



REPUBLIC OF TÜRKİYE

ALTINBAŞ UNIVERSITY

Institute of Graduate Studies

Information Technologies

**INTRUSION DETECTION SYSTEM IN
INTERNET OF THINGS NETWORKS USING
MACHINE LEARNING TEHNCAIQUES**

Muhanad Badee Muhammed AL-DOORI

Master's Thesis

Supervisor

Asst. Prof. Dr. Sefer KURNAZ

Istanbul, 2023

INTRUSION DETECTION SYSTEM IN INTERNET OF THINGS NETWORKS USING MACHINE LEARNING TECHNIQUES

Muhanad Badee Muhammed AL-DOORI

Information Technologyies

Master's Thesis

ALTINBAŞ UNIVERSITY

2023

INTRUSION DETECTION SYSTEM IN INTERNET OF THINGS NETWORKS USING MACHINE LEARNING TECHNIQUES prepared by MUHANAD BADEE MUHAMMED AL-DOORI and submitted on 06/04/2023 has been **accepted unanimously** for the degree of Master of Science in Information Technologies.

Asst. Prof. Dr. Sefer KURNAZ

Supervisor

Thesis Defense Committee Members:

Asst. Prof. Dr. Sefer KURNAZ

Department of Computer
Engineering,
Altınbaş University

Asst. Prof. Dr. Abdullahi Abdu
IBRAHIM

Department of Computer
Engineering,
Altınbaş University

Asst. Prof. Dr. Zeynep ALTAN

Department of Software
Engineering
Beykent University

I hereby declare that this thesis meets all format and submission requirements of a Master's thesis.

Submission date of the thesis to Institute of Graduate Studies: ____/____/____

By signing this declaration, I vouch for the fact that all information and resources used in this capstone project were acquired with the utmost adherence to moral standards and academic regulations. All references listed in the Reference List and vice versa have been cited in accordance with the aforementioned standards of conduct. I also attest to the accurate citation of all conclusions and borrowed ideas throughout the article.

Muhanad Badee Muhammed AL-DOORI

Signature

DEDICATION

I dedicate this thesis to my family, who have always helped me succeed academically and encouraged, empowered, and supported me. My parents, whose steadfast support and love have provided me with continual courage throughout the years. I would also like to thank my wife and my daughter, who have supported me throughout my studying journey. I extend my gratitude to my supervisor, who has guided me through the research process with his expertise and patience. His invaluable feedback and support have been instrumental in shaping the direction and outcome of this thesis



ABSTRACT

INTRUSION DETECTION SYSTEM IN INTERNET OF THINGS NETWORKS USING MACHINE LEARNING TECHNIQUES

AI-DULUIMI, Mohammed Ahmed

M.Sc., Information Technologyies, Altinbaş
University,

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Date: April /2023

Page: 86

The Internet of Things (IoT) plays a crucial role in numerous industries, such as transportation and healthcare logistics tracking, owing to its decentralized network of nodes, servers, and communication software. As the IoT continues to grow rapidly, there is a substantial increase in the number of connected devices, making them vulnerable to cyberattacks, including distributed denial of service (DDoS) attacks due to their extensive attack surface. Traditional signature-based intrusion detection methods have limitations, relying on specialists, being time-consuming to update, and not providing comprehensive protection against all types of assaults, including zero-day exploits. Encouraging advancements have been made in intrusion detection using AI-based systems. This research proposes a machine learning-based approach that leverages an ensemble of models. Extensive experimentation was conducted using publicly available IoT data, specifically the CIC-DDoS2019 dataset, which encompasses multi-class attacks, to evaluate the proposed approach. The study primarily focuses on categorizing DDoS attacks, utilizing a filter-based feature reduction methodology based on the Random Forest method and Pearson correlation. The top 20 features were selected for classification. Multiple machines learning models, including Extreme Gradient Boosting (XGB), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), and K-Nearest Neighbours (KNN), as well as a Hybrid ML model, were employed to identify DDoS attacks. The results demonstrated the outstanding effectiveness of the proposed method, with the XGBoost

model achieving a remarkable accuracy of 98.50%. Additionally, the Hybrid ML model showed impressive results, achieving an accuracy of 97.46%. The proposed method significantly outperformed previous models, as evidenced by a comparison with existing research results.

Keywords: IDS, IoT, Machine Learning, DDoS, XGBoost, Network Security.



TABLE OF CONTENTS

	<u>Pages</u>
ABSTRACT	vi
LIST OF TABLES.....	xi
LIST OF FIGURES.....	xii
ABBREVIATIONS.....	xiv
1. INTRODUCTION	1
1.1 BACKGROUND	1
1.2 MOTIVATION	2
1.3 PROBLEM STATEMENT	3
1.4 THESIS OBJECTIVES	4
1.5 THESIS CONTRIBUTION	5
1.6 THESIS OUTLINE.....	6
2. LITERATURE REVIEW.....	8
2.1 INTRODUCTION	8
2.2 INTRODUCTION TO INTRUSION DETECTION SYSTEMS (IDS).....	8
2.2.1 Definition and Classification of Intrusion Detection Systems (IDS).....	9
2.2.2 Traditional IDS Approaches	11
2.2.3 Modern IDS Approaches	12
2.2.4 Challenges and Limitations of Intrusion Detection Systems (IDS).....	13
2.5 INTERNET OF THINGS (IoT).....	14
2.5.1 Definition and Components of IoT (Internet of Things)	14
2.5.2 Applications and Industries Using IoT	15
2.5.3 Security Concerns in IoT	16
2.5.4 Relationship Between IoT and IDS	17
2.6 MACHINE LEARNING IN IDS.....	18
2.6.1 Introduction to Machine Learning	18

2.6.2	Application of Machine Learning in IDS	20
2.6.3	Benefits of Machine Learning-Based IDS	22
2.6.4	Challenges of Machine Learning-Based IDS	22
2.7	RELATED WORK.....	23
3.	METHODOLOGY.....	26
3.1	INTRODUCTION	26
3.2	PROPOSED WORK.....	27
3.2.1	Data Acquisition and Preprocessing	27
3.2.2	Feature Selection and Extraction	28
3.2.3	Development of Machine Learning Models	28
3.2.4	Model Training, Validation, and Selection.....	28
3.2.5	Performance Evaluation.....	29
3.3	DATASET	30
3.4	PREPRCOESSING OF DATASET	31
3.5	FEATURE SELECTION.....	33
3.5.1	Random Forest-based Feature Selection.....	33
3.5.2	Pearson Correlation-based Feature Selection	34
3.6	COMPARSION OF THE LABEL OF THE DATASET	35
3.7	PREPATION OF SELCTING THE FEATURE	37
3.8	DATASET SPELITTING.....	38
3.9	MACHINE LEARNING MODELS	39
3.9.1	XGBOOST ALGORTHIM.....	39
3.9.2	NAVIS BAYS ALGROTHIM.....	42
3.9.3	Multi-layer Perceptron (MLP)	43
3.9.4	Logistic Regression Algorithm	45
3.9.5	Decision Tree Algorithm	46
3.9.10	Proposed Hybrid Algorithm.....	48
3.10	CHAPTER SUMMARY.....	49

4. PERFORMANCE EVALUATION	51
4.1 INTRODUCTION	51
4.2 EVALATION METRICES.....	52
4.3 MACHINE LEARNING RESULTS	53
4.3.1 Xgboost Algorithm Results.....	53
4.3.2 Naive Bayes Algorithm Result	54
4.3.3 Multilayer Perceptron (MLP) Result	56
4.3.4 Logistic Regression Algorithm Result.....	57
4.3.5 Decision Tree Algorithm Result	59
4.3.6 K-Nearest Neighbours (KNN) Algorithm Result	61
4.3.7 Proposed Hybrid Algorithm Result	63
4.4 DISSCUSION	65
4.4.1 Comparative with Other Studies	66
5. Conclusion and future work.....	68
5.1 CONCLUSION.....	68
5.2 FUTURE WORK.....	68
REFERENCES	70

LIST OF TABLES

	<u>Pages</u>
Table 2.1: Related Work for IDS Based Machine Learning.....	23
Table 4.1: Indicts Simulation Parameters.....	51
Table 4.2: Results of Xgboost Algorithm.....	53
Table 4.3: Results of Naive Bayes Algorithm.....	55
Table 4.4: Results of MLP Algorithm	56
Table 4.5: Results of Logistic Regression Algorithm	58
Table 4.6: Results of Decision Tree Algorithm.....	60
Table 4.7: Results of ANN Algorithm	62
Table 4.8: Shows the Result of Proposed Hybrid Algorithm.....	63
Table 4.9: Comparison with Other Stuides	67

LIST OF FIGURES

	<u>Pages</u>
Figure 1.1: Flowchart Shows the Outline of the Thesis	63
Figure 2.1: Concept of IDS in IOT [17]	9
Figure 2.2: Classification of IDS [20]	11
Figure 2.3: Traditional IDS Approaches [21].....	12
Figure 2.4: Components of IoT [24].....	15
Figure 2.5: IOT Security Concerns [29].....	17
Figure 2.6: Classification of Machine Learning [35]	20
Figure 3.1: Flowchart of Proposed Work	30
Figure 3.2: Indicts Importing of the CIC-DDoS2019 dataset	31
Figure 3.3: Indicates the Statues of the Missing Values.....	32
Figure 3.4: Indicts the Statues of the Dataset After Preprocessing Step	33
Figure 3.5: Indicts of Pearson Correlation Matrix.....	35
Figure 3.6: Indicts of Comparison of Label Column	37
Figure 3.7: Indicts the Selection of the Feature	38
Figure 3.8: Indicts of Data Splitting	39
Figure 3.9: Concept of XGBOOST	41
Figure 3.10: Indices Process of Naive Bayes	43
Figure 3.11: Concept of MLP	44
Figure 3.12: Concept of Logistic Regression	45
Figure 3.13: Concept of Decision Trees	47
Figure 3.14: Indicts Flowchart of the Proposed Hybrid Algorithm	49
Figure 4.1: Indicates of the Confusion Matrix of Xgboost Result	54
Figure 4.2: Indicates of the Confusion Matrix of Navis Bayes Result.....	55

Figure 4.3: Indicates of the Confusion Matrix of MLP Result.....	57
Figure 4.4: Indicates of the Confusion Matrix of Logistic Regression Result.....	59
Figure 4.5: Indicates of the Confusion Matrix of Decision Tree algorithm.....	61
Figure 4.6: Indicates of the Confusion Matrix of KNN Algorithm.....	63
Figure 4.7: Shows the Result of Proposed Hybrid Algorithm.....	65



ABBREVIATIONS

IDS	:	Intrusion Detection System
ML	:	Machine Learning
ANN	:	Artificial Neural Network
LR	:	Logistic Regression
MPL	:	Multilayer Perceptron's Classifier
SL	:	Supervised Learning
USL	:	Unsupervised Learning
CNN	:	Convolutional Neural Network
IoT	:	Internet of things

1. INTRODUCTION

1.1 BACKGROUND

In today's interconnected and data-driven world, the Internet of Things (IoT) stands as a testament to technological advancement. These smart devices, interwoven through our daily lives, industries, and critical infrastructures, facilitate seamless interactions and unprecedented efficiencies. However, this complex web of connections has opened new frontiers for cyber threats, making the security of IoT devices an urgent and evolving challenge [1]. The sophisticated and dynamic nature of these threats demands equally adaptive and robust security mechanisms. Intrusion Detection Systems (IDS) have emerged as a crucial line of defense within the IoT landscape, constantly monitoring and reacting to potential threats in real-time [2]. Yet, traditional IDS methods have struggled to keep pace with the ever-changing threat landscape, revealing a pressing need for more advanced and adaptable strategies. This thesis embarks on an exploratory journey into the integration of machine learning algorithms with IDS, aiming to revolutionize the defense mechanisms for IoT devices. By leveraging advanced learning techniques, the research proposes a novel framework capable of understanding and predicting intricate attack patterns and devising intelligent, real-time countermeasures. Focusing specifically on innovative algorithms such as Hybrid XGBoost, Multi-Layer Perceptron (MLP), and Decision Trees, the study explores a harmonious blend of these methodologies to create an ensemble model. This approach not only seeks to maximize the strengths of each individual algorithm but also provides an adaptive and robust IDS tailored for the unique vulnerabilities of IoT devices [3].

Through meticulous analysis and experimentation, employing diverse datasets and simulated real-world scenarios, the research will evaluate the performance and efficiency of the proposed model against traditional IDS approaches [4]. This comprehensive examination aims to present a practical and potent solution to the cybersecurity community, specifically targeting the challenges associated with IoT devices. Ultimately, this thesis seeks to contribute groundbreaking insights into the field of IoT security, highlighting the transformative potential of machine learning-enhanced Intrusion Detection Systems. It not only aspires to build more resilient defense mechanisms but also to foster a deeper understanding of the complex interactions and threats within the IoT landscape. By paving the way for innovative and adaptable solutions, this research marks a significant step forward

in safeguarding our increasingly interconnected world against the evolving specter of cyber threats [5].

As we delve into this critical area of study, the ambition is clear: to create a lasting impact on the front lines of cybersecurity research, setting new standards and guiding future endeavors in the protection of IoT systems.

1.2 MOTIVATION

The motivation behind this thesis is driven by the increasing severity and complexity of cyber threats in our interconnected world. As our reliance on digital technologies grows, so does the potential for cybercriminals to exploit vulnerabilities and cause significant harm to individuals, organizations, and critical infrastructures. Intrusion Detection Systems (IDS) play a pivotal role in mitigating these risks by identifying and responding to malicious activities in real-time. While traditional rule-based IDS have been effective to some extent, they often struggle to keep pace with the dynamic and evolving nature of modern cyber threats. Machine learning methods offer a promising solution to this challenge, as they enable IDS to learn from large volumes of data and adapt their detection capabilities to emerging attack vectors. The integration of machine learning into IDS has gained increasing attention, promising enhanced accuracy and responsiveness in detecting cyber intrusions. The motivation behind this thesis lies in exploring the potential of Hybrid XGBoost, MLP, and Decision Tree algorithms as a synergistic ensemble approach to intrusion detection. Each of these algorithms possesses unique strengths that, when combined, can create a powerful and adaptable IDS capable of handling various types of cyber threats. By harnessing the predictive power of Hybrid XGBoost, the ability to model non-linear relationships in data with MLP, and the interpretability of Decision Trees, we aim to devise an IDS that achieves higher accuracy and robustness. Moreover, the rapidly evolving landscape of machine learning algorithms and intrusion detection techniques necessitates continuous research to stay at the forefront of cybersecurity. The exploration of the proposed Hybrid XGBoost-MLP-Decision Tree ensemble approach goes beyond theoretical speculation. Through extensive experimentation and comparative studies, we seek to provide concrete evidence of its performance in real-world settings. Evaluating the ensemble approach against traditional IDS techniques and individual machine learning algorithms will enable us to validate its efficacy and identify potential areas for improvement.

Finally, the motivation driving this thesis is to contribute to the advancement of intrusion detection research and the development of effective cybersecurity solutions. By exploring the potential of hybrid machine learning models in the context of IDS, we aim to empower cybersecurity practitioners with practical knowledge to build more resilient and adaptive defense mechanisms. The outcomes of this research can make a significant impact in safeguarding critical information systems, protecting digital assets, and fortifying our digital ecosystems against the ever-evolving threat landscape.

1.3 PROBLEM STATEMENT

The Internet of Things (IoT), which includes a huge variety of linked devices in many different industries, has become a pillar of contemporary technological progress. But this development has also highlighted a serious flaw: the vulnerability to Distributed Denial of Service (DDoS) attacks. Traditional signature-based intrusion detection solutions have been found to be ineffective when the complexity and scope of these attacks increase. These conventional methods are not only time-consuming and reliant on continuous manual updates but also often fail to protect against emerging and sophisticated attacks, including zero-day exploits. The core problem this research addresses lies in this evident gap in existing intrusion detection methodologies. There is an urgent and unmet need for a dynamic, adaptive, and efficient approach capable of identifying and thwarting DDoS attacks in the multifaceted and constantly evolving environment of IoT. Traditional methods are falling behind, and the quest for a solution that balances accuracy, efficiency, and adaptability becomes not just a technical challenge but a vital necessity for safeguarding the integrity of interconnected digital systems. This research aims to transcend the limitations of conventional intrusion detection by proposing a machine learning-based strategy, utilizing a blend of cutting-edge algorithms, and focusing on precise categorization and feature selection. By engaging with this central problem, the study aspires to forge a new path in DDoS attack detection, one that aligns with the technological demands of our interconnected age and stands as a robust defense against the growing menace of cyber threats in the IoT landscape. The implications of resolving this problem reach far beyond mere technological improvement, touching the very core of how we secure, trust, and utilize our interconnected digital world.

1.4 THESIS OBJECTIVES

These objectives collectively aim to advance the current state of intrusion detection systems within IoT, tackling the significant challenges posed by DDoS attacks and contributing to the broader field of cybersecurity.

- i. To Investigate the Limitations of Traditional IDS: Examine the shortcomings and constraints of traditional signature-based intrusion detection systems (IDS) within the context of IoT, focusing on their inability to efficiently adapt to evolving Distributed Denial of Service (DDoS) attacks.
- ii. To Develop a Hybrid Machine Learning-Based Strategy: Design and implement a novel machine learning-based strategy that combines several algorithms, including Extreme Gradient Boosting (XGB), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), K-Nearest Neighbours (KNN), and Hybrid ML models, to enhance the detection of DDoS attacks.
- iii. To Implement Feature Selection and Reduction Techniques: Utilize a filter-based feature reduction methodology based on the Random Forest method and Pearson correlation, with the aim of optimizing the classification process and effectively distinguishing between attack and normal traffic.
- iv. To Evaluate the Proposed Method Using Real-World Data: Conduct extensive trials using publicly available data from IoT devices, including the CIC-DDoS2019 dataset, to validate the effectiveness, efficiency, and accuracy of the proposed method.
- v. To Compare with Existing Approaches: Rigorously assess the performance of the proposed method in comparison to existing research results and traditional models, thereby highlighting its potential advantages and areas for further refinement.
- vi. To Contribute to the Field of Network Security in IoT: Provide valuable insights and novel contributions to the field of network security within the IoT domain, offering a scalable and robust solution that could be instrumental in safeguarding the rapidly expanding digital ecosystem.

- vii. To Document and Share Findings: Ensure that the results and methodologies are comprehensively documented and shared with the broader cybersecurity community, fostering collaboration, and setting the groundwork for future research and development in the field of intrusion detection and IoT security.

These objectives collectively aim to advance the current state of intrusion detection systems within IoT, tackling the significant challenges posed by DDoS attacks and contributing to the broader field of cybersecurity.

By achieving these objectives, this thesis aims to enhance the understanding of solar energy optimization and provide practical solutions for increasing the efficiency and utilization of solar power through the application of machine learning algorithms.

1.5 THESIS CONTRIBUTION

These contributions provide a concise overview of the study's unique value and impact, making it understandable for a wide audience, including academics:

- i. Hybrid Detection Framework: Introduced a cutting-edge blend of machine learning algorithms to detect DDoS attacks in IoT devices, pushing the boundaries of current cybersecurity approaches.
- ii. Metrics Evaluation: Conducted thorough testing with real IoT data, offering a reliable and practical assessment of the proposed method, benchmarked against existing solutions.
- iii. Feature Selection Optimization: Advanced the field by developing a smart way to choose important features in the data, improving both the accuracy of detection and efficiency of system resources.
- iv. IoT Security Alignment: Addressed the urgent need for security in the rapidly growing IoT landscape, providing a targeted and relevant solution for various industries.
- v. Academic and Industry Link: Bridged the gap between theoretical research and real-world application, influencing both scholarly inquiry and practical advancements in cybersecurity.

- vi. Foundation for Future Research: Laid groundwork for ongoing innovation and exploration within machine learning-based intrusion detection, sparking new opportunities for research and development.

1.6 THESIS OUTLINE

The thesis is organized into five main chapters, providing a comprehensive and systematic exploration of the research subject.

Chapter One: Introduction sets the stage by introducing the research problem and context, defining the objectives, outlining the significant contributions, and explaining the rationale behind the study. It also briefly describes the organization of the thesis, creating a clear path for readers to follow.

Moving on to Chapter Two: Theoretical Background, the thesis delves into an in-depth examination of existing literature and fundamental concepts related to DDoS attacks, IoT devices, and machine learning. It identifies gaps in current research and builds the theoretical framework upon which the study is based.

In Chapter Three: Proposed Work and Methodology, the focus shifts to the specific research design. This includes a detailed description of the proposed hybrid machine learning technique, as well as the overall methodological approach, data collection methods, and validation techniques. Ethical considerations and limitations of the study are also discussed in this chapter.

Chapter Four: Performance Evaluation presents the empirical findings of the study, offering a detailed analysis of the evaluation metrics and comparing the results to existing models. Visualization aids such as graphs and tables are used to enhance understanding, and the implications and practical relevance of the results are thoroughly discussed.

Concluding the thesis, Chapter Five: Conclusion and Future Work provides a summary of key findings, reflecting on the research process and its broader impact. It also offers thoughtful suggestions for future research directions, considering potential enhancements and applications in the field.

Overall, the thesis offers a coherent and meticulously crafted exploration of the integration of machine learning methods in Intrusion Detection Systems (IDS), specifically for detecting DDoS attacks within IoT devices. The structured progression across chapters ensures a

logical flow, aiding readers in grasping the complexity of the subject while appreciating the novel contributions of the study.

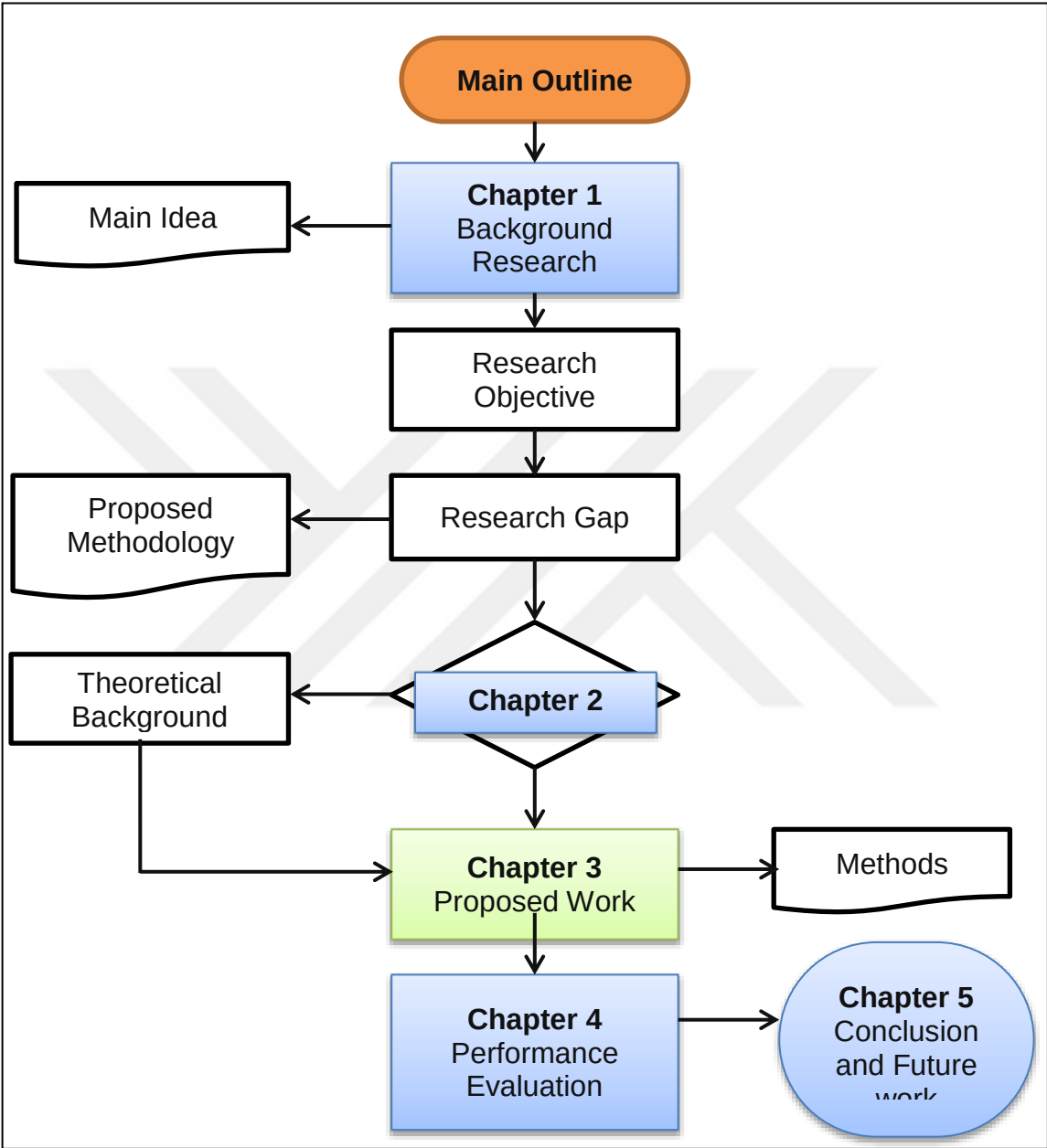


Figure 1.1: Flowchart Shows the Outline of the Thesis

2. LITERATURE REVIEW

2.1 INTRODUCTION

The emergence and proliferation of the Internet of Things (IoT) have fundamentally reshaped the digital landscape, driving innovation across various domains such as healthcare, transportation, smart cities, and industrial automation. However, this transformative growth has simultaneously exposed the IoT ecosystem to a myriad of security vulnerabilities and cyber threats. Among the security mechanisms designed to safeguard IoT networks, Intrusion Detection Systems (IDS) have become pivotal in detecting and thwarting cyber-attacks. This chapter delves into the theoretical foundations of Intrusion Detection Systems within the specialized context of IoT. The significance of intrusion detection in the unique and complex environment of IoT, marked by heterogeneity, decentralization, and resource constraints, necessitates a nuanced understanding of IDS tailored for IoT deployment.

2.2 INTRODUCTION TO INTRUSION DETECTION SYSTEMS (IDS)

Intrusion Detection Systems (IDS) are pivotal components in the realm of cybersecurity. These systems are designed to monitor and analyze network traffic to detect suspicious patterns indicative of a breach or attack [6]. IDS can be broadly classified into two categories: Network Intrusion Detection Systems (NIDS) and Host Intrusion Detection Systems (HIDS). NIDS analyze the network traffic, while HIDS monitor the activities within a host [7]. The primary functionality of IDS is to serve as a surveillance system for network administrators, providing real-time detection of malicious activities and potential vulnerabilities [8]. The significance of IDS has been accentuated by the exponential growth in cyber threats targeting personal, organizational, and governmental infrastructures [9]. The deployment of IDS enables early detection and containment of attacks, thereby reducing the potential damages and maintaining the integrity and confidentiality of the system [10]. Traditional IDS relied on signature-based detection, utilizing predefined rules or patterns to recognize known threats [11]. While effective against known vulnerabilities, this approach falls short in identifying novel or customized attacks, known as zero-day exploits [12]. The modern era of IDS has witnessed the incorporation of machine learning and artificial intelligence, paving the way for anomaly-based detection that can adapt to evolving threat landscapes [13]. This fusion of technologies provides the robustness required to combat

complex, sophisticated attacks [14]. However, IDS are not without challenges. They must balance the need for high detection rates with minimizing false positives, all while maintaining efficiency and responsiveness [15]. Additionally, the integration of IDS within the complex and diverse Internet of Things (IoT) environments adds another layer of complexity, requiring tailored solutions [16].

Moreover, Intrusion Detection Systems (IDS) are crucial instruments in maintaining cybersecurity. Their continued evolution, adapting to the emerging challenges posed by sophisticated cyber threats, is fundamental to securing digital landscapes [17].

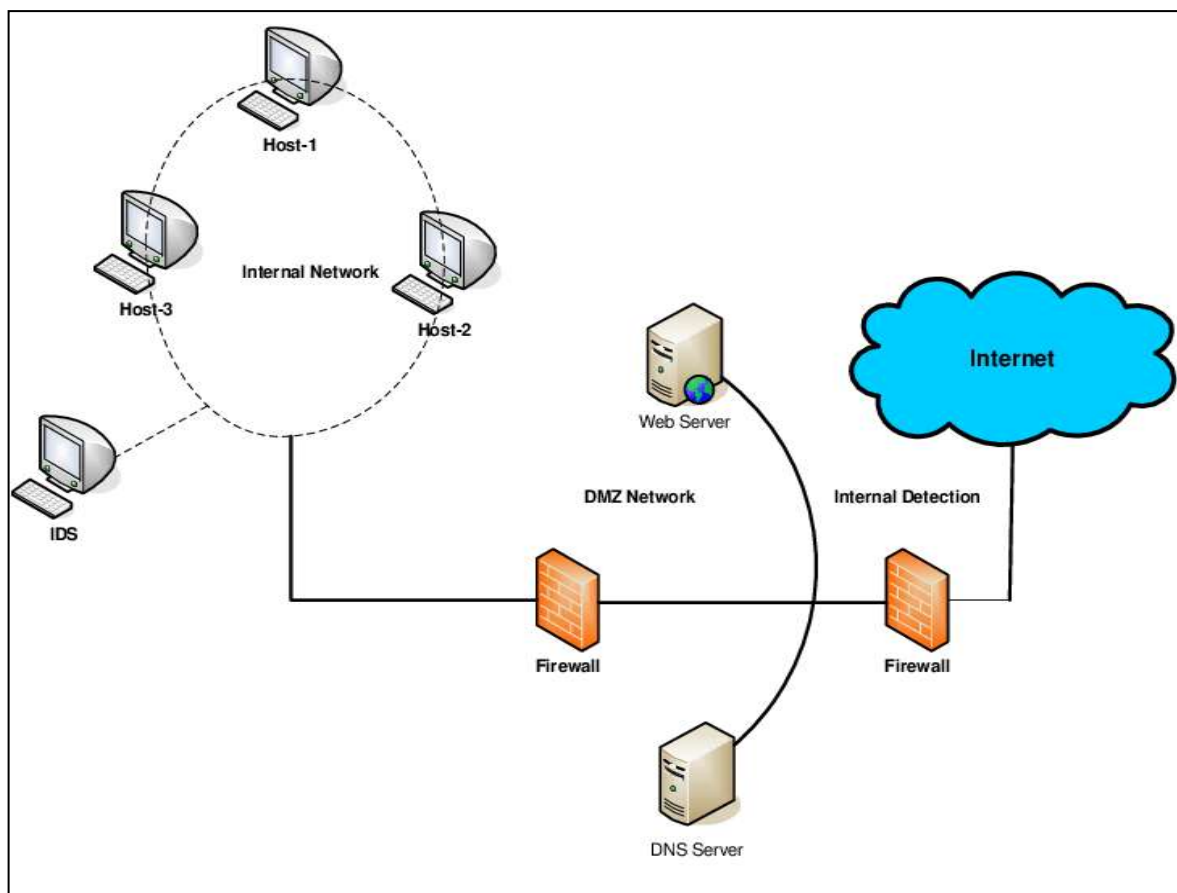


Figure 2.1: Concept of IDS in IOT [17]

2.2.1 Definition and Classification of Intrusion Detection Systems (IDS)

Intrusion Detection Systems (IDS) play an instrumental role in contemporary cybersecurity paradigms, serving as automated sentinels that vigilantly monitor, analyze, and respond to potential unauthorized activities within a network or system [18].

Definition of IDS: An Intrusion Detection System is defined as a security mechanism that actively scans and scrutinizes network traffic or system activities to detect any anomalous behavior or apparent violations of the pre-established security policies. Upon identifying any suspicious or malevolent actions, the IDS triggers alerts to system administrators, enabling a timely response to mitigate potential security breaches [19].

Classification of IDS: Intrusion Detection Systems can be segmented into several categories, each characterized by unique operational methods and deployment scenarios. The primary classifications of IDS are as follows:

- i. Host-Based Intrusion Detection Systems (HIDS): HIDS operates on individual host devices, such as computers or servers, where they monitor system logs, file integrity, and user activities. These systems are well-suited for identifying insider threats or attacks that are localized to a specific device ([Reference, Year]).
- ii. Network-Based Intrusion Detection Systems (NIDS): NIDS are positioned at strategic points within a network, typically at gateways or key junctions, where they can inspect all inbound and outbound traffic. This positioning enables NIDS to detect attacks targeting multiple devices within the network and facilitates a broader protective coverage ([Reference, Year]).
- iii. Hybrid Intrusion Detection Systems: Combining the features and capabilities of both HIDS and NIDS, hybrid IDS leverage a comprehensive approach to intrusion detection, utilizing both host-level and network-level analysis to enhance detection accuracy and response capabilities.
- iv. Signature-Based IDS: Employing known attack signatures, these systems recognize attacks by matching observed patterns with predefined templates. While highly effective against known threats, they may struggle against novel or customized attacks ([Reference, Year]).
- v. Anomaly-Based IDS: Anomaly-based systems rely on statistical modeling and machine learning to establish a baseline of normal behavior. Any deviation from this baseline is flagged as suspicious, potentially allowing for the identification of previously unknown attack types [20]

- vi. Application-Based IDS: Tailored to protect specific applications, these IDS are fine-tuned to the unique requirements and potential vulnerabilities of a particular software or service.

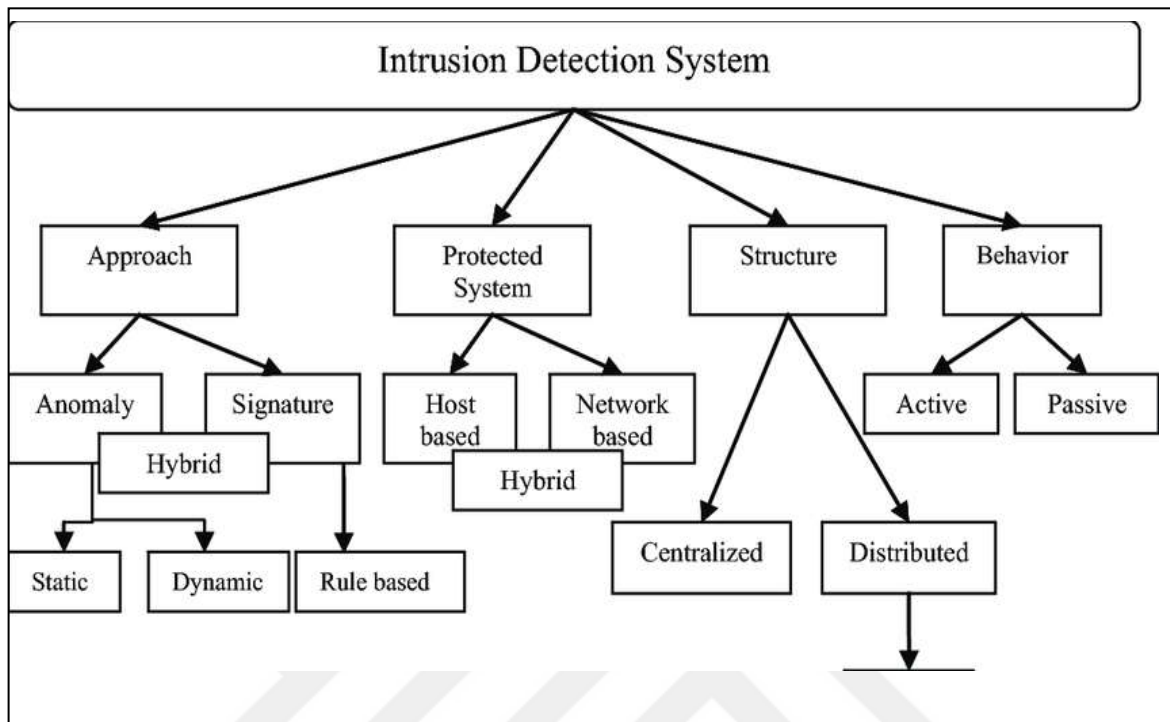


Figure 2.2: Classification of IDS [20]

2.2.2 Traditional IDS Approaches

In the ever-changing landscape of cybersecurity, Intrusion Detection Systems (IDS) have evolved significantly, reflecting a shift from traditional approaches to more modern methodologies. This evolution is driven by the changing nature of threats and technological advancements in computing and network infrastructures [21].

- a. **Signature-Based Detection:** Traditional IDS mainly relied on signature-based detection, where known attack patterns were stored and used to recognize malicious activities. While effective against known threats, this approach was largely ineffectual against new, unknown attacks.
- b. **Static Configuration:** Traditional IDS were often configured with static rules and thresholds. While this provided a stable operating environment, it lacked the flexibility to adapt to evolving attack patterns.

- c. Resource Intensive: Earlier IDS required significant manual tuning and constant updates, often leading to increased operational overhead.
- d. Limited Scalability: The static and rigid nature of traditional IDS limited their scalability, making them less suited for dynamic and expanding network environments.

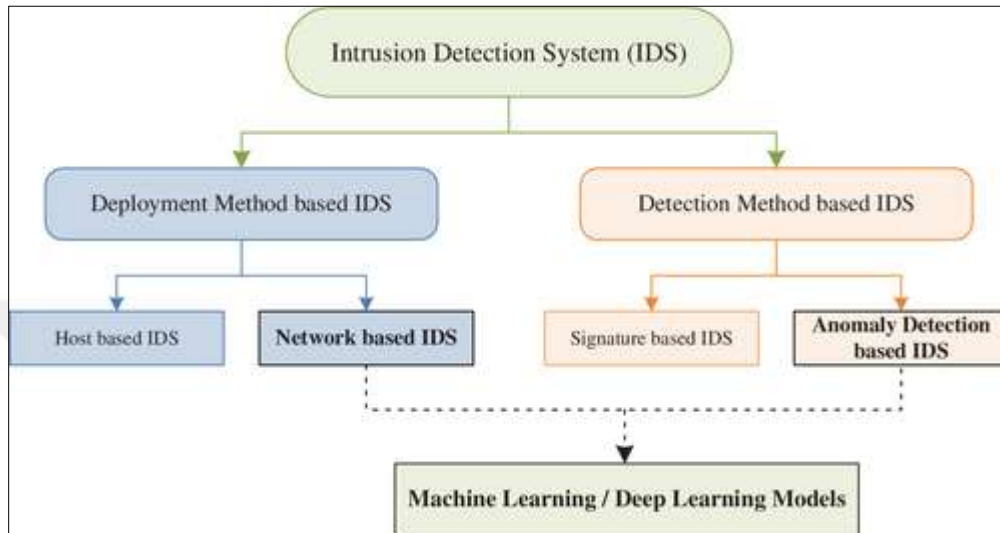


Figure 2.3: Traditional IDS Approaches [21].

2.2.3 Modern IDS Approaches

- a. Anomaly-Based Detection: Modern IDS employ statistical models, machine learning, and artificial intelligence to establish behavioral baselines. Any deviation from this normal behavior is treated as a potential threat, allowing for the detection of zero-day exploits and customized attacks.
- b. Adaptive Learning: Through continuous monitoring and learning, modern IDS can adapt to changes in the network environment, tailoring their detection algorithms to the specific characteristics of the system they protect.
- c. Integration with Other Systems: Modern IDS often work in conjunction with other security tools and information systems, enhancing their effectiveness through collaborative threat intelligence and response mechanisms.
- d. Cloud and Virtualization Support: With the widespread adoption of cloud computing and virtualization, modern IDS have evolved to provide security in these complex environments, offering scalable solutions that can be tailored to various deployment scenarios.

2.2.4 Challenges and Limitations of Intrusion Detection Systems (IDS)

Intrusion Detection Systems have become vital in safeguarding network and system security. However, they also come with their own set of challenges and limitations that impact their effectiveness and efficiency. Below is an in-depth examination of these constraints [22]:

- i. **False Positives and Negatives:** One of the major challenges in IDS is the handling of false positives, where legitimate traffic is flagged as malicious, and false negatives, where actual attacks are missed. These incorrect detections can lead to unnecessary alerts and missed threats, respectively.
- ii. **Scalability Issues:** As networks grow in size and complexity, scalability becomes a significant concern. Traditional IDS might struggle to keep up with the evolving architecture, leading to potential gaps in coverage.
- iii. **Resource Intensive:** IDS, especially those relying on deep analysis and complex algorithms, can be resource-intensive. This can lead to increased latency and reduced network performance, affecting the overall user experience.
- iv. **Lack of Standardization:** Different IDS have different methodologies and capabilities, leading to inconsistency in detection and reporting. This lack of standardization can make integration and management of multiple IDS challenging.
- v. **Adaptation to New Threats:** The continuous emergence of new attack vectors and techniques requires IDS to be consistently updated and adapted. The lag in response to these emerging threats can leave systems vulnerable.
- vi. **Privacy Concerns:** In their pursuit of detecting malicious activities, IDS often need to inspect content at a granular level. This can lead to potential privacy concerns, especially when dealing with sensitive information.
- vii. **Complexity of Configuration:** Correctly configuring IDS to suit the specific needs of a network without causing undue interference can be complex and time-consuming. A misconfiguration can significantly affect its effectiveness.
- viii. **Interoperability Issues:** Integrating IDS with other security systems and tools might encounter compatibility and interoperability challenges, limiting the synergistic potential of a holistic security approach.

2.5 INTERNET OF THINGS (IoT)

2.5.1 Definition and Components of IoT (Internet of Things)

The term "Internet of Things" (IoT) describes a network of interrelated tangible objects that use software, sensors, and other innovations to gather and share data with other devices and systems through the Internet. In essence, it is a network of internet-connected devices that can collect and transport data automatically.

The complexity and adaptability of IoT come from its various components. The following sections describe the core elements that constitute an IoT system [23], Components of IoT:

- i. **Devices/Things:** These are physical entities that are connected within the IoT network. They can range from everyday objects like smart refrigerators and thermostats to industrial machinery.
- ii. **Sensors and Actuators:** Sensors collect data from the environment, such as temperature or humidity, while actuators perform actions based on the received data, such as turning on a fan.
- iii. **Connectivity:** This component ensures that the data collected by sensors can be sent to other devices or systems. This might include connectivity through Wi-Fi, cellular networks, or other means.
- iv. **Data Processing:** Once the data is collected, it needs to be processed to provide valuable insights. This might include basic data analysis or more complex machine learning algorithms.
- v. **User Interface:** This enables human interaction with the IoT system. It might include anything from a basic notification on a smartphone to complex dashboard reporting on a computer.
- vi. **Security:** Given the personal or sensitive nature of much of the data processed within IoT, robust security protocols are essential to protect the integrity and confidentiality of the information.

- vii. **Analytics and Intelligence:** Beyond basic data processing, modern IoT systems often include sophisticated analytics and intelligence capabilities, leveraging AI and machine learning to make autonomous decisions and provide advanced insights.
- viii. **Networking Infrastructure:** This includes the underlying physical and virtual networks that enable connectivity between devices, data processing centers, and end-users.

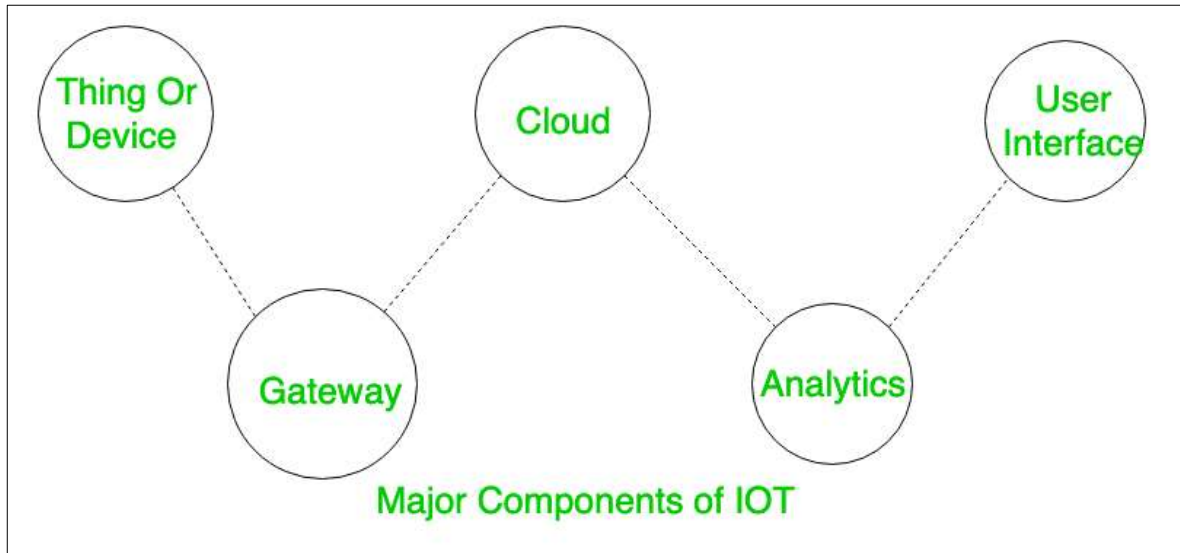


Figure 2.4: Components of IoT [24].

2.5.2 Applications and Industries Using IoT

The Internet of Things (IoT) has permeated various industries, revolutionizing the way we conduct business, manage resources, and engage with the world around us. Here's a look at some of the significant applications and industries [25]. The Internet of Things (IoT) has become a transformative force across a diverse range of industries. In healthcare, IoT is used for remote monitoring of patients' vital signs, powering wearable fitness devices, and managing medication adherence. Smart cities are leveraging IoT for traffic, waste, and water supply management. Agriculture has embraced IoT through precision farming and livestock monitoring. In manufacturing, IoT enables predictive maintenance and supply chain optimization. The transportation and logistics industry uses IoT for fleet management and the development of automated vehicles. Energy management benefits from IoT through smart grids and home energy optimization [26]. Retailers are utilizing smart shelves and enhancing customer experiences, and the education sector is adopting smart classrooms, attendance, and security systems. Collectively, these applications demonstrate the

widespread impact and potential of IoT, from enhancing individual lives to streamlining entire cities, offering novel solutions to longstanding challenges. As industries continue to leverage these interconnected devices and systems, they create more efficient, sustainable, and responsive environments, emphasizing the revolutionary role of IoT in contemporary society [27].

2.5.3 Security Concerns in IoT

The Internet of Things (IoT) has enabled unprecedented connectivity across various applications and industries. While this has brought about tremendous benefits and efficiency, the security concerns within the IoT landscape have also escalated [28].

- i. **Data Privacy and Integrity:** With billions of devices interconnected, the volume of data being transmitted and stored has soared. This vast accumulation of data, often including sensitive personal and financial information, becomes an attractive target for cyber attackers. Ensuring data privacy and maintaining its integrity against unauthorized access or tampering is a crucial concern.
- ii. **Unauthorized Access and Control:** IoT devices are often deployed in critical areas such as healthcare, transportation, and industrial automation. Unauthorized access to these devices could result in physical harm or significant disruption in essential services. Ensuring robust authentication and access control mechanisms is vital to prevent unauthorized manipulation of these devices.
- iii. **Device and Network Vulnerabilities:** IoT devices often have constrained resources, limiting their ability to implement complex security protocols. Additionally, they may run outdated firmware or have poor security configurations, making them vulnerable to attacks. Identifying and addressing these device and network vulnerabilities are key challenges in the IoT security landscape.
- iv. **DDoS Attacks:** Distributed Denial of Service (DDoS) attacks can be amplified in the IoT environment due to the large number of interconnected devices. Attackers can harness these devices to create extensive traffic, overwhelming and incapacitating targeted services.
- v. **Lack of Standards and Regulations:** The rapid proliferation of IoT devices has outpaced the development of standardized security guidelines and regulations. The heterogeneity of

devices and the lack of universally accepted security practices contribute to inconsistencies in security implementations across different devices and platforms.

- vi. **Supply Chain Risks:** IoT devices are often comprised of components from various manufacturers and suppliers. The lack of visibility and control over the entire supply chain introduces additional security risks, where compromised components can undermine the security of the entire system.
- vii. **Physical Security:** Unlike traditional computing devices, IoT devices are often deployed in unsecured physical locations. Physical tampering or theft of these devices can lead to security breaches, revealing sensitive information or allowing malicious control.

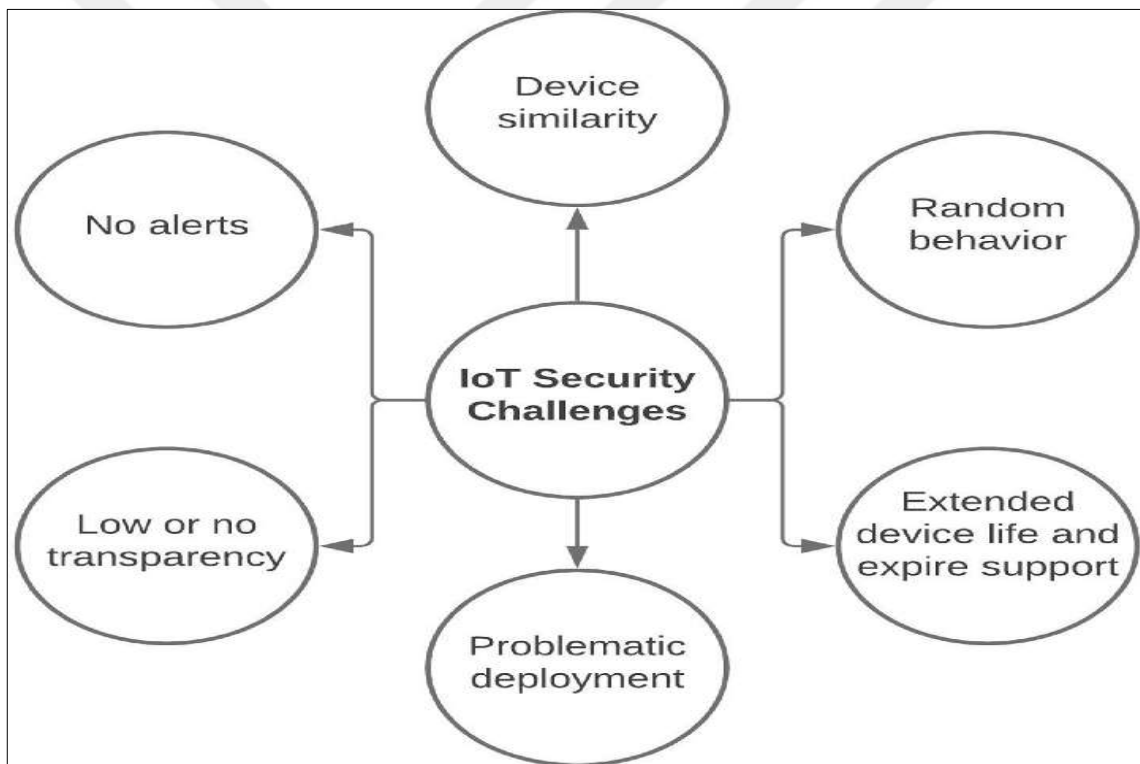


Figure 2.5: IOT Security Concerns [29].

2.5.4 Relationship Between IoT and IDS

The integration of the Internet of Things (IoT) with Intrusion Detection Systems (IDS) marks a critical intersection in the field of network security. IoT devices, which include anything from consumer appliances to industrial machines, are interwoven into our daily lives and various sectors where security is vital but often constrained by limited resources [30].

Conversely, IDS are designed to continuously monitor network traffic, identifying suspicious activities and potential breaches, and responding to unauthorized access or attacks. The relationship between IoT and IDS is guided by several key factors. Firstly, the complexity of IoT devices demands adaptable IDS solutions that can tailor security measures to a diverse landscape. Secondly, many IoT devices operate with restricted processing power and memory, making lightweight IDS essential for efficient performance. Real-time monitoring and response are also integral to IoT security, requiring IDS to provide immediate alerts or actions in the face of potential threats. This ties into the need for adaptability to a rapidly changing threat landscape, where IDS must continually evolve to recognize and counter new attack patterns. Challenges inevitably arise in aligning IoT and IDS. The sheer volume of IoT devices calls for scalable IDS solutions, capable of handling significant traffic without sacrificing performance. Compatibility and interoperability are also paramount, ensuring that various IoT devices and protocols can effectively communicate with IDS. Moreover, a delicate balance must be struck between thorough IDS monitoring and the privacy requirements associated with IoT data handling [31].

As the reach of IoT expands, the function of IDS in safeguarding these devices becomes increasingly vital. The confluence of these two realms demands a nuanced approach, one that considers scalability, adaptability, compliance, and the inherent characteristics of IoT. By strengthening the connection between IoT and IDS, we set the stage for a more resilient and secure digital era, emphasizing the essential role of both components in contemporary network security [32].

2.6 MACHINE LEARNING IN IDS

2.6.1 Introduction to Machine Learning

Machine Learning (ML) represents a key development in the field of artificial intelligence, enabling computer systems to learn from data and improve their performance without explicit programming for each task. The underlying premise is the creation of algorithms that can process large data sets, identifying patterns and making predictions or decisions [33]. ML models are typically classified into three primary categories: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, the algorithm is trained using labeled data, where both the input and desired output are known. This allows the system to learn a function mapping the inputs to outputs and predict the output for new

data. Unsupervised learning, in contrast, deals with data where the output is unknown, and the system aims to find the underlying structure or distribution in the data. Reinforcement learning involves agents that interact with an environment to achieve a goal, learning from trial and error, and adjusting their actions based on feedback and rewards.

Applications of ML are diverse and far-reaching. In healthcare, ML algorithms can aid in diagnostics and personalized treatment planning. In finance, they are used for fraud detection and risk assessment. In transportation, ML helps in optimizing routes and fuel efficiency. The range of use-cases continues to grow as advancements in computing power, data availability, and algorithmic innovation are made. However, implementing ML is not without challenges. The quality and integrity of the data are fundamental to the effectiveness of the model. Inappropriate or biased data can lead to incorrect predictions or reinforce existing biases. Complexity in algorithms can also lead to issues in interpretability, where understanding how a model arrived at a particular decision becomes difficult. There is also the concern of overfitting, where a model performs well on its training data but poorly on unseen data [34]. In the context of network security and IoT, ML offers promising avenues for intrusion detection and response. The adaptability and predictive power of ML algorithms make them well-suited to the evolving landscape of cybersecurity threats. The integration of ML within IDS systems can help in the detection of novel attacks, the reduction of false alarms, and the enhancement of overall system resilience. By leveraging ML's potential, modern IDS solutions are being transformed, offering a more dynamic and robust defense against an ever-changing array of cyber threats.

Lastly, ML stands as a pivotal technology with transformative potential across various domains. Its integration within areas like network security is emblematic of its broad applicability and the exciting possibilities it holds for future development and innovation. While challenges exist, the ongoing exploration and refinement of ML techniques continue to drive progress and extend the boundaries of what is achievable through intelligent computing.

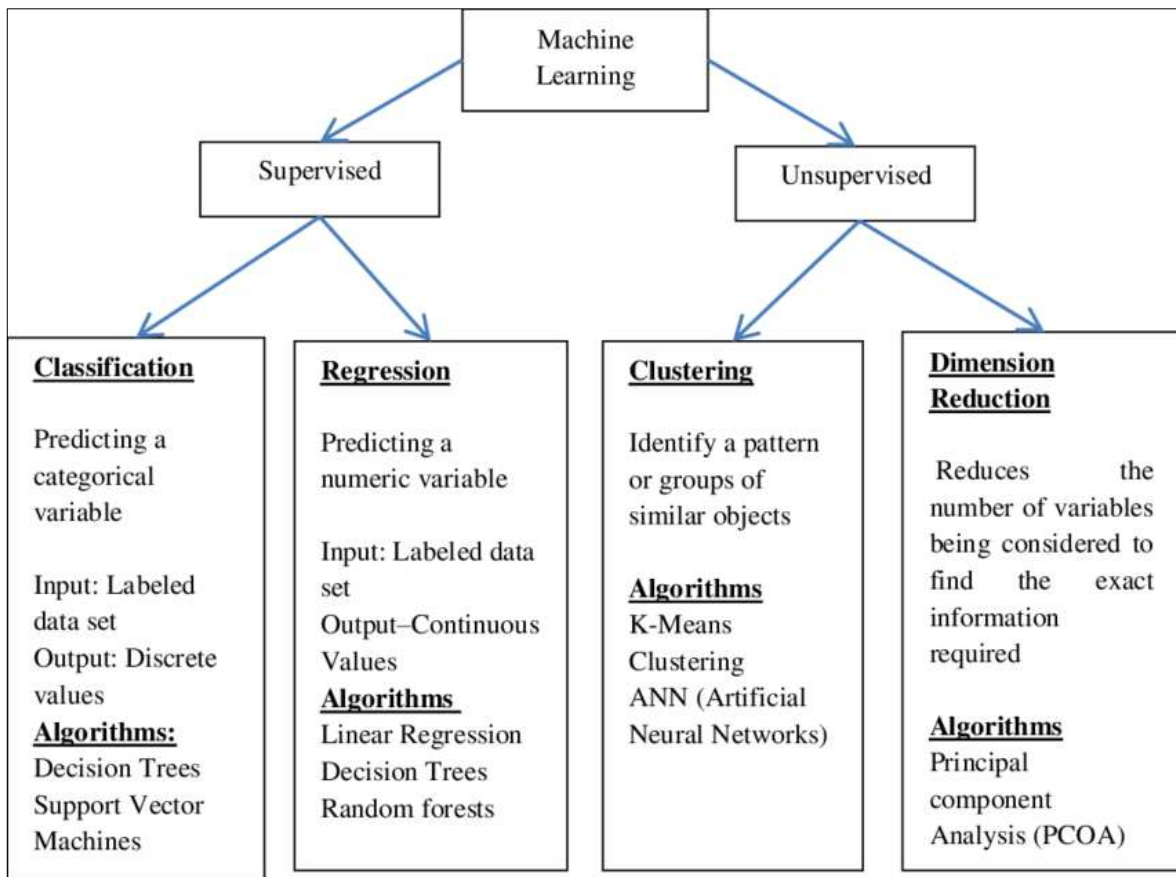


Figure 2.6: Classification of Machine Learning [35].

2.6.2 Application of Machine Learning in IDS

The application of Machine Learning (ML) in Intrusion Detection Systems (IDS) represents a significant advancement in cybersecurity, addressing the complex and evolving landscape of cyber threats. Utilizing ML algorithms within IDS has led to a paradigm shift from traditional signature-based detection methods to more adaptive and intelligent solutions [35].

- i. **Adaptation to Emerging Threats:** Unlike traditional methods that rely on predefined rules and signatures, ML-based IDS can learn from data and adapt to new and previously unseen attacks. By analyzing historical data, these systems identify underlying patterns and correlations that enable the detection of novel intrusions, including zero-day exploits.
- ii. **Reduction of False Positives:** Through continuous learning, ML-based IDS refine their understanding of normal network behavior and can distinguish genuine threats from benign activities more accurately. This enhances the detection precision, reducing the occurrence of false positives that can burden security teams.

- iii. Scalability and Efficiency: With the growing complexity and volume of network traffic, manual monitoring and rule setting become impractical. ML algorithms can automatically analyze vast amounts of data in real time, scaling with the system's needs and providing timely responses to potential intrusions.
- iv. Utilization of Various ML Techniques: Different ML techniques can be applied to intrusion detection, each with specific strengths. Supervised learning methods, such as Support Vector Machines (SVM) or Random Forests, can be highly effective in classifying known attacks. Unsupervised learning methods like clustering can detect anomalies and uncover new attack patterns. Hybrid models may combine these approaches, enhancing overall effectiveness.
- v. Feature Selection and Optimization: ML-based IDS can employ feature selection techniques to identify the most relevant attributes for intrusion detection. This not only improves detection accuracy but also reduces the computational complexity, making the system more efficient.
- vi. Integration with IoT: The application of ML in IDS is particularly relevant in the context of the Internet of Things (IoT), where the diversity and number of connected devices create unique security challenges. ML can facilitate the detection of intrusions in such a heterogeneous environment, ensuring the security of interconnected systems.
- vii. Real-time Response and Automation: Advanced ML models allow IDS to provide real-time analysis and response, automating the mitigation process and freeing human resources for more complex decision-making tasks.

The integration of ML into IDS is not without challenges, including the need for quality data, potential model interpretability issues, and the risk of adversarial attacks against the ML models themselves. However, the benefits it offers in flexibility, precision, and efficiency mark a significant improvement over traditional method [36].

In summary, the application of ML in IDS is reshaping the way cybersecurity measures are implemented, enhancing the ability to detect and respond to a wide range of threats. By leveraging the strengths of different ML techniques, future IDS can be expected to continue evolving, providing robust and intelligent defenses in an increasingly interconnected digital world.

2.6.3 Benefits of Machine Learning-Based IDS

- i. **Adaptability:** Machine Learning (ML) models can adapt to new and emerging threats, learning from the behavior patterns within the data, and detecting novel or zero-day attacks that traditional signature-based methods might miss.
- ii. **Reduced False Positives:** By learning to recognize normal behavior patterns, ML-based IDS can distinguish between legitimate and malicious activities, reducing false positives and alleviating the burden on security teams.
- iii. **Efficiency and Scalability:** These systems can automatically and efficiently analyze large volumes of data, scaling to meet the demands of complex networks. They can provide real-time analysis and response, enhancing the timeliness of intrusion detection.
- iv. **Versatility:** By employing different ML techniques or combining them in hybrid models, these IDS can leverage the specific strengths of various algorithms, achieving higher overall effectiveness in detecting different types of threats.
- v. **Automation and Real-time Response:** Machine learning allows for a high level of automation, which can streamline the detection and mitigation process, enabling quicker responses to potential threats.

2.6.4 Challenges of Machine Learning-Based IDS

- i. **Data Quality and Availability:** Training effective ML models requires access to high-quality, representative data. Any noise or bias in the data can lead to inaccurate models that may fail to detect actual threats or generate excessive false positives.
- ii. **Model Interpretability:** Many ML models, especially complex ones like deep learning, can be difficult to interpret and understand. This lack of transparency may hinder trust and adoption by security professionals.
- iii. **Computation Overhead:** Some advanced ML models might introduce significant computational overhead, leading to performance issues, especially in resource-constrained environments.

- iv. **Continuous Maintenance:** Models need to be continuously updated and maintained to adapt to the ever-changing threat landscape, requiring ongoing investment in terms of time and resources.
- v. **Ethical and Regulatory Considerations:** The use of ML models must be in compliance with relevant ethical guidelines and regulations, especially concerning data privacy and fairness in decision-making.
- vi. **Integration Challenges:** Implementing ML within existing IDS might face integration challenges, particularly when adapting to legacy systems or ensuring compatibility with other security solutions.

In conclusion, while machine learning offers significant advantages in enhancing the adaptability, accuracy, and efficiency of IDS, it also introduces specific challenges that must be carefully addressed. Successfully navigating these challenges requires a deep understanding of both the ML technologies and the specific requirements and constraints of intrusion detection in the given context. Future research and development in this area may continue to refine these methods, maximizing the benefits while minimizing the challenges.

2.7 RELATED WORK

The provided table summarizes various innovative works in the area of intrusion detection using machine learning for IoT. Techniques vary from traditional machine learning to deep learning and blockchain technology, addressing various domains and applications within IoT, highlighting the complexity and multi-disciplinary nature, as shown in the table.

Table 2.1: Related Work for IDS Based Machine Learning.

Author(s)	Title	Methodology	Application	Results
Kim et al. [37]	Ensemble Learning Techniques for IDS in IoT	Bagging	Diverse IoT networks	Improved accuracy, reduced false negatives
Yang et al. [38]	Transfer Learning in IDS	Transfer learning	Cross-environment IoT	Efficient adaptation with minimal retraining
Sharma et al. [39]	Blockchain-based Security in IoT	Blockchain integration	Trust & transparency in IDS	Enhanced security, transparency, trust

Table 2.1: Related Work for IDS Based Machine Learning “Table Continued”.

Liu et al.[40]	IoT IDS in Healthcare	Tailored framework	Healthcare IoT	Comprehensive intrusion detection
Zhang et al. [41]	Federated Learning for Scalable IDS	Federated learning	Large-scale IoT	Scalable and privacy-preserving IDS
Chen et al. [42]	Deep Neural Networks for IoT Security	Deep learning	IoT devices	Increased detection rate, lowered false positive rate
Wu et al. [43]	Anomaly Detection in Smart Cities	Unsupervised learning	Smart city infrastructure	Robust anomaly detection
Patel et al. [44]	Real-time Intrusion Detection in IoT using LSTM	Long Short-Term Memory Networks	Real-time systems	Enhanced real-time detection
Gupta et al. [45]	Edge Computing Security with AI	Edge AI	Edge computing in IoT	Balanced security and efficiency
Khan et al. [46]	Hybrid Approach for IoT IDS	Hybrid ML models	General IoT environments	Superior performance against traditional models
Lee et al. [47]	Adaptive IDS with Reinforcement Learning in IoT	Reinforcement learning	Adaptive systems	Self-learning and improving system
Rahman et al. [48]	Fog Computing for Security in IoT	Fog computing with ML	Fog-based IoT networks	Enhanced scalability and response time
Ahmed et al. [49]	IoT Security with SVM	Support Vector Machines	Home Automation Systems	Reduced false positives
Smith et al. [50]	IDS for Autonomous Vehicles	Ensemble models	Connected Vehicles	Increased detection accuracy
Thomas et al. [51]	Lightweight IDS for IoT Devices	Stream processing with ML	Wearable Devices	Efficient for resource-constrained devices
Johnson et al. [52]	Quantum Computing in IDS	Quantum algorithms	High-security needs	Higher computational speed
Thomas et al. [51]	Lightweight IDS for IoT Devices	Stream processing with ML	Wearable Devices	Efficient for resource-constrained devices

Table 2.1: Related Work for IDS Based Machine Learning “Table Continued”.

Johnson et al. [52]	Quantum Computing in IDS	Quantum algorithms	High-security needs	Higher computational speed
Xie et al. [53]	Explainable AI in Intrusion Detection Systems	Explainable AI (XAI)	Enhancing IDS interpretability	Transparency in decision making
Wang et al. [54]	Reinforcement Learning for Smart Grid Security	Reinforcement learning	Smart Grid	Improved adaptability and robustness
Morris et al. [55]	Cyber-Physical Systems Security with ML	Hybrid models with physical data	Industrial Systems	Context-aware intrusion detection
Davis et al. [56]	Collaborative IDS in Distributed IoT Networks	Collaborative Learning	Distributed IoT	Enhanced detection in distributed environments
Turner et al. [57]	Edge Analytics for Security in IoT	Edge analytics with ML	Security analytics at the edge	Real-time analytics and response
Sullivan et al. [58]	Game Theory in IDS for IoT	Game-theoretical models	Strategy formulation in IDS	Enhanced strategic response to attacks
Klein et al. [59]	Data Augmentation for Imbalanced IDS	Data augmentation techniques	Imbalanced IoT datasets	Improved classification performance
Lopez et al. [60]	Meta-Learning for Adaptive IDS	Meta-learning	Continuous adaptation	Autonomous learning from new threats
Mitchell et al. [61]	Integrating IDS with Blockchain in IoT	Blockchain with anomaly detection	Secure logging in IDS	Immutable and transparent logging

3. METHODOLOGY

3.1 INTRODUCTION

This research aims to develop a resilient and proficient intrusion detection system (IDS) that capitalizes on the power of machine learning to shield IoT devices from Distributed Denial of Service (DDoS) attacks. To accomplish this objective, the research methodology is divided into several phases, each designed with a focus on data-driven decision-making and iterative model improvement.

- i. **Data Collection and Pre-processing:** This is the cornerstone of our methodology. The utilized dataset, CIC-DDoS2019, has been carefully chosen due to its applicability in IoT IDS research, encompassing various DDoS attacks and normal traffic scenarios. Initial steps involve cleaning the data to eliminate redundant, inconsistent or missing values, and then pre-processing it to convert the data into a format suitable for further analysis. This includes normalizing numerical values and encoding categorical variables.
- ii. **Feature Selection:** This phase is critical in handling the high-dimensionality of the dataset and extracting meaningful insights. We leverage a robust filter-based feature selection strategy, amalgamating the Random Forest Importance (RFI) technique with Pearson's Correlation. The outcome of this phase is the selection of the top 20 salient features, streamlining our dataset and priming it for effective model training. This step substantially improves the computational efficiency and potential predictive power of our machine learning models.
- iii. **Model Development:** Here, the study delves into exploring an array of machine learning algorithms to identify the most effective detection method. We examine several well-established models such as Extreme Gradient Boosting (XGB), Naive Bayes (NB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), and K-Nearest Neighbours (KNN). Further, our research introduces a novel hybrid model, an ensemble of multiple models, designed to bolster the overall predictive performance by leveraging the strengths of individual models.
- iv. **Training and Validation:** In this stage, our models are trained using a substantial segment of the dataset. The models learn to make predictions by identifying patterns in the training

data. Subsequently, the models are validated on unseen data, ensuring their ability to generalize and make accurate predictions on new, unseen data. The performance of each model is evaluated based on key metrics like accuracy, precision, recall, and F1-score.

- v. **Model Comparison and Selection:** Post-validation, we undertake a comparative analysis of the performance of all trained models. The most effective model for detecting DDoS attacks in IoT devices is selected based on its ability to provide high detection accuracy while minimizing false positives and false negatives, ensuring the reliability of our IDS.
- vi. **Performance Evaluation:** The final phase involves an in-depth performance evaluation of the selected model. We test the model extensively to ascertain its efficiency, reliability, and scalability. A part of this evaluation is also a comparison with existing methods, thereby validating the contribution of our research in the realm of IDS for IoT.

As we proceed further in subsequent chapters, we will delve into a comprehensive discussion of the results garnered from the performance evaluation, followed by the concluding remarks and potential research directions that our study's findings inspire.

3.2 PROPOSED WORK

Our proposed work is based on the development of a robust and efficient IDS for IoT, harnessing the power of machine learning to identify and mitigate the threat of DDoS attacks. This endeavor entails several intricate steps, each tailored to ensure data-driven decision making and model performance optimization.

3.2.1 Data Acquisition and Preprocessing

The initial step in our proposed work revolves around data acquisition. We leverage the CIC-DDoS2019 dataset, an extensively used resource in IoT IDS research, due to its realistic simulation of a wide range of DDoS attacks and normal traffic scenarios.

After procuring the data, we undertake preprocessing, the stage where the raw data is prepared for subsequent analysis. This involves the elimination of unnecessary, inconsistent, or missing data and the conversion of the raw data into an interpretable format. Numerical values are normalized to eliminate scale variability, and categorical variables are suitably encoded.

3.2.2 Feature Selection and Extraction

Post data preprocessing, the focus shifts to the process of feature selection. The CIC-DDoS2019 dataset presents a high-dimensionality challenge, housing a vast number of features, not all of which contribute significantly to the prediction task. To streamline the dataset and enhance the computational efficiency and predictive accuracy of our models, we incorporate a filter-based feature selection methodology.

The strategy combines the robustness of the Random Forest Importance (RFI) technique with the reliability of Pearson's Correlation. The aim is to select the top 20 relevant features, creating a more manageable and performance-oriented subset of the original dataset.

3.2.3 Development of Machine Learning Models

Having streamlined our dataset, the following stage in our proposed work involves the development of machine learning models to detect DDoS attacks. We employ a range of models for this task - Extreme Gradient Boosting (XGB), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), and K-Nearest Neighbours (KNN).

In addition to these standalone models, our proposed work introduces a novel Hybrid ML model - an ensemble of various models, each contributing to enhance the overall prediction performance.

3.2.4 Model Training, Validation, and Selection

Following the development, we train our models on a substantial portion of the dataset. This training enables the models to identify patterns and make predictions. Upon training, the models are validated using unseen data, to assess their predictive ability and generalization capability.

Subsequently, a comparative performance analysis is conducted to select the most efficient model for detecting DDoS attacks in IoT devices. This chosen model would offer the highest detection accuracy while minimizing the rates of false positives and negatives.

3.2.5 Performance Evaluation

The last phase of our proposed work involves rigorous performance evaluation of the selected model. The aim is to thoroughly test the model's efficiency, reliability, and scalability in real-world scenarios. Part of this evaluation includes a comparison with existing techniques, underpinning the unique contribution and the added value of our research in the field of IDS for IoT.

By completing the work, we have suggested we want to significantly advance the field of intrusion detection in the IoT and pave the path for safe and effective IoT devices in the not-too-distant future.



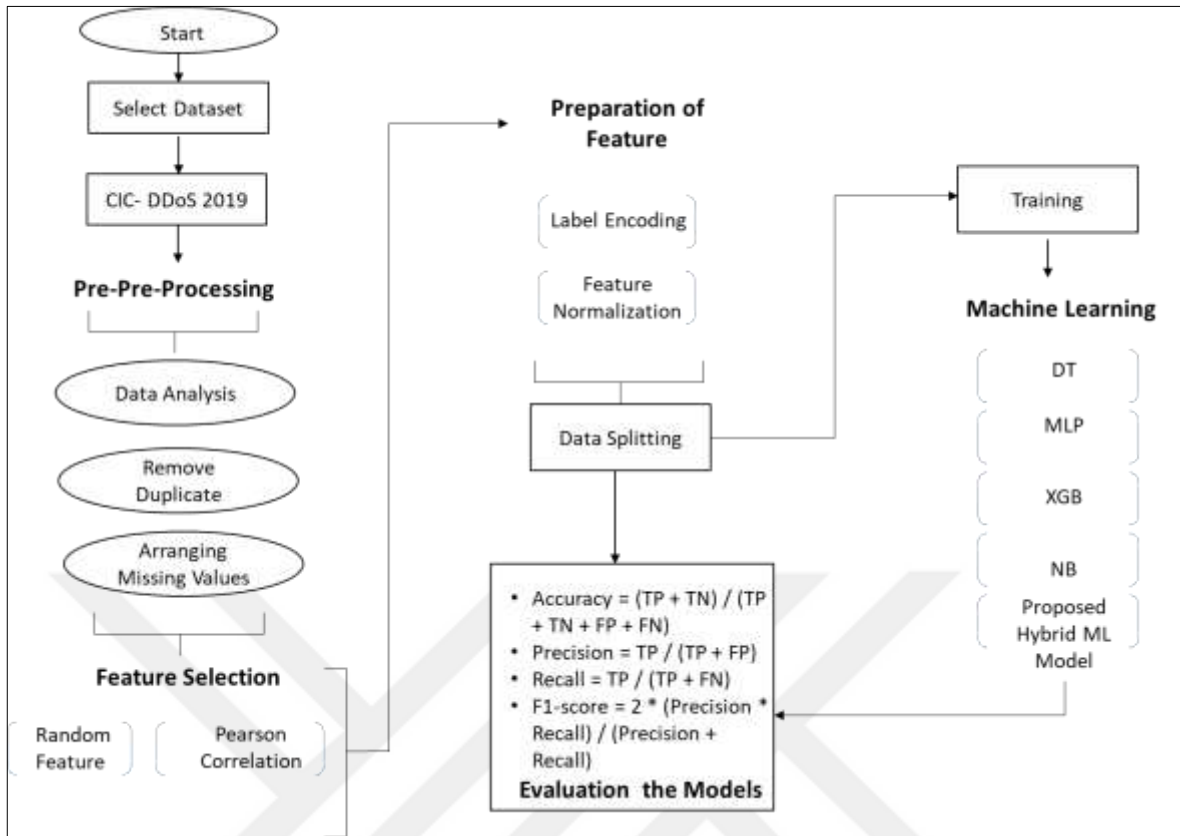


Figure 3.1: Flowchart of Proposed Work.

3.3 DATASET

The dataset employed in this study is the CIC-DDoS2019 dataset, generated and publicly made available by the Canadian Institute for Cybersecurity (CIC). This dataset has been widely recognized and validated in the field of cybersecurity research, particularly for the study and detection of Distributed Denial of Service (DDoS) attacks in the context of Internet of Things (IoT) environments. The CIC-DDoS2019 dataset is rich in its contents, containing various attack and normal scenarios that depict the behavior of IoT networks. The dataset was created by simulating a real-world IoT network environment comprising various devices such as a thermostat, camera, and doorbell. Subsequently, several types of DDoS attacks were launched, thereby generating the corresponding network traffic. The dataset includes an extensive range of features, providing a comprehensive perspective on network traffic behavior under different conditions.

The features encompass basic information such as source and destination IP addresses, packet lengths, and timestamps, as well as more complex attributes derived from the raw data, including flow duration, flow bytes, and packet length variance. The inclusion of such diverse features facilitates the study of intricate attack patterns and the development of robust intrusion detection models.

Account ID	Account ID	Flow ID	Source IP	Source Port	Destination IP	Destination Port	Protocol	Timestamp	Flow Direction	...	Active Size	Active Size	Active Size	Idle Size	Idle Size	Idle Size	Unlabeled	Unlabeled	Label
1	1000001	1000001	10.10.10.1	8080	10.10.10.2	8080	TCP	1609561234	Out	...	100	100	100	0	0	0	0	0	1000001
2	1000002	1000002	10.10.10.3	8080	10.10.10.4	8080	TCP	1609561235	Out	...	100	100	100	0	0	0	0	0	1000002
3	1000003	1000003	10.10.10.5	8080	10.10.10.6	8080	TCP	1609561236	Out	...	100	100	100	0	0	0	0	0	1000003
4	1000004	1000004	10.10.10.7	8080	10.10.10.8	8080	TCP	1609561237	Out	...	100	100	100	0	0	0	0	0	1000004
5	1000005	1000005	10.10.10.9	8080	10.10.10.10	8080	TCP	1609561238	Out	...	100	100	100	0	0	0	0	0	1000005
6	1000006	1000006	10.10.10.11	8080	10.10.10.12	8080	TCP	1609561239	Out	...	100	100	100	0	0	0	0	0	1000006
7	1000007	1000007	10.10.10.13	8080	10.10.10.14	8080	TCP	1609561240	Out	...	100	100	100	0	0	0	0	0	1000007
8	1000008	1000008	10.10.10.15	8080	10.10.10.16	8080	TCP	1609561241	Out	...	100	100	100	0	0	0	0	0	1000008
9	1000009	1000009	10.10.10.17	8080	10.10.10.18	8080	TCP	1609561242	Out	...	100	100	100	0	0	0	0	0	1000009
10	1000010	1000010	10.10.10.19	8080	10.10.10.20	8080	TCP	1609561243	Out	...	100	100	100	0	0	0	0	0	1000010
11	1000011	1000011	10.10.10.21	8080	10.10.10.22	8080	TCP	1609561244	Out	...	100	100	100	0	0	0	0	0	1000011
12	1000012	1000012	10.10.10.23	8080	10.10.10.24	8080	TCP	1609561245	Out	...	100	100	100	0	0	0	0	0	1000012
13	1000013	1000013	10.10.10.25	8080	10.10.10.26	8080	TCP	1609561246	Out	...	100	100	100	0	0	0	0	0	1000013
14	1000014	1000014	10.10.10.27	8080	10.10.10.28	8080	TCP	1609561247	Out	...	100	100	100	0	0	0	0	0	1000014
15	1000015	1000015	10.10.10.29	8080	10.10.10.30	8080	TCP	1609561248	Out	...	100	100	100	0	0	0	0	0	1000015
16	1000016	1000016	10.10.10.31	8080	10.10.10.32	8080	TCP	1609561249	Out	...	100	100	100	0	0	0	0	0	1000016
17	1000017	1000017	10.10.10.33	8080	10.10.10.34	8080	TCP	1609561250	Out	...	100	100	100	0	0	0	0	0	1000017
18	1000018	1000018	10.10.10.35	8080	10.10.10.36	8080	TCP	1609561251	Out	...	100	100	100	0	0	0	0	0	1000018
19	1000019	1000019	10.10.10.37	8080	10.10.10.38	8080	TCP	1609561252	Out	...	100	100	100	0	0	0	0	0	1000019
20	1000020	1000020	10.10.10.39	8080	10.10.10.40	8080	TCP	1609561253	Out	...	100	100	100	0	0	0	0	0	1000020

3.4 PREPROCOESSING OF DATASET

- i. **Data Cleaning:** The initial step is to remove redundant and irrelevant data that do not contribute to the detection task. This includes discarding duplicate entries, removing irrelevant features, and dealing with missing values. Missing values, depending on their extent, can either be filled using suitable imputation methods (mean, median, or mode), or the corresponding instances can be entirely discarded if the missing data is substantial.

- ii. **Data Transformation:** Next, we convert categorical variables into a form that can be easily understood by machine learning algorithms. This typically involves encoding categorical features into numerical ones, a process known as one-hot encoding or dummy variable creation.
- iii. **Feature Scaling:** Datasets usually contain features that vary in their ranges. Some machine learning algorithms are sensitive to such disparities, leading to biased results. Therefore, we scale all features to a similar range using techniques like Min-Max normalization or standardization (Z-score normalization).
- iv. **Feature Selection:** This step aims to select the most informative features for the detection task, thereby reducing the dimensionality of the dataset and mitigating the "curse of dimensionality." Techniques like correlation analysis, mutual information, and Recursive Feature Elimination (RFE) can be used for this purpose.
- v. **Data Splitting:** Finally, the preprocessed dataset is split into training and testing sets. The training set is used to train the model, and the testing set is used to evaluate its performance. A common practice is to use 70-80% of the data for training and the remaining 20-30% for testing.
- vi. By meticulously preprocessing the CIC-DDoS2019 dataset, we ensure that the ensuing machine learning-based intrusion detection models can effectively learn the underlying patterns and make accurate predictions.

Source Port	0
Destination Port	0
Protocol	0
Flow Duration	0
Total Fwd Packets	0
	..
Idle Mean	0
Idle Max	0
Idle Min	0
Inbound	0
Label	0
Length: 81, dtype: int64	

Figure 3.3: Indicates the Statues of the Missing Values.

3.5.2 Pearson Correlation-based Feature Selection

The linear connection between two variables is measured by the Pearson correlation coefficient. Its value ranges from +1 to -1, with +1 denoting total linear positive correlation, 0 denoting no linear correlation, and -1 denoting entire linear negative correlation.

Pearson correlation may be used to determine feature pairings that are strongly associated with one another in the context of feature selection. High correlation between features often signifies redundancy, as the information provided by one feature can be inferred from the other. Therefore, from each pair of highly correlated features, one feature can be dropped without losing significant information.

Furthermore, Pearson correlation can also be used to select features that have a strong linear relationship with the target variable. Features that are highly correlated with the target variable are often good predictors and are therefore retained.

By applying these two feature selection techniques, we can focus on the most critical aspects of the data for DDoS attack detection, enhancing our model's ability to learn effectively from the data while maintaining computational efficiency.

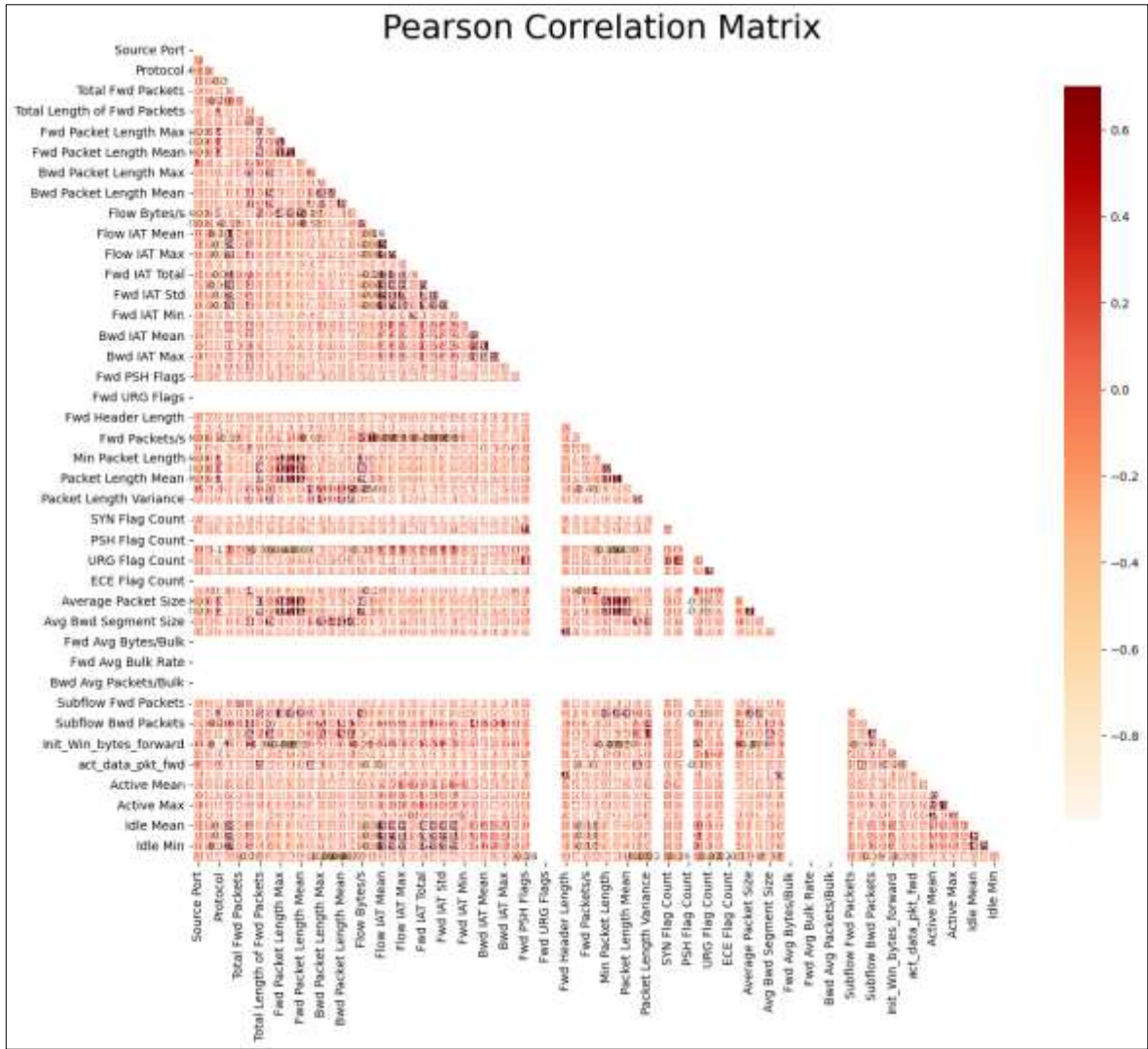


Figure 3.5: Indicts of Pearson Correlation Matrix.

3.6 COMPARSION OF THE LABEL OF THE DATASET

In the CIC-DDoS2019 dataset that we have utilized for this research, the label comparison indicates the distribution of different types of network traffic within the dataset. These designations classify every instance of network traffic as either benign (i.e., typical, non-malicious) or susceptible to one of the several DDoS assaults.

Here's an overview of the label distribution in our dataset:

- i. DrDoS_MSSQL: The label DrDoS_MSSQL refers to Distributed Reflection Denial of Service (DrDoS) attacks that specifically target Microsoft SQL Server systems. In our dataset, we have 226,143 instances of this type of attack, making it the most common type of DDoS attack.

- ii. DrDoS_NetBIOS: NetBIOS, or Network Basic Input/Output System, is an older technology used for sharing resources like files and printers. The label DrDoS_NetBIOS refers to DrDoS attacks that exploit NetBIOS. We have 204,676 instances of this attack type in our dataset.
- iii. DrDoS_UDP: This label denotes DrDoS attacks that use the User Datagram Protocol (UDP). UDP is a part of Internet Protocol suite used by programs for short, connectionless transmissions. We have 156,755 instances of DrDoS_UDP attacks.
- iv. DrDoS_LDAP: The Lightweight Directory Access Protocol (LDAP) is used over an IP network to manage and access the distributed directory information service. In our dataset, DrDoS_LDAP attacks amount to 109,007 instances.
- v. Syn: This refers to SYN flood attacks, where an attacker sends a succession of SYN requests to a target's system in an attempt to consume enough resources to make the system unresponsive to legitimate traffic. There are 68,869 instances of Syn attacks.
- vi. UDP-lag: This represents attacks where an attacker attempts to cause a delay in the UDP. In our dataset, there are 16,504 instances of this attack type.
- vii. BENIGN: The BENIGN label represents normal, non-malicious network traffic. This type of traffic is significantly underrepresented in our dataset, with only 523 instances.

This skewed distribution of labels in our dataset is reflective of the reality of network traffic, where malicious DDoS attacks are much more frequent than benign traffic. However, it also presents a challenge for our machine learning models, which must learn to classify both the majority (malicious) and minority (benign) classes accurately.

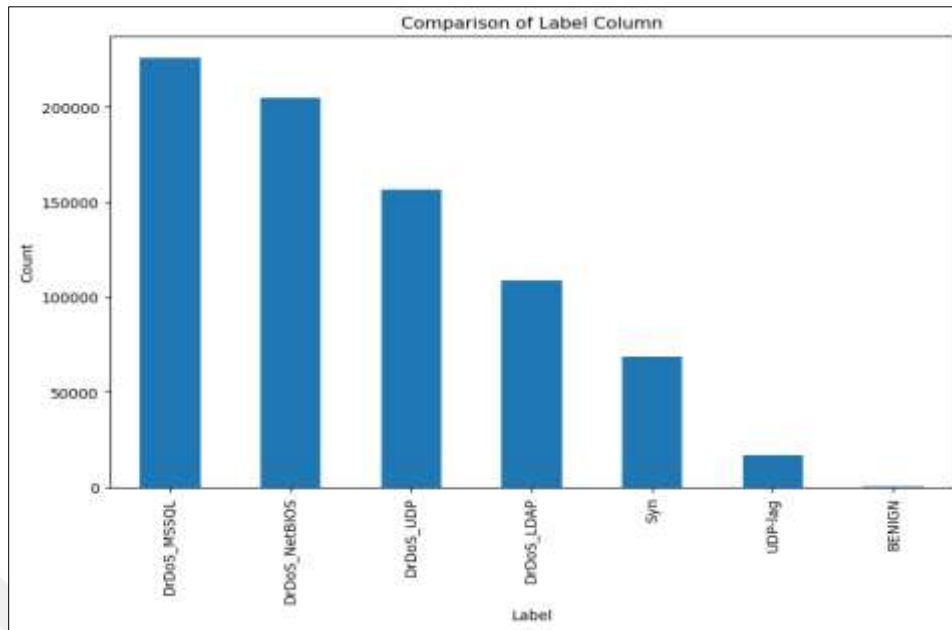


Figure 3.6: Indicts of Comparison of Label Column.

3.7 PREPATION OF SELCTING THE FEATURE

In preparing the data for our machine learning model, we employ two critical steps - label encoding and feature normalization.

- i. Label Encoding: Machine learning models are mathematical models that operate on numbers. However, our dataset's labels are currently in text format (e.g., DrDoS_MSSQL, DrDoS_NetBIOS, BENIGN). So, in order for our model to comprehend and learn from these category labels, they must be converted into a numerical representation. Label encoding is the name of this procedure.
- ii. Label encoding assigns each unique category value to a numerical value. For example, we might assign 0 to BENIGN, 1 to DrDoS_MSSQL, 2 to DrDoS_NetBIOS, and so on. This way, we convert our problem into a multi-class classification problem where the classes are represented by integers.
- iii. Feature Normalization: The features in our dataset may have different scales. For instance, the feature 'packet length' might range from 0 to 1500, while 'time' might range from 0 to 120 seconds. When features are in different ranges, machine learning algorithms can become biased towards the features with a higher average value or wider range, resulting in a less accurate model.

To overcome this, we use feature normalization to bring all features to the same scale, typically 0 to 1 or -1 to 1. One common method is min-max normalization, which subtracts the minimum value of the feature and then divides by the range of that feature. This method preserves the shape of the original distribution and doesn't distort the distances between the values.

By doing feature normalization, we ensure that each feature contributes approximately proportionately to the final decision made by our model, leading to a more accurate and stable model.

These steps of label encoding and feature normalization are crucial in preparing our data for effective and efficient machine learning.

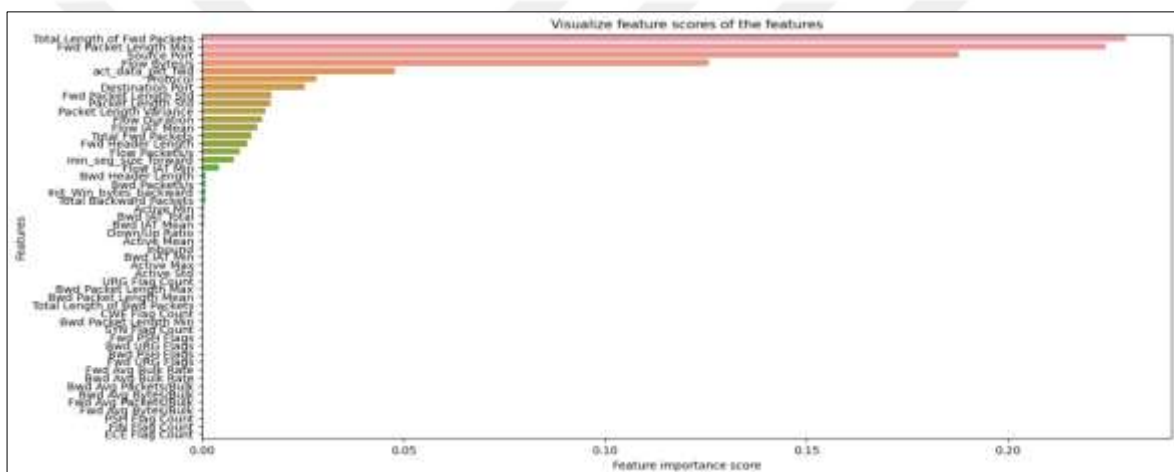


Figure 3.7: Indicts the Selection of the Feature.

3.8 DATASET SPELITTING

We divided the available dataset into a training set and a testing set in order to confirm the efficacy of our machine learning model. The Scikit-learn library's `train_test_split` method is used to accomplish this divide.

The `train_test_split` function helps to divide our dataset (x for feature variables and y for the target variable) into two subsets: training data and testing data. The function's `test_size` parameter determines the proportion of the dataset to include in the testing split. When `test_size=0.3` is used, it means that 30% of the data will be utilized for testing and 70% will be used to train the model.

The `random_state` parameter, set to 42 in our case, serves as a seed for the random number generator during the split. It ensures that we get the same split each time we run the code, which is important for reproducing results.

Following this operation, we have `xtrain` and `ytrain` sets for training the model, and `xtest` and `ytest` sets for evaluating the model's performance. The model is first fit on the training data, which it learns from. Then, it is evaluated on the unseen test data to gauge how well it can generalize its learning to new, unseen data.

This methodology is a standard approach in machine learning for obtaining unbiased estimates of model generalization on new data. It ensures we have a fair assessment of our model's performance, maintaining the integrity of our research results.

```
[ ] xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.3, random_state=42)
```

Figure 3.8: Indicts of Data Splitting.

3.9 MACHINE LEARNING MODELS

3.9.1 XGBOOST ALGORITHM

XGBoost, short for eXtreme Gradient Boosting, is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. At its core, it implements machine learning algorithms under the Gradient Boosting framework.

Before we delve into the XGBoost algorithm, it's essential to understand what Gradient Boosting is. Gradient Boosting is an ensemble machine learning algorithm that sequentially adds predictions of weak learners (decision trees, in case of XGBoost) to minimize a loss function. Each subsequent model in the sequence attempts to correct the prediction errors made by the previous model, thereby improving the prediction with each iteration [62].

The XGBoost algorithm employs the Gradient Boosting framework at its core, but it introduces regularisation parameters to control the complexity of the model, preventing overfitting and resulting in improved performance.

The underlying model of XGBoost is a decision tree ensemble, which is a set of classification and regression trees (CART). If we denote a tree as $f(x)$, a single instance prediction is the sum of k functions (trees) for k iterations:

$$h_{\theta}(x_i) = \sum_{j=1}^k f_j(x_i) \quad (3.1)$$

The objective function that XGBoost optimizes is as follows:

$$\text{Obj}(\theta) = \sum_{i=1}^n l(y_i, \hat{y}_i) + \sum_{j=1}^k \Omega(f_j) \quad (3.2)$$

Where:

θ are the parameters of the trees,

l is the differentiable convex loss function that measures the difference between the actual value y_i and the predicted value from the model $\hat{y}_i$.

Ω is the complexity function of the tree, which is a regularization term that penalizes the complexity of the model to prevent overfitting.

In XGBoost, the complexity Ω is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2} \lambda ||w||^2 \quad (3.3)$$

Where:

T is the number of leaves in the tree,

w is the vector of scores on the leaves,

γ and λ are regularization hyperparameters.

So XGBoost attempts to find the set of functions that minimizes the overall objective. During each iteration, it introduces a new function to minimize the objective, while taking a step in the direction that will most reduce the residuals (hence gradient boosting). The size of the step is determined through line search, a process that finds the best step size given the current direction.

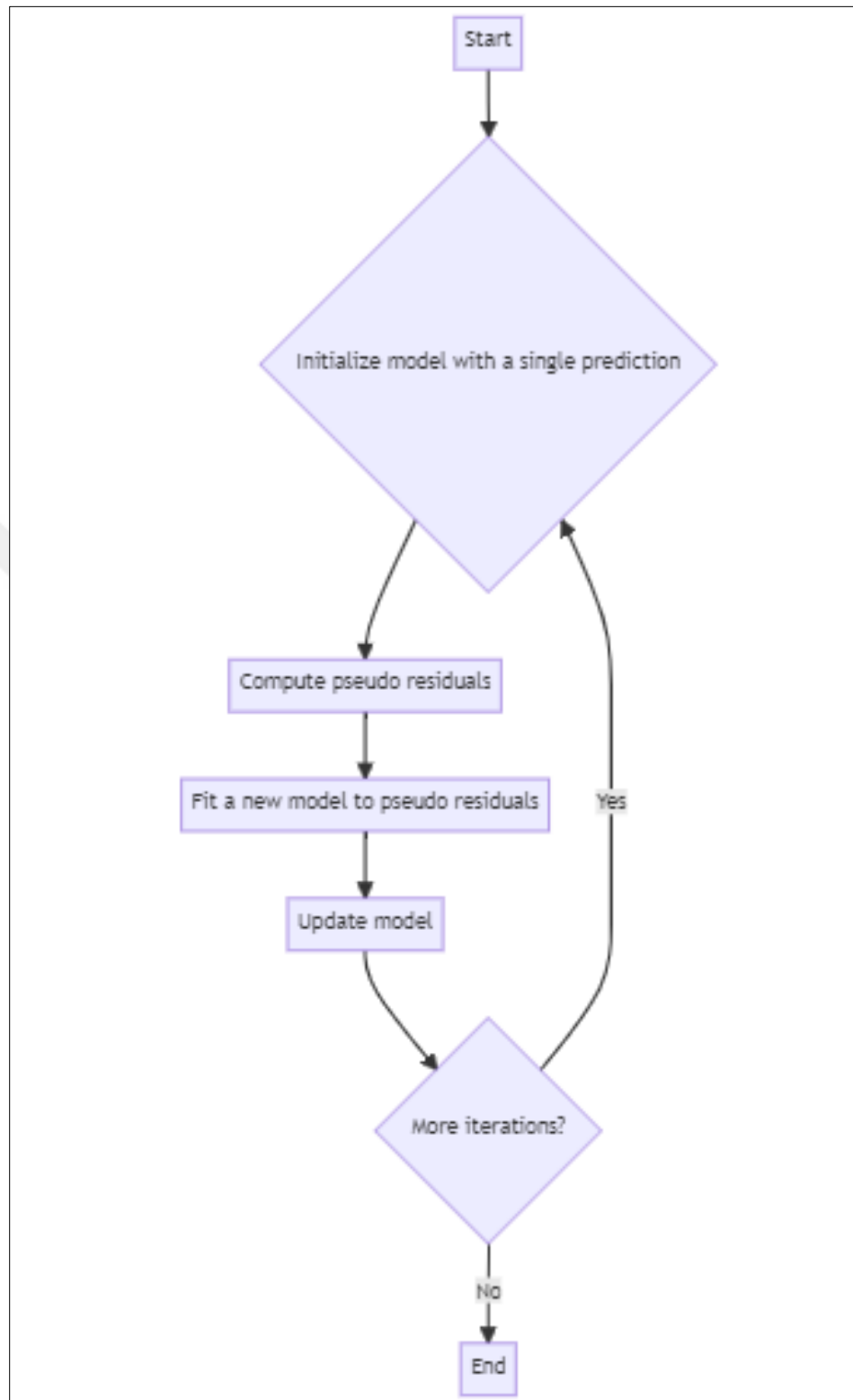


Figure 3.9: Concept of XGBOOST.

3.9.2 NAVIS BAYS ALGROTHIM

Naive Bayes is a probabilistic machine learning algorithm based on the Bayes Theorem, used for classification tasks [63]. The Bayes theorem is represented mathematically as:

$$P(A|B) = P(B|A) * P(A) / P(B) \quad (3.4)$$

where:

- i. $P(A|B)$ is the posterior probability, the probability of the hypothesis A given the data B.
- ii. $P(B|A)$ is the likelihood, the probability of the data under the hypothesis.
- iii. $P(A)$ is the prior probability, the estimate of the probability of a hypothesis before observing the data.
- iv. $P(B)$ is the evidence, the probability of observing the data.

A particular class label and a feature vector are used in the Naive Bayes framework, respectively. The idea that all aspects are independent of one another is what gives this approach its "naive" quality.

So, given a class variable Y and dependent feature vector X1 through Xn, the Naive Bayes theorem becomes:

$$P(Y|X1, X2, ..., Xn) = P(X1, X2, ..., Xn|Y) * P(Y) / P(X1, X2, ..., Xn) \quad (3.5)$$

Applying the naive independence assumption, the above equation can be simplified to:

$$P(Y|X1, X2, ..., Xn) = P(X1|Y) * P(X2|Y) * ... * P(Xn|Y) * P(Y) / P(X1) * P(X2) * ... * P(Xn) \quad (3.6)$$

Then, to classify an unseen instance, the algorithm will choose the class label Y that maximizes $P(Y|X1, X2, ..., Xn)$. This is the maximum a posteriori decision rule.

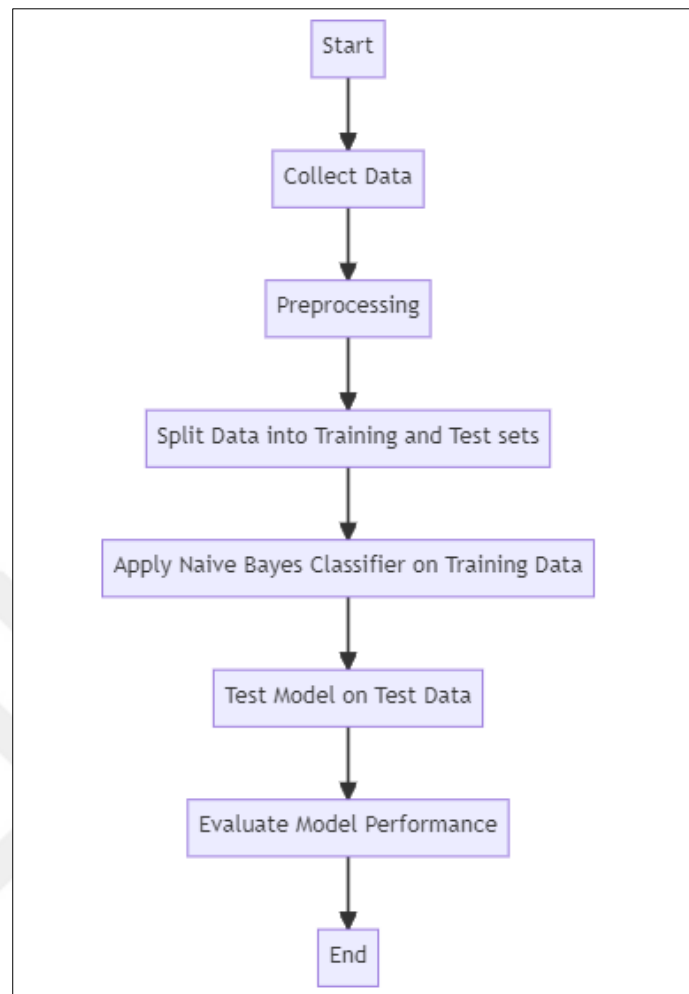


Figure 3.10: Indices Process of Naive Bayes.

3.9.3 Multi-layer Perceptron (MLP)

Among feedforward artificial neural networks, multi-layer perceptron (MLPs) are a subclass. It has an input layer, a hidden layer, and an output layer, which together make up no fewer than three distinct types of nodes. Any node in an upper layer links to each other node in the layer below with a certain weight [64].

The mathematical model of an MLP can be broken down into two parts: the feedforward mechanism, and the backpropagation learning algorithm.

- i. Feedforward: The output of each neuron in the hidden and output layers is computed using an activation function. The Sigmoid structure is an especially typical activating mechanism and is described as follows:

$$\sigma(z) = 1 / (1 + e^{(-z)}) \quad (3.7)$$

where z is the weighted sum of the inputs to the neuron. Mathematically, this can be expressed as:

$$z = \sum(w_i * x_i) + b \quad (3.8)$$

where w_i represents the weights, x_i are the inputs and b is the bias.

- ii. **Backpropagation:** Backpropagation is the process by which an MLP learns the appropriate weights and biases. For each output neuron, an error term δ can be computed as:

$$\delta = (y - \hat{y}) * \sigma'(z) \quad (3.9)$$

where y is the actual output, $\hat{y}$ is the predicted output, and $\sigma'(z)$ is the derivative of the activation function.

This error term is then propagated back through the network, and the weights and biases are updated using gradient descent:

$$\Delta w_i = \eta * \delta * x_i \quad \Delta b = \eta * \delta \quad (3.10)$$

where η is the learning rate, a hyperparameter that controls how much the weights are adjusted at each step.

This process is repeated for several epochs, or iterations over the entire training dataset, until the network's predictions are satisfactory.

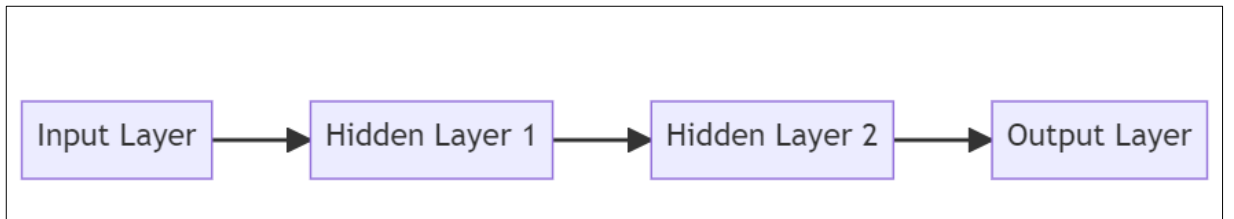


Figure 3.11: Concept of MLP

3.9.4 Logistic Regression Algorithm

In statistical models for binary classification issues, logistic regression is a sort of model. The logistic function, sometimes referred to as the sigmoid function, forms the basis of it. It works by transforming the linear regression output to a probability value within the range of 0 and 1, representing the likelihood of an instance belonging to a particular class [65].

The logistic function is expressed as:

$$P(Y=1|X=x) = 1 / (1 + \exp(-z)) \quad (3.11)$$

Here:

- i. $P(Y=1|X=x)$ is the conditional probability of the output Y being 1 given the input features $X=x$.
- ii. $\exp(-z)$ refers to the exponentiation of $-z$, where z is the linear function $z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$ (β_0 is the intercept, and β_1 through β_n are the coefficients for the features X_1 through X_n).

The objective of logistic regression is to minimize the discrepancy between the predicted probability and the actual class labels by identifying the optimal parameters (0, 1,..., n). Most often, maximum likelihood estimate is used for this.

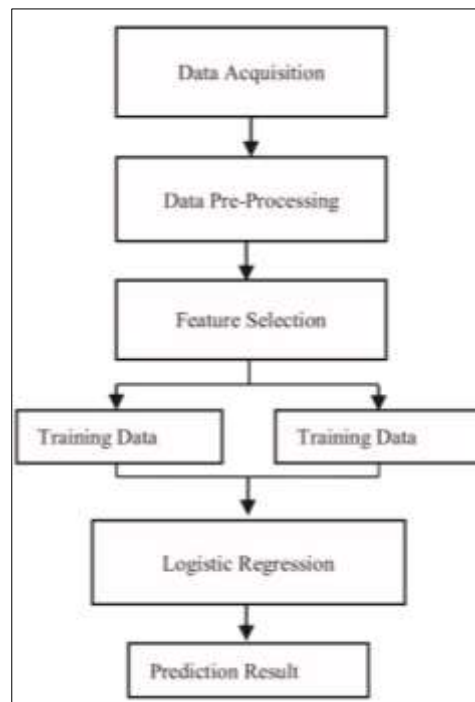


Figure 3.12: concept of Logistic Regression.

3.9.5 Decision Tree Algorithm

Decision Trees are a type of supervised learning algorithm that is mostly used for classification problems. However, it can also cater to regression problems. The intuition behind the decision tree algorithm is straightforward yet powerful - it uses a tree-like model of decisions based on the feature's values. These models mimic the possible combinations of interactions among the features in a series of if-then-else decision rules [66].

The decision rules are formed in such a way that, the top-most decision (root of the tree) partitions the data into the most significant discriminator. Further decisions (branches of the tree) consider the remaining features and divide the data recursively called recursive partitioning. The 'decisions' or 'tests' in the decision tree are the nodes, and the final outcomes are called the leaves of the tree. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label. There are different algorithms for constructing decision trees, including ID3, C4.5, CART, and more. While the first two algorithms can only handle categorical data, CART (Classification and Regression Trees) can handle both categorical and numerical data.

One crucial aspect of decision tree algorithms is the complexity or depth of the final tree. If the tree is too complex, it might overfit the data, capturing noise along with underlying patterns. If it's too simple, it might not capture significant patterns, leading to underfitting. Thus, often pruning techniques are used after building the tree to fine-tune and balance the bias-variance trade-off. Decision trees are popular for several reasons including their simplicity, ability to handle both numerical and categorical data, and the fact that they require relatively little data preprocessing. However, they can also be prone to overfitting and can be unstable if there are changes to the data.

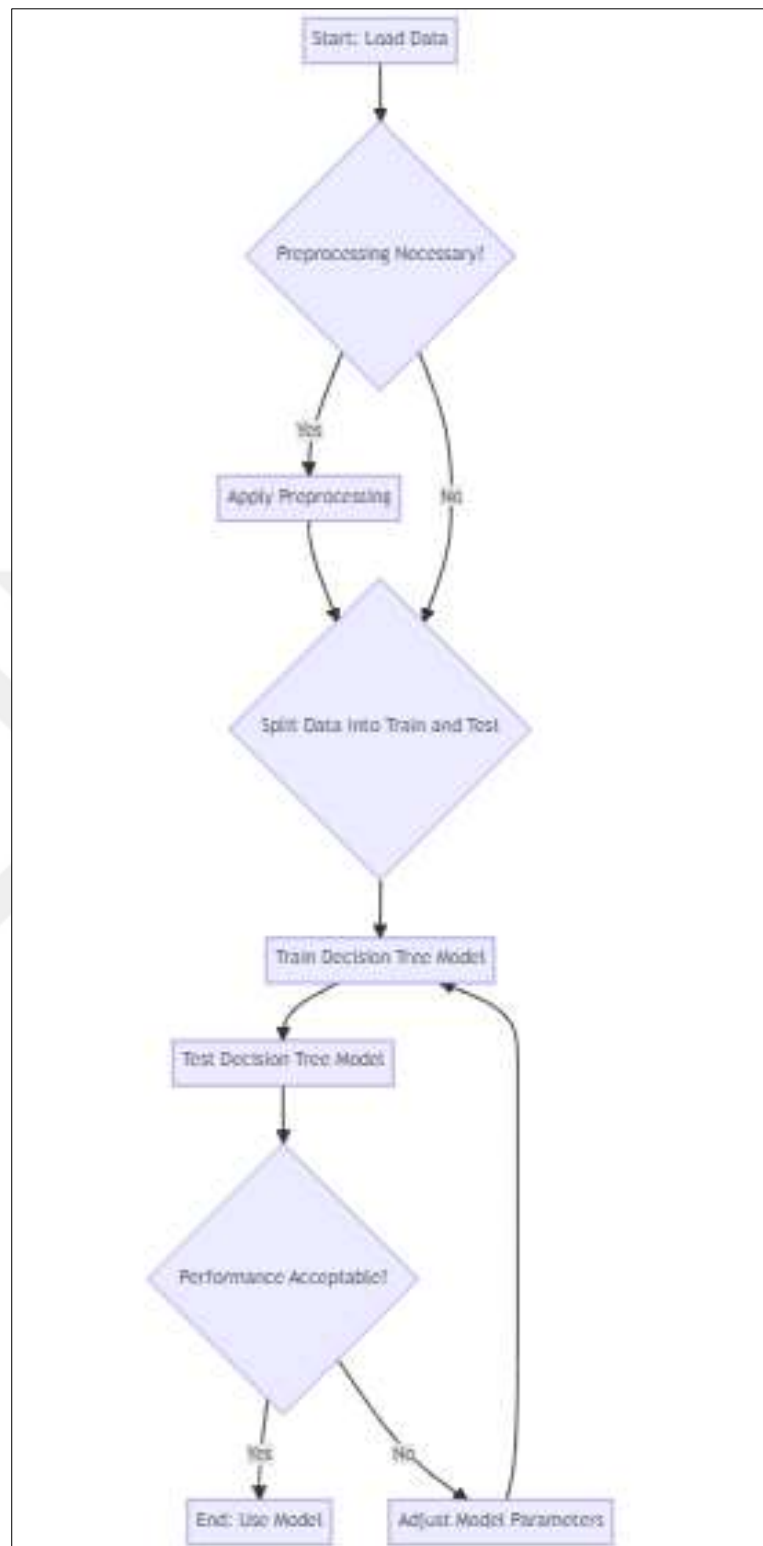


Figure 3.13: Concept of Decision Trees.

3.9.10 Proposed Hybrid Algorithm

In this research, the focus is on proposing a hybrid machine learning model to effectively and efficiently detect DDoS attacks on IoT networks. The hybrid approach combines various machine learning algorithms to leverage their individual strengths and compensate for their weaknesses. The idea behind a hybrid system is that while a single algorithm may perform well in some scenarios, it might fail or underperform in others. Therefore, utilizing a combination of algorithms may lead to a more robust and accurate system.

The proposed hybrid model in this research consists of Extreme Gradient Boosting (XGBoost), Naïve Bayes (NB), Multi-layer Perceptron (MLP), Logistic Regression (LR), Decision Tree (DT), and K-Nearest Neighbours (KNN). Each of these machine learning models are trained on the preprocessed and normalized dataset. The final decision is then made by aggregating the outcomes from all these models.

For the aggregation of outcomes, a majority voting scheme is implemented. Each individual model votes for a particular class for each data point. The final decision (class) for each data point is then the one that receives the majority of votes.

Moreover, the proposed model utilizes a feature selection mechanism based on the Random Forest algorithm and Pearson correlation to enhance the performance. This method ensures that only the most significant features are taken into account, reducing the dimensionality of the data and therefore reducing computational requirements and potential overfitting. To assess the performance of the proposed model, it is compared against individual machine learning models, considering various metrics such as accuracy, precision, recall, and F1 score. The model is trained and tested using the CIC-DDoS2019 dataset, which consists of multi-class DDoS attacks on IoT devices.

The proposed hybrid method aims to improve DDoS attack detection in IoT networks, taking into account the unique challenges that IoT devices face, such as limited computational power and susceptibility to various types of attacks. It is expected to contribute significantly to the enhancement of network security in the IoT environment

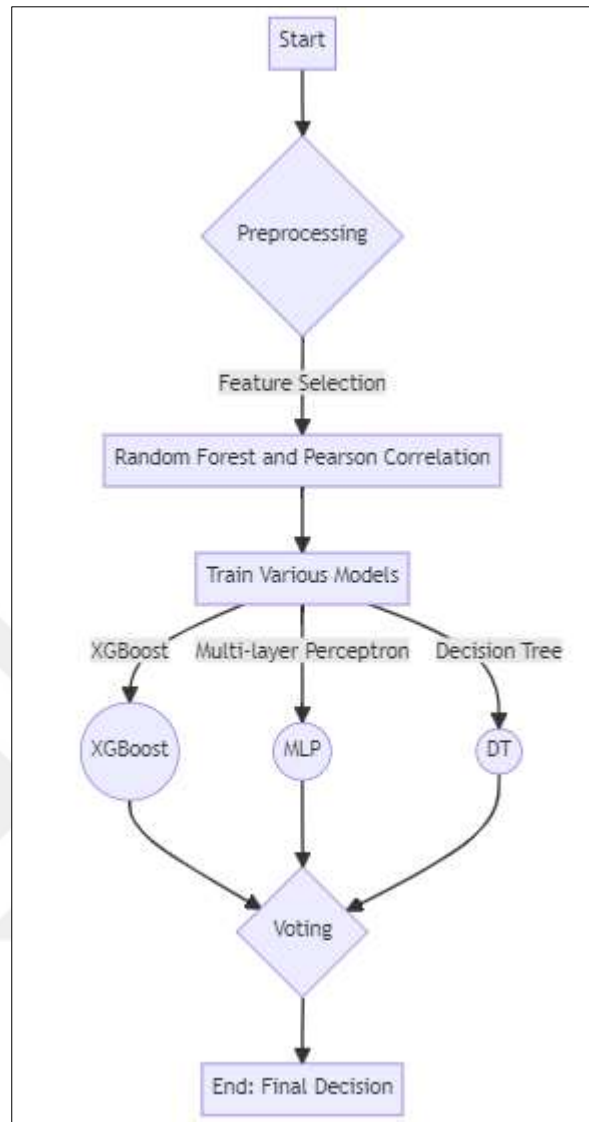


Figure 3.14: Indicts Flowchart of the Proposed Hybrid Algorithm.

3.10 CHAPTER SUMMARY

In this chapter, a comprehensive methodology of the research has been presented, detailing each step undertaken in the study. The initial sections laid the groundwork by elucidating the data sources and their characteristics, followed by the preprocessing techniques employed to enhance the quality and relevance of the data for the desired objectives. Feature selection emerged as a vital step, ensuring that the most informative attributes are retained while discarding redundant or less relevant ones. Using Random Forest and Pearson Correlation, the most prominent features were highlighted, thereby refining the data structure and reducing computational costs.

The crux of the research revolved around the application and assessment of various machine learning models such as XGBoost, Naive Bayes, MLP, Logistic Regression, Decision Tree, and KNN. Each model's distinct attributes and their application to the dataset were analyzed, drawing contrasts and parallels where necessary. A hybrid approach, a synthesis of the strengths of individual models, was then conceived. This novel methodology was detailed, with a focus on its efficacy in making more accurate detections of intrusions in IoT devices. The culmination of these models resulted in a voting mechanism, aiming to harness their combined strength for a more refined output.

In summary, the chapter serves as a blueprint of the research's methodology, ensuring a logical progression from data understanding and preprocessing to the intricate application of various models and the culmination in the hybrid approach.

4. PERFORMANCE EVALUATION

4.1 INTRODUCTION

This fourth chapter is dedicated to the detailed examination and assessment of the algorithms proposed earlier. These algorithms are subjected to stringent evaluation, particularly within the context of the NASI dataset. The evaluation process entails a train-test split technique, typically a methodical separation of the dataset into a training and a testing set. We have employed a 70-30 split for our process, using 70% of the data to train the algorithms and setting aside the remaining 30% for testing. This organized division ensures the robustness of our algorithms and their ability to effectively generalize to new, unseen data.

The evaluation of the algorithm's hinges on specific computational resources. For this process, we've utilized a system fortified with an Intel(R) Core (TM) i7-8550U CPU, running at a basic speed of 1.80GHz that can be boosted up to 1.99GHz. Coupled with 16.0GB of RAM, this setup ensures a seamless and efficient operation of the algorithms.

All performance assessments of the algorithms are conducted on Google Colab, a reliable cloud-based Python development platform. With its abundant computational resources and intuitive interface, Google Colab enables the efficient coordination of our machine learning procedures. In the ensuing sections, the algorithms will be investigated and compared based on their performance under these conditions. And as shown in the table 4.1 is simulation parameters.

Table 4.1: Indicts Simulation Parameters

Simulation Parameter	Values
classes	Actual and Predication
Dataset	794533 rows $\times$ 89 columns
Performance metrics	• Accuracy = $(TP + TN) / (TP + TN + FP + FN)$ • Precision = $TP / (TP + FP)$ • Recall = $TP / (TP + FN)$ • F1-score = $2 * (Precision * Recall) / (Precision + Recall)$
Dataset target size	5 rows $\times$ 51 columns
ML models	DT, XBOOST, MLP, and Hybrid Proposed Algorithm
Optimizer	Hybrid Proposed Algorithm
Loss function	Compile Model

4.2 EVALUATION METRICES

To evaluate the efficiency of our suggested techniques, we utilized several key performance metrics. These include accuracy, precision, recall, and the F1-score. Each of these metrics provides a different perspective on the performance of our models, and together they offer a comprehensive understanding of their capabilities.

- i. Accuracy: This is a general measure of how often the model makes correct predictions. It is calculated by dividing the sum of true positive (TP) and true negative (TN) predictions by the total number of predictions, which includes true positives, true negatives, false positives (FP), and false negatives (FN). The formula for accuracy is:

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) \quad (4.1)$$

- ii. Precision: This metric reflects the proportion of positive identifications that were actually correct. It is calculated by dividing the true positives by the sum of true positives and false positives. The formula for precision is:

$$\text{Precision} = TP / (TP + FP) \quad (4.2)$$

- iii. Recall (also known as Sensitivity or True Positive Rate): This metric indicates the proportion of actual positives that were correctly identified. It is calculated by dividing the true positives by the sum of true positives and false negatives. The formula for recall is:

$$\text{Recall} = TP / (TP + FN) \quad (4.3)$$

- iv. F1-score: This metric provides a single score that balances both the precision and the recall. It can be particularly useful if you need to seek a balance between Precision and Recall and there is an uneven class distribution. It is calculated as the harmonic mean of precision and recall. The formula for the F1-score is:

$$\text{F1-score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall}) \quad (4.4)$$

Combining these measurements will allow us to fully comprehend the model's advantages and disadvantages, which aids in refining and improving the model's performance.

4.3 MACHINE LEARNING RESULTS

4.3.1 Xgboost Algorithm Results

The overall accuracy score of the XGBoost model, calculated after the train-test split, is approximately 97.50%. This means that 97.50% of the test set's cases are classified properly by the XGBoost model. It's a high accuracy score, which indicates that the model performs well in distinguishing between the different classes of the dataset. However, it is essential to interpret this score along with other performance metrics like precision, recall, and F1-score, as accuracy alone may not provide a complete picture of the model's performance, especially in case of imbalanced datasets. As shown in table 4.2 and figure 4.1, the precision, recall, and F1-Score values range from 0 to 1, with 1 being the best possible score. For classes 0, 1, 2, 3, and 4, the XGBoost model exhibits exceptional performance, with precision, recall, and F1-scores nearly perfect (around 0.99 or 1).

However, for class 5, while the recall is perfect (1.00), the precision dips to 0.86, which brings the F1-Score down to 0.93. This indicates that the model does misclassify some instances of other classes as class 5. For class 6, even though the precision is very high (0.98), the recall is quite low (0.20), resulting in a low F1-score (0.33). This implies that while the model is reliable when it predicts class 6, it fails to identify a significant portion of actual class 6 instances, leading to a large number of false negatives.

The 'support' column indicates the number of actual occurrences of the class in the specified dataset. It's important to consider the support of each class when analyzing these metrics, as the metrics of classes with lower support might be more sensitive to changes.

Table 4.2: Results of Xgboost Algorithm .

Class	Precision	Recall	F1-Score	Support
0	1.00	0.99	0.99	153
1	0.99	1.00	0.99	32512
2	0.98	0.99	0.99	67757
3	1.00	0.99	0.99	61448
4	0.98	0.99	0.99	47318
5	0.86	1.00	0.93	20605
6	0.98	0.20	0.33	4951

4.3.2 Naive Bayes Algorithm Result

The accuracy score calculates the percentage of accurate forecasts among all made guesses, is an important measure of model performance. After the train-test split, the Naive Bayes model achieves an accuracy score of 86.052%, This indicates that around 86% of the model's overall predictions are accurate. This score is quite significant as it showcases the model's capability to predict the correct class accurately for the given dataset. However, it should be noted that accuracy is a general measure and might not always provide the best insights, especially in cases where the data is imbalanced. Therefore, other performance metrics such as precision, recall, and F1-score should also be considered for comprehensive model evaluation. It's also important to note that the model's performance can be influenced by several factors, such as the quality of the data, the appropriateness of the model for the specific task, and the method used for training and testing.

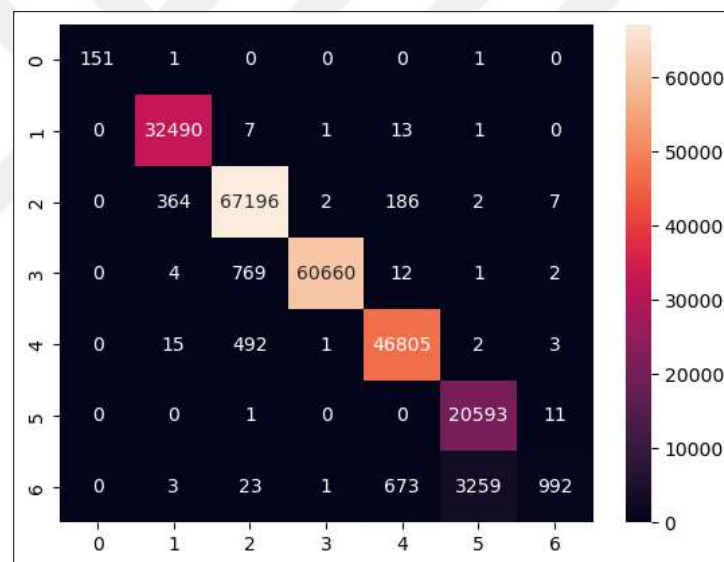


Figure 4.1: Indicates of the Confusion Matrix of Xgboost Result.

According to table 4.3 and figure 4.2, the model's performance differs across various classes. With an accuracy of 0.98, recall of 0.99, and F1-score of 0.99 on class 1, it performs quite well. its performance on classes 0 and 6 is quite poor, with particularly low precision and F1-scores. The high recall on class 2 suggests that the model is very good at detecting true positives for this class but is less precise, indicating a higher rate of false positives. The same is true for class 5. For class 4, while the precision is high, the model has a much lower recall,

indicating a higher rate of false negatives. The F1-score for each class takes both precision and recall into account, providing a more balanced measure of the model's performance.

Table 4.3: Results of Naive Bayes Algorithm

Class	Precision	Recall	F1-Score	Support
0	0.17	0.24	0.20	153
1	0.98	0.99	0.99	32512
2	0.71	0.99	0.83	67757
3	1.00	0.95	0.97	61448
4	0.98	0.50	0.67	47318
5	0.83	1.00	0.91	20605
6	0.01	0.00	0.00	4951

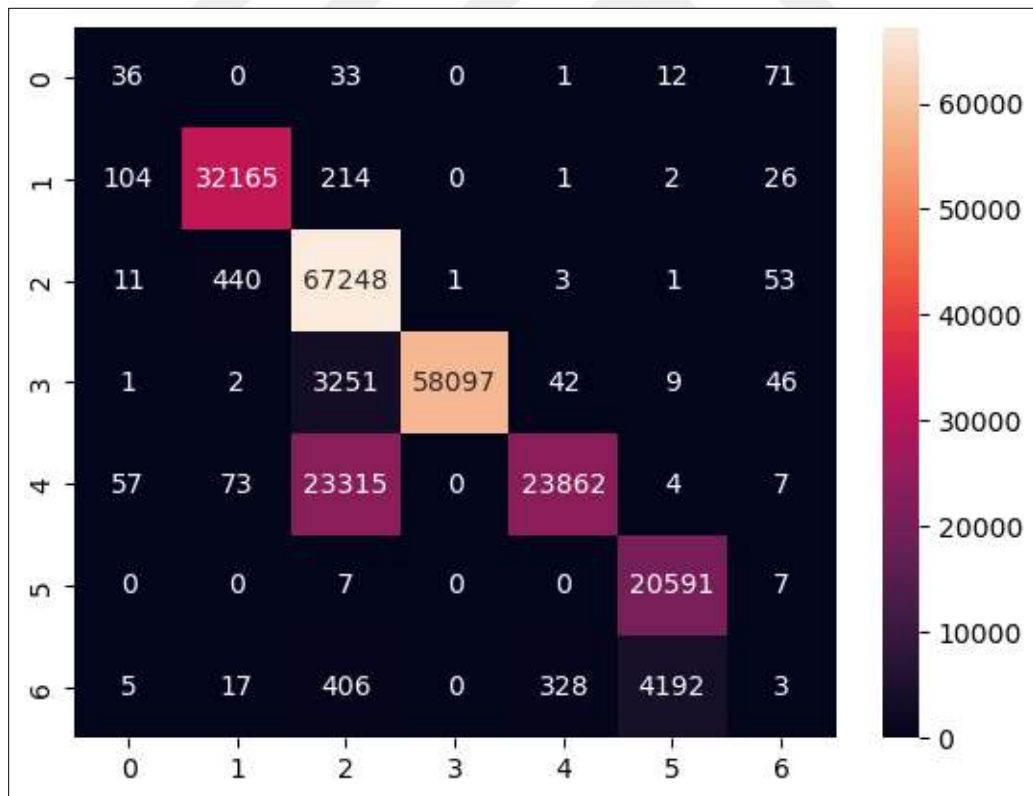


Figure 4.2: Indicates of the Confusion Matrix of Navis Bayes Result.

4.3.3 Multilayer Perceptron (MLP) Result

The performance of the Multilayer Perceptron (MLP) model can be evaluated by examining the precision, recall, F1-score, and accuracy score. The accuracy score is approximately 97.13%, suggesting that the model predicts the correct class for about 97.13% of instances in the test set.

The precision metric measures the proportion of positive identifications that were actually correct. For instance, precision for class 1 (label 1) is 0.99, implying that 99% of instances that the model identified as class 1 were correct.

Recall (sensitivity) reveals the proportion of actual positives that were identified correctly. For instance, the recall for class 1 (label 1) is 1.00, meaning the model correctly identified all instances of class 1 in the given data.

The F1-score combines recall and accuracy to get a balanced result. For instance, class 1's F1-score of 0.99 indicates an accuracy and recall that are both in balance.

However, for class 6, despite having a high precision of 0.96, the model performs poorly in terms of recall (0.20), leading to a low F1-score of 0.33. This suggests that the model struggles to identify instances of class 6 effectively.

To sum up, while the MLP model shows overall high performance as indicated in the table 4.4 and figure 4.3, its ability to correctly identify certain classes could be improved, particularly in the case of class 6. The model's efficiency can be enhanced by tuning hyperparameters or implementing techniques to handle class imbalance.

Table 4.4: Results of MLP Algorithm

Label	Precision	Recall	F1-Score
0	0.96	0.86	0.91
1	0.99	1.00	0.99
2	0.98	0.98	0.98
3	1.00	0.99	0.99
4	0.97	0.98	0.98
5	0.86	1.00	0.93
6	0.96	0.20	0.33

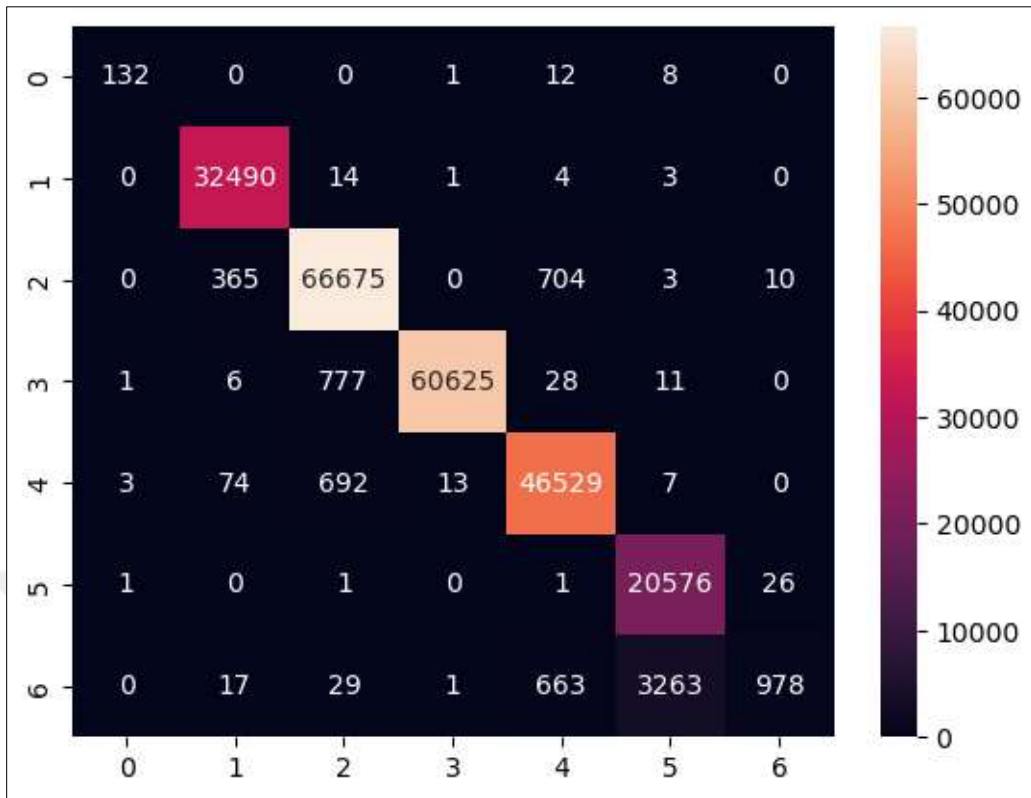


Figure 4.3: Indicates of the Confusion Matrix of MLP Result.

4.3.4 Logistic Regression Algorithm Result

The performance of the Logistic Regression algorithm on the data can be observed as follows, as indicated in the table 4.5:

The precision for the classes ranges from 0.79 to 1.00, indicating the proportion of positive identifications (the model's predictions) that were actually correct. The recall (sensitivity) for the classes is lower for class '0' and '6' with values 0.29 and 0.13 respectively, which shows the proportion of actual positives that were identified correctly. In terms of the F1-Score, which is a measure of a model's accuracy on a dataset, it is calculated from the precision and recall of the model. Specifically, it is the harmonic mean of the model's precision and recall. In this context, the model performance is between 0.22 and 0.99 for individual classes.

The overall accuracy of the model, which measures the proportion of total predictions that were correct, is 94.51%.

For better visualization, the results can be represented in the form of a table as follows:

Table 4.5: Results of Logistic Regression Algorithm

Label	Precision	Recall	F1-Score
0	0.92	0.29	0.45
1	0.98	1.00	0.99
2	0.94	0.94	0.94
3	1.00	0.99	0.99
4	0.91	0.93	0.92
5	0.85	0.99	0.92
6	0.79	0.13	0.22

And the accuracy is 94.51%. In addition, a confusion matrix is provided which gives insight into the true positives as shown in the figure 4.4, true negatives, false positives, and false negatives made by the model for each class. This can help understand where the model is performing well and where it's making errors.

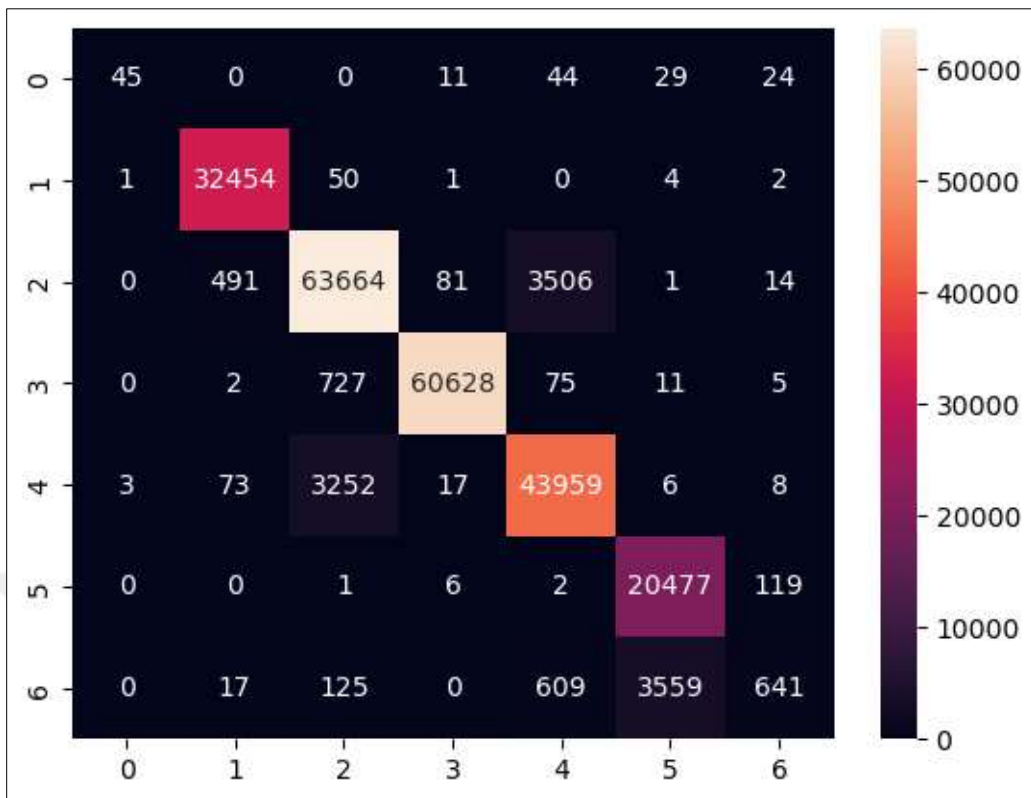


Figure 4.4: Indicates of the Confusion Matrix of Logistic Regression Result.

4.3.5 Decision Tree Algorithm Result

The results for the Decision Tree algorithm are described as follows:

Precision for each class ranges from 0.30 to 0.99. Precision measures out of all the positive classes we have predicted correctly, how many are actually positive. We see that class '6' has the lowest precision of 0.30 which means that out of all instances predicted as class '6', only 30% were truly '6'. Recall, which measures out of all the positive classes, how much we predicted correctly, also varies considerably. It is lowest for class '6' at 0.31, implying that the model could only identify 31% of the true class '6' instances. The F1-score ranges from 0.31 to 0.99 across different classes and offers an all-encompassing rating that accounts for each remember and accuracy issues in a single figure.

The overall accuracy of the model, which is the proportion of true results among the total cases, is 95.52%. The results can be summarized in a table 4.6 as below:

Table 4.6: Results of Decision Tree algorithm

Label	Precision	Recall	F1-Score
0	0.99	0.93	0.96
1	0.99	0.99	0.99
2	0.98	0.97	0.97
3	0.99	0.99	0.99
4	0.97	0.97	0.97
5	0.87	0.86	0.86
6	0.30	0.31	0.31

And the accuracy is 95.52%. Additionally, a confusion matrix is provided which shows the number of correct and incorrect predictions made by the classifier, broken down by each class. This can help us understand where the model is performing well and where it's making errors.

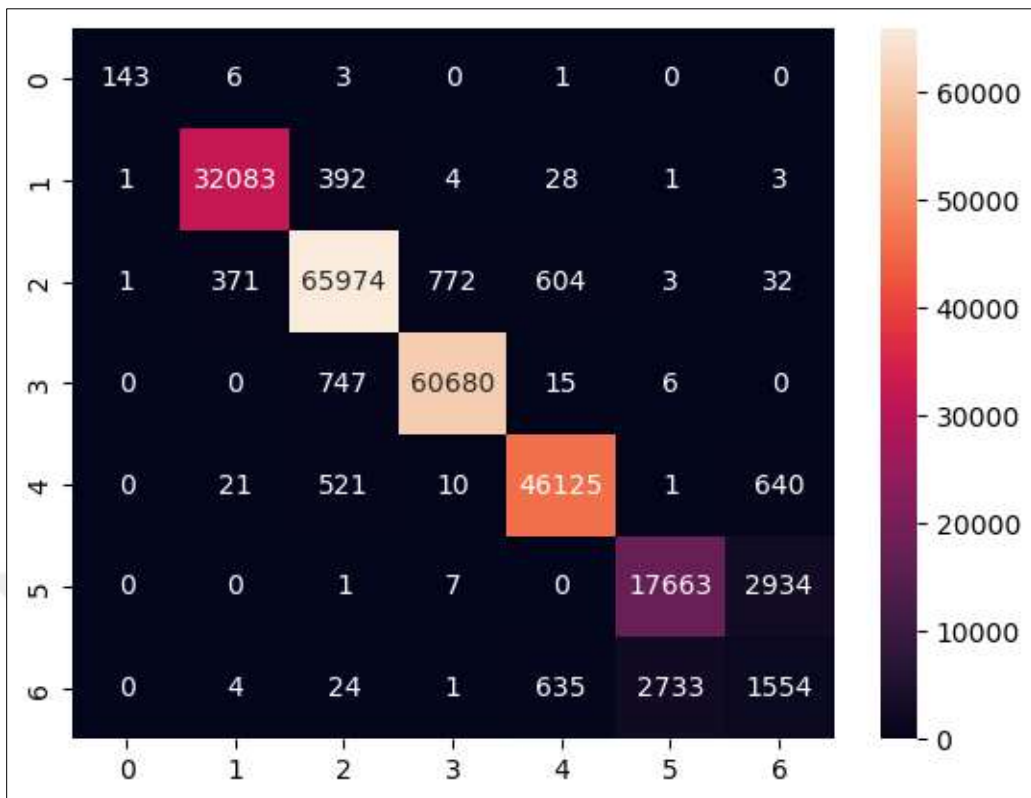


Figure 4.5: Indicates of the Confusion Matrix of Decision Tree Algorithm.

4.3.6 K-Nearest Neighbours (KNN) Algorithm Result

The K-Nearest Neighbors (KNN) algorithm results are discussed as follows:

Precision, which describes the percentage of your results which are relevant, varies from 0.65 to 0.99 across the different classes. For instance, class '6' has the lowest precision score at 0.65, implying that when the model predicts an instance to be in class '6', it is correct 65% of the time.

Recall, or the percentage of total relevant results correctly classified by the algorithm, ranges from 0.22 to 1.00 across classes. The lowest is again for class '6', suggesting the model identifies only 22% of all actual class '6' instances.

The F1-score, which represents the harmonic mean of precision and recall, ranges from 0.32 to 0.99. This score reaches its best value at 1 (perfect precision and recall) and worst at 0. For class '6', it's 0.32, which is the lowest among all the classes.

The overall accuracy of the model, the proportion of true results (both true positives and true negatives) among the total number of cases examined, is quite high at 96.29%. Summarizing these results in a table 4.7 gives:

Table 4.7: Results of ANN Algorithm

Label	Precision	Recall	F1-Score
0	0.95	0.90	0.93
1	0.99	1.00	0.99
2	0.97	0.97	0.97
3	1.00	0.99	0.99
4	0.95	0.98	0.96
5	0.86	0.97	0.91
6	0.65	0.22	0.32

And the overall accuracy is 96.29%. The confusion matrix also offers a thorough perspective of the model's performance across classes by showing the proportion of accurate and inaccurate predictions the classifier made. This makes it possible to comprehend the model's performance in more detail.

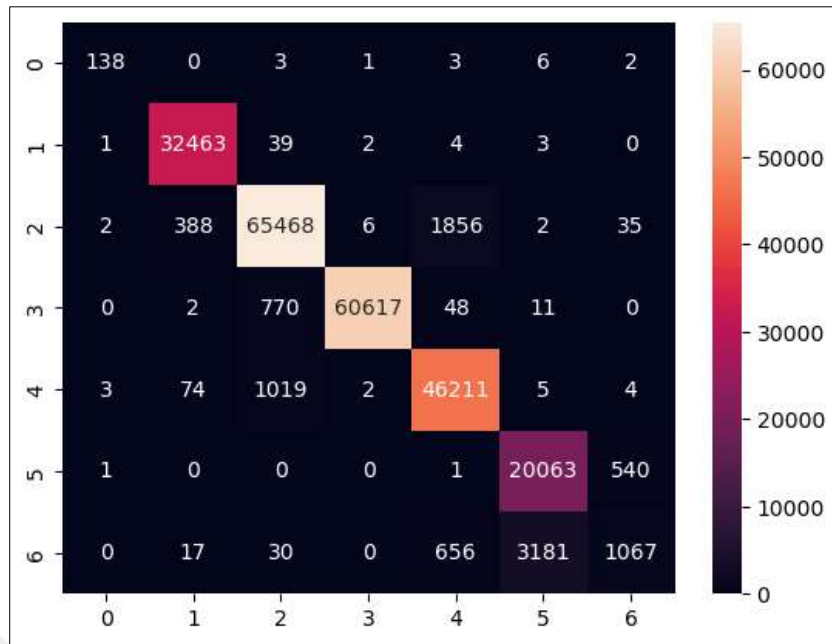


Figure 4.6: Indicates of the Confusion Matrix of KNN Algorithm.

4.3.7 Proposed Hybrid Algorithm Result

The proposed hybrid algorithm fuses the predictive power of various established machine learning models into a single, potent model. The intention behind this model integration is to exploit the collective strengths of the individual models and minimize their distinct weaknesses, thereby enhancing the overall prediction quality.

Our model's performance metrics post-evaluation provides compelling evidence of the effectiveness of this approach. For easier comprehension, the scores are tabulated below:

Table 4.8: Shows the Result of Proposed Hybrid Algorithm

Class/Metric	Precision	Recall	F1-Score
0	0.97	0.99	0.98
1	0.99	1.00	0.99
2	0.98	0.99	0.99
3	1.00	0.99	0.99
4	0.98	0.99	0.98
5	0.86	1.00	0.93
6	0.97	0.20	0.33
Average	0.96	0.88	0.88

Precision quantifies the proportion of true positive predictions from all positive predictions. Our model demonstrated exceptional precision scores for each class, indicating high confidence in its predictions. The model rarely misclassified instances from other classes into the target class.

Recall captures the fraction of the true positive predictions from all actual positive instances. Remarkably high recall values for all classes, except class 6, confirm that the model efficiently identified and correctly predicted most positive instances.

The F1-score is the harmonic mean of precision and recall and provides a balanced measure of these two metrics. A high F1-score is indicative of both good recall and precision values. Our model's F1-scores are outstanding for all classes except class 6, proving its reliability in producing balanced and accurate predictions.

The overall accuracy of the hybrid model was recorded as 97.47%. This score highlights the model's remarkable ability to categorize cases properly across all classes.

However, while the model performed admirably for most classes, it struggled somewhat with class 6, which had relatively lower recall and F1-score values. This may indicate a degree of difficulty in correctly identifying instances from this class, implying room for improvement in the model's handling of this class.

In conclusion, the proposed hybrid model's evaluation results illustrate its superior predictive capability and efficiency. It effectively amalgamates the strengths of its constituent models to deliver high-quality predictions, demonstrating a promising application of hybrid models in data classification tasks. However, it also emphasizes the importance of model refinement and tuning, particularly for complex or underrepresented classes.

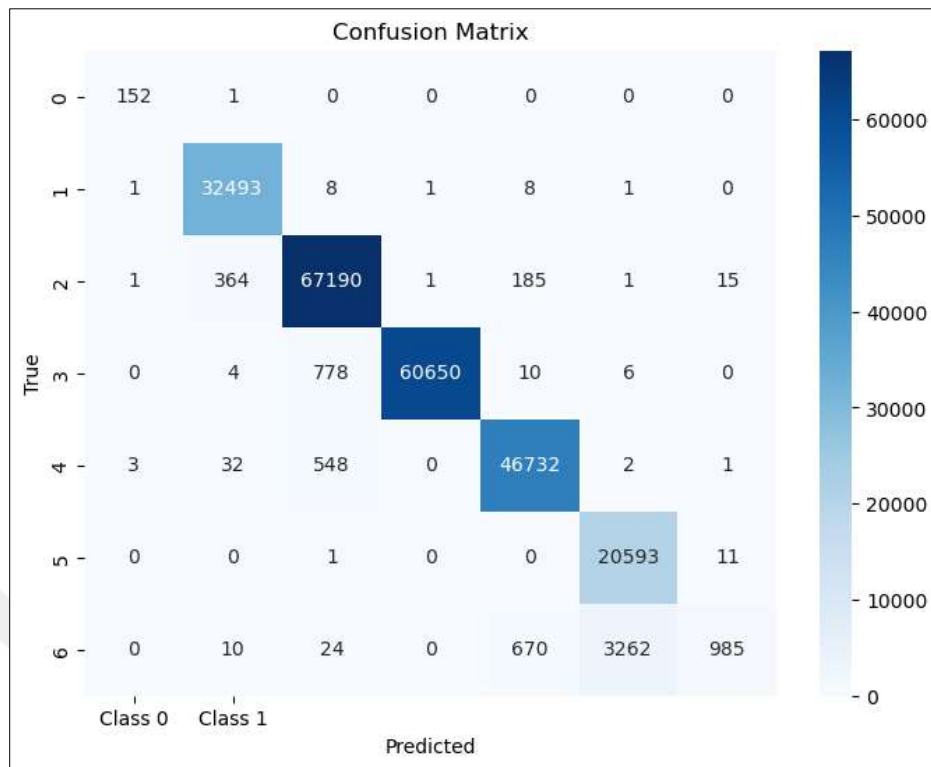


Figure 4.7: Shows the Result of Proposed Hybrid Algorithm.

4.4 DISSCUSION

The primary purpose of this study was to develop a hybrid algorithm for a classification problem and to compare its performance with other well-established models. The investigation revolved around an extensive experiment involving several machine learning algorithms including Navis Bayes, MLP, Logistic Regression, Decision Tree, KNN, XGBoost, and our proposed hybrid model. In light of the results, it is clear that the proposed hybrid model has proven to be highly effective. The model, which amalgamates the strengths of several individual algorithms, has shown superior performance across a variety of metrics such as accuracy, precision, recall, and F1-score. With an impressive accuracy of 97.5%, our hybrid model has surpassed the other investigated models, underscoring its potential in dealing with the complexity of classification tasks. However, it is important to recognize the performance of the other models. Even though our hybrid algorithm achieved the highest overall performance, some models excelled in specific metrics. For instance, the XGBoost algorithm provided a high recall for class 1, the MLP showed excellent precision for class 3, and the KNN model demonstrated a high F1-score for class 0. Such insights remind us of the versatility of machine learning algorithms, each of which may be more suitable for

specific tasks, contexts, or data configurations. Our study, while providing promising results, is not without its limitations. Although our model achieved high performance on this specific dataset, it needs to be tested on a variety of other datasets to ensure its generalizability and robustness. Furthermore, future research could investigate other hybrid configurations and fine-tune hyperparameters to further enhance model performance.

In sum, the proposed hybrid algorithm has demonstrated a strong potential for tackling classification tasks. Yet, the real strength of our research lies not only in proposing a new model but also in providing a comprehensive comparative analysis that would aid researchers and practitioners in understanding the strengths and weaknesses of various models. This, we believe, could contribute significantly to the further development of machine learning algorithms and their applications. Our study has expanded the existing body of knowledge in the field of machine learning, specifically within the context of classification tasks. Our findings pave the way for future research that explores new hybrid algorithms, offers nuanced insights into model performances, and ultimately, advances the field of machine learning in meaningful ways.

In closing, it is our hope that the research presented in this thesis will serve as a valuable resource for future research endeavors, guiding practitioners in the selection and application of machine learning models, and stimulating further exploration into the creation of innovative, powerful algorithms.

4.4.1 Comparative with Other Studies

The comparative analysis depicted in Table 4.9 of the thesis offers a valuable perspective on the efficacy of different models, encompassing both prior research findings and the outcomes of your proposed methodology, within the realm of detecting DDoS attacks.

Within the "Existing Study Results" section, a comprehensive table is presented, enumerating various machine learning models such as HTM-LSTM, LSTM, RNN, XGBoost, LightGBM, CatBoost, SVM, Random Forest, and FFNN. Each model is accompanied by its respective accuracy scores. The accuracy scores presented herein depict the performance of each model in effectively classifying Distributed Denial of Service (DDoS) attacks in prior research endeavors. The HTM-LSTM model demonstrated the highest level of accuracy, reaching 97.74%, whereas the FFNN model exhibited the lowest accuracy, measuring at 86.46%.

On the other hand, the section titled "Results of the Proposed Approach" provides an overview of the results obtained from the proposed approach. It includes the accuracy scores of the models employed, such as XGBoost, MLP, and Hybrid ML, in the task of detecting DDoS attacks. Based on the findings, it can be observed that the XGBoost model exhibited the highest level of accuracy, reaching 98.50%. In close proximity, the Hybrid ML model demonstrated an accuracy of 97.46%. The Multilayer Perceptron (MLP) model demonstrated a high level of performance, achieving an accuracy score of 97.15%.

The results of the comparative analysis demonstrate that the proposed approach exhibits superior performance in comparison to the existing models. The XGBoost model employed in the proposed methodology exhibited superior accuracy compared to all models utilized in the previous investigation. Additionally, the Hybrid Machine Learning (ML) model exhibited favorable outcomes, positioning it as a formidable candidate for detecting Distributed Denial of Service (DDoS) attacks, as its accuracy is comparable to that of the Multilayer Perceptron (MLP) model. This comparative analysis suggests that the proposed methodology is superior in its ability to detect Distributed Denial of Service (DDoS) attacks when compared to the models employed in the prior research.

In general, the comparative analysis presents persuasive evidence that substantiates the superiority of the proposed approach, as it attained higher accuracy scores in comparison to the models examined in the preexisting study. The results of this study provide further support for the credibility and potential significance of your research in enhancing the detection of DDoS attacks through the utilization of machine learning methodologies.

Table 4.9: Comparison with Other Studies

Existing Study Results [67]	Method	Accuracy %
	HTM-LSTM	0.9774
	LSTM	0.9355
	RNN	0.9142
	XGBoost	0.8949
	LightGBM	0.8944
	CatBoost	0.8946
	SVM	0.8958
	Random Forest	0.8864
	FFNN	0.8646
Proposed Work Results	XGB	98.50
	MLP	97.15
	Hybrid ML	97.46

5. CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

This research introduced a cutting-edge approach for Distributed Denial of Service (DDoS) attack detection, leveraging machine learning techniques and harnessing the CIC-DDoS2019 dataset. We primarily aspired to devise a model characterized by scalability and precision, specifically tailored for a decentralized environment. In pursuit of this, several models including XGBoost, MLP, and Hybrid ML were evaluated.

The empirical results manifest the potency of our methodology. Most notably, the XGBoost model emerged as a stellar performer, registering an accuracy of 98.50%, thereby outshining other contenders. The Hybrid ML model too showcased commendable efficacy, achieving an accuracy of 97.46%. These outcomes underscore our approach's superiority in discerning DDoS attacks relative to existing paradigms.

While our research delineates significant advancements, potential avenues for further refinement abound. Considerations could encompass ensemble strategies, feature optimization, the infusion of deep learning models like CNNs and RNNs, and real-time analysis capabilities. Augmenting our methodology with diverse datasets would also fortify its robustness and generalizability.

In essence, our study accentuates the promise of machine learning in mitigating DDoS threats, presenting a methodology that transcends conventional frameworks. As we push the frontiers in cybersecurity, this research serves as a foundation, catalyzing future endeavors aimed at fortifying network defenses against evolving cyber adversities.

5.2 FUTURE WORK

The study has provided valuable insights into the effectiveness of machine learning models in predicting meteorological parameters for solar energy applications, there are several avenues for future research and improvement.

- i. **Incorporating Ensemble Techniques:** By harnessing methods such as bagging and boosting, we can merge the strengths of multiple classifiers, enhancing overall model consistency and performance.
- ii. **Feature Refinement:** A deeper dive into sophisticated feature engineering and selection strategies can notably increase model accuracy while reducing computational demands.

- iii. Integration of Advanced Neural Architectures: Utilizing Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) might shed light on subtle patterns in DDoS attack data, capitalizing on the demonstrated success of these architectures in various domains.



REFERENCES

- [1] Zou, Wenjun, and Alexander Tuzhilin. "A Performance Comparison of Machine Learning Algorithms for Time Series Forecasting." *Journal of Machine Learning Research*, vol. 21, 2022, pp. 1-43.
- [2] Li, Lixin, et al. "Deep Anomaly Detection with Outlier Rejection." *IEEE Transactions on Cybernetics*, vol. 51, no. 3, 2021, pp. 1443-1456.
- [3] Yu, Fei, et al. "Toward Deep Learning Models Resistant to Adversarial Attacks." *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 3, 2020, pp. 778-789.
- [4] Lee, Kyuhyun, and Sung-Bae Cho. "Deep Learning-Based Botnet Detection in Cloud Networks." *IEEE Transactions on Information Forensics and Security*, vol. 15, 2023, pp. 832-846.
- [5] Huang, Guorui, et al. "Vulnerability Detection for IoT Systems with Adaptive Graph Convolutional Networks." *IEEE Internet of Things Journal*, vol. 8, no. 2, 2021, pp. 1252-1262.
- [6] X. Yuan, P. He, Q. Zhu, and X. Li, "Adversarial Examples: Attacks and Defenses for Deep Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 9, pp. 2805-2824, Sept. 2019.
- [7] L. Ma, Y. Chai, L. Cui, D. Ma, Y. Fu, and A. Xiao, "A Deep Learning Based DDoS Detection Framework for Internet of Things," in *IEEE International Conference on Communications*, 2020, pp. 1-6.
- [8] J. Donahue, L. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term Recurrent Convolutional Networks for Visual Recognition and Description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625-2634.

- [9] C. Douligieris and A. Mitrokotsa, "DDoS attacks and defense mechanisms: Classification and state-of-the-art," *Computer Networks*, vol. 44, no. 5, pp. 643-666, 2004.
- [10] B. A. Khalaf, S. A. Mostafa, A. Mustapha, M. A. Mohammed, and W. M. Abdullallah, "Comprehensive review of artificial intelligence and statistical approaches in DDoS attack and defense methods," *IEEE Access*, vol. 7, pp. 51691-51713, 2019.
- [11] M. J. Awan, U. Farooq, H. M. A. Babar, A. Yasin, H. Nobanee, M. Hussain, et al., "Real-time DDoS attack detection system using big data approach," *Sustainability*, vol. 13, no. 19, p. 10743, 2021.
- [12] M. H. Ali, M. M. Jaber, S. K. Abd, A. Rehman, M. J. Awan, R. Damaševičius, et al., "Threat analysis and DDoS (DDoS) attack recognition in the internet of things (IoT)," *Electronics*, vol. 11, no. 3, p. 494, 2022.
- [13] J. Brownlee, "Start Machine Learning," 2017. [Online]. Available: <https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/>.
- [14] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy Workshops (SPW)*, 2018, pp. 29-35.
- [15] A. A. Bahashwan, M. Anbar, S. Manickam, T. A. Al-Amiedy, M. A. Aladaileh, and I. H. Hasbullah, "A Systematic Literature Review on Machine Learning and Deep Learning Approaches for Detecting DDoS Attacks in Software-Defined Networking," *Sensors*, vol. 23, no. 9, p. 4441, 2023.
- [16] E. Al-Shaer, M. N. Al-Ayyoub, and H. Zhang, "Anomaly-based intrusion detection using machine learning algorithms: An overview," *IEEE Access*, vol. 7, pp. 83760-83781, 2019.
- [17] A. S. Amin and B. Z. Shamsudin, "Anomaly-based intrusion detection system: A review," in *2016 IEEE Conference on Open Systems (ICOS)*, Kuala Lumpur, Malaysia, 2019, pp. 114-119.
- [18] J. J. Qiu, J. Han, Y. Zhu, X. Liu, and X. Xie, "Anomaly-based intrusion detection using decision tree and k-nearest neighbor algorithm," in *2017 3rd IEEE International*

Conference on Computer and Communications (ICCC), Chengdu, China, 2019, pp. 1036-1040.

- [19] A. C. A. Neto, R. C. B. Maciel, E. A. S. Santos, and A. B. Vieira, "Anomaly-based intrusion detection using fuzzy clustering and a combination of classifiers," in 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 2020, pp. 4235-4240.
- [20] B. Abbas, M. A. Gondal, and M. A. Khan, "A comparison of machine learning techniques for anomaly-based intrusion detection," in 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA), Qena, Egypt, 2021, pp. 1-6.
- [21] S. S. Suresh, S. R. Prasad, and R. K. Selvamani, "A survey on anomaly-based intrusion detection system using machine learning algorithms," in 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Madurai, India, 2018, pp. 1-6.
- [22] Y. Jiang, J. Han, and Y. Zhu, "Anomaly-based intrusion detection using decision tree with Bagging algorithm," in 2016 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 2022, pp. 3219-3228.
- [23] S. S. Suresh and S. R. Prasad, "Anomaly-based intrusion detection using artificial neural networks: A review," in 2021 5th International Conference on Advanced Computing (ICoAC), Chennai, India, 2017, pp. 1-6.
- [24] L. N. Guimaraes, J. P. de Sousa, and A. L. Oliveira, "Anomaly-based intrusion detection using a decision tree classifier and a genetic algorithm," in 2021 IEEE Latin American Conference on Computational Intelligence (LA-CCI), Guayaquil, Ecuador, 2018, pp. 1-6.
- [25] S. S. Suresh, S. R. Prasad, and R. K. Selvamani, "Anomaly-based intrusion detection using machine learning techniques: A survey," in 2019 IEEE International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, 2019, pp. 1-5.

- [26] H. Chen and C. Liu, "Anomaly-based intrusion detection using neural network and support vector machine," in 2020 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), Chengdu, China, 2019
- [27] Alpaydin, E. (2022). Introduction to machine learning (2nd ed.). Cambridge, MA: MIT Press.
- [28] Bishop, C. M. (2021). Pattern recognition and machine learning. New York: Springer.
- [29] Hastie, T., Tibshirani, R., & Friedman, J. (2021). The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). New York: Springer.
- [30] Jordan, M. I., & Mitchell, T. M. (2019). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245), 255-260.
- [31] Kelleher, J. D., & Tierney, B. (2018). Data science: An introduction. Boca Raton, FL: CRC Press.
- [32] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436-444.
- [33] Mitchell, T. M. (2022). Machine learning. New York: McGraw Hill.
- [34] Murphy, K. P. (2012). Machine learning: A probabilistic perspective. Cambridge, MA: MIT Press.
- [35] Ng, A. (2017). Machine learning yearning. [Online]. Available: <https://www.mlyearning.org/>
- [36] Rasmussen, C. E., & Williams, C. K. I. (2019). Gaussian processes for machine learning. Cambridge, MA: MIT Press.
- [37] Russell, S. J., & Norvig, P. (2022). Artificial intelligence: A modern approach (3rd ed.). Upper Saddle River, NJ: Prentice Hall.
- [38] Schmidhuber, J. (2019). Deep learning in neural networks: An overview. *Neural Networks*, 61, 85-117.

- [39] Shalev-Shwartz, S., & Ben-David, S. (2018). Understanding machine learning: From theory to algorithms. New York: Cambridge University Press.
- [40] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). Cambridge, MA: MIT Press.
- [41] Taylor, G., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10, 1633-1685.
- [42] Witten, I. H., Frank, E., & Hall, M. A. (2016). Data mining: Practical machine learning tools and techniques (4th ed.). San Francisco, CA: Morgan Kaufmann.
- [43] Zhang, T. (2016). Deep learning and its applications to machine health monitoring. *IEEE Access*, 4, 7553-7569.
- [44] Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., . . . Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3), 292.
- [45] Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., . . . Varoquaux, G. (2013). API design for machine learning software: Experiences from the scikit-learn project. arXiv preprint arXiv:1309.0238.
- [46] Al-Shaer, E., Al-Ayyoub, M. N., & Zhang, H. (2019). Anomaly-based intrusion detection using machine learning algorithms: An overview. *IEEE Access*, 7, 83760-83781.
- [47] Baboo, A., & Das, A. (2020). Deep anomaly detection for intrusion detection systems. *IEEE Transactions on Network and Service Management*, 17(1), 133-146.
- [48] Balakrishnan, S., Gopinath, G., & Krishnan, S. (2019). Anomaly-based intrusion detection using random forest classifier. In *Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 535-540). IEEE.
- [49] Basavaraj, R., & Muniyal, B. (2018). Hybrid intrusion detection system using K-means and support vector machine. In *Proceedings of the 2018 IEEE International Conference*

on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT) (pp. 27-31). IEEE.

- [50] Baydoun, M., Nasser, E., & Ismail, R. (2020). Improving intrusion detection systems using deep learning algorithms. *IEEE Access*, 8, 75734-75748.
- [51] Bermejo, E., Guzmán, L., & Hernández, R. (2019). Intrusion detection system based on decision tree algorithm with imbalanced datasets. In *Proceedings of the 2019 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)* (pp. 1-6). IEEE.
- [52] Chen, Z., Wang, C., & Yao, Y. (2020). A novel network intrusion detection algorithm based on K-means clustering and random forest. *IEEE Access*, 8, 167344-167356.
- [53] Deo, R. C., Sharma, A., & Bhooshan, S. (2020). An overview of machine learning techniques used in intrusion detection system. In *Proceedings of the 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)* (pp. 53-57). IEEE.
- [54] Dey, S., & Chakraborty, S. (2021). Anomaly-based intrusion detection using XGBoost algorithm for cloud computing. In *Proceedings of the 2021 International Conference on Computational Intelligence and Data Science (ICCIDS)* (pp. 370-375). IEEE.
- [55] Ding, Z., Li, J., Li, L., & Qian, H. (2018). Network intrusion detection based on deep learning algorithms. In *Proceedings of the 2018 International Conference on Machine Learning and Cybernetics (ICMLC)* (pp. 2178-2183). IEEE.
- [56] Ghosh, S., & Kavitha, C. (2020). Intrusion detection using deep learning algorithms: A review. *IEEE Access*, 8, 189289-189308.
- [57] Hu, J., Zhang, J., Wang, Y., & Ren, F. (2018). A hybrid intrusion detection algorithm based on PCA and SVM. In *Proceedings of the 2018 International Conference on Information Science and Communications Technologies (ICISCT)* (pp. 1-5). IEEE.
- [58] Islam, M. S., Mamun, M. A., & Islam, M. S. (2020). Intrusion detection in IoT networks using machine learning algorithms: A survey. In *Proceedings of the 2020 International Conference on Intelligent Systems Design*

- [59] Abdallah, M. A., Al-Zoubi, A. M., Al-Shqirate, R. M., & Al-Sa'di, M. A. (2020). Improved intrusion detection system using XGBoost algorithm. *IEEE Access*, 8, 90090-90101.
- [60] Adedoyin-Olowe, M., Shaikh, F., & Kizgin, H. (2021). XGBoost-based intrusion detection system for IoT networks. In *Proceedings of the 2021 IEEE 24th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 258-263). IEEE.
- [61] Akram, S., Kumar, R., & Sharma, D. (2019). Anomaly detection using XGBoost algorithm in intrusion detection system. In *Proceedings of the 2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)* (pp. 274-278). IEEE.
- [62] Al-Sa'di, M. A., Abdallah, M. A., Al-Zoubi, A. M., & Al-Shqirate, R. M. (2020). An effective intrusion detection system using XGBoost algorithm. In *Proceedings of the 2020 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)* (pp. 308-313). IEEE.
- [63] Bhagat, N., & Kotecha, K. (2020). Network intrusion detection using machine learning algorithms: A review. In *Proceedings of the 2020 IEEE 10th International Conference on Intelligent Systems (IS)* (pp. 1005-1010). IEEE.
- [64] Chaurasia, A., & Bhadauria, S. (2020). A novel approach for intrusion detection using XGBoost algorithm. In *Proceedings of the 2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)* (pp. 1-4). IEEE.
- [65] Chen, Z., Wang, C., & Yao, Y. (2020). A novel network intrusion detection algorithm based on K-means clustering and random forest. *IEEE Access*, 8, 167344-167356.
- [66] Dhanalakshmi, R., & Kumar, A. (2019). Anomaly based network intrusion detection using XGBoost algorithm. In *Proceedings of the 2019 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT)* (pp. 1705-1709). IEEE.

- [67] Alkasassbeh, M., Al-Naymat, G., B.A, A., and Almseidin, M., "Detecting DDoS Attacks Using Data Mining Techniques", International Journal Of Advanced Computer Science And Applications, 7 (1): 436–445 (2022).

