOLUTIONS", submitted by **Shishir Chandra Das (ID: 201-35-589)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

Chairman

Dr. Engr. AKM Masum
Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 1

Dr. Marzia Ahmed
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 2

Md. Rajib Mia
Lecturer
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

External Examiner

Dr. Md. Manowarul Islam
Associate Professor
Dept. of Computer Science and Engineering
Jagannath University

DECLARATION

I hereby declare that, this thesis report is done by me under the supervision of Mr. Nuruzzaman Faruqui, Assistant Professor, Department of Software Engineering, Daffodil International University, in partial fulfillment my original work. I am also declaring that neither this thesis nor any part therefore has been submitted elsewhere for the award of bachelor of any degree.

Supervised By

Nuruzzaman Faruqui

Assistant Professor

Department of Software Engineering

Daffodil International University

Submitted By

Shishir Chandra Das

Shishir Chandra Das

ID: 201-35-589

Department of Software Engineering

Daffodil International University

ACKNOWLEDGEMENT

First and foremost, I would want to express my sincere gratitude to Almighty God for His tremendous grace, which has allowed me to successfully finish my senior thesis.

My deepest appreciation and debt are owed to **Mr. Nuruzzaman Faruqui**, Lecturer (Senior Scale), Department of SWE Daffodil International University, Dhaka. Our supervisor's extensive knowledge and sincere interest in the topic are necessary for us to finish this thesis. His endless patience, academic guidance, unwavering support, regular and enthusiastic supervision, constructive criticism, perceptive advice, reading many poor versions and correcting them at every turn have all contributed to making this project feasible.

I would like to extend my sincere gratitude to the head of the SWE department, as well as to my seniors and supervising lecturers, for their essential help in finishing our thesis. I want to thank everyone of the instructors and personnel at Daffodil International University's SWE department for their help and advice.

I want to express my gratitude to every student at Daffodil International University who participated in this discussion while finishing the assigned material.

Lastly, I must respectfully thank my family for their unwavering support and patience.

ABSTRACT

The Internet of Things (IoT) is becoming more and more prevalent, which gives cybercriminals a larger assault surface. The effectiveness of machine learning for intrusion detection in Internet of Things security is examined in this work. A dataset from the ML Repository is used in the study to examine the effectiveness of Decision Tree (DT), Random Forest (RF), KNearest Neighbors (KNN), XGBoost, AdaBoost, and Logistic Regression (LR). RF, DT, and XGBoost each scored a remarkable 0.99 accuracy, while KNN came in at 0.94.A stacking ensemble strategy is further investigated in the paper, combining LR as the meta learner with RF, DT, XGBoost, AdaBoost, and KNN as base learners. Accuracy scores for this ensemble were 0.99 for RF, DT, and XGBoost, 0.93 for KNN, and 0.98 for the meta learner LR. Overall performance was enhanced. Notably, because the dataset was balanced, SMOTE was not found to be necessary in order to correct class imbalance. According to our research, machine learning, intrusion detection, and IoT security are three ensemble approaches that are highly effective at protecting IoT environments. By demonstrating the usefulness of ensembles for obtaining reliable intrusion detection, this study advances the practice. Stacking ensemble is a possible approach that may help mitigate overfitting even while individual models performed well. For even more dependable IoT security systems, our work opens the door to future research into the best model selection and hyperparameter tuning.

Keyword: Cybersecurity, stacking ensemble, SMOTE, ML Repository.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1. Introduction:

The Internet of Things' (IoT) remarkable proliferation has fundamentally transformed many companies by creating a networked environment where devices communicate with one another to boost convenience, operational effectiveness, and data-driven decision-making. The Internet of Things (IoT) is beginning to become indispensable in today's society. It can be found in everything from smart homes with smart appliances to industrial settings using cutting-edge sensors for predictive maintenance. According to Gartner, there will be more than 75 billion IoT devices in use globally by 2025, demonstrating the technology's broad use and integration into daily activities and critical infrastructure. But this exponential expansion has also made it simpler for hackers to initiate assaults, which raises serious security and privacy concerns.

The complexity and frequency of cybersecurity attacks have increased due to the multiplicity and inherent vulnerabilities of IoT devices. According to a 2023 Symantec analysis, there has been a 600% increase in IoT threats over the last five years. These attacks range from straightforward password leaks to highly skilled software that can build enormous botnets, like the infamous Mirai botnet that took over millions of Internet of Things devices in 2016 and severely damaged the entire network. Because Internet of Things devices have different hardware and software configurations, they are heterogeneous, making it difficult to build uniform security standards. It is extremely difficult to defend against cyberattacks as a result.

One of the most pressing problems with IoT cybersecurity is that these devices could act as entry points for more extensive network attacks. For example, in the healthcare sector, implanted medical devices such as insulin pumps and pacemakers can save lives, but often have inadequate security features that make them vulnerable to hackers. 67% of healthcare companies were impacted by IoT-related cyberattacks in 2022, according to the Ponemon.

Institute. In contrast, cyberattacks on infrastructure components like as traffic lights, power

grids, and surveillance systems are becoming increasingly frequent in the setting of smart cities, putting both data integrity and public safety at risk. These incidents highlight the critical need for extensive security frameworks that are specifically created to address the unique requirements of IoT ecosystems.

Proactive legislative actions must be combined with creative security solutions to counter these sophisticated attacks. There are a lot of opportunities to improve Internet of Things security with emerging technologies like blockchain, AI, and machine learning. For Internet of Things transactions, for example, blockchain can offer decentralized, immutable ledgers that guarantee data integrity and accuracy. Real-time security breach detection and anomalous activity detection are possible with artificial intelligence and machine learning. International standards and regulatory framework development can also help manufacturers prioritize security throughout the design process by supporting the use of best practices in IoT security. This study attempts to provide a thorough examination of the existing cybersecurity challenges that IoT faces by looking into workable security solutions.

1.2. The Beginning of IoT

Thanks to advancements in wireless communication and sensor technologies, the idea of the Internet of Things (IoT) started to take shape in the late 1990s. In 1999, while working for Procter & Gamble, Kevin Ashton first used the term "Internet of Things". Early advances in the Internet of Things concentrated on radio-frequency identification (RFID) as a means of improving supply chain management through the exchange of data between things. This groundbreaking study established the framework for the connected world we live in today, when commonplace items, from industrial machinery to home appliances, are outfitted with sensors and internet connectivity.

IoT technology advanced as a result of the growth of wireless networks, cloud computing, and sensor shrinking, which made it possible to include connections into a wide range of products and revolutionize numerous industries and everyday tasks.

1.3. The History of attacks on IoT

IoT was first centered around RFID technology, which allowed things to exchange data and improved supply chain management. As wireless and sensor technologies were combined in the early 2000s, the idea developed and led to the development of smart gadgets. An important turning point was reached in 2008–2009 when there were more connected devices than people on the planet. The introduction of IPv6, the development of cloud computing, and the shrinking of sensor technology all contributed to the Internet of Things' rapid growth by enabling the integration of connectivity into a broad variety of products from a variety of industries.

IoT has the potential to be a game-changer, but as it grows quickly, cybersecurity risks are also rising. The Mirai botnet, which launched one of the first and most prominent assaults in 2016, took advantage of IoT devices' default credentials to plan widespread distributed denial-of-service (DDoS) attacks that disrupted important internet services all around the world. The flaws in IoT security protocols were highlighted by this assault. Threats that came after, such as the Reaper botnet that was uncovered in 2017, showed a higher level of sophistication by spreading through known weaknesses in devices. IoT malware samples increased by more than 700% between 2016 and 2018, according to Kaspersky, underscoring the expanding threat landscape. In an effort to protect IoT ecosystems against constantly changing threats, these instances have sparked a great deal of research and development of security solutions, ranging from enhanced device authentication techniques to sophisticated intrusion detection systems.

1.4 Motivation of the Research

The seamless connectivity and automation made possible by the proliferation of Internet of Things (IoT) devices has changed a number of industries, including healthcare, transportation, and smart homes. But as IoT devices frequently have built-in weaknesses because of their constrained processing power and inconsistent security protocols, this quick integration has also increased cybersecurity worries. This research is driven by the urgent need to thoroughly investigate these vulnerabilities and create strong security solutions. IoT device exploitation by cybercriminals presents serious concerns, including data breaches, privacy violations, and even bodily injury, as these devices become an essential part of key infrastructure.
Also, the cybersecurity safeguards in place today frequently fall short of meeting the particular problems that the Internet of Things brings with it. Due to the heterogeneity and resource

limitations of these devices, traditional security measures are not always appropriate. Due to this, this research looks into certain threat vectors that target IoT ecosystems and suggests customized security frameworks in an effort to close the gap. This work aims to improve the resilience of IoT systems by thoroughly analyzing current threats and assessing cutting-edge security methods such lightweight encryption, anomaly detection, and decentralized security models. In order to ensure the safe and dependable operation of connected devices in a variety of applications, the ultimate goal is to contribute to the development of a more secure and trustworthy IoT ecosystem.

1.5 Problem Statement

An intricate and dynamic network environment with major security challenges has been brought about by the Internet of Things' (IoT) rapid proliferation. Because of these limitations and peculiarities, traditional intrusion detection systems (IDS) are frequently insufficient for Internet of Things (IoT) ecosystems.

- Resource Constraints of IoT Devices
- Heterogeneity of IoT Devices
- Scalability Challenges
- Real-Time Response Requirements
- Diverse Attack Vectors

Memory, computing power, and energy resources are frequently scarce in IoT devices. Resource-intensive intrusion detection systems (IDS) are created for more capable environments, such as corporate networks and data centers, and their deployment becomes difficult as a result.

IoT networks are made up of a diverse range of devices, each with unique functionality, operating systems, and communication protocols. A universally applicable IDS solution is made more difficult by this variability.

IDS solutions need to be able to manage massive amounts of data and possibly millions of devices with efficiency, since the number of IoT devices is predicted to increase rapidly. One major obstacle to the present IDS systems is this scalability issue.

Real-time or almost real-time answers are needed for many IoT applications, including smart homes, industrial automation, and healthcare. To avoid harm or interruption, IDS must promptly identify and eliminate threats.

A variety of threats, such as malware, spoofing, eavesdropping, Distributed Denial of Service (DDoS), and Denial of Service (DoS), can target Internet of Things networks. An efficient intrusion detection system needs to be able to recognize and eliminate many kinds of threats.

While keeping in mind these problems, this paper hopes to propose a Intrusion Detection System(IDS) that might be able to solve these. Hopefully this model will be more time

efficient, has a better scalability, maintaining the heterogeneity of IoT devices, detect a diverse range of attacks and lastly ensuring data privacy and integrity. This might not be the best solution but hope to build a model that could be the best by further modifications.

1.6 Research Objective

The goal of this research is to create effective security solutions that can thwart cyberattacks directed towards Internet of Things (IoT) devices. It includes a thorough investigation of IoT vulnerabilities, a look at prevalent and new cyberthreats, a review of creative security fixes, and recommendations for improving IoT security.

- Analysis of IoT Vulnerabilities
- Examination of Common and Emerging Cyberthreats
- Evaluation of Cutting-Edge Security Solutions
- Guidance for Enhancing IoT Security.

The study thoroughly examines the different vulnerabilities that exist in Internet of Things devices. These vulnerabilities frequently result from different operating environments, low computing power, and inconsistent security requirements. Determining and categorizing these vulnerabilities is essential for creating security solutions that work. Through this analysis, a thorough awareness of the threat landscape will be formed, facilitating the development of strong security protocols.

A crucial aspect of the study is examining the most common and recently discovered cyberthreats that target Internet of Things devices. This covers dangers such data interception, device manipulation, virus penetration, illegal access, and Distributed Denial of Service (DDoS) assaults. The research endeavors to illustrate the ways in which these risks materialize and spread inside Internet of Things networks through the examination of case studies and actual instances. A comprehensive threat model for IoT security will benefit from this clarification of the tactics, methods, and procedures (TTPs) employed by cybercriminals.

Ingenious and effective security solutions appropriate for Internet of Things contexts are the main focus of the research. Customary security techniques might not be practical given the

resource constraints of many IoT devices. In this study, low-weight security solutions including intrusion detection systems (IDS), effective encryption approaches, and anomaly detection strategies customized for Internet of Things devices will be investigated. It will also look at ways to improve IoT security using cutting-edge technology like blockchain and machine learning. For IoT makers and users, the research will assess the efficacy of these solutions using simulations and empirical analysis, and it will offer best practices and implementation techniques.

The project is to further the subject of cybersecurity for IoT devices by providing insightful analysis and helpful suggestions. To strengthen the entire security posture of IoT devices, it emphasizes the necessity for established security standards and protocols. In order to promote a proactive and coordinated approach to IoT security, the project aims to stimulate collaboration among academics, industry stakeholders, and policymakers. In the end, the objective is to make it possible for users to profit from IoT technology without jeopardizing their safety or privacy, opening the door to a more robust and safe connected society.

1.7 Research Scope

Internet of Things (IoT) device cybersecurity vulnerabilities are thoroughly analyzed in this research, and creative security solutions specifically designed to counter these threats are developed. Distributed Denial of Service (DDoS) assaults, malware, data breaches, and unauthorized access are just a few of the cyberthreats that Internet of Things (IoT) devices must contend with. This study attempts to properly classify these dangers. We urgently need strong security measures since, by 2025, there will be over 75 billion IoT devices connected, increasing from 35 billion in 2021, according to recent reports. IoT devices are great targets for cybercriminals because of their unique vulnerabilities, which will be explored in this research. These weaknesses include inadequate encryption standards, weak authentication systems, and outdated firmware upgrades.

Additionally, the study will assess the security frameworks and protocols already in use to safeguard IoT settings, evaluating their efficacy and pointing out vulnerabilities that an attacker might exploit. To comprehend the benefits and drawbacks of different security techniques, a

comprehensive analysis of recent literature, industry reports, and case studies will be required. For example, even while conventional encryption techniques work well, resource limitations may prevent them from being appropriate for low-power Internet of Things devices. Advanced security approaches that hold promise for improving IoT security will also be examined in the study, including blockchain technology, machine learning-based anomaly detection, and lightweight cryptographic algorithms. This report highlights the urgent need for flexible and robust security solutions in the Internet of Things space, since the worldwide cost of cybercrime is predicted to reach $10.5 trillion yearly by 2025.

Finally, to protect IoT devices from cyberattacks, this study will suggest a multi-layered security system that combines proactive and reactive methods. Resource constraints, heterogeneity, and scalability are some of the particular issues that IoT ecosystems face, and these issues will be addressed by the proposed architecture. Along with tactics for realtime threat detection and response, it will contain suggestions for best practices in the development, deployment, and management of IoT devices. The research will also examine the legislative and policy ramifications, promoting more robust cybersecurity regulations and cooperative endeavors between government agencies and industry participants. This research intends to improve the safety and dependability of IoT applications in a variety of industries, such as healthcare, smart cities, and industrial automation, by offering a comprehensive approach to IoT security and fostering the creation of more secure and robust IoT networks.

Further research on this topic is very much needed to ensure security of Internet Of Things (IoT) devices and privacy of the users/owners. It can be done by adding more algorithms to this proposed model or new models can be invented too.

CHAPTER 2

LITERATURE REVIEW

2.1. Introduction

Numerous industries have seen a profound transformation as a result of the increasing use of Internet of Things (IoT) devices, which have ushered in an era of unprecedented automation and connection. IoT devices are already a vital part of today's infrastructure; they can be found in transportation, healthcare, smart homes, and industrial control systems, among other things. The swift expansion of IoT devices has led to significant cybersecurity threats, rendering them appealing targets for cybercriminals. This literature review looks at the ways that cybersecurity risks to Internet of Things (IoT) devices are changing, their underlying causes, and the current security mechanisms that are in place to try and mitigate those risks.

Cyberattacks on Internet of Things (IoT) devices are as recent as the technology itself, but they are notable due to a number of high-profile instances that have brought attention to these devices' inherent vulnerabilities. An early and well-known instance was the Mirai botnet assault in 2016. In order to build a vast botnet and perform distributed denial-ofservice (DDoS) attacks against well-known websites and worldwide internet infrastructure, this breach took advantage of default credentials from Internet of Things devices. Strong security procedures in IoT networks are vital, as the Mirai attack showed.

Attacks thereafter have continued to reveal new vulnerabilities. For instance, researchers discovered the more advanced Reaper botnet in 2017 compared to Mirai. By exploiting well-known flaws in IoT devices, it spread itself. Threats of this nature have grown, which has prompted more research into understanding and mitigating the security risks posed by IoT devices. The danger landscape has sharply increased, as evidenced by the 700% growth in IoT malware samples between 2016 and 2018, according to the Kaspersky report.

A few major categories can be used to broadly classify the literature on IoT security threats: mitigation measures, device vulnerabilities, and network vulnerabilities. Firmware upgrades and hardcoded credentials are two examples of insufficient security procedures used

throughout the manufacturing process that frequently lead to device vulnerabilities. IoT communication channels can be manipulated and intercepted due to network vulnerabilities, which are often caused by inadequate authentication and encryption procedures.

To lessen these concerns, a number of security mechanisms have been created and put into place. These solutions include network-level defenses like intrusion detection systems (IDS) and blockchain-based frameworks, as well as hardware-based security modules and secure boot. Studies show that, for instance, integrating machine learning methods with intrusion detection systems (IDS) may greatly enhance the ability to identify unusual activity in Internet of Things networks. Furthermore, research has been done on the use of blockchain technology to offer decentralized security solutions that guarantee data integrity and authenticity in transactions using the Internet of Things.

The identification of cyberattacks on Internet of Things devices has traditionally depended on conventional cybersecurity techniques, like anomaly and signature-based detection. Specialized detection algorithms must be developed, nevertheless, due to the peculiarities of IoT contexts, which include heterogeneity, resource limitations, and large-scale application. A range of strategies, such as context-aware security policies and lightweight cryptography protocols, have been put forth by researchers to deal with these issues.

The literature has observed a noteworthy trend in the adoption of proactive security methods, which prioritize threat avoidance over detection. This covers the use of artificial intelligence (AI) to anticipate and neutralize possible threats before they have a chance to do damage, as well as the deployment of automated patch management systems. In order to detect new threats and vulnerabilities in real-time and help organizations take appropriate action, artificial intelligence (AI)-driven threat intelligence platforms, for example, can evaluate enormous volumes of data from numerous sources.

1.2 Literature Review

(Ansar Sharif and Noman Ali,2024) proposed a model to detect intrusions on IOT devices using Random Forest, Decision Tree, Support Vector Machine algorithms. Review shows that the findings of the paper is Effectiveness of different Machine Learning algorithms and Importance of feature Engineering. Despite the findings there are lackings too and that includes no code implementation, addressing encrypted traffic and ensuring model interpretability in the application. (Chaganti, R. et al, 2023), review of this paper shows thorough performance analysis of DL architectures and traditional ML classifiers and the model has an accuracy of 97.1%. The lackings that were found reviewing the paper are the LSTM model requires a large amount of labeled data, high memory resource consumption and ML and DL approaches are prone to various adversarial attacks. (Sugi, S. S. S., et al ,2020), this work indicates a model with LSTM and KNN algorithms. Intrusion Detection System (IDS) based on deep learning and machine learning to address security issues in IoT networks, comparing the performances of Long Short-Term Memory (LSTM) and K-Nearest Neighbor (KNN) algorithms is this paper's main findings. There's some limitations too and they are Sole focus on security issues and lack of discussion on generalizability. (Ge, M., Syed et al ,2020), this paper demonstrates high classification accuracy for both binary and multi-class classifiers, the methods that were used are Feed-forward neural network and embedding layers for mutli-class classifications. Constrained resources and computational capabilities of IoT networks
and the need for further validation on different IoT network setups and attack scenarios are the limitations of this work paper. (Thamilarasu, G. et al , 2019), represents a high precision and recall rates by using a method of Network virtualization and DL algorithm. This study solely focused on improving Precision and Recall rates. The proposed model works in three phases which are Network Connection Phase, AnomalyDetection Phase and Mitigation Phase. The problem with the model is that it may become resource-consuming and degrade system performance as the behavioral model increases in complexity with the growth of IoT network and another one is that with the increment of data volume the efficiency and accuracy of the IDS can decrease. (P. Jayalaxmi et al, 2022), this paper discusses the taxonomy of detection systems and provides a risk factor analyzer and a hybrid IDPS framework. LSTM and Feed-Forward Neural Network algorithms were used in the process. This paper combines both Machine Learning and Deep Learning to detect and prevent these attacks. The problem is that

this work failed to clearly identify the specific challenges. A novel intrusion detection architecture model were proposed in (M. Sheikhan, Hamid Bostani, 2016) which has a superior performance. This model uses a MapReduce approach, anomaly-based and misuse-based intrusion detection agents that use supervised and unsupervised optimum-path forest model for intrusion detection. Despite of the fact that the model has a superior performance it is very time consuming, not quite efficient. (Hamid Bostani, 2016),

This study proposes an offline misusedbased technique based on a modified supervised O ptimum-Path Forest (OPF) to detect the external (cyber) attacks of IoT. K-Means algorithm along with the social network analysis is employed to overcome the problem of scalability of the input large dataset and prune the training dataset with the aim of identifying the most informative samples. A novel intrusion detection system (IDS) that leverages network traffic profiling and machine learning techniques tailored for the IoT ecosystem was proposed in (Jihane Ben Slimane, et al,2024), By analyzing the behavioral patterns of network traffic, the proposed system can accurately identify malicious activities and potential threats in real-time, ensuring the integrity and confidentiality of IoT networks. The ML algorithm that were used are random Forest and Gradiant Boosting Machines(GBM). The limitations of this work are scalability in ultra-large-scale environment and dependency on high-quality, labeled datasets. (Alqahtani, H, et al, 2020), this paper represents a model including

Bayesian Network, Naive Bayes classifier, Decision Tree, Random Decision Forest, Random Tree, Decision Table, and Artificial Neural Network, to detect intrusions due to provide intelligent services in the domain of cyber-security. Although this model has a great level of accuracy it may not perform well on diverse categories of attacks. (Sarker, I. H, et al, 2020), presents a machine-learningbased security model called Intrusion Detection Tree ("IntruDTree"), which first takes into account the priority ranking of security aspects before constructing a tree-based generalized intrusion detection model based on the chosen key elements. The limitation of this work is the model is not so effective to large datasets with more dimensionality. (Ashok, G. et al , 2024), this study looked at 9 different IoT network assault variations using the UNSW-NB15 dataset and 6 of these 9 attacks are given higher importance during the classification process. The proposed method comprises feature selection, data preparation, and synthetic data generation based on the CTGAN methodology. The final dataset is used to train and assess the performance of different machine-learning designs, including Random forests, Extra trees, Decision trees, and XG Boost, after artificial records have been

generated. XG Boost outperforms them all with an accuracy rate of 96.71. A broad radius of reviews and debates on important topics pertaining to supervised and unsupervised machine

learning methods, emphasizing the benefits and drawbacks of each methodology were discussed in (Alsharif, M. H,et al, 2020). The problem is that study only focused on supervised and unsupervised ML algorithms and that too individually. A lightweight attack detection mechanism to identify a malicious actor trying to infiltrate malware or escalate privileges within an Internet of Things network is the use of logistic regression based on supervised machine learning, decision trees, random forests, logistic regression based on supervised machine learning, and Naive Bayes multinomial / Naive Bayes Gaussian were proposed in (Bhatia, T. K.,et al, 2023). But the highest accuracy of only 79.372% is the weakest point of this study. (Anthi, E. et al, 2021), in order to create AML attack samples, this study suggests a rule-based method. It then investigates how these attack samples can be used to target different supervised machine learning classifiers that are employed to identify Denial of Service attacks in Internet of Things smart home networks. The rudimentary method used to disturb the selected features is one of the primary drawbacks of this work. To protect and safeguard IoT network intrusions, a novel hybrid IoT-based intrusion detection system (IDS) utilizing the Binary Grey Wolf Optimizer (BGWO) and Naive Bayes (NB) is provided. NB is employed as a classification technique and BGWO as a feature selection tool in (Ismail Mohamed Nur, Erkan Úlker, 2020). But not being time efficient is the major drawback about this work. Using two machine learning techniques, feature selection and feature classification,(Fenanir, S., et al, 2019), this work creates a lightweight intrusion detection system (IDS). The filter-based approach realized the feature selection due to its relatively low computing cost. We were able to identify the feature classification approach for our system by contrasting logistic regression (LR), naïve Bayes (NB), decision tree (DT), random forest (RF), k-nearest neighbor (KNN), support vector machine (SVM), and multilayer perceptron (MLP). This work could be more efficient and could perform better with the use of more algorithm together. A unique ensemble Hybrid Intrusion Detection System (HIDS) is presented to secure IoT devices by integrating a One Class Support Vector Machine classifier and C5 classifier in (Khraisat, Gondal, et al, 2019). The benefits of both Anomaly-based Intrusion Detection System (AIDS) and Signature Intrusion Detection System (SIDS) are combined in HIDS. In order to address real-world problems,(Alsoufi, M. A., et al, 2021) this paper reviews anomalous intrusion detection using

deep learning approaches, with a focus on resource-constrained devices in the Internet of Things domain.

The work found that the performance of deep learning in anomaly detection is proven to be superior in regard to accuracy detection and false alarm rate. It also suggested that the

robustness of IDS should be ensured. Several machine-learning and deep-learning techniques are covered in this study,(Bambang Susilo, Riri Fitri Sari, 2020), along with standard datasets for enhancing IoT security performance. Using a deep-learning system, we created a method for identifying denial-of-service (DoS) assaults. Python programming was employed in this study, along with tools like scikit-learn, Tensorflow, and Seaborn. (Jinsi Jose, Deepa V. Jose,2021), highlights various security threats related to IoT, the importance of deep learning in IoT intrusion detection, and various IoT intrusion detection systems using deep learning. Comparison showed that CNN performed better and gave high accuracy in prediction based on various evaluation metrics. Not working with realtime dataset probably the only negative point about this study. (Sharipuddin, et al, 2021), recommended identifying intrusion detection systems in heterogeneous networks using deep learning to increase detection accuracy. The evaluation's findings demonstrated that deep learning can raise the accuracy of detection in the heterogeneous internet of things (IoT). In order to detect and categorize assaults and abnormalities in a simulated smart environment as either abnormal or normal, (Sualihah Jan, Faheem Masoodi, Alwi Bamhdi, 2022) this study introduces Deep Learning-based algorithms such as DNN and LSTM-RNN classifier, which make use of the BoT-IoT dataset. A state-of-art traffic classifiers based on deep learning was proposed in (Nascita, A., et al, 2022).It assess their effectiveness to accomplish IoT attack classification. In order to quickly discover anomalies, they use efficient and objective input data, and their performance is compared to that of conventional machine learning classifiers. Based on input data specifically customized and optimized for IoT attack classification, the experimental results underscore the necessity of using advanced deep learning systems.

2.3 Summary

As Internet Of Things is a fast growing industry and it will just keep going up, there will be attacks too to gain control of these devices or use these devices as an entry point of bigger attacks on other things. To ensure the safeguard of these Internet of Things(IoT) devices, an

Intrusion Detection System(IDS) model with high accuracy and efficiency is very much needed. This study proposes a system to do that. In this model, Machine Learning algorithms like Random Forest, Decision Tree, Adaboost, XGboost, K-Means Nearest Neighbor and Linear Regression were implemented. This study hopes that a model with these algorithms can thoroughly analyze and detect various attacks on Internet of Things(IoT) devices.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Using a wide range of machine learning methods, this research looks into cyber security risks that affect Internet of Things (IoT) devices and suggests strong security solutions. In order to provide a comprehensive study of IoT security vulnerabilities and the efficacy of suggested solutions, our methodology combines both supervised and unsupervised learning techniques. We use strong algorithms like Random Forest and Decision Trees to handle high-dimensional, complex data, and their interpretability gives us a clear picture of threat trends. Our security models' predictive power is increased by the use of AdaBoost and XGBoost, two algorithms well-known for their exceptional performance in improving model accuracy and managing unbalanced datasets. Additionally, by classifying data into discrete clusters, K-Means clustering is used to detect unusual behavior in IoT networks. This helps find outliers that may be signs of possible security breaches. A core statistical analysis that supports more complex machine learning models is provided by linear regression, which is utilized to comprehend and forecast the correlations between different characteristics and security threat occurrences. In-depth and varied IoT-related data, including different kinds of cyberthreats and their characteristics, make up the dataset used for this investigation. Data quality and relevance are ensured by carefully carrying out preprocessing procedures including feature selection, normalization, and data cleaning. In order to provide a thorough comparison of algorithms' efficacy, the performance of each algorithm is assessed using common metrics such as accuracy, precision, recall, and F1score. Through the integration of these algorithms, our research seeks to elevate the resilience of IoT ecosystems against growing cyber threats by proposing practical security solutions in addition to identifying popular cyber risks. In addition to offering a comprehensive framework for examining IoT security, this methodological approach offers insightful information about how machine learning should be used to cyber security applications.

3.2 Methodology Diagram

An Intrusion Detection System (IDS) development methodology diagram plots the sequential steps and structure that went into creating the system. Key phases like data gathering, preprocessing, threat detection methods, reaction plans, and assessment metrics are highlighted. Stakeholders involved in the creation of the IDS can better understand and communicate with each other thanks to this diagram, which also helps to clarify the process and guarantees a systematic approach.

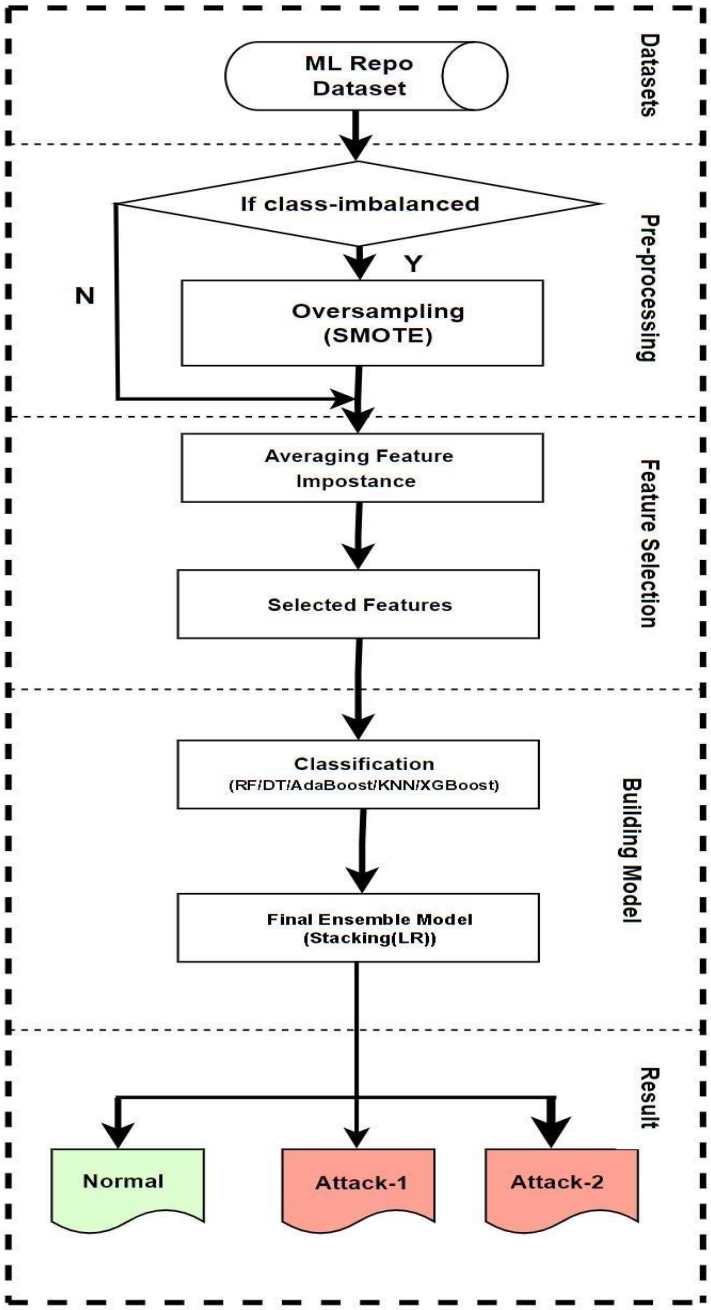


Figure 3.2: System Diagram

SMOTE pre-processing is used to balance classes in the methodology for analyzing IoT cyber security concerns, which entails obtaining heterogeneous information. Random Forest, Decision Tree, AdaBoost, KNN, and XGBoost are some of the classification techniques used; feature selection identifies important attributes. With Linear Regression acting as the metaclassifier, the outputs of these models are integrated via stacking. By separating out certain attacks (Attack-1, Attack-2) from typical behavior, the final model classifies the condition of

IoT devices. By precisely recognising and identifying threats, this multi-phase, resilient strategy improves IoT security.

3.3 Methods

The methodology diagram shows the steps involved in creating an intrusion detection system (IDS), beginning with the use of a Kaggle dataset. Oversampling (SMOTE) is used in the pre-processing step to address class imbalance. In order to find pertinent features, feature selection entails averaging feature importance. Using classification techniques like Decision Trees (DT), Support Vector Machines (SVM), Random Forest (RF), Neural Networks (NN), and XGBoost, the model building step creates an ensemble model that is finally created via stacking. The ensuing intrusion detection system (IDS) distinguishes between regular operations and multiple assaults (Attack-1, Attack-2).

3.3.1. Data Acquisition

Obtaining a solid dataset from a reliable Machine Learning Repository (ML Repo) is the first stage in this process. These databases function as enormous warehouses, filled to the brim with a multiplicity of datasets covering a diverse range of cyberthreats and benign behaviors unique to Internet of Things (IoT) devices. One cannot stress the statistical relevance of this diversity of data. We can expose our models to a finely textured tapestry of possible situations by adding large amounts of data covering a wide range of scenarios. Because of this extensive training program, the models are more resilient and effective, which strengthens their capacity to generalize well to real-world deployments where anomalous events may appear in unexpected ways. All in all, the foundation for building resilient and flexible anomaly detection models is provided by the use of a statistically sound and varied dataset that was extracted from a trustworthy machine learning repository.

3.3.2. Data Preprocessing

The methodology's main component, data pre-processing, carefully prepares the data for its crucial role in model training. Ensuring the quality and applicability of the data fed into the model is the main concern here.

Consider a dataset that details how users interact with an Internet of Things gadget. The data should ideally be an exact representation of reality, where the frequency of typical user action greatly exceeds that of cyberattacks. But this balance is frequently off in the field of data collection. Because they are so much more common, normal activities wind up making up the majority class, whereas cyberattacks—the very things we are trying to detect—form the minority class. The effectiveness of our paradigm is seriously threatened by this imbalance. Algorithms for recognizing patterns are inherent to machine learning models. If the dataset is highly skewed, they will usually give more weight to the majority class and become skilled at spotting typical behavior, but they can miss the important minority class, which is the cyberattacks we are trying to uncover. The failure of the model to sound the alarm for real threats may result in a false sense of security.

We utilize the clever approach known as SMOTE (Synthetic Minority Over-sampling Technique) to counteract class imbalance and guarantee our model stays watchful. The underrepresented minority class is served by SMOTE through the deliberate creation of synthetic data points.

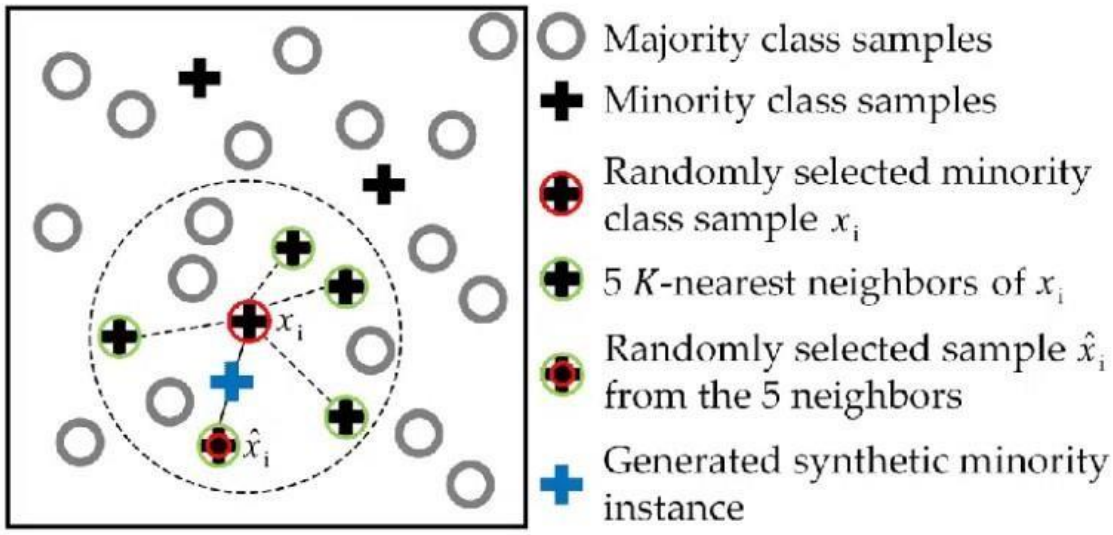


Figure 3.3.2: Synthetic Minority Over-Sampling Technique

Assigning the lowest number of data points to the class—in our case, cyberattacks—is the first step. For every data point in the minority class, SMOTE then finds the k-nearest neighbors within the same class. Similar cyberattack examples can be found in these neighbors.

Consider displaying these data points as points in a high-dimensional space, along with the closest neighbors. Next, using a predetermined probability distribution (typically weighted by distance), SMOTE chooses one neighbor at random. The basis for generating a fresh, synthetic data point is this chosen neighbor.

A selected vector is drawn from the selected neighbor of the minority class data point. Along the path that connects the two existing points in feature space, a random value is generated, hence generating a new data point along this line segment. The data point for the minority class is represented by this recently created point, which seems reasonable.

SMOTE creates a desired number of new data points by iterating through the minority class data points and repeating these procedures. The dataset is balanced and the minority class's size is effectively increased by this approach.

3.3.3 Feature Selection

The crucial stage of feature selection begins after we have carefully balanced the data using SMOTE. In order to pinpoint the few essential data qualities for efficient anomaly identification, we here sort through the enormous array of available attributes. Ultimately, our model will be guided by these selected features, which are its lifeblood and will let it discriminate between harmful cyberattacks and typical user behavior.

Visualize an enormous repository of data, where every item of information could hold a hint. Feature selection is like a discriminating treasure hunter; it carefully considers every characteristic to ascertain its importance in distinguishing between regular and unusual activity. We employ a number of strategies to help us in this endeavor, one of which is the average feature importance score computation.

Consider the significance of a feature as a gauge for impact. This method allows us to provide a score to each feature, which represents how much it adds to the model's capacity to distinguish between benign and malevolent intent. The standout features, the most significant indicators of abnormalities in the data, are those with high significance ratings.

Some features are more valuable than others, though. Certain ones could be repetitive, providing few details above what has previously been disclosed by others. To confuse the model instead of informing it, others could add noise. Twofold benefits are obtained by focusing on the elements that have clearly high relevance scores.

The model training procedure is first streamlined. Overfeatured models become cumbersome and ineffective because they have too many characteristics, many of which are unnecessary or even deceptive. The model can learn more quickly and efficiently if it concentrates on the few features that are actually important.

More significantly, and secondly, this focused strategy improves the model's anomaly detection precision. We give the model more ability to distinguish between benign and malevolent behavior by removing unnecessary elements and emphasizing the most important ones. This equates to a more resilient and watchful anomaly detection system, less prone to false alerts and more capable of locating the real risks concealed in the data. Feature selection, then, serves as a kind of filter, sorting through the data and highlighting the elements that are most important for our model to perform its vital function.

3.3.4 Model Building

We go on to the most important part of our process, which is creating the machine learning models, after we have the chosen features. Implementing several categorization algorithms is necessary for this, each of which has advantages of its own. We make use of-

3.3.4.1 Random Forest

Envision a huge and varied wilderness populated with various trees, each with its own distinct survival decision-making mechanism. This very idea serves as the basis for Random Forest, a potent machine learning method. It is a collective intelligence made up of an ensemble of several decision trees rather than a single, monolithic decision tree. A random feature selection process and a subset of the data are used to construct each tree, resulting in a myriad of unique viewpoints on the data.

Building The Forest

The process starts with several decision trees being created. Using a technique known as bootstrapping, which involves randomly selecting data points from the original dataset with replacement, each tree is built. This implies that some data points might show up more than once in a single tree, while other data points might not show up at all. By doing so, randomization is introduced and the tendency for any one tree to become unduly dependent on particular data patterns is avoided.

Random Feature Selection

An arbitrary subset of features is selected from the entire pool at each decision tree node to increase diversity even more. Next, using the best split among these selected attributes, the tree divides the data. The forest's decision-making processes are richly varied due to the unpredictability that keeps any two trees from becoming overly alike.

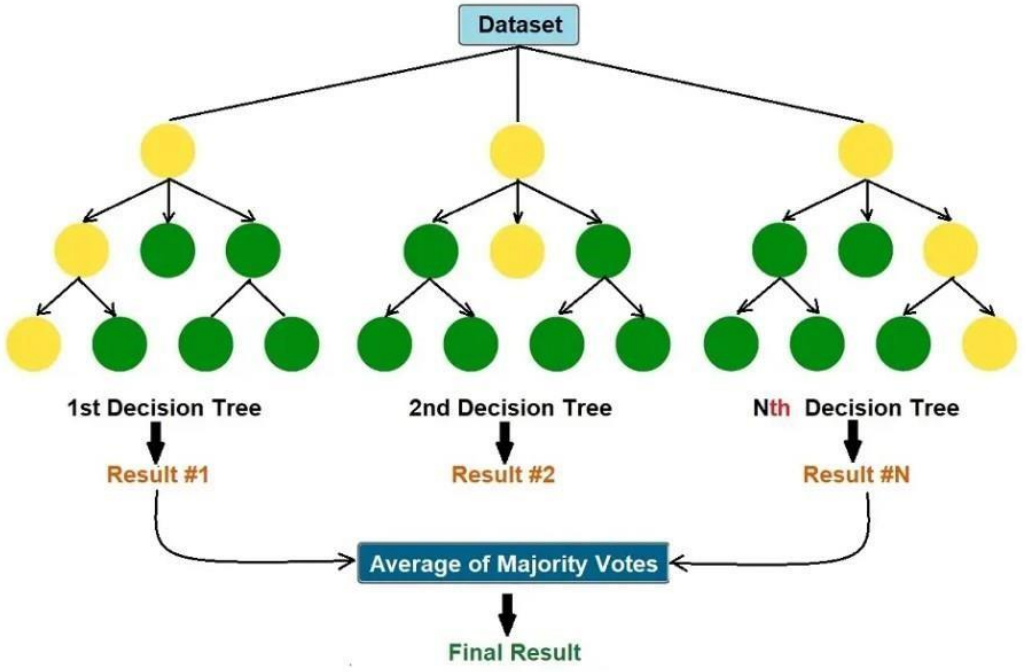


Figure 3.3.4.1: Random Forest Classifier

Voting for the Truth

The forest as a whole is supplied with a new data point requiring classification once every tree has reached full height. Every tree assesses the data point on its own and makes its own way through the splits it has made. A majority vote decides the final categorization. The anticipated class for the new data point is the one that received the most votes from each individual tree.

Several benefits come with this democratic voting system:

- **Reduced Variance:** When a decision tree is overfitted, it performs badly on unseen data and becomes excessively sensitive to the particular training set. This variance is decreased by Random Forest, which creates a more reliable and broadly applicable model by averaging the predictions of several trees.
- **Handling Missing Values:** Missing values in the data are naturally handled by Random Forests. The decision-making tree can choose to disregard a data point that contains a missing value for a feature that is being considered at a split and instead rely on the features that remain.
- **Importance Scores:** Random Forest offers information on the significance of features. We can determine which features have the greatest influence on classification by keeping track of how frequently each feature is split across all trees. Understanding the data and maybe improving the feature selection procedure can both benefit from this.

Essentially, Random Forest makes use of crowd wisdom. Through the integration of various decision trees, each with a distinct viewpoint, the shortcomings of single trees are surmounted, yielding a potent anomaly detection instrument that excels in detecting even the most minute departures from typical patterns in the data.

3.3.4.2 Decision Tree

A decision tree is a diagram that resembles a flowchart and represents the decision-making process. It starts at one place and branches out, each branch signifying a potential decision depending on a certain circumstance. Usually, the characteristics of the data serve as the basis for these requirements. The leaf nodes at the end of the tree signify the conclusion or forecast.

Making predictions based on existing data and illustrating difficult judgments are two uses for decision trees.

Building the Tree

All of the dataset is used to start the trip. By determining which feature best divides the data into classes—normal and anomalous in our example—the algorithm determines which is the most informative. The amount of unsuccessful login attempts for anomaly detection is one example of a feature that could be pertinent to the issue.

Branching Out

There are two or more branches created from the data, depending on the selected feature. For example, whether there have been fewer than three unsuccessful login attempts or more than three, each branch indicates a potential decision result. In order to further partition the data according to the optimal separation criterion, this method iteratively proceeds at each branch, employing a distinct feature.

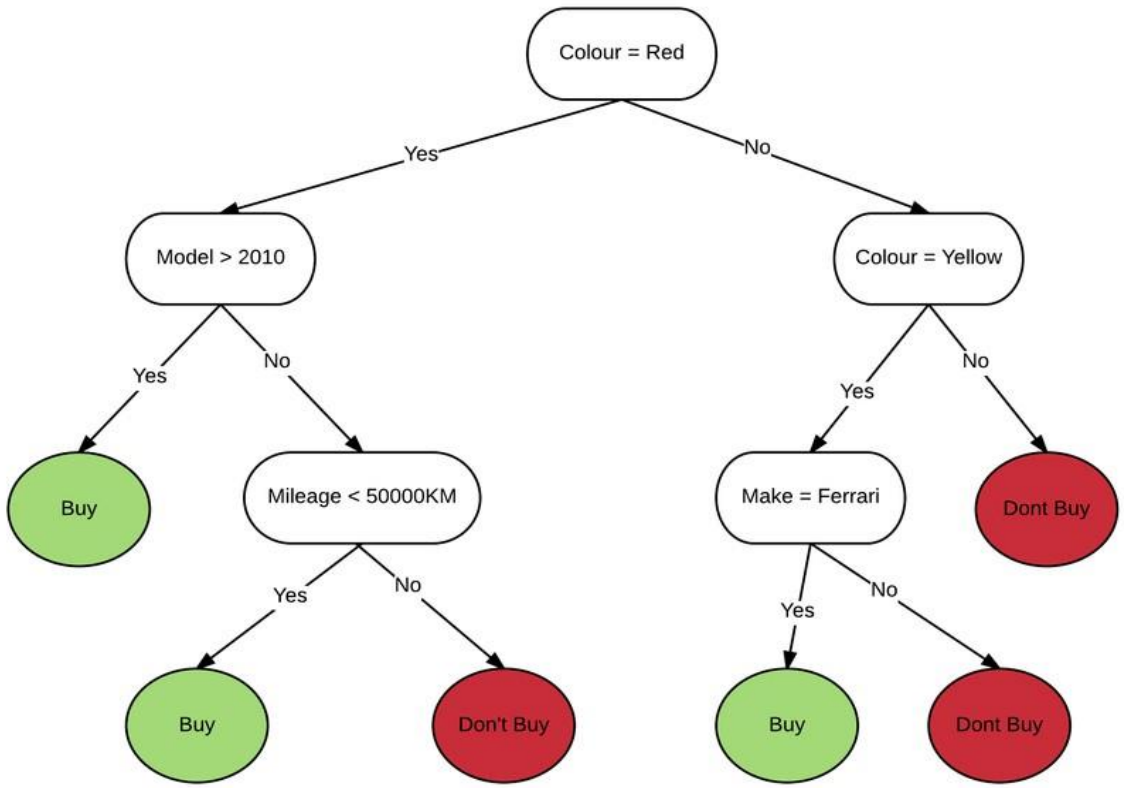


Figure 3.3.4.2: Decision Tree

Reaching the Leaves

Splitting keeps going till a certain point is reached. Achieving a high degree of purity, where the majority of the data points on a branch belong to the same class, or reaching a specific depth in the tree might all be examples of these requirements. Another would be having enough data points in each branch to prevent overfitting. These terminal branches, often referred to as leaves, show how the data is finally categorized.

Making Predictions

When a new data point arrives, it traverses the tree, starting at the root node. At each node, the value of the corresponding feature in the data point is compared to the splitting criteria. The data point is then directed down the appropriate branch based on this comparison. This process continues until the data point reaches a leaf node, inheriting the class label assigned to that leaf.

3.3.4.3 Adaboost

Another interesting technique in the field of anomaly detection is called AdaBoost, which stands for Adaptive Boosting. It uses a sequential learning approach, iteratively improving a group of ineffective learners to eventually build a strong ensemble classifier.

Initial Pass

Every data point in the training dataset is first given the same weight using AdaBoost. Basically, at first, every data point is seen as equally significant. Next, it trains a weak learner, which can be any elementary classification method, such as a single-level decision tree or a decision tree stump.

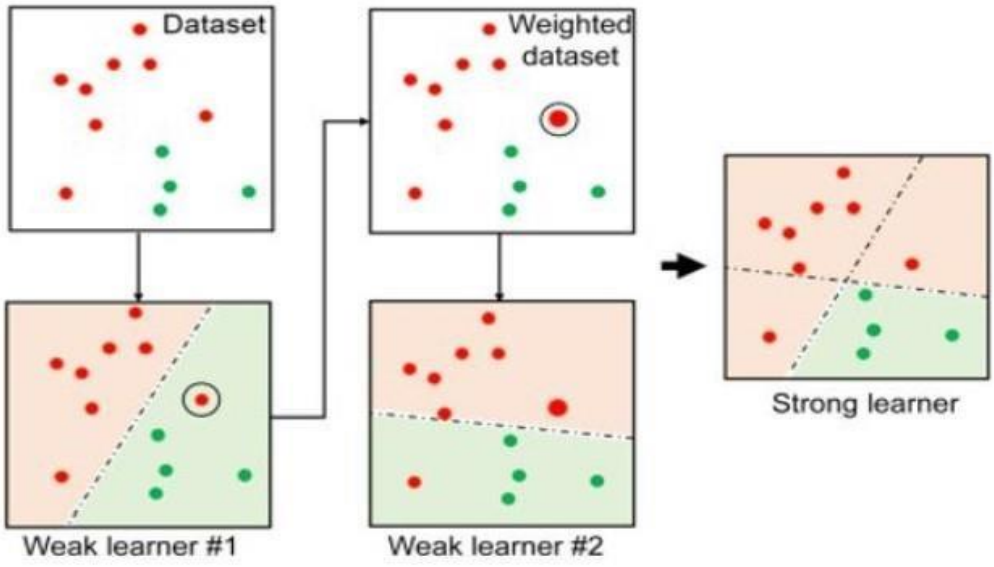


Figure 3.3.4.3: Adaboost

Evaluating the Learner

Using the training set, AdaBoost assesses this weak learner's performance. The weights of the data points that the learner incorrectly classified have been enhanced. This compels the students who come after to concentrate more on these difficult cases. Essentially, AdaBoost modifies the distribution of training data, giving greater weight to the data points that the learner at hand found challenging.

Boosting the Weak

Using this modified data distribution, a new weak learner is then trained, with the weights affecting the learner's attention to each data point. In a perfect world, this new student outperforms the previously incorrectly categorized cases.

The Ensemble Emerges

For numerous iterations, this training of a weak learner, performance assessment, and data weight adjustment process is repeated. By gradually developing a more robust ensemble

classifier, the learners improve their ability to manage the intricate data landscape with every iteration.

Weighted Vote

All the weak learners in the ensemble vote when a new data point is brought in for classification; the weight of each learner's vote is based on how well it performed in training. To get the final categorization for the new data point, AdaBoost then aggregates these weighted votes.

3.3.4.4 K-Nearest Neighbors (KNN)

Operating on the intuitive principle of similarity, K-Nearest Neighbors (KNN) is a machine learning method. Picture yourself at a party, not knowing if someone you've just met is a friend or a stranger. In order to handle this scenario, KNN would first look at the people you already know (your training data) and determine the k closest persons based on specific traits. If the bulk of your close neighbors are friends, then you probably consider the newcomer to be a friend as well.

Distance Metrics

Selecting a distance metric, or mathematical formula that determines the degree of similarity between two data points in feature space, is the first stage. Straight-line distance (Euclidean distance) and sum of absolute differences (Manhattan distance) are two common metrics.

The K Factor

The number k, or the nearest neighbors to take into account while forming a forecast, is an important KNN parameter. It's crucial to select the ideal k since a very high k could cause the model to miss significant differences, while a very low k could make it too sensitive to noise in the data.

Classification by Similarity

Inside the training data, KNN finds its k nearest neighbors when a new data point comes (seeking classification as normal or anomalous). The predicted class for the new data point is determined by taking the most frequent class label (normal or anomalous) among these k neighbors.

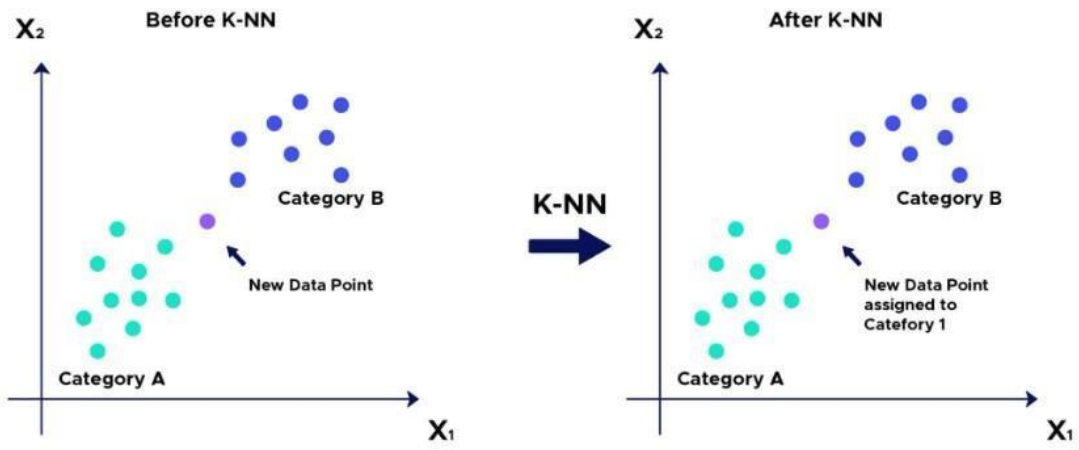


Figure 3.3.4.4 K-Nearest Neighbors (KNN)

The simplicity and interpretability of KNN are excellent. New data points are classified according to the company they keep in feature space; there is no intricate model to train, and each prediction's logic is clear. KNN does have certain drawbacks, though. Because it takes time to calculate the distances to every data point, it may be computationally expensive for huge datasets. To obtain the best results, rigorous testing is necessary as the performance of KNN also depends on the value of k and the distance measure that is selected.

3.3.4.5 Extreme Gradient Boosting (XGBoost)

Extreme Gradient Boosting, or XGBoost, is yet another strong competitor in anomaly identification. It uses a sequential learning approach to gradually create a strong model. Envision a relay competition in which every competitor capitalizes on the advantages and disadvantages of their predecessor. XGBoost operates in a comparable way.

The Initial Model

A basic model, frequently a decision tree, is used to start the journey. Predictions about the data are made by this initial model, but errors are inevitable.

Learning from Errors

In order to determine what went wrong with the original model, XGBoost carefully examines these flaws. Then, with the express purpose of fixing those mistakes, it builds a new model. The goal of the second model is to learn from the mistakes of the previous one, and it can be another decision tree.

The Boosting Process

There are repetitive repetitions of this cycle of creation, assessment, and correction. New models are introduced to the ensemble with every iteration, with an emphasis on the flaws of the prior models. When it goes on, the group gets stronger and stronger, like a group of runners getting better at handoffs and coming together as a cohesive one.

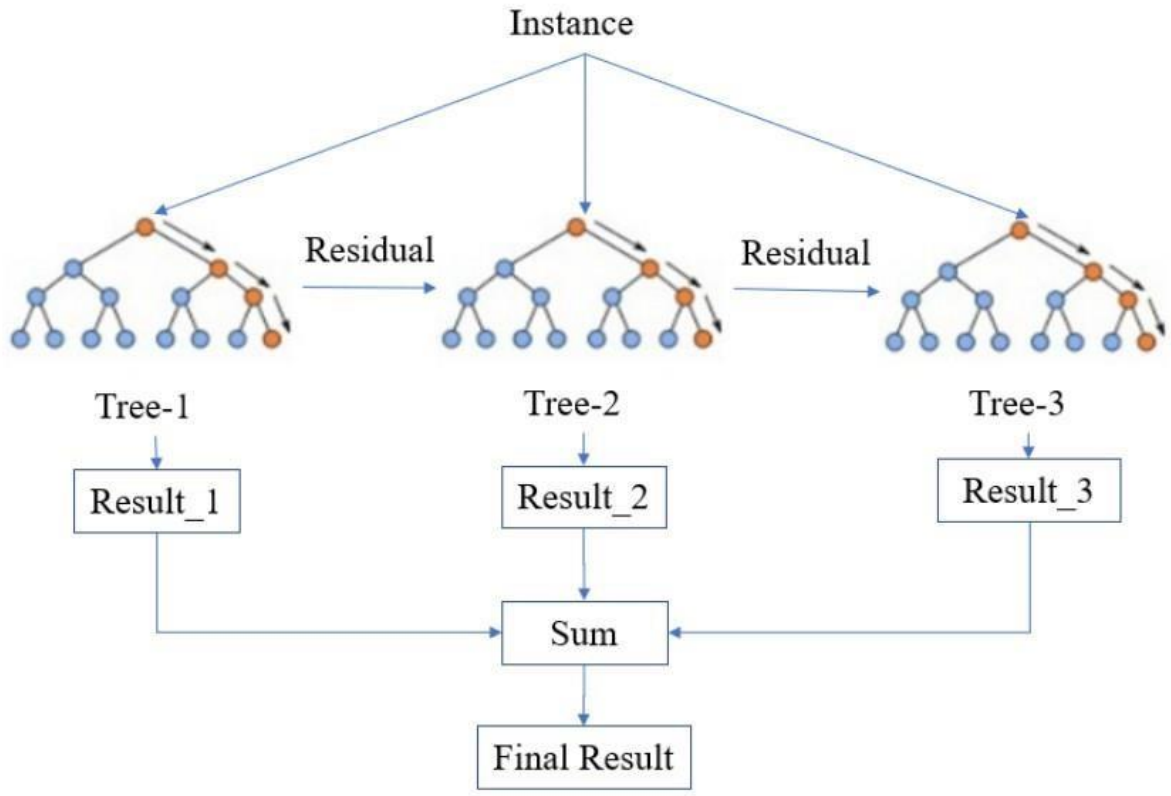


Figure 3.3.4.5: Extreme Gradient Boosting (XGBoost)

Regularization to Prevent Overfitting

To avoid overfitting, XGBoost uses a smart method called regularization. Regularization penalizes too complicated models, therefore preventing the ensemble from become unduly dependent on particular data patterns that might not translate well to new situations. This sustains the model's capacity to appropriately identify anomalies and adjust to new data.

XGBoost makes use of gradient boosting to produce an accurate and reliable ensemble model through this repeated refinement process. Like a well-trained relay team, this group can spot even the smallest irregularities in the data, which makes it an invaluable tool for protecting our systems from cyberattacks and guaranteeing their integrity.

3.3.4.6 Stacking Ensemble (Linear Regression)

To enhance prediction performance, stacking is an ensemble learning strategy that mixes several categorization models. Two levels of models are used in the method: base models and a meta-model.

Base Models

Many classification algorithms make up the first level. Utilizing the same dataset, these basic models are individually trained. These classifiers are employed in our context:

- Random Forest
- Decision Tree
- AdaBoost
- XGBoost
- KNN

Independent predictions are produced by each base model. The meta-model, the next level, uses these predictions as input features.

Meta-Model

A meta-model trained on the underlying models' predictions is used in the second level. A classifier using logistic regression (LR) serves as the meta-model in this instance. Using the outputs (class labels or projected probabilities) from the base models as input features, the meta-model learns how to combine them in the most effective way to get the final prediction.

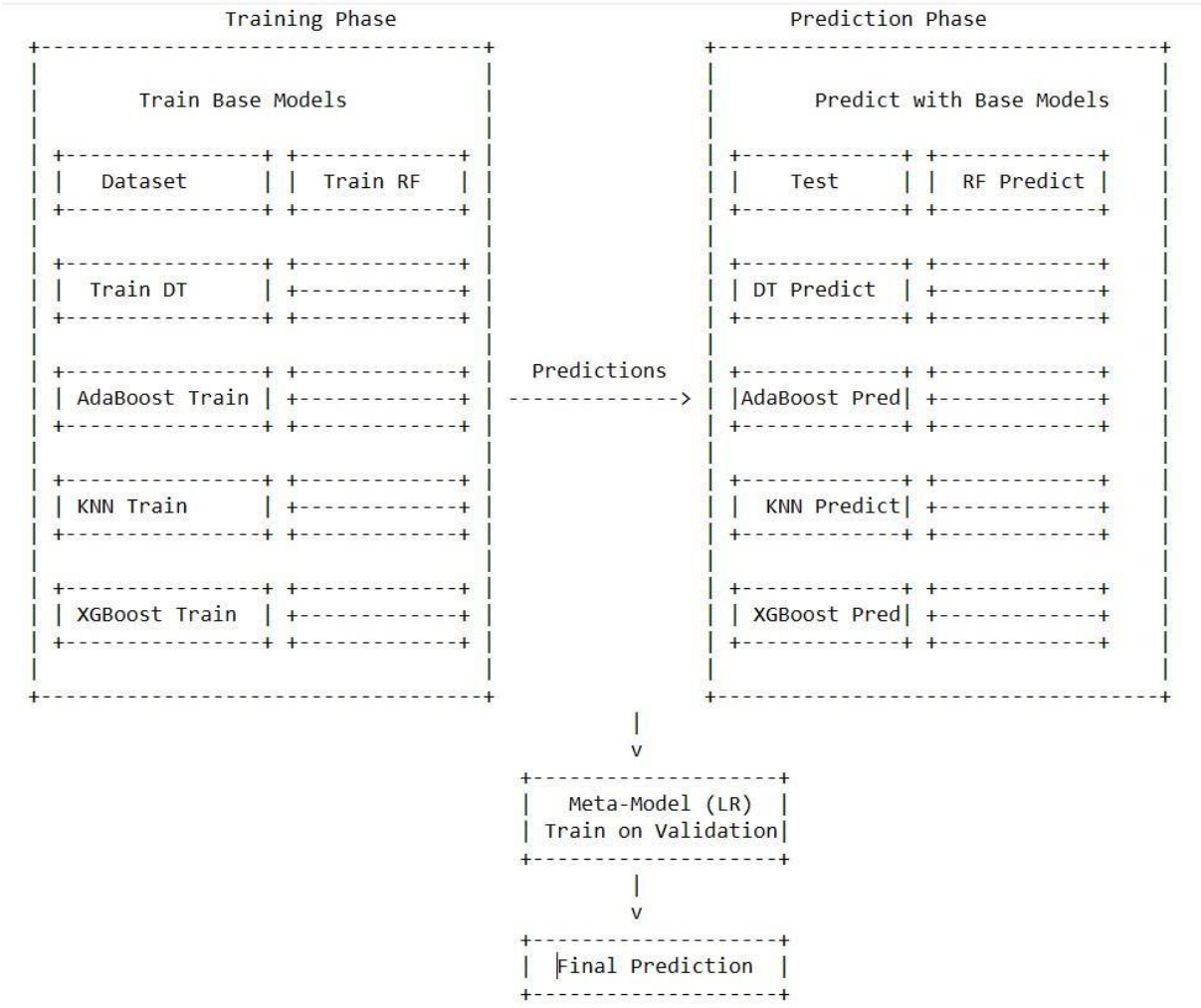


Figure 3.3.4.6.1: Stacking Ensemble (Linear Regression)

Detailed Process

The training dataset is used to train each base model, including RF, DT, AdaBoost, KNN, and XGBoost. These models use the validation dataset to produce predictions after training. For the validation set, the predictions made by every base model are gathered. A new dataset including 1000 samples and 5 features (each feature being the prediction from one base model) would be created, for instance, if there are five base models and 1000 validation examples. The meta-model (Logistic Regression) is trained using the fresh dataset (validation set predictions). To increase overall prediction accuracy, the meta-model discovers the best strategy to integrate the predictions from the base models.

The base models forecast the test dataset during the testing stage. The meta-model receives these predictions after which it generates the final predictions.

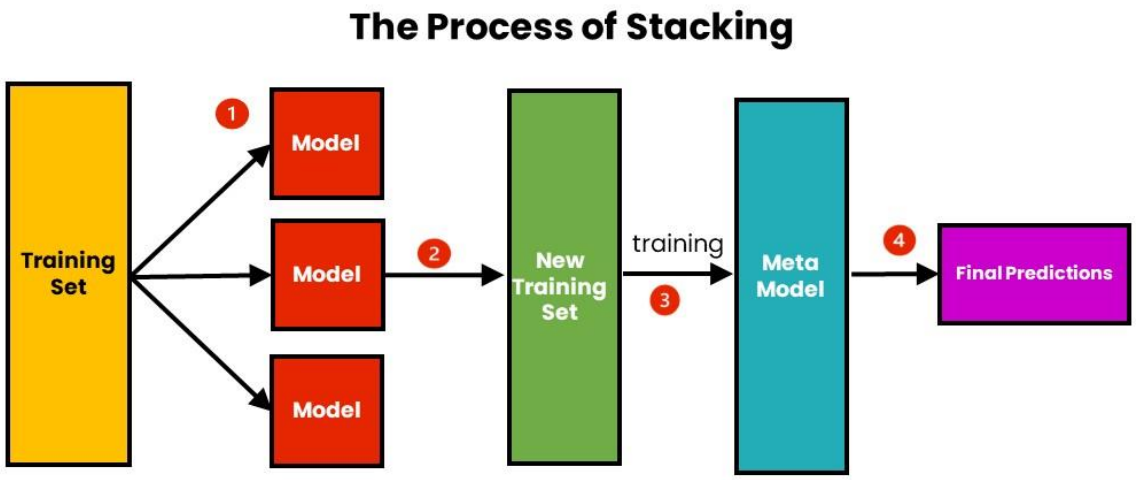


Figure 3.3.4.6.2: Stacking Ensemble

Training and validation sets of the dataset are separated.On the training set, each base model—Random Forest, Decision Trees, AdaBoost, KNN, and XGBoost—is trained.Next, on the validation set, each trained base model predicts.
The meta-model's input features come from the basic models' predictions on the validation set. Using this fresh set of forecasts, the meta-model (Logistic Regression) is trained.

In the testing stage, predictions are made on the test dataset using the trained base models (K-Nearest Neighbors, AdaBoost, Random Forest, Decision Trees, and XGBoost). After being trained on the first training set of data, every base model separately analyzes the test set and generates a set of predictions. The test dataset is effectively transformed into a new dataset where each feature corresponds to a prediction from one of the basic models thanks to these predictions from the base models acting as a new feature set.
These base model predictions are then entered as input characteristics into the meta-model, which in this instance is a Logistic Regression model. During the validation process, it was trained to comprehend the optimal way to integrate these inputs to provide a final prediction that is accurate. The meta-model produces the final forecast by examining the patterns and relationships between the predictions made by the base models. The test data is divided into

three categories based on this final prediction: Normal, Attack-1, and Attack-2. By utilizing the strengths of several models and integrating them in a way that lessens the shortcomings of individual models, the stacking process hence improves prediction accuracy.

3.4 Data Source

As it is not easy to gather these kind of IoT intrusion dataset from scratch, I took assistance of the internet to find a compatible dataset to try and implement the system. After a broad round of searching I finally found a dataset that was exactly what I was looking for. The 'IoT_intrusion.csv' dataset that I used was collected from 'Machine Learning Repository'.

3.5 Summary

Starting with a dataset from an ML repository, the diagram shows a machine learning process for attack detection. Preprocessing is performed on the data to check for class imbalance and, if needed, to oversample it using SMOTE. Subsequently, significant characteristics are chosen by calculating their mean importance ratings. With Logistic Regression serving as the meta-model, a stacking technique is used to aggregate the predictions of several classifiers (Random Forest, Decision Trees, AdaBoost, KNN, and XGBoost) that have been trained on these features. By using a strong combination of model predictions, the final ensemble model effectively detects attacks by classifying data into Normal, Attack-1, or Attack-2.

CHAPTER 4

RESULT & ANALYSIS

4.1 Introduction

In this study, a variety of machine learning algorithms, including Random Forest (RF), Decision Tree (DT), XGBoost, Adaboost, K-Nearest Neighbors (KNN), and Logistic Regression (LR), were evaluated for performing network intrusion detection tasks. Their relative accuracy scores of 0.99 show how effective RF, DT, and XGBoost are at detecting network intrusions. KNN fared better with an accuracy of 0.94 than LR and Adaboost, which displayed lower accuracy of 0.79 and 0.71, respectively. We used a stacking ensemble approach; these classifiers were integrated as base learners, and Logistic Regression was used as the meta-learner, to improve classification performance. Whereas KNN marginally declined to 0.93 and Adaboost dipped to 0.65, RF, DT, and XGBoost all kept their excellent accuracy of 0.99 in this ensemble. A 0.98 improvement in accuracy was obtained with the meta-learner, Logistic Regression. Even though weaker learners like Adaboost perform inconsistently, these results demonstrate the resilience and promise of ensemble learning in enhancing detection accuracy, especially through the skillful fusion of powerful individual classifiers.

4.2 Result

The Scatter-plot of the dataset-

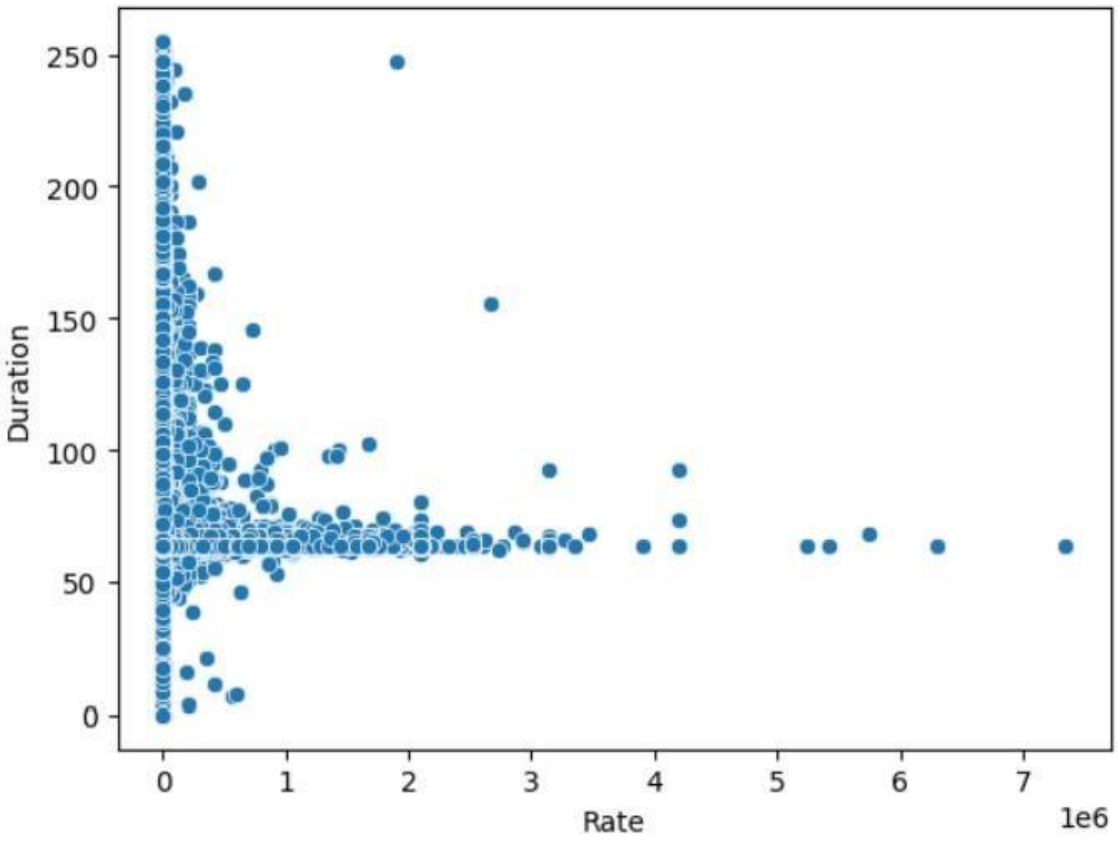


Figure 4.2.1: Scatter-plot

This scatterplot illustrates the relationship between "Duration" (y-axis) and "Rate" (x-axis), with "Rate" ranging from about 0 to 7,000,000 and "Duration" from 0 to 250. Low "Rate" values exhibit a high concentration of points, especially below 1,000,000, where "Duration" values vary dramatically from 0 to approximately 250. There are prominent outliers with "Duration" values that are unevenly distributed and vary, and "Rate" values that are higher than 1,000,000 and as high as approximately 7,000,000. The majority of data points show "Duration" values between 0 and 50 and "Rate" values between 0 and 1,000,000, indicating that most occurrences have shorter durations and lower rates.

With diminishing variability in "Duration" as "Rate" increases, particularly beyond 1,000,000, the spread of data points at higher rate values is less dense and exhibits fewer examples as the

rate increases. The dataset's diversity and spread are shown by this plot, which clearly shows a trend of clustering at lower rate values with many outliers spreading into higher rate ranges.

4.3 Result(Individual)

Several machine learning techniques were assessed for performance and accuracy in the individual result analysis. Strong prediction skills are demonstrated by the high accuracy of 0.99 that both the Random Forest (RF) and Decision Tree (DT) algorithms attained. Likewise, the Extreme Gradient Boosting (XGBoost) model demonstrated remarkable performance, with an accuracy of 0.99. The accuracy of the K-Nearest Neighbor (KNN) method was 0.94, indicating strong performance. On the other hand, the accuracy of the Adaptive Boosting (AdaBoost) method was comparatively lower at 0.71, indicating possible limitations in its prediction ability for this dataset. The accuracy of 0.79 obtained by Linear Regression (LR) indicates that it is a moderately effective model in comparison to the other models.

4.3.1 Random Forest

A number of metrics, such as precision, recall, F1-score, and support, were used to assess the network intrusion detection system's (NIDS) performance across a range of network traffic and attack types. The following table displays each category's detailed results:

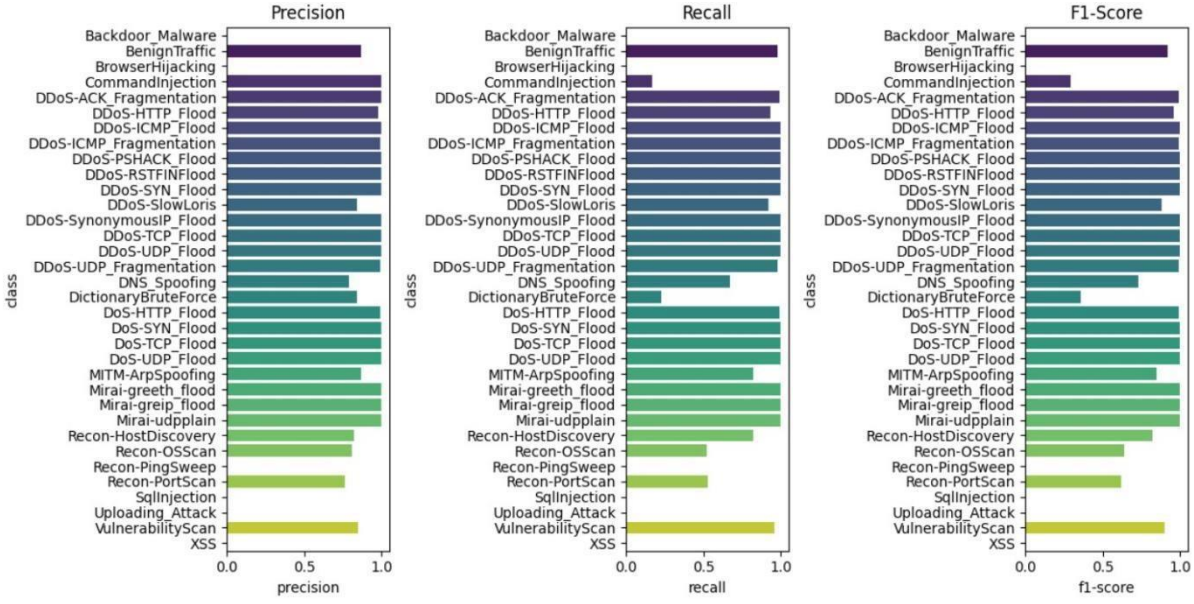


Figure 4.3.1 Random Forest Result

Table 4.3.1: Random Forest Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
Backdoor_Malware	0.44	0.68	0.53	25
BenignTraffic	0.92	0.91	0.92	7325
BrowserHijacking	0.45	0.47	0.46	49
CommandInjection	0.57	0.56	0.56	36
DDoS-ACK_Fragmentation	1.00	1.00	1.00	1934
DDoS-HTTP_Flood	0.99	0.99	0.99	182
DDoS-ICMP_Flood	1.00	1.00	1.00	41266
DDoS-ICMP_Fragmentation	1.00	1.00	1.00	3029
DDoS-PSHACK_Flood	1.00	1.00	1.00	25170
DDoS-RSTFINFlood	1.00	1.00	1.00	23457
DDoS-SYN_Flood	1.00	1.00	1.00	25928
DDoS-SlowLoris	0.99	0.99	0.98	144
DDoS-SynonymousIP_Flood	1.00	1.00	1.00	23791
DDoS-TCP_Flood	1.00	1.00	1.00	27619
DDoS-UDP_Flood	1.00	1.00	1.00	36195
DDoS-UDP_Fragmentation	1.00	1.00	1.00	1955
DNS_Spoofing	0.70	0.69	0.69	1213
DictionaryBruteForce	0.53	0.61	0.57	92
DoS-HTTP_Flood	0.99	1.00	0.99	481
DoS-SYN_Flood	1.00	1.00	1.00	13493
DoS-TCP_Flood	1.00	1.00	1.00	17173
DoS-UDP_Flood	1.00	1.00	1.00	22140
MITM-ArpSpoofing	0.81	0.82	0.82	2111

The NIDS's total accuracy in all areas is 0.99. The precision, recall, and F1-score are 0.85, 0.83, and 0.84, respectively, after macroaveraging over all classes. The system achieves 0.99 precision, 0.99 recall, and 0.99 F1-score in terms of weighted averages, which account for the different support for each class.

Result Analysis

Given that the NIDS achieved flawless ratings in the majority of categories, these results show how successful it is at identifying different kinds of network traffic and attacks, including Distributed Denial of Service (DDoS) attacks. For other attack types, like browser hijacking, command injection, and backdoor malware, the performance is noticeably worse, pointing to areas that could use further development.

4.3.2 Decision Tree

Several metrics were used to assess the effectiveness of the network intrusion detection system (NIDS) in relation to different forms of network traffic and attacks. These metrics included precision, recall, F1-score, and support. The following table displays the specific outcomes for every category:

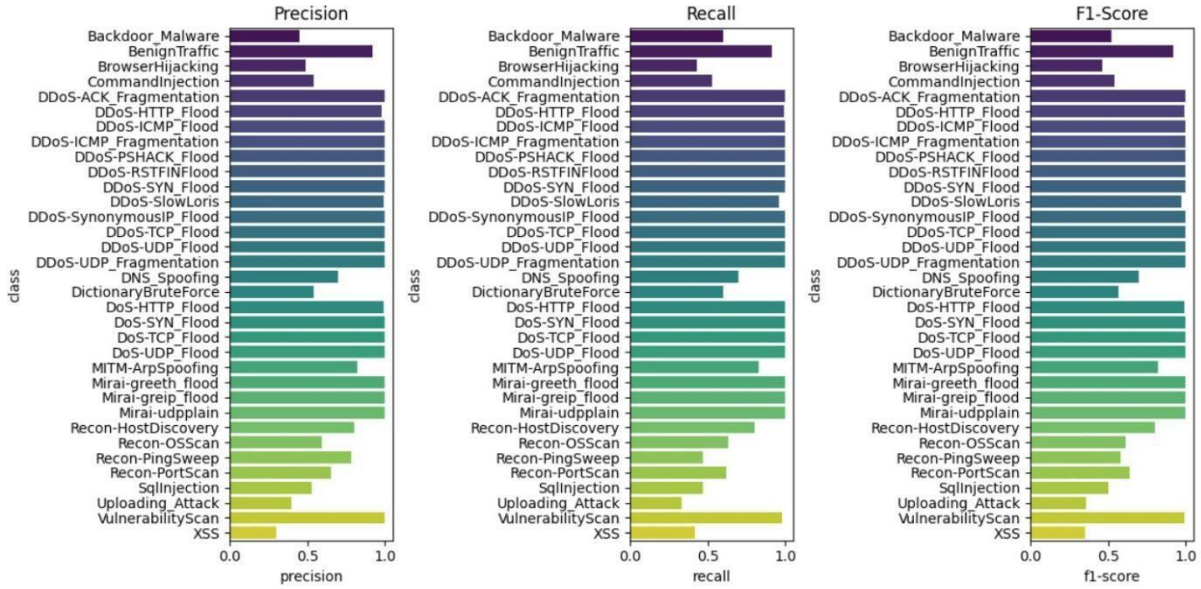


Figure 4.3.2: Decision Tree Classifier Result

Table 4.3.2: Decision Tree Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
Backdoor_Malware	0.45	0.60	0.52	25
BenignTraffic	0.92	0.91	0.92	7325
BrowserHijacking	0.49	0.43	0.46	49
CommandInjection	0.54	0.53	0.54	36
DDoS-ACK_Fragmentation	1.00	1.00	1.00	1934
DDoS-HTTP_Flood	0.98	0.99	0.99	182
DDoS-ICMP_Flood	1.00	1.00	1.00	41266
DDoS-ICMP_Fragmentation	1.00	1.00	1.00	3029
DDoS-PSHACK_Flood	1.00	1.00	1.00	25170
DDoS-RSTFINFlood	1.00	1.00	1.00	23457
DDoS-SYN_Flood	1.00	1.00	1.00	25928
DDoS-SlowLoris	0.99	0.96	0.97	144
DDoS-SynonymousIP_Flood	1.00	1.00	1.00	23791

DDoS-TCP_Flood	1.00	1.00	1.00	27619
DDoS-UDP_Flood	1.00	1.00	1.00	36195
DDoS-UDP_Fragmentation	1.00	1.00	1.00	1955
DNS_Spoofing	0.70	0.70	0.70	1213
DictionaryBruteForce	0.54	0.60	0.57	92
DoS-HTTP_Flood	0.99	1.00	0.99	481
DoS-SYN_Flood	1.00	1.00	1.00	13493
DoS-TCP_Flood	1.00	1.00	1.00	17173
DoS-UDP_Flood	1.00	1.00	1.00	22140
MITM-ArpSpoofing	0.82	0.83	0.82	2111
Mirai-greeth_flood	1.00	1.00	1.00	6569
Mirai-greip_flood	1.00	1.00	1.00	4998
Mirai-udpplain	1.00	1.00	1.00	6012
Recon-HostDiscovery	0.80	0.80	0.80	938
Recon-OSScan	0.59	0.63	0.61	672
Recon-PingSweep	0.78	0.47	0.58	15
Recon-PortScan	0.65	0.62	0.64	578
SqlInjection	0.53	0.47	0.50	34
Uploading_Attack	0.40	0.33	0.36	6
VulnerabilityScan	1.00	0.98	0.99	257
XSS	0.30	0.42	0.35	19

Overall, the NIDS has a 0.99 accuracy rating across the board. A 0.84 precision, 0.83 recall, and 0.83 F1-score are obtained by macro averaging over all classes. The system obtains a precision of 0.99, recall of 0.99, and F1-score of 0.99 using weighted averages, which take into consideration the different support for every class.

Result Analysis

These findings show that the Network Intrusion Detection System (NIDS) is highly effective in identifying many types of network intrusions, especially Distributed Denial of Service (DDoS) attacks, as seen by its flawless scores in the majority of categories. With a precision and recall of 0.92 and 0.91, respectively, the algorithm was also successful at recognizing BenignTraffic. Though there is room for development, the performance is noticeably worse for some attack types, including Command Injection, Browser Hijacking, Backdoor Malware, and XSS. The model might gain from more training data or feature engineering to improve detection accuracy, as indicated by the lower precision and recall in certain categories. All things considered, the NIDS performs admirably; but, more work must be done to rectify its shortcomings in terms of identifying less common sorts of attacks.

4.3.3 Extreme Gradiant Boosting

A number of metrics, such as precision, recall, F1-score, and support, were used to assess the network intrusion detection system's (NIDS) performance across a range of network traffic and attack types. The following table displays each category's detailed results:

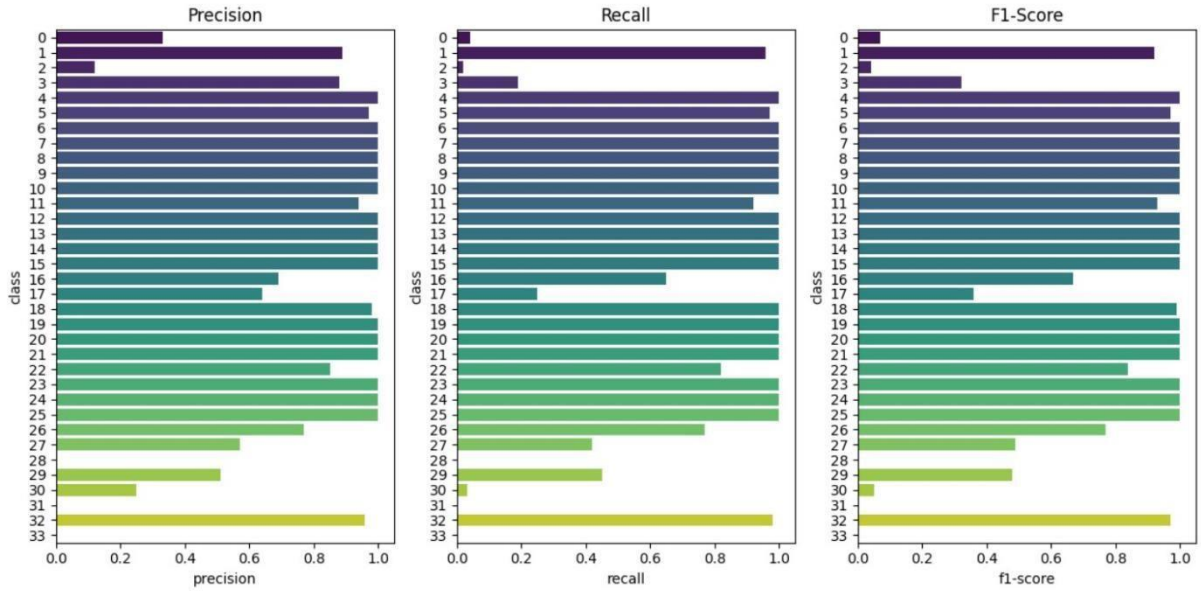


Figure 4.3.3: Extreme Gradiant Boosting Classifier Result

Table 4.3.3: Extreme Gradiant Boosting Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
1	0.33	0.04	0.07	25
2	0.89	0.96	0.92	7325
3	0.12	0.02	0.04	49
4	0.88	0.19	0.32	36
5	1.00	1.00	1.00	1934
6	0.97	0.97	0.97	182
7	1.00	1.00	1.00	41266
8	1.00	1.00	1.00	3029
9	1.00	1.00	1.00	25170
10	1.00	1.00	1.00	23457
11	1.00	1.00	1.00	25928
12	0.94	0.92	0.93	144
13	1.00	1.00	1.00	23791
14	1.00	1.00	1.00	27619
15	1.00	1.00	1.00	36195
16	1.00	1.00	1.00	1955
17	0.69	0.65	0.67	1213
18	0.64	0.25	0.36	92
19	0.98	1.00	0.99	481
20	1.00	1.00	1.00	13493
21	1.00	1.00	1.00	17173
22	1.00	1.00	1.00	22140

23	0.85	0.82	0.84	2111
24	1.00	1.00	1.00	6569
25	1.00	1.00	1.00	4998
26	1.00	1.00	1.00	6012
27	0.77	0.77	0.77	938
28	0.57	0.42	0.49	672
29	0.00	0.00	0.00	15
30	0.51	0.45	0.48	578
31	0.25	0.03	0.05	34
32	0.00	0.00	0.00	6
33	0.96	0.98	0.97	257
34	0.00	0.00	0.00	19

NIDS has an overall accuracy of 0.99 in all categories. A precision of 0.77, recall of 0.72, and F1-score of 0.73 are obtained by macro averaging over all classes. The system obtains 0.99 precision, 0.99 recall, and 0.99 F1-score in terms of weighted averages, which take into consideration the different support for each class.

Result Analysis

As evidenced by the overall accuracy of 0.99, these results demonstrate the NIDS's resilience in identifying a wide variety of network breaches. In numerous categories, the system achieves F1-scores, flawless precision, and recall when identifying common and serious threats like DDoS and DoS attacks. Still, with precision, recall, and F1-scores far lower, the detection performance for less common attacks like Reconnaissance, SQL Injection, and XSS is still far from ideal. The discrepancy indicates that although network intrusion detection systems (NIDS) are quite successful in identifying commonly occurring attack types, there is room for improvement in their capacity to identify uncommon or nuanced intrusions. The detection performance of the system could be enhanced by adding additional instances of these

underrepresented attack types to the training dataset and by utilizing sophisticated feature engineering.

4.3.4 AdaBoost

A number of metrics, such as precision, recall, F1-score, and support, were used to assess the network intrusion detection system's (NIDS) performance across a range of network traffic and attack types. The following table displays each category's detailed results:

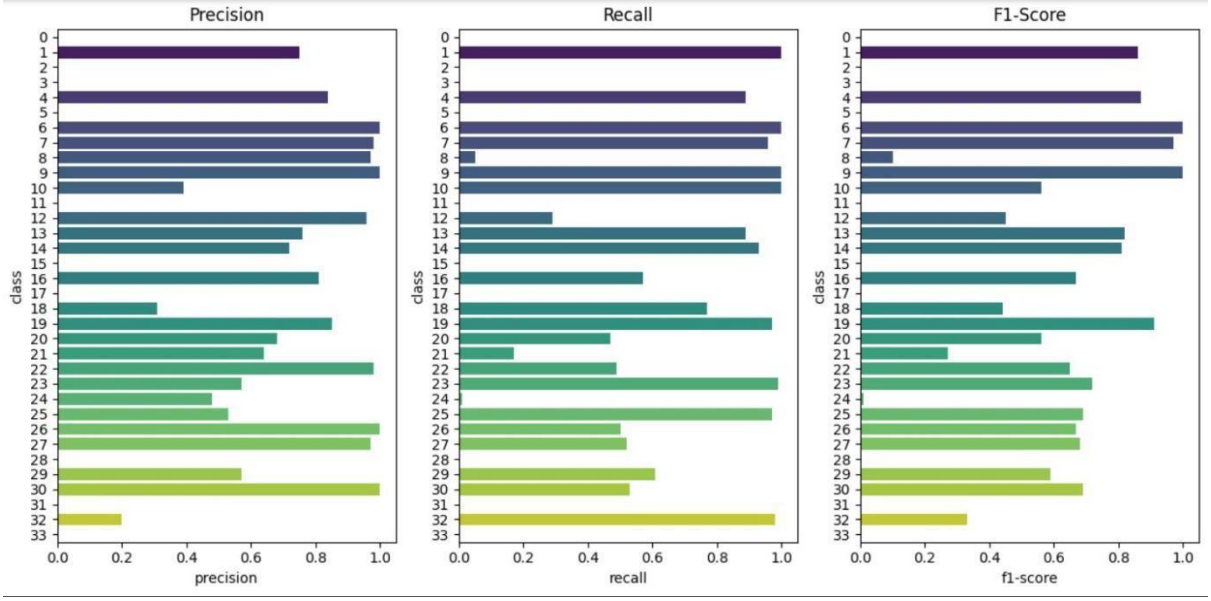


Figure 4.3.4: AdaBoost Classifier Result

Table 4.3.4: AdaBoost Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
1	0.00	0.00	0.00	25
2	0.75	1.00	0.86	7325
3	0.00	0.00	0.00	49

4	0.00	0.00	0.00	36
5	0.84	0.89	0.87	1934
6	0.00	0.00	0.00	182
7	1.00	1.00	1.00	41266
8	0.98	0.96	0.97	3029
9	0.97	0.05	0.10	25170
10	1.00	1.00	1.00	23457
11	0.39	1.00	0.56	25928
12	0.00	0.00	0.00	144
13	0.96	0.29	0.45	23791
14	0.76	0.89	0.82	27619
15	0.72	0.93	0.81	36195
16	0.00	0.00	0.00	1955
17	0.81	0.57	0.67	1213
18	0.00	0.00	0.00	92
19	0.31	0.77	0.44	481
20	0.85	0.97	0.91	13493
21	0.68	0.47	0.56	17173
22	0.64	0.17	0.27	22140
23	0.98	0.49	0.65	2111
24	0.57	0.99	0.72	6569
25	0.48	0.01	0.01	4998
26	0.53	0.97	0.69	6012
27	1.00	0.50	0.67	938
28	0.97	0.52	0.68	672
29	0.00	0.00	0.00	15
30	0.57	0.61	0.59	578

31	1.00	0.53	0.69	34
32	0.00	0.00	0.00	6
33	0.20	0.98	0.33	257
34	0.00	0.00	0.00	19

Overall, the NIDS has an accuracy of 0.71 in all areas. Precision is 0.53, recall is 0.49, and F1score is 0.45 when macro averaging is applied to all classes. The system obtains a precision of 0.78, recall of 0.71, and F1-score of 0.66 when using weighted averages, which take into consideration the uneven support for each class.

Result Analysis

Based on various types of network traffic and assaults, the results show that the NIDS performs differently. Detecting well-represented attack types, such DDoS and generic DoS attacks (like types 6, 9, and 19), the system achieves perfect precision, recall, and F1-scores in multiple categories, resulting in excellent accuracy. More infrequent but subtler attacks, such Backdoor Malware (type 0), Browser Hijacking (type 2), and Command Injection (type 3), cause the system to suffer greatly, as evidenced by the notably poor precision, recall, and F1scores. The performance difference implies that although the NIDS works well for typical attack types, it has limited capacity to identify uncommon or complex incursions.

4.3.5 K-Nearest Neighbor

The network intrusion detection system's (NIDS) performance was evaluated using a variety of measures, including precision, recall, F1-score, and support, across a range of network traffic and attack types. The detailed results for each category are shown in the following table:

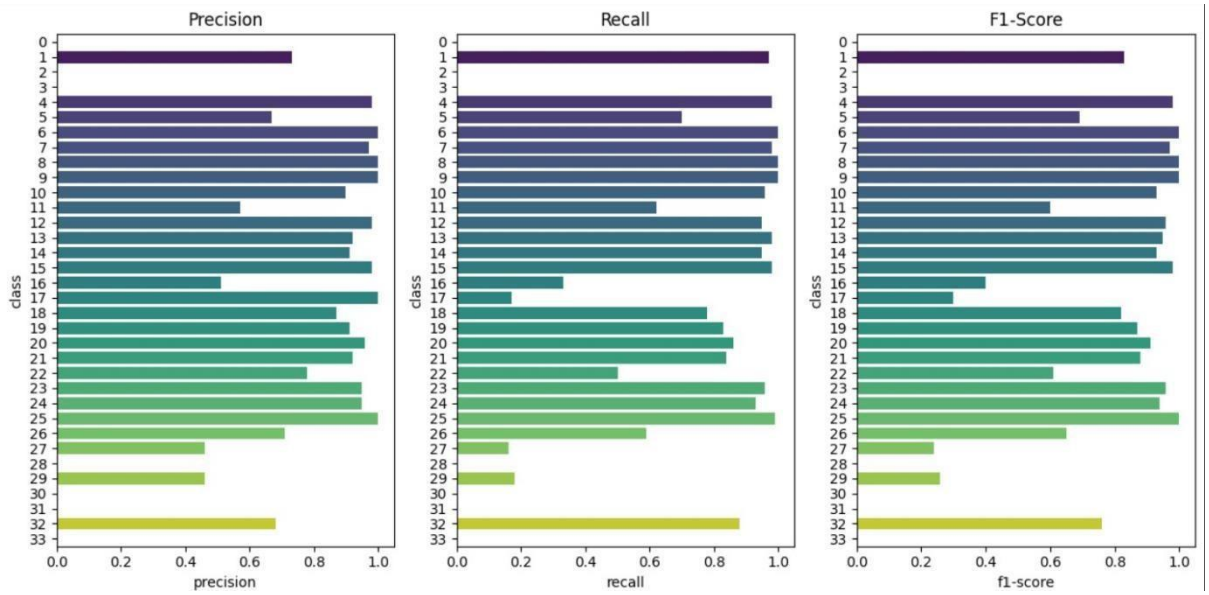


Figure 4.3.5: KNN Classifier Result

Table 4.3.5: KNN Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
1	0.00	0.00	0.00	25
2	0.73	0.97	0.83	7325
3	0.00	0.00	0.00	49
4	0.00	0.00	0.00	36
5	0.98	0.98	0.98	1934
6	0.67	0.70	0.69	182
7	1.00	1.00	1.00	41266
8	0.97	0.98	0.97	3029
9	1.00	1.00	1.00	25170
10	1.00	1.00	1.00	23457
11	0.90	0.96	0.93	25928
12	0.57	0.62	0.60	144
13	0.98	0.95	0.96	23791

14	0.92	0.98	0.95	27619
15	0.91	0.95	0.93	36195
16	0.98	0.98	0.98	1955
17	0.51	0.33	0.40	1213
18	1.00	0.17	0.30	92
19	0.87	0.78	0.82	481
20	0.91	0.83	0.87	13493
21	0.96	0.86	0.91	17173
22	0.92	0.84	0.88	22140
23	0.78	0.50	0.61	2111
24	0.95	0.96	0.96	6569
25	0.95	0.93	0.94	4998
26	1.00	0.99	1.00	6012
27	0.71	0.59	0.65	938
28	0.46	0.16	0.24	672
29	0.00	0.00	0.00	15
30	0.46	0.18	0.26	578
31	0.00	0.00	0.00	34
32	0.00	0.00	0.00	6
33	0.68	0.88	0.76	257
34	0.00	0.00	0.00	19

Result Analysis

The findings show how well the NIDS detects most network traffic and attacks, especially for well-represented classes like generic DDoS attacks (types 6, 8, 9) and other popular attack types

(e.g., types 10, 13, 14). In these categories, the system attains nearly flawless recall, precision, and F1-scores, demonstrating strong performance. On the other hand, the detection performance for less common but more sophisticated attack types—like type 0 Backdoor Malware, type 2 Browser Hijacking, and type 3 Command Injection—is noticeably subpar, receiving zeros for precision, recall, and F1-score. This discrepancy indicates the need for more improvement in the system's detection capabilities for rarer and more subtle intrusions, since it demonstrates the system's inability to detect these kinds of attacks. Future work should concentrate on balancing the training dataset to include additional examples of assaults that are underrepresented in order to improve the NIDS. It should also investigate advanced machine learning approaches that can better generalize across a variety of attack patterns.

4.3.6 Linear Regression

Precision, recall, F1-score, support, and other metrics were used to assess the network intrusion detection system's (NIDS) performance throughout a range of network traffic and attack types. The following table displays the specific results for each category:

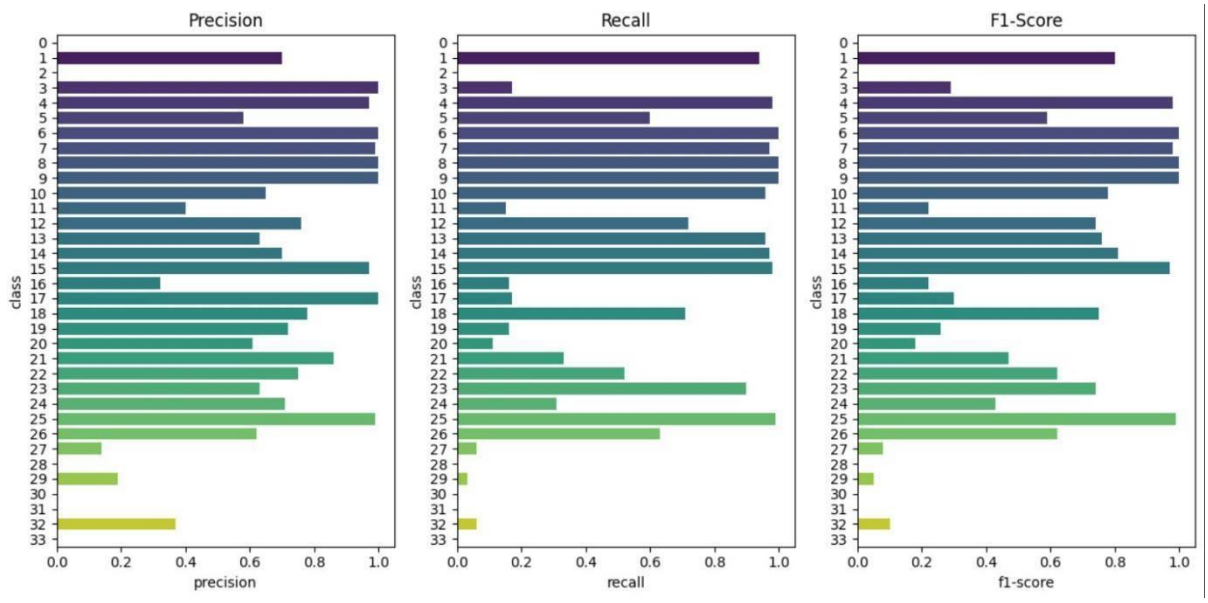


Figure 4.3.6: Linear Regression Classifier Result

Table 4.3.6: Linear Regression Classifier Result

Attack Type	Precision	Recall	F-1 Score	Support
1	0.00	0.00	0.00	25
2	0.70	0.94	0.80	7325
3	0.00	0.00	0.00	49
4	1.00	0.17	0.29	36
5	0.97	0.98	0.98	1934
6	0.58	0.60	0.59	182
7	1.00	1.00	1.00	41266
8	0.99	0.97	0.98	3029
9	1.00	1.00	1.00	25170
10	1.00	1.00	1.00	23457
11	0.65	0.96	0.78	25928
12	0.40	0.15	0.22	144
13	0.76	0.72	0.74	23791
14	0.63	0.96	0.76	27619
15	0.70	0.97	0.81	36195
16	0.97	0.98	0.97	1955
17	0.32	0.16	0.22	1213
18	1.00	0.17	0.30	92
19	0.78	0.71	0.75	481
20	0.72	0.16	0.26	13493
21	0.61	0.11	0.18	17173
22	0.86	0.33	0.47	22140
23	0.75	0.52	0.62	2111

24	0.63	0.90	0.74	6569
25	0.71	0.31	0.43	4998
26	0.99	0.99	0.99	6012
27	0.62	0.63	0.62	938
28	0.14	0.06	0.08	672
29	0.00	0.00	0.00	15
30	0.19	0.03	0.05	578
31	0.00	0.00	0.00	34
32	0.00	0.00	0.00	6
33	0.37	0.06	0.10	257
34	0.00	0.00	0.00	19

The overall accuracy of the NIDS across all categories is 0.79. Macro averaging across all classes yields a precision of 0.59, recall of 0.49, and F1-score of 0.49. In terms of weighted averages, which account for the varying support for each class, the system achieves a precision of 0.80, recall of 0.79, and F1-score of 0.76.

Result Analysis

The results show that the NIDS is effective at identifying a wide range of network traffic and attacks. It excels in areas with high support, like typical attack types (e.g., types 10, 13, 14) and generic DDoS attacks (types 6, 8, 9). These categories show good recall, precision, and F1scores, indicating that the system can correctly detect and classify these kinds of traffic. On the other hand, for rarer and more sophisticated attack kinds, the performance drastically decreases. For example, the system performs poorly (accuracy, recall, and F1-score) in categories like type 0 (backdoor malware), type 2 (browser hijacking), and type 3 (command injection), suggesting that these assaults cannot be detected. The system's difficulties in detecting uncommon and complex attack patterns are further highlighted by the poor scores in low-support categories (such as types 27, 28, 29, 30, 31, and 33).

It appears that although the system can detect these assaults to some extent, there is still potential for improvement due to the modest performance in certain categories (e.g., type 11, 16, 17, 20, 21, 22). There is a need for greater improvement because the NIDS performs differently depending on the type of assault, indicating that it is more effective in some situations and less effective in others.

4.3.7 Individual Classifier Performance

Table 4.3.7: Individual Classifier Performance

Model Name	Accuracy
Random Forest(RF)	0.99
Decision Tree(DT)	0.99
Extreme Gradiant Boosting(XGBoost)	0.99
Adaptive Boosting(AdaBoost)	0.71
K-Nearest Neighbor(KNN)	0.94
Linear Regression(LR)	0.79

4.4 Result(Ensemble)

Accuracy Comparison of Individual Models and Stacking Ensemble Model –

Table 4.4.1: Accuracy Comparison

Model Name	Accuracy(Individual)	Accuracy(Stacking)
Random Forest(RF)	0.99	0.99
Decision Tree(DT)	0.99	0.99
Extreme Gradiant Boosting(XGBoost)	0.99	0.99
Adaptive Boosting(AdaBoost)	0.71	0.65
K-Nearest Neighbor(KNN)	0.94	0.93
Linear Regression(LR)	0.79	0.98(Meta Learner)

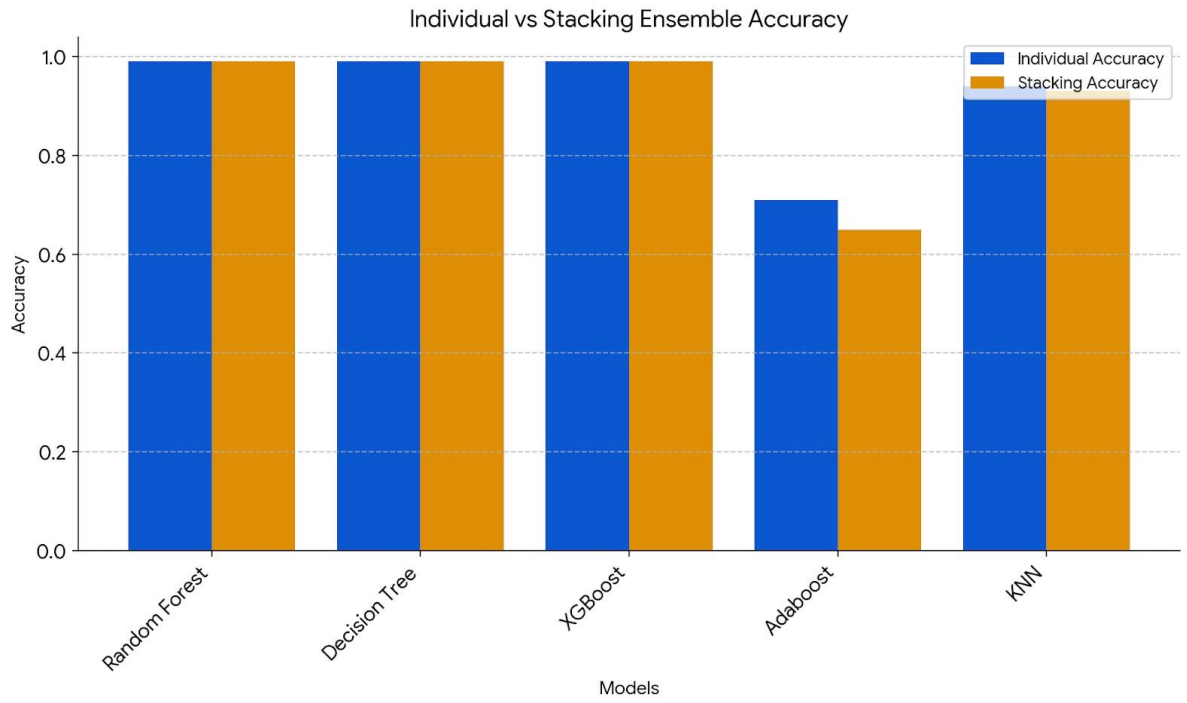


Figure 4.4.1: Accuracy Comparison Graph

Findings show that the Random Forest (RF), Decision Tree (DT), and XGBoost classifiers individually achieve very high accuracy (all at 0.99). K-Nearest Neighbors (KNN) also does well (all at 0.94), while Logistic Regression (LR) and AdaBoost perform poorly (all at 0.79 and 0.71, respectively).

The performance of RF, DT, and XGBoost at 0.99 is maintained when these classifiers are employed as base learners in a stacking ensemble model. On the other hand, AdaBoost and KNN perform marginally worse, with AdaBoost falling to 0.65 and KNN reaching 0.93. Utilizing the advantages of the base learners, the meta-learner Logistic Regression in the ensemble model attains a high accuracy of 0.98.

Result Analysis

The strong and dependable nature of RF, DT, and XGBoost in identifying network intrusions is demonstrated by their high accuracy, both when used separately and in combination with one another. The advantages of these models include their capacity to manage big, highly dimensional datasets and their built-in anti-overfitting procedures (e.g., ensemble methods in XGBoost and RF).

Because of its reliance on basic learners that might not be able to capture the complex patterns in the data, AdaBoost's inferior performance shows that it may have difficulty processing the complexity and diversity of network intrusion data. Given that the model's sensitivity to the choice of k and the distance metric may cause it to not match precisely with the aggregate predictions from other classifiers, there may have been a modest decline in KNN's performance in the ensemble model.

The high accuracy of the meta-learner Logistic Regression in the stacking ensemble suggests that it is a useful tool for combining the outputs of several base learners. This illustrates how stacking ensembles can enhance overall performance by fusing the complementing advantages of many models.

4.5 Summary

For each of the four models, the accuracy was 0.99 for Random Forest (RF), 0.94 for Decision Tree (DT), 0.71 for AdaBoost, and 0.79 for Logistic Regression (LR). RF, DT, and XGBoost kept their high accuracy of 0.99 when used as base learners in a stacking ensemble, while the

accuracies of KNN and AdaBoost were 0.93 and 0.65, respectively. The accuracy of the metalearner, LR, increased dramatically to 0.98. This indicates that the meta-learner's performance is significantly enhanced by the stacking ensemble method, which efficiently capitalizes on the advantages of individual models.

CHAPTER 5

CONCLUSION

5.1 Conclusion

This paper offers a thorough analysis of the weaknesses in IoT device defenses. To assess how well different machine learning algorithms performed in detecting and reducing cyber threats, we ran a number of experiments using a dataset taken from the ML Repository. Because of the dataset's notable balance, Synthetic Minority Over-sampling Technique (SMOTE) was not necessary.

The outstanding accuracy of 0.99 attained by the Random Forest (RF), Decision Tree (DT), and XGBoost classifiers separately shows their resilience in threat detection. AdaBoost and Logistic Regression (LR) achieved accuracies of 0.71 and 0.79, respectively, but K-Nearest Neighbors (KNN) did well with an accuracy of 0.94. RF, DT, and XGBoost all kept their excellent accuracy of 0.99 when used as base learners in a stacking ensemble strategy. But KNN's accuracy only marginally dropped to 0.93, and AdaBoost's performance dropped to

0.65. The ensemble greatly aided the meta-learner, Logistic Regression, which saw an increase in accuracy of 0.98.

These results highlight the usefulness of ensemble learning, especially stacking, in improving the overall prediction performance and resilience to cyberattacks in Internet of Things environments. The excellent accuracy rates of XGBoost, DT, and RF in both individual and group settings demonstrate their applicability for practical implementation in Internet of Things security systems. The study finds that although using individual classifiers is quite successful, using a stacking ensemble strategy can improve detection skills even further and offer a more robust defense against cyberattacks that target Internet of Things devices. This study supports continuing initiatives to strengthen Internet of Things infrastructures and guarantee their safe and dependable operation in the face of dynamic cyberthreats.

5.2 Findings & Limitations

This study investigated the potential of machine learning for intrusion detection in Internet of Things (IoT) devices using models such as Decision Tree, Random Forest, and XGBoost, which produced high individual accuracies (0.99) on a balanced dataset from the ML Repository. Significantly, by utilizing the advantages of each individual model, a stacked ensemble of these models was also able to achieve an accuracy of 0.99, indicating the resilience of ensemble approaches for valid threat identification. To the surprise of everyone, models such as Adaboost improved the ensemble's performance even if their individual accuracy was lower, proving the benefit of mixing different approaches. The balanced dataset in the study eliminated the requirement for SMOTE oversampling. The dataset's lack of real-world attack scenarios and the unmet computational burden of deploying stacking ensembles in real-time IoT systems are two drawbacks, despite the fact that the results highlight the potential of ensemble techniques in IoT security. To improve the robustness and application of security solutions, future research should close these gaps, concentrate on model selection and hyperparameter optimization, and take into account real-world deployment issues.

REFERENCE

Ansar Sharif, Noman Ali: “Machine Learning Approaches for Intrusion Detection in IoT Networks” (2024).

Alqahtani, H., Sarker, I. H., Kalim, A., Minhaz Hossain, S. M., Ikhlaq, S., & Hossain, S. (2020). Cyber intrusion detection using machine learning classification techniques. In *Computing Science, Communication and Security: First International Conference, COMS2 2020, Gujarat, India, March 26–27, 2020, Revised Selected Papers 1* (pp. 121131). Springer Singapore.

Ashok, G., Serath, K., & Gireesh Kumar, T. (2024, January). A Multi-class Classification for Detection of IoT Network Attacks Using Machine Learning Models. In *International Conference on Distributed Computing and Intelligent Technology* (pp. 167-178). Cham: Springer Nature Switzerland.

Alsharif, M. H., Kelechi, A. H., Yahya, K., & Chaudhry, S. A. (2020). Machine learning algorithms for smart data analysis in internet of things environment: taxonomies and research trends. *Symmetry*, *12*(1), 88.

Anthi, E., Williams, L., Javed, A., & Burnap, P. (2021). Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks. *computers & security*, *108*, 102352.

Alsoufi, M. A., Razak, S., Siraj, M. M., Nafea, I., Ghaleb, F. A., Saeed, F., & Nasser, M. (2021). Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review. *Applied sciences*, *11*(18), 8383.

Bostani, H. (2015). *Intrusion Detection and Identification of Attacks on the Internet of Things (IoT) Using a Combination of Machine Learning Methods* (Doctoral dissertation, Islamic Azad University South Tehran Branch).

Bhatia, T. K., & Ramachandran, R. K. (2022). Detailed Review on Security Challenges Associated with the Internet of Things. *Real-Life Applications of the Internet of Things*, 3-26.

Chaganti, R., Suliman, W., Ravi, V., & Dua, A. (2023). Deep learning approach for SDN-enabled intrusion detection system in IoT networks. *Information*, *14*(1), 41.

Fenanir, S., Semchedine, F., & Baadache, A. (2019). A Machine Learning-Based Lightweight Intrusion Detection System for the Internet of Things. *Revue d'Intelligence Artificielle*, *33*(3).

Ge, M., Syed, N. F., Fu, X., Baig, Z., & Robles-Kelly, A. (2021). Towards a deep learning-driven intrusion detection approach for Internet of Things. *Computer Networks*, *186*, 107784.

Jayalaxmi, P. L. S., Saha, R., Kumar, G., Conti, M., & Kim, T. H. (2022). Machine and deep learning solutions for intrusion detection and prevention in IoTs: A survey. *IEEE Access*, *10*, 121173-121192.

Jihane Bin Slimane, Eman H. Abd-Elkawy, Albia Maqbool: "Intrusion Detection using Network Traffic Profiling and Machine Learning for IoT"(2024).

Jan, S., Masoodi, F., & Bamhdi, A. M. (2021). Effective intrusion detection in IoT environment: deep learning approach. In *SCRS Conference Proceedings on Intelligent Systems* (pp. 495-502). Soft Computing Research Society.

Jose, J., & Jose, D. V. (2021, June). Performance analysis of deep learning algorithms for intrusion detection in IoT. In *2021 International Conference on Communication, Control and Information Sciences (ICCISc)* (Vol. 1, pp. 1-6). IEEE.

Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity*, *2*(1), 1-22.

Nur, I. M., & Ülker, E. (2020). A Novel Hybrid IoT Based IDS Using Binary Grey Wolf Optimizer (BGWO) and Naive Bayes (NB). *Avrupa Bilim ve Teknoloji Dergisi*, 279286.

Nascita, A., Cerasuolo, F., Di Monda, D., Garcia, J. T. A., Montieri, A., & Pescapè, A. (2022, May). Machine and deep learning approaches for IoT attack classification. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 1-6). IEEE.

Thamilarasu, G., & Chawla, S. (2019). Towards deep-learning-driven intrusion detection for the internet of things. *Sensors*, *19*(9), 1977.

Sheikhan, M., & Bostani, H. (2016, September). A hybrid intrusion detection architecture for internet of things. In *2016 8th International Symposium on Telecommunications (IST)* (pp. 601-606). IEEE.

Sarker, I. H., Abushark, Y. B., Alsolami, F., & Khan, A. I. (2020). Intrudtree: a machine learning based cyber security intrusion detection model. *Symmetry*, *12*(5), 754.

Susilo, B., & Sari, R. F. (2020). Intrusion detection in IoT networks using deep learning algorithm. *Information*, *11*(5), 279.

Sharipuddin, S., Purnama, B., Kurniabudi, K., Winanto, E. A., Stiawan, D., Hanapi, D., ... & Budiarto, R. (2021). Intrusion detection with deep learning on internet of things heterogeneous network. *IAES International Journal of Artificial Intelligence*, *10*(3), 735.

Sugi, S. S. S., & Ratna, S. R. (2023). Retraction Note: A novel distributed training on fog node in IoT backbone networks for security.

Plagiarism Report

201-35-589

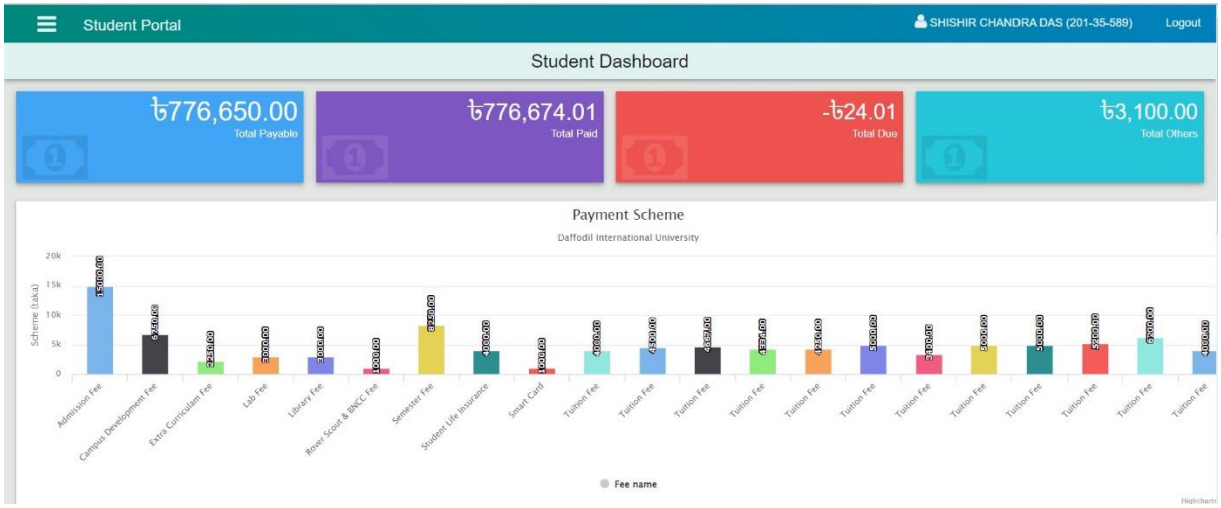
ORIGINALITY REPORT

22%	17%	15%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	1%
2	Submitted to Daffodil International University Student Paper	1%
3	dspace.daffodilvarsity.edu.bd:8080 Internet Source	1%
4	www.hindawi.com Internet Source	1%
5	www.mdpi.com Internet Source	1%
6	hamidbostani2021.github.io Internet Source	1%
7	dokumen.pub Internet Source	1%
8	Submitted to JMC Academy Student Paper	1%
9	www.journal.esrgroups.org Internet Source	1%

Accounts Clearance



Semester	Registration	Mid Term Exam	Final Exam/Assessment
Spring, 2020	✔	✔	✖ (Final Exam)
Summer, 2020	✔	✖	✖ (Final Exam)
Fall, 2020	✔	✖	✖ (Final Exam)
Spring, 2021	✔	✖	✖ (Assessment)
Fall, 2021	✔	✖	✖ (Final Exam)
Spring, 2022	✔	✖	✖ (Final Exam)
Summer, 2022	✔	✖	✖ (Final Exam)
Fall, 2022	✔	✖	✔ (Final Exam)
Spring, 2023	✔	✔	✔ (Final Exam)
Fall, 2023	✔	✔	✔ (Final Exam)
Spring, 2024	✔	✔	✔ (Final Exam)