FACULTY OF SCIENCE & TECHNOLOGY

MSc Internet of Things with Cyber Security
July 2024

Machine Learning-Driven Intrusion detection System for monitoring IoT Devices in Healthcare

by

Opajobi Ti-Oluwani

Faculty of Science & Technology

Department of Computing and Informatics

Individual Masters Project

Abstract

In the evolving landscape of healthcare, the integration of the Internet of Medical Things (IoMT) presents new opportunities and challenges, particularly in the realm of cybersecurity. This project explores the comparative performance of deep learning (DL) and machine learning (ML) algorithms in detecting anomalies within healthcare IoT datasets. Specifically, we assess Long Short-Term Memory (LSTM), Artificial Neural Networks (ANN), and Random Forest (RF) algorithms using two distinct datasets: the WUSTL EHMS 2020 dataset and the ECU-IoHT dataset. The analysis involves training and testing each algorithm on both datasets, followed by a repeated evaluation to ensure consistency and reliability. A novel ensemble algorithm combining Random Forest and Logistic Regression is developed to enhance detection accuracy. The performance metrics used to evaluate the models are accuracy, precision, recall, and F1-score. The results demonstrate the strengths and weaknesses of each approach, providing insights into their applicability in real-time healthcare security systems. The top-performing algorithm is deployed on a web application to facilitate real-time monitoring and anomaly detection in healthcare IoT environments. This deployment underscores the practical implications of the research, enabling healthcare providers to proactively manage security threats. This research contributes to the field by highlighting the effectiveness of ensemble techniques and addressing the challenges associated with emerging datasets in IoMT environments.

Dissertation Declaration

I agree that, should the University wish to retain it for reference purposes, a copy of my dissertation may be held by Bournemouth University normally for a period of 3 academic years. I understand that once the retention period has expired my dissertation will be destroyed.

Confidentiality

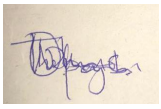
I confirm that this dissertation does not contain information of a commercial or confidential nature or include personal information other than that which would normally be in the public domain unless the relevant permissions have been obtained. Any information which identifies a particular individual's religious or political beliefs, information relating to their health, ethnicity, criminal history, or sex life has been anonymised unless permission has been granted for its publication from the person to whom it relates.

Copyright

The copyright for this dissertation remains with me.

Requests for Information

I agree that this dissertation may be made available as the result of a request for information under the Freedom of Information Act



Signed: _____.

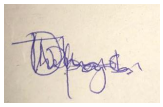
Name: Opajobi Ti-Oluwani

Date: 24/07/2024

Programme: MSc Internet of Things with Cyber Security

Original Work Declaration

This dissertation and the project that it is based on are my own work, except where stated, in accordance with University regulations.



Signed: _____.

Name: Opajobi Ti-Oluwani

Date: 24/07/2024

Acknowledgments

I am also grateful to the faculty and staff at Bournemouth University for providing the necessary resources and a conducive environment for my research.

I would like to extend my appreciation to the creators and maintainers of the WUSTL EHMS 2020 dataset and the ECU-IoHT dataset, whose data was pivotal for my research.

Finally, I am deeply thankful to my family and friends for their constant encouragement, patience, and understanding during the challenging times of my research journey.

Thank you all for your support and contributions to this work.

Best regards,

Opajobi Ti-Oluwani

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 Background of the study

The evolution of technology, from mainframe computers to the pervasive Internet of Things (IoT), has significantly transformed healthcare. IoT in healthcare, often referred to as the Internet of Medical Things (IoMT), enables continuous patient monitoring, enhances diagnostic accuracy, and improves healthcare efficiency (Oliveira et al., 2024). However, this technological advancement also introduces security and privacy challenges in healthcare networks as the integration of IoT devices in healthcare settings creates potential entry points for cyber attackers, raising concerns about data breaches, compromised patient safety, and disruption of healthcare services (Ahmed et al., 2024) .

Due to the sensitivity of the data involved, the healthcare sector has become a major target for cyber-attacks. As a result, there is a critical need for robust techniques to detect and mitigate network intrusions with precision, flexibility, and consistency.(Umamaheswaran et al., 2024). Implementing advanced Intrusion Detection Systems (IDS) tailored for the healthcare environment can play a crucial role in safeguarding patient data, ensuring the integrity of healthcare operations, and maintaining trust in digital healthcare systems. The integration of machine learning and deep learning models in IDS offers the potential to enhance detection capabilities by accurately identifying threats and attack patterns. This approach is essential in the increasingly connected and digitized landscape of healthcare, where the protection of sensitive information is paramount (Zhang et. al., 2024).

The process of monitoring, detecting, and identifying malicious activity on a network or device is known as intrusion detection (Attou et. al 2023). Intrusion Detection Systems (IDS) are categorized into two main types based on their information processing methods: signature-based IDS and anomaly-based IDS. (Kizza 2024). The signature based method identifies the attack based on a dictionary of system parameters and attack signatures but it fails to recognize zero-day attacks while the Anomaly IDS is a method that identifies and flags abnormal or suspicious activities within a network that deviate from established patterns of behavior. Unlike traditional IDS systems that rely on known attack signatures, anomaly IDS employs statistical models, machine learning algorithms, or deep learning techniques to detect deviations from normal network behavior, thus enabling the detection of unknown and zero-day attacks (Abdulganiyuet al 2024).

To address this challenge, this project proposes an anomaly-based Intrusion Detection System (IDS) utilizing Machine Learning (ML) and Deep Learning (DL) techniques. This system predicts potential cyber-attacks by continuously monitoring network traffic for anomalous and malicious

patterns. This ML-powered IDS aims to mitigate targeted threats within the healthcare sector, ultimately improving overall cyber security posture (Attou et al., 2023).

1.2 Statement of the problem

While significant progress has been made in enhancing the accuracy and innovation of Intrusion Detection Systems (IDS) for Internet of Medical Things (IoMT) healthcare networks, many existing studies have notable limitations. A considerable number of research efforts in healthcare intrusion detection rely on outdated IoT datasets, such as KDDCup 99, NSL-KDD, and UNSW-NB15 and lack patient biometric data. Models trained on these older datasets may be ineffective in detecting emerging cyber threats and may fail to represent the actual conditions of healthcare environments accurately. Some studies that have achieved promising results lack a thorough comparative analysis of both models and datasets. This gap in research hinders a complete understanding of the strengths and weaknesses of different approaches and their applicability to various real-world scenarios, especially in the fast-changing field of healthcare cybersecurity (Zhang et. Al, 2024).

Despite substantial research on intrusion detection for IoMT networks, there remains a significant gap between theoretical breakthroughs and the practical implementation of Intrusion Detection Systems (IDS).Many studies have concentrated on enhancing detection accuracy and developing new methodologies, often neglecting the crucial elements of real-world implementation and scalability. Once the best-performing algorithm is identified, the study will advance to deploying it on a web application.

To address these issues, we propose a comprehensive comparative study of intrusion detection in healthcare systems by investigating and comparing the performance of four machine learning (ML) models—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—along with two deep learning (DL) algorithms—Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) and an innovative ensemble algorithm that combines the strengths of Extreme Gradient Boosting, Random Forest, and Support Vector Machine (SVM).

This study will involve training and testing these models on two recent and relevant Internet of Medical Things (IoMT) datasets: the WUSTL EHMS 2020 dataset and the ECU-IoHT dataset. The WUSTL EHMS 2020 dataset, which includes 43 features, contains both network and patient-related biometric features which is crucial for designing an efficient and accurate Intrusion Detection System (IDS) for healthcare systems. In contrast, the ECU-IoHT dataset contains nine (9) network features, providing a different perspective on intrusion detection within IoMT environments. This dual-dataset approach aims to provide a more thorough evaluation of the models' capabilities and their applicability to real-world healthcare systems.

The objective is to provide a comprehensive analysis of intrusion detection in healthcare systems by determining the most effective approach for each dataset and assessing the consistency of ML and DL model performance across both datasets. This strategy aims to bridge the gap between research and practical application, ensuring that the most effective IDS model is not only theoretically sound but also prepared for real-world implementation in healthcare environments.

1.3 Aim and objectives of the study

This project utilizes two distinct datasets to evaluate the effectiveness of various machine learning algorithms for anomaly-based Intrusion Detection Systems (IDS) in IoT healthcare environments. The aim is to identify the most effective ML or DL approach and evaluate the consistency of their performance across different datasets. Following the evaluation, the top-performing method will be deployed in a web application to strengthen the security of IoMT devices in healthcare settings. This deployment will enhance the efficiency, accessibility, and manageability of the IDS, ensuring scalable, real-time, and comprehensive protection against security threats.

The objectives are as follows:

- Investigate the current usage of Internet of Medical Things (IoMT) devices within the healthcare sector, identify the inherent vulnerabilities in these devices, and explore the various cyber attacks they are susceptible to.
- Review the existing methods for detecting intrusions, with a focus on the necessity of employing machine learning (ML) and deep learning (DL) models in Intrusion Detection Systems (IDS).
- Develop predictive models using four different Machine Learning (ML) algorithms—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—along with two Deep Learning (DL) algorithms, Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). Additionally, create an ensemble algorithm that combines Extreme Gradient Boosting, Random Forest, and Support Vector Machine to accurately detect and classify malicious traffic targeting IoT devices in healthcare environments using the WUSTL EHMS 2020 and the ECU-IoHT datasets.
- Evaluate the performance of the predictive models to assess their effectiveness in identifying security threats. This evaluation will utilize established performance metrics, including precision, recall, F1-score, confusion matrix, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and the Area Under the Curve of the Receiver Operating Characteristic (AUC-ROC) to measure the models' accuracy and reliability in detecting and classifying security threats
- Deploy the most accurate predictive model within a real-time web application framework using Django, a Python-based framework to ensure seamless integration and practical use of the IDS. This deployment will enhance accessibility, real-time detection, alerting, log monitoring

and user-friendliness, allowing users to manage and interact with the IDS through its intuitive interface.

- Evaluate the Machine Learning-based anomaly IDS web application designed for IoMT in healthcare environments by conducting interviews with end users of the app in real-world clinical settings to get valuable insights into the application's usability, functionality, and overall effectiveness in managing healthcare-specific security threats.

1.4 Research Questions

- What are the existing techniques to detect malicious traffic in IoMT healthcare networks.
- To what extent can machine learning and deep learning algorithms identify security threats targeting IoMT devices in healthcare environments?
- How does the predictive model's performance vary when trained and tested on different datasets (WUSTL EHMS 2020 Dataset and ECU-IoHT Dataset)?

1.5 Significance of the study

The study underscores the critical need for a robust Intrusion Detection System (IDS) for the Internet of Medical Things (IoMT), given their growing significance in patient monitoring, diagnostics, and treatment. As IoMT devices become increasingly integral to modern healthcare, they face significant security threats, such as hacking, data breaches, and other malicious activities that could jeopardize patient data and disrupt healthcare services. To mitigate these vulnerabilities, the study explores the application of machine learning techniques in intrusion detection systems. By assessing a variety of machine learning and deep learning algorithms across two distinct datasets—one of which includes patient biometric data—the research aims to identify the most effective IDS solutions for real-world IoT healthcare settings. This analysis ensures that the algorithms are robust and consistent across different datasets. The deployment of the top-performing model within a web application framework represents a substantial step forward, improving the security, efficiency, accessibility, and scalability of IDS solutions for IoMT devices. This implementation is intended to provide real-time, adaptable protection against emerging security threats, thereby safeguarding patient well-being and preserving the integrity of the healthcare system.

1.6 Structure of the Project

This project is structured to develop an intelligent system aimed at predicting and mitigating cyber attacks targeting IoT devices within healthcare networks.

Chapter One introduces the project's background and significance, emphasizing the growing reliance on IoT devices in healthcare and the associated cybersecurity risks. It outlines the background, statement of problem, research questions scope of the study, and significance of the study

Chapter Two offers an in-depth literature review, synthesizing current knowledge on healthcare IoT. It investigates the deployment of Internet of Medical Things (IoMT) devices within the healthcare sector, highlighting their inherent vulnerabilities and exploring the range of cyber threats they face. The chapter also reviews existing intrusion detection methods, emphasizing the critical role of machine learning (ML) and deep learning (DL) models in enhancing Intrusion Detection Systems (IDS). This comprehensive review establishes a foundational understanding of the state-of-the-art technologies and identifies key research gaps.

Chapter Three delves into the methodology employed for the research, starting with an overview of the approach. It details the implementation of Agile methodology and the application of quantitative research techniques. The chapter describes the datasets used, including the ECU-IoHT and WUSTL-EHMS-2020 datasets, and outlines the experimental setup for both machine learning training and IDS web application. It further explains the procedure for simulation, covering data loading, processing, feature selection, and handling imbalanced data with SMOTE. The chapter then justifies the selection of various machine learning as well as deep learning algorithms and concludes with the evaluation metrics, deployment of the web application.

Chapter four presents the results and findings of the research, beginning with an overview of the outcomes. It provides a detailed comparison of model performance on the ECU-IoHT dataset and the WUSTL-EHMS 2020 dataset, highlighting key metrics and insights. The chapter further compares the performance of models across both datasets and assesses the proposed system against related works. This comprehensive analysis aims to demonstrate the effectiveness and comparative advantage of the proposed method in the context of existing research.

Chapter five focuses on the artifact of the research, specifically evaluating the Machine Learning Anomaly Intrusion Detection System (IDS) web application. It begins with an assessment of the web application's performance, incorporating feedback from interviews. The chapter includes a detailed analysis of responses to the interview questions, reflecting on the practical implications of the IDS, emphasizing how the system performs in real-world IoMT settings. Visual aids included in the chapter further illustrate the application’s design and user experience, rounding out the chapter’s focus on the IDS’s real-world applicability and effectiveness.

Chapter 6 highlights the conclusion and future work of the project.

2 LITERATURE REVIEW

2.1 Overview of IoT in healthcare networks

The development of the Internet of Things (IoT) is ushering in unparalleled innovation in healthcare transforming the practice of healthcare, and fostering progress in medical research. (Singh and Kaunert 2024). Wearable fitness trackers, smartwatches, and medical sensors and actuators are used to gather critical health data such as heart rate, blood pressure, and body temperature, playing a key role in health monitoring, medication adherence and early disease detection. These devices collect and transmit clinical data from remote locations for healthcare, facilitating a shift from a hospital-centric model to a more patient-centric approach (Yadav, 2024). The integration of the Internet of Medical Things (IoMT), remote patient monitoring (RPM), electronic health records (EHR), cloud infrastructure, and imaging centers has transformed healthcare systems into a centralized hub for all patient information and medical data. This synergy enables real-time monitoring, automated data collection, and improved disease management. As a result, healthcare providers can actively monitor patients' well-being and intervene when necessary, significantly enhancing disease management, reducing hospital admissions, and improving patient access to care (Arun, 2023).

2.2 IoT devices deployed in healthcare

IoT technologies, such as wearables and sensors, enable continuous patient monitoring, prompt detection of potential health problems, and swift interventions to treat or prevent health complications (Vats et. Al, 2023).

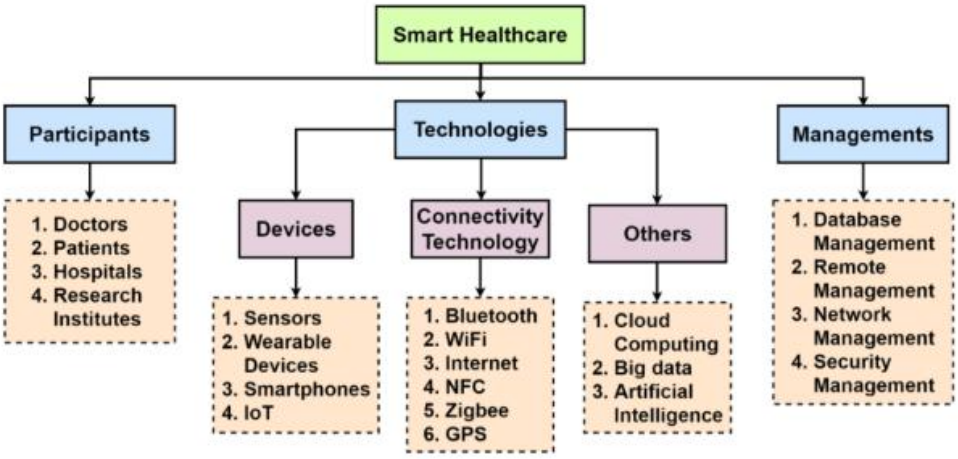


Figure 1: Chart Showing the components of smart healthcare; source: (Singh et. al, 2021).

Some IoMT devices that are deployed in healthcare are listed below;

- Wearable Devices: These are devices, worn on the body as watches, earbuds, or glasses to collect data on physiological, environmental, location, and behavioral measures. Some are designed for specific conditions like diabetes, heart disease, or respiratory issues to monitor heart rate, respiration, sleep patterns, and activity levels in real-time, aiding in diagnosis and health monitoring (Hosseini et. al, 2023).
- Telemedical IoMTDevices: These devices are during remote healthcare procedures for medical imaging, videoconferencing, and messaging to connect patients with healthcare providers, facilitating remote consultations and care (Suleiman and Adinoyi 2023).
- IoMT electronic healthcare record (EHR) devices: These devices are used to enhance healthcare by continuously collecting and transmitting detailed healthcare data providing a comprehensive view of healthcare data (Chopade et. al, 2023).
- Ingestible Medical Sensors: These sensors can be ingested as pills, implanted into the body, or placed around the patient to gather health information and monitor their condition in real-time (Anand et. al, 2024).

2.3 Healthcare's IoT Security Landscape

The rise in deployment of IoT devices in healthcare despite its unparalleled benefits introduces vulnerabilities that expose healthcare organizations, patients, and practitioners to cyber attacks (Patil et. al, 2024). As new devices proliferate and integrate with the Internet of Things (IoT), they automate the process of generating and exchanging valuable patient information and medical records over the Internet (Tyagiet. al, 2024). While automation has improved patient care workflows and reduced costs, integrating IoT devices into healthcare systems raises significant concerns about the confidentiality, integrity, availability, security, and privacy of healthcare data (Atadoga et. al, 2024). The increasing use of connected medical devices (IoT), telemedicine, and electronic health records (EHRs) expands the attack surface, making them prime targets for cybercriminals seeking to steal valuable data (Patil et. al, 2024).

According to (Arun et. Al, 2023), IoT device manufacturers, in their effort to deliver cost-effective solutions on tight timelines, often focus functionality and don't prioritize the security of these devices. these devices frequently have inadequate security measures, lacking fundamental authentication, trust, and anomaly detection methods, thus creating vulnerabilities in the devices, making them susceptible to potential exploitation by cyber attackers.

According to (Zhang et. Al, 2024) In May 2017, the "WannaCry" ransomware attack struck 48 NHS-affiliated healthcare facilities in the UK, causing significant disruptions such as hospital shutdowns, surgery cancellations, and emergency patient redirections. During the COVID-19 pandemic, there was a dramatic increase in cyberattacks, with incidents rising fivefold compared to

the previous year. Ransomware attacks targeting healthcare institutions, including hospitals, clinics, outpatient centers, and laboratories, surged by 45%.

The security of IoMT is a critical concern that must be addressed from the outset to ensure its effective adoption and continued exponential growth in healthcare. Cyber threats have evolved with attackers using increasingly sophisticated methods. Reactive security measures, which respond only after an incident has occurred, often struggle to keep pace with the speed and adaptability of modern cyber threats. Proactive security measures, on the other hand, aim to identify and address vulnerabilities before they can be exploited by attackers, thereby enhancing resiliency. As cyber threats become increasingly sophisticated, healthcare systems must ensure that their security measures are not only effective in real-time but also adaptable to evolving attack vectors. This dual focus on immediate response and long-term fortification is crucial to safeguarding patient data, maintaining the integrity of medical services, and ensuring continuous care delivery.(Okoli et. al, 2024).

2.4 Common vulnerabilities of IoT devices in healthcare

The deployment of Internet of Things (IoT) devices in healthcare has ushered in a new era of interconnected medical systems, enhancing patient monitoring, diagnostics, and treatment. However, this technological advancement is accompanied by significant vulnerabilities that must be carefully managed to ensure patient safety and data security. Some vulnerabilities associated with deploying IoT devices in healthcare include:

- Vulnerable Firmware and Software: IoMT devices often suffer from irregular updates and patching practices. These outdated or unpatched firmware and software exposes them to vulnerabilities which could be exploited by attackers (Bajpai et. al, 2024).
- Physical Security Risks: IoMT devices deployed in healthcare can be physically tampered with if they are located in publicly accessible areas that are not secure (Aditya et al., 2024).
- Weak, Guessable, or Hard-Coded Passwords: Some IoMT devices come with default passwords that are easy to guess or discover, while others have hard-coded passwords that cannot be changed, making them vulnerable to attackers (Shambharkar and Sharma, 2024).
- Weak Authentication and Authorization: IoMT devices often have insufficient mechanisms for verifying user identities and granting access which can lead to unauthorized access to sensitive data and system controls (Ryan and Rozier, 2024).
- Lack of Proper Encryption: Some IoT devices lack adequate encryption. Poorly encrypted data transmitted between devices can be intercepted and manipulated by attackers, leading to data breaches and unauthorized access to sensitive information (Ryan and Rozier, 2024).

2.5 Common attacks on IoMT devices in healthcare

Common cyberattacks targeting healthcare IoMTare listed below;

- Data Breaches: IoMT devices in healthcare gather sensitive patient information, including vital signs, medications, and treatment plans. Hackers exploit security vulnerabilities to access this confidential data, resulting in major breaches. The stolen information is frequently used for identity theft, sold on black markets, or employed in blackmail against patients and healthcare providers. (Siwakoti et. al, 2023).
- Man-in-the-Middle (MITM) Attacks: In MITM attacks, an attacker intercepts communications between two parties without their knowledge, the attacker can alter or steal sensitive data being exchanged between IoMT devices and healthcare systems. (Siwakoti et. al, 2023).
- Denial-of-Service (DoS) Attacks: IoT devices can be targeted in DoS attacks where they are flooded with excessive requests to exhaust their resources thereby causing performance degradation or failure leading to disruption of critical services, delay patient care, and denial of access to health services. (Siwakoti et. al, 2023).
- Malicious Firmware Updates: Attackers distribute compromised firmware updates that introduce vulnerabilities or give them control over connected devices. This malicious firmware enables attackers to manipulate data, disrupt healthcare operations, or facilitate further attacks within the network. (Inuwa and Das, 2024).
- Supply Chain Attacks: Counterfeit or substandard IoMT devices in the healthcare supply chain can embed vulnerabilities like malware or backdoors. These compromised devices can be exploited by attackers to gain access to healthcare networks and sensitive data, undermining the integrity of the entire system (Inuwa and Das, 2024).
- Physical Attacks: Physical access to IoMT devices can allow attackers to bypass security measures, extract sensitive information, or disrupt healthcare services (Inuwa and Das, 2024).

2.6 Existing Techniques to detect malicious traffic in IoMT

Existing techniques to detect malicious traffic in the Internet of Medical Things (IoMT) encompass a range of approaches, including signature-based, anomaly-based, and hybrid detection methods. Signature-based detection relies on known patterns to quickly identify threats but can be less effective against novel attacks. Anomaly-based detection monitors network traffic for deviations from normal behavior, allowing for the identification of new threats but with the risk of false positives. Hybrid methods combine these techniques, often incorporating machine learning and deep learning models to improve detection accuracy and adaptability to emerging threats. In addition to these, technologies like intrusion detection systems (IDS), intrusion prevention system (IPS), Firewalls, deep packet inspection (DPI), network traffic analysis tools, log monitoring, and threat management systems are widely employed to detect malicious traffic. IDS, in particular, is a modern and effective tool for identifying various network attacks and security issues in IoMT, leveraging a combination of behavioral analysis, machine learning, and threat intelligence to provide robust security in dynamic healthcare environments. (Shambharkar and Sharma, 2024).

2.7 Intrusion Detection Systems

Intrusion is an attempt to breach or gain unauthorized access to a system, network, or data. This act is often part of a series of actions designed to threaten the integrity, confidentiality, or availability of an information resource. Intrusion Detection System (IDS) continuously monitors networks for suspicious activities that could indicate an ongoing or imminent cyber attack by analyzing network traffic, system logs, and device behavior to identify anomalies that deviate from established patterns (Altulaihan, 2024).

As the digital landscape becomes more interconnected and complex, healthcare must shift from merely responding to incidents to actively anticipating and preventing potential threats.
This research presents an innovative approach to strengthening network security by leveraging machine learning and deep learning-based threat intelligence. Traditional signature-based Intrusion Detection Systems (IDS) face substantial limitations in detecting zero-day attacks and adapting to the evolving threat landscape (Makanju et al., 2024). In contrast, machine learning and deep learning technologies excel at identifying zero-day threats by recognizing intricate patterns and relationships within data. This study explores the application of these advanced algorithms as proactive tools for detecting anomalous patterns and behaviors that signal potential cyber threats, thereby enhancing cybersecurity within healthcare environments (Bai and Fang, 2024).

IDS can be categorized based on their deployment type as either host-based or network-based, and based on their approach to attack detection as either signature-based or anomaly-based (Kizza, 2024).

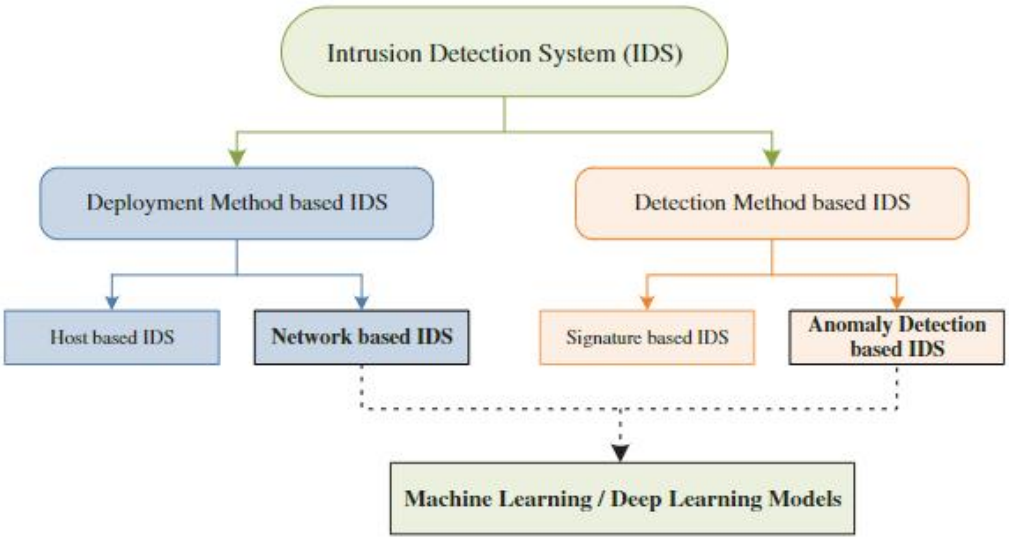


Figure 2: Chart showing IDS framework; Source: (Ahmad et. al, 2021)

Network Intrusion Detection System (NIDS) and Host-based Intrusion Detection System (HIDS) are two main ways of deploying IDS. NIDS is usually a standalone device for this purpose or

embedded into firewalls to monitor network traffic for malicious activity. The HIDS is installed on endpoint devices or servers to analyze activity on that specific device (Ghadermazi and Bastian, 2024).

Signature-based and Anomaly-based are two approaches that are used to detect cyber attacks. An IDS detects attacks by either searching for signatures of known threats or identifying deviations from a predefined profile of normal activity (Satilmiş et al., 2024). Signature-based IDS, or "knowledge-based intrusion detection," identifies attacks by matching patterns against a stored database of known signatures. If incoming data matches an existing signature, the system flags it as malicious. This method is effective at identifying known attacks with minimal false positives but fails to detect novel or zero-day attacks due to its reliance on the existing database. In contrast, anomaly-based IDS, or "behavior-based IDS," detects attacks by looking for deviations from normal system behavior. While this method can identify new and zero-day attacks, it tends to generate more false positives, which can limit its effectiveness. (Abdulganiyu et. al, 2023).

2.8 Related Works

Based on the analysis of previously published papers, this section discusses the significant findings and ongoing research challenges concerning ML/DL-based intrusion detection methods.

(Kumaar et al, 2022) highlighted the advantages of integrating machine learning-based intrusion detection systems (ML-IDSs) into smart healthcare environments. They emphasized that such systems could significantly bolster the security of medical IoT devices and, consequently, improve patient outcomes. (Neto et al. 2023) investigated machine learning (ML) solutions for enhancing IoT security in healthcare, emphasizing the critical need for new datasets to advance innovative methods for addressing emerging threats. They noted that the lack of standardized security protocols across diverse IoT devices and platforms complicates the creation of a uniform Intrusion Detection System (IDS). Additionally, they pointed out that the scarcity of publicly available datasets reflecting cyberattacks on the Internet of Healthcare Things (IoHT) is largely due to privacy concerns.

(Tauqeer al. 2023) demonstrated that incorporating both network and biometric measurements as features enhances IDS performance. (Zhang et. Al, 2023) highlighted the effectiveness of feature selection in creating a robust Intrusion Detection System (IDS) for healthcare environments and emphasized the critical importance of incorporating patient biometric features into IDS to enhance detection accuracy and system performance.

(Hady et al. 2023) used the WUSTL-EHMS-2020 dataset to assess various machine learning models and classifier algorithms. Their approach achieved an accuracy of 91.56% and an AUC of

0.8748. However, they identified that the low AUC was due to class imbalance issues within the dataset that were not addressed. The study highlighted the critical need for balancing samples to enhance the performance of intrusion detection systems.

(Kattankulathur et. Al, 2023) presented an advanced LSTM-based intrusion detection system (IDS) designed for IoT networks, utilizing advanced deep learning techniques to effectively detect network intrusions and enhance security. The study emphasized the importance of meticulous data preprocessing methods, including the synthetic minority over-sampling technique (SMOTE) to address data imbalance and recursive feature elimination (RFE) for optimized feature selection, which enabled precise differentiation between normal and malicious activities. The LSTM architecture achieved notable accuracy rates of 99.34% on CICIDS2017, 99.67% on NSL-KDD, and 98.31% on UNSW-NB15. The research underscored the significance of effective data preparation and deep learning methodologies in IoT intrusion detection.

(Altulaihan et. al, 2024) utilized four supervised classification algorithms—Decision Tree (DT), Random Forest (RF), K Nearest Neighbor (kNN), and Support Vector Machine (SVM)—with two feature selection techniques, Correlation-based Feature Selection (CFS) and Genetic Algorithm (GA). Using the IoTID20 dataset, DT and RF classifiers had the best performances when trained with features chosen by GA. The findings suggest that machine learning approaches can effectively detect anomalies with adequate training, though performance may differ depending on the specific algorithm used.

(Khan and Alkhathami, 2024) used the Canadian Institute for Cybersecurity (CIC) CICIDS2017 IoT dataset to model machine learning techniques for the detection of anomalous network traffic using Random Forest, Adaptive Boosting, Logistic Regression, Perceptron, and Deep Neural Networks. The Random Forest model demonstrated optimal performance for both binary and multiclass classification of IoT attacks, achieving approximately 99.55% accuracy. This high accuracy was accompanied by a reduced computational response time, highlighting the necessity of low latency for effective real-time attack detection and response in healthcare settings.

(Obeidat et al. 2024) tested five machine learning classifiers on the NSL-KDD dataset, which has known limitations in representing contemporary network traffic. The classifiers evaluated included Decision Tree, Random Forest, Multilayer Perceptron, Naïve Bayes, and Bayes Network. They found that the Random Forest classifier achieved the best result and outperformed the other classifiers in their study.

(More et. al, 2024) applied various machine learning algorithms to the UNSW-NB15 network traffic dataset and found that the Random Forest model was the most effective for detecting cyber-

attacks. Their performance and confusion matrix results revealed that this model achieved an impressive F1 score of 97.80%, an accuracy of 98.63%, and a low false alarm rate of 1.36%. These results highlight the Random Forest model as a strong candidate for enhancing IDS system security. (Binbusayyis et. alm 2022) compared five machine learning algorithms—Naive Bayes (NB), K-Nearest Neighbors (KNN), Decision Tree (DT), Artificial Neural Networks (ANN), and Support Vector Machine (SVM) using the Bot-IoT datasets to evaluate their effectiveness for developing Intrusion Detection Systems (IDS) in Internet of Medical Things (IoMT) networks. Their findings revealed that the Decision Tree (DT) algorithm outperformed the others in terms of intrusion detection.

2.9 Research Gap and Contribution

Although many advanced studies have investigated intrusion detection in healthcare systems, they frequently utilize outdated IoT datasets such as KDDCup 99, NSL-KDD, and UNSW-NB15, which lack essential patient biometric data, including temperature, blood pressure, heart rate, and ECG. Consequently, models trained on these older datasets may be ineffective in detecting emerging cyber threats and may fail to represent the actual conditions of healthcare environments accurately. These datasets often contain features that do not contribute to the intrusion detection process, and some suffer from imbalances in their feature distributions. Some studies that have achieved promising results lack a thorough comparative analysis of both models and datasets. This gap in research hinders a complete understanding of the strengths and weaknesses of different Intrusion detection system approaches and their applicability to various real-world scenarios, especially in the fast-changing field of healthcare cybersecurity .

Despite significant advancements in research aimed at improving the precision and novelty of Intrusion Detection Systems (IDS) for Internet of Medical Things (IoMT) networks, a major gap remains between theoretical breakthroughs and practical implementation. Current research often prioritizes enhancing detection accuracy and exploring new methodologies, while largely overlooking crucial aspects related to real-world deployment and scalability. This oversight hampers the effective integration of IDS into practical IoMT healthcare environments, where the ability to deploy and scale solutions effectively is as important as their detection capabilities.

To bridge the gap between theoretical advancements and practical application in Intrusion Detection Systems (IDS) for Internet of Medical Things (IoMT) networks, this study conducts a comparative evaluation using four Machine Learning (ML) algorithms—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—along with two Deep Learning (DL) algorithms, Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). Additionally, an innovative ensemble algorithm combining Extreme Gradient Boosting, Random Forest, and Support Vector Machine is tested. This evaluation is

carried out on two recent healthcare datasets: the WUSTL EHMS 2020 dataset, which integrates both network and patient biometric features, and the ECU-IoHT dataset, which focuses solely on network features. This approach provides a comprehensive analysis of intrusion detection within IoMT environments, aiming to identify the most effective method for each dataset and assess the consistency of model performance across both datasets. By leveraging this dual-dataset strategy, the study aims to ensure that the IDS models are not only theoretically sound but also ready for practical implementation in real-world healthcare settings.

Recursive Feature Elimination (RFE) combined with the Random Forest classifier (RFC) was utilized to pinpoint the most important features for intrusion detection in datasets with an excessive number of features and the Synthetic Minority Over-sampling Technique (SMOTE) was applied to resample the data effectively. to address class imbalance within the datasets.

Furthermore, this study deployed an anomaly-based IDS on a web application, enhancing its accessibility and real-time detection capabilities. This deployment aims to provide an intuitive interface for users to manage and interact with the IDS, improving its usability, scalability, and effectiveness in real-world healthcare environments. By automating responses, alerts, and threat containment, the system aims to offer dynamic, real-time protection against cybersecurity incidents in IoMT networks.

2.10 Summary

In this chapter, I provided a comprehensive overview of the Internet of Things (IoT) in healthcare networks, examining the various IoT devices deployed within this sector and their significant impact on healthcare delivery. I explored the security landscape of healthcare IoT, identifying common vulnerabilities and the types of attacks that target the Internet of Medical Things (IoMT) devices. These insights underscore the critical need for robust security measures. I reviewed existing techniques for detecting malicious traffic in IoMT, with a particular focus on Intrusion Detection Systems (IDS). The discussion highlighted the growing importance of Machine Learning (ML) and Deep Learning (DL) based IDS in safeguarding healthcare IoT environments, given their ability to adapt to evolving threats. The chapter also examined related works in the field, identifying key approaches and methodologies previously applied to IoT security in healthcare. Through this review, I identified research gaps. This chapter set the stage for the research by outlining the current state of IoT security in healthcare, the limitations of existing solutions, and the contributions our work aims to make in addressing these challenges.

3 METHODOLOGY

3.1 Overview

This chapter outlines the methodology used to develop the anomaly intrusion detection system for Internet of Medical Things (IoMT) in healthcare within this research project. It starts with an introduction to the study's framework and then covers an overview of the dataset employed, data preprocessing, feature selection, model selection, and evaluation techniques. Each step is designed to address the specific characteristics of the dataset to ensure precise and dependable anomaly detection. The research paradigm for this project is aimed to give a systematic approach to predicting potential cyber attacks on IoT devices in healthcare networks. This paradigm combines quantitative and agile methodology (extreme programming) to build and evaluate machine learning models for the ensemble anomaly IDS.

3.2 Implementation of Agile Methodology

The evolution of cyber threats has outpaced traditional security measures, which struggle to keep up with the rapid progression in complexity, diversity, and sophistication . From simple malware to advanced persistent threats, the threat landscape has become increasingly complex, necessitating continuous innovation in cybersecurity practices (Zaid and Garai, 2024). This project aims to employ agile framework for deploying an anomaly intrusion detection system (IDS) to address these dynamic and adaptive security challenges. Agile is a customer-centric project management and software development approach that emphasizes flexibility, collaboration, and continuous improvement / continuous development (CI/CD) prioritizing adaptability over rigid planning and embracing change throughout the project life-cycle (Lakshminarasimham, 2024). Agile prioritizes delivering working software that provides value to the customer early and often, allowing for continuous feedback and refinement. Projects are broken down into smaller, iterative cycles that involves repeated cycles of planning, executing, and reviewing small, manageable increments of work to continuously improve and adapt to changing requirements (Wood, 2024). There are various frameworks that implement the core principles of Agile (Daraojimba et. al, 2024). Some popular examples include:

- Scrum is a framework designed to address complex problems by providing structured solutions. It involves key roles such as the Scrum Master, who oversees the process; the Product Owner, who represents the stakeholders; and the Development Team, which is dedicated to producing functional product increments. Scrum operates in short, time-boxed cycles called sprints, allowing for the delivery of functional increments. Its flexibility makes it well-suited for projects with changing requirements(Kalyani & Mehta, 2023).
- Kanban, in contrast to Scrum, emphasizes continuous delivery without prescribing time-boxed iterations. It uses a visual board to track workflow and tasks, making it ideal for environments with frequently changing work requirements (Camara and Marinho, 2024).

- Extreme Programming (XP) focuses on engineering practices to enhance software quality and adapt to evolving customer requirements. XP practices include test-driven development (TDD), continuous integration, continuous development which involves frequently integrating code changes, and refactoring. XP prioritizes customer involvement and feedback to collect input and make necessary adjustments to softwares.(Fernandez, 2024)

In my project, I adopted Extreme Programming (XP) to ensure a high-quality, adaptive, and robust development process for the machine learning anomaly intrusion detection system. By adhering to XP principles, I implemented small, frequent releases of the IDS, deploying each algorithm individually for thorough evaluation before integrating them for ensemble model and deploying to the web application. I utilized test-driven development (TDD) to ensure the robustness of the algorithms and models. Continuous integration practices facilitated the seamless incorporation of new features and models. This iterative approach enabled the development of a sophisticated and adaptive machine learning-based IDS capable of effectively predicting anomalies in real-time.
‘

3.3 Quantitative Research

This research applies quantitative research methodologies , using machine learning and algorithms to predict potential cyber attacks on IoT devices in healthcare network (Inuwa and Das, 2024). This approach involves the systematic collection and analysis of two secondary IoT Healthcare datasets ECU-IoHT dataset and WUSTL-EHMS-2020 dataset to train and test Machine learning algorithms within the IDS framework. The performance of the algorithms were further evaluated using operational metrics such as accuracy, precision, recall, f1-score, confusion matrix, true positive, false positive, true negative, false negative, and Area Under the Curve - Receiver Operating Characteristic (AUC-ROC) to compare the performance of different algorithms, with different datasets to determine which combination yields optimal and consistent results. Leveraging on quantitative research principles, this framework provides empirical evidence for comparing classifiers and establishing effective strategies for predicting potential cyber attacks on IoT devices inside healthcare networks contributing to advancements in cybersecurity defenses against evolving threats (Neto et. al, 2023).

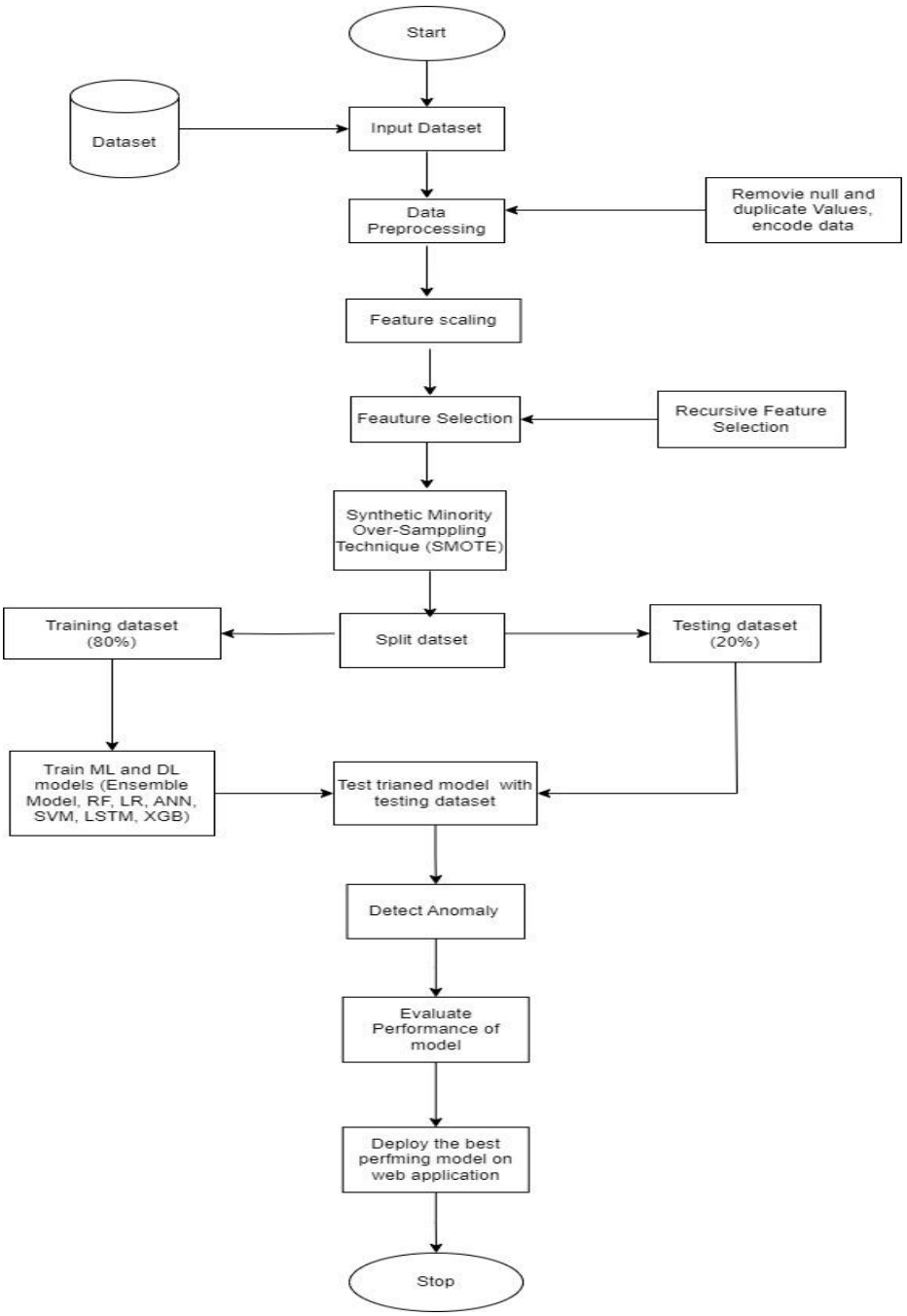


Figure 3: Flowchart architecture of this Project

3.4 Datasets Used

Two recent publicly available realistic datasets were used; ECU-IoHT dataset and WUSTL EHMS 2020. The WUSTL EHMS 2020 dataset entails network flow and biometrics information which are used to create realistic intrusion analysis while the ECU-IoHT contains solely network data. These datasets provide an extensive set of features and reflect current trends and threats in IoT healthcare security, making them highly relevant for the healthcare sector's unique security challenges. (Alsalman, 2024).

3.5 Description of Datasets Used

3.5.1 ECU-IoHT dataset

The ECU-IoHT is a healthcare-related dataset built in the Internet of Health Things (IoHT) environment. It has 9 features which is fewer compared to the WUSTL-EHMS 2020 dataset. The ECU-IoHT dataset is a valuable resource for for analyzing healthcare networks and is accessible via o https://ro.ecu.edu.au/datasets/48/

Measurement	Value	Percentage
Number of normal samples	23,453	21.09%
Number of attack samples	87,754	78.91%
Total number of samples	111,207	100%

Table 1: showing the category of samples in the ECU-IoHT dataset

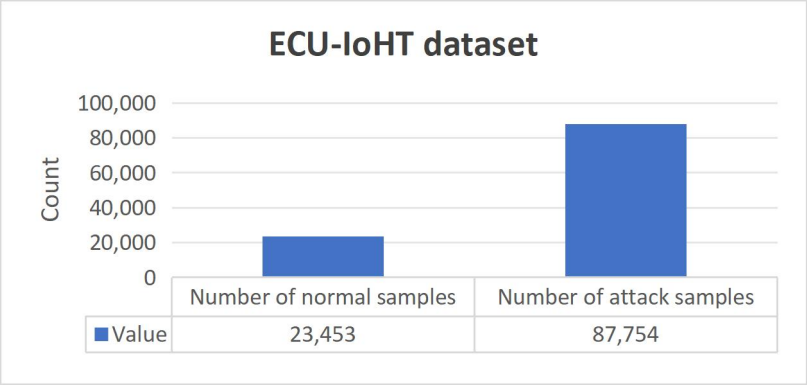


Figure 4: showing the distribution of the target feature (Type) in the ECU-IoHT dataset

Feature	Description
No	Number of the traffic entry
Time	The timestamp indicating when the network packet or data flow was captured.
Source	The IP address of the originating device where the data packet was sent.
Destination	The IP address of the target device where the data packet is intended to be sent.
Protocol	The communication protocol used for the data transfer
Length	The size of the data packet or message in bytes. This feature can be useful in detecting anomalies or unusual traffic patterns.
Info	Information about the traffic content
Type	Classification of attack, normal or attack
Type of attack	The category of the detected intrusion or anomaly, such as DDoS, phishing,

	malware

Table 2: Showing the feature and description of ECU-IoHT dataset

3.5.2 WUSTL-EHMS-2020 dataset

The WUSTL-EHMS-2020 dataset, developed by Washington University in St. Louis (WUSTL) using a real-time EHMS test-bed, includes not only network features but also crucial patient biometric data, such as temperature, blood pressure, heart rate, SpO2, and ECG. This combination of data is vital for creating an effective and accurate Intrusion Detection System (IDS) for healthcare environments. The dataset contains 16,318 records detailing attack statuses (labeled as "Attack" or "Normal"), with 35 features representing network flow metrics and 8 features corresponding to patient biometrics. The WUSTL-EHMS-2020 dataset is a valuable resource for understanding network behaviors in the context of the Internet of Things (IoT) deployed in healthcare and is accessible via https://www.cse.wustl.edu/~jain/ehms/index.html

Measurement	Value	Percentage
Number of normal samples	14,272	87.5%
Number of attack samples	2,046	12.5%
Total number of samples	16,318	100%

Table 3: showing the category of samples in the WUSTL-EHMS-2020 dataset

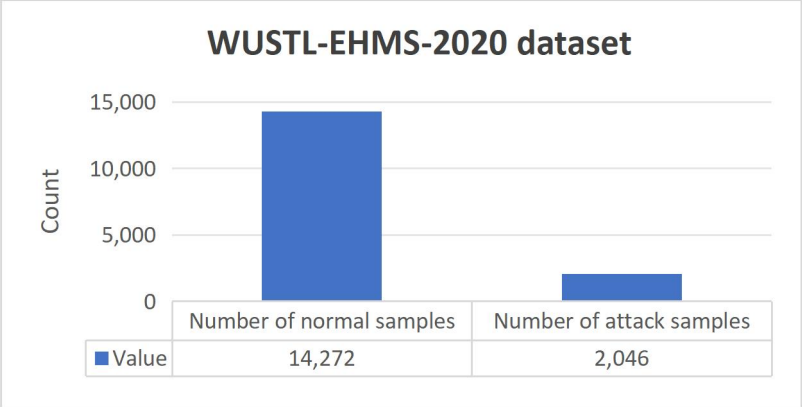


Figure 5: showing the distribution of the target feature (Type) in the WUSTL-EHMS 2020 dataset

Feature	Description
Dir	Direction of the traffic (e.g., incoming or outgoing).
Flgs	Flags indicating various conditions or attributes of the traffic (e.g., SYN, ACK flags in TCP).
SrcAddr	Source IP address from which the traffic originated.

DstAddr	Destination IP address to which the traffic is directed.
Sport	Source port number from which the traffic originated.
Feature	**Description**
Dport	Destination port number to which the traffic is directed.
SrcBytes	Number of bytes sent from the source.
DstBytes	Number of bytes sent to the destination.
SrcLoad	Load or bandwidth usage by the source.
DstLoad	Load or bandwidth usage by the destination.
SrcGap	Gap or delay between packets from the source.
DstGap	Gap or delay between packets to the destination.
SIntPkt	Source inter-packet arrival time.
DIntPkt	Destination inter-packet arrival time.
SIntPktAct	Source inter-packet active time.
DIntPktAct	Destination inter-packet active time.
SrcJitter	Variation in packet arrival time from the source.
DstJitter	Variation in packet arrival time at the destination.
sMaxPktSz	Maximum packet size sent from the source.
dMaxPktSz	Maximum packet size sent to the destination.
sMinPktSz	Minimum packet size sent from the source.
dMinPktSz	Minimum packet size sent to the destination.
Dur	Duration of the traffic flow.
Trans	Type of transmission protocol (e.g., TCP, UDP).

TotPkts	Total number of packets in the traffic flow.
TotBytes	Total number of bytes in the traffic flow.
Feature	**Description**
Load	Overall load or bandwidth usage in the traffic flow.
Loss	Number of lost packets in the traffic flow.
pLoss	Percentage of packets lost.
pSrcLoss	Percentage of packets lost from the source.
pDstLoss	Percentage of packets lost to the destination.
Rate	Rate of packet transmission.
SrcMac	Source MAC address from which the traffic originated.
DstMac	Destination MAC address to which the traffic is directed.
Packet_num	Packet sequence number in the traffic flow.
Temp	Temperature measurement (possibly related to a sensor).
SpO2	Oxygen saturation level (possibly related to a health monitoring sensor).
Pulse_Rate	Pulse rate (possibly related to a health monitoring sensor).
SYS	Systolic blood pressure (possibly related to a health monitoring sensor).
DIA	Diastolic blood pressure (possibly related to a health monitoring sensor).
Heart_rate	Heart rate measurement (possibly related to a health monitoring sensor).
Resp_Rate	Respiration rate (possibly related to a health monitoring sensor).
ST	Possibly referring to segment time or status (context-specific interpretation required).
Attack	Indicates if the traffic flow is associated with an attack (yes/no or type of attack).
Category	Category of the traffic flow, which may relate to the type of application, service, or traffic class.

Label	Label indicating the nature of the traffic flow (e.g., normal, suspicious, attack type).

Table 4: Showing the feature and description of WUSTL-EHMS-2020 dataset

3.6 Experimental Setup

The experimental work was conducted on a Windows 10 (64-bit) system with the following hardware specifications:

RAM: 16 GB

CPU: Intel Core i5-7200U @ 2.50 GHz

3.6.1 Experimental setup for ML training and evaluation

The ML model training and evaluation was implemented as a simulation on Google Colab, leveraging its cloud-based environment to run Python code without local installation requirements. This allowed for easy access to powerful computational resources and facilitated collaboration. Python was used as the programming language due to its popularity and robust ecosystem for data analysis and machine learning. The following libraries were utilized for the MLtraining and evaluation:

- Pandas: A powerful library for data analysis and manipulation, especially useful for working with structured data such as csv files and Data frames.
- Scikit-learn: A comprehensive library that provides a wide range of machine learning algorithms, tools for model training, evaluation, and preprocessing tasks.
- NumPy: A fundamental library for numerical computations, providing support for large, multi-dimensional arrays and matrices.
- Seaborn: A statistical data visualization library built on top of Matplotlib, enhancing the capability to create informative graphics.
- Matplotlib: A widely used library for creating static, animated, and interactive visualizations in Python.
- TensorFlow: A deep learning framework that allows you to build and train neural networks.

3.6.2 Experimental set up for IDS web application

The IDS web application was developed using Visual Studio Code and implemented with Django, a Python framework. The templates for each page, the routing and rendering logic, the necessary libraries, the machine learning ensemble model, and the server were all implemented locally. After development, the application was deployed on PythonAnywhere, a cloud hosting platform. The ML anomaly IDS can be accessed via https://cyberattackiotprediction.pythonanywhere.com. The following libraries were used to develop the IDS web application:

- asgiref (3.8.1): provides an ASGI (Asynchronous Server Gateway Interface) reference implementation, in Django for handling asynchronous requests.
- Django (5.0.8): A high-level Python web framework that encourages rapid development and clean, pragmatic design.

- joblib (1.4.2):Used to run Python functions as parallel tasks and speed up the execution of code, especially in machine learning.
- numpy (2.0.1): A basic package for numerical computing in Python, providing support for large, multi-dimensional arrays and matrices.
- scikit-learn (1.5.1): A machine learning library in Python, offering simple and efficient tools for data mining and data analysis.
- scipy (1.14.0): Python library for scientific and technical computing, built on NumPy,
- sqlparse (0.5.1): A non-validating SQL parser for Python, used to format and parse SQL queries.
- threadpoolctl (3.5.0): Controls the number of threads used by libraries relying on multi-threading.
- typing_extensions (4.12.2): To provide backported and experimental type hints for Python, helping with type annotations in Python code.
- tzdata (2024.1): Provides timezone data for Python

3.7 Procedure for carrying out the simulation

3.7.1 Data Loading

The dataset in csv format was uploaded into a Pandas Data frame using the read csv method.

3.7.2 Data Processing

Raw data was transformed into a format that is suitable for training machine learning models. This process includes handling missing values, removing duplicate rows, removing space from column names, merging infrequent classes, and using one-hot encoding to convert categorical strings into numerical values for compatibility with ML algorithms (Ozdogan, 2024). Effective preprocessing is essential for ensuring the reliability and accuracy of results, as it minimizes the risk of classifiers being influenced by irrelevant or redundant data, thus enhancing both performance and validity. Domain expertise is valuable in this process, as it helps ensure that preprocessing is performed correctly to improve the quality of the model (Lin et al, 2023).

3.7.3 Feature Scaling

This process involves scaling the values of all features to a uniform range, ensuring that each feature contributes equally to prevent the machine learning algorithms from being biased towards features with larger values. Feature scaling was used to standardize numerical data, providing a consistent scale across all features. This improves model efficiency by eliminating biases caused by differing distributions of individual features. (Talukder et. al, 2024).

3.7.4 RFE feature Selection

For the WUSTL-EHMS-2020 dataset, Recursive Feature Elimination (RFE) with Random Forest classifiers was employed to optimize the feature space. This approach systematically ranked and

eliminated less important features, retaining only the most influential ones for intrusion detection. By reducing the feature set, the IDS was able to process data more efficiently, which in turn improved the speed of intrusion detection. Focusing on the most informative features enhanced the IDS's accuracy in distinguishing normal network traffic from potential cyber threats (Zhang et.al, 2024).

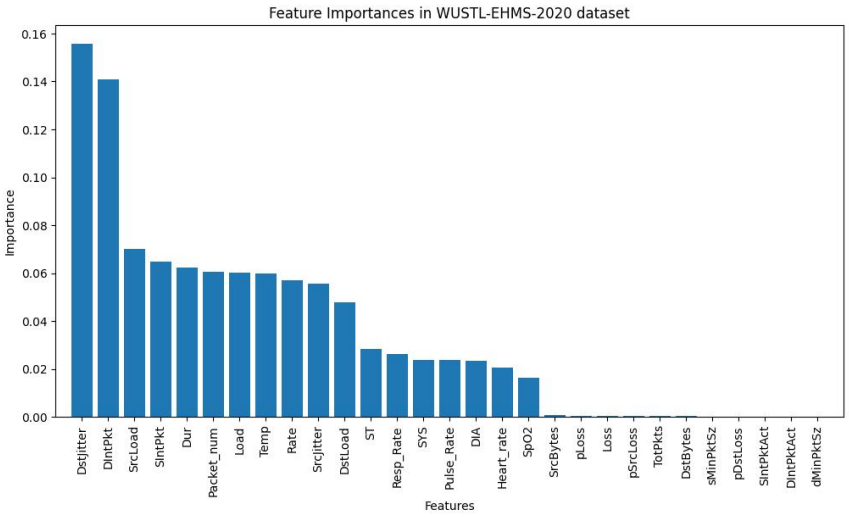


Figure 6: Feature importance ranking for the WUSTL-EHMS-2020 dataset

3.7.5 Synthetic Minority Over-sampling Technique (SMOTE)

In the two datasets used, there is a notable imbalance in the target feature, where one class significantly outnumbers the other. This imbalance can skew model performance, causing the model to favor predicting the majority class. To address this issue, the Synthetic Minority Over-sampling Technique (SMOTE) is utilized. SMOTE creates synthetic data to rebalance the class distribution by oversampling the minority class. By generating new instances that mimic real attack patterns, SMOTE increases the number of minority class samples, thereby improving the balance of the class distribution. This technique helps reduce the impact of class imbalance, enhances the model's generalization capabilities, and boosts its effectiveness in detecting genuine attacks within network traffic (Sayegh et. al, 2024)

3.7.6 Data Splitting

The datasets were divided using Scikit-learn's `train_test_split` function with an 80/20 ratio, where 80% of the data was allocated for training the model, and the remaining 20% was set aside for evaluating its performance. This method ensures the model is trained on a significant portion of the data, enabling it to learn different patterns and anomalies. The 20% test set, which the model has not seen during training, is used to evaluate its ability to detect new, previously unseen anomalies (Turukmane and Devendiran, 2024).

3.8 Model Training

Various machine learning models, including [insert specific models], were trained on the training set

3.8.1 Justification for deploying Random Forest Algorithm

Random Forest (RF) model is ideal for an IoHT Intrusion Detection System (IDS) due to its ability to build multiple decision trees and aggregate their predictions, which enhances accuracy and mitigates overfitting which is a common issue in complex data environments like IoHT. By using a majority vote from multiple trees and randomly selecting subsets of features and samples, RF reduces variance and improves generalization. This robustness makes RF highly effective at detecting anomalies in complex network data, ensuring reliable and accurate intrusion detection in IoHT systems (Rahman, 2024).

3.8.2 Justification for deploying Extreme Gradient Boosting (XGB)

Deploying Extreme Gradient Boosting (XGBoost) for the Intrusion Detection System (IDS) project is highly advantageous due to its superior predictive accuracy and efficiency. XGBoost excels in handling imbalanced datasets, which is crucial for detecting rare anomalies in network traffic. Its built-in feature importance analysis aids in optimizing the feature set, while its scalability and robustness to overfitting ensure effective performance on large, complex datasets making it a reliable choice for developing a robust IDS capable of accurately identifying and responding to cyber threats in IoHT healthcare networks (Sarkar et. Al, 2024).

3.8.3 Justification for deploying Support Vector Machines (SVMs):

Support Vector Machines (SVMs) are effective for IoHT Intrusion Detection Systems (IDS) because they create a hyperplane that maximizes the separation between classes which is crucial for distinguishing between normal and malicious traffic. SVMs handle both linear and non-linear data using kernel functions making them adept at capturing complex patterns necessary for detecting sophisticated cyber threats. Their ability to generalize well in high-dimensional spaces ensures reliable performance, essential for maintaining the security and integrity of IoHT systems. (Altulaihan, 2024) .

3.8.4 Justification for deploying Logistic Regression Algorithm

Logistic Regression (LR) is an effective choice for an IoHT Intrusion Detection System (IDS) due to its simplicity, high interpretability, and efficiency in binary classification tasks. It uses the sigmoid function to model the probability of a binary outcome, making it well-suited for distinguishing between normal and anomalous network traffic. LR's computational efficiency also ensures quick decision-making, which is crucial for real-time anomaly detection in IoHT environments (Bacha et al., 2024).

3.8.5 Justification for deploying Artificial Neural Networks

Artificial Neural Networks (ANNs) are ideal for an IoHT Intrusion Detection System (IDS) due to their ability to model complex, non-linear relationships in data without relying on assumptions about data distribution. Comprising multiple layers and utilizing activation functions like ReLU and sigmoid, ANNs learn and capture intricate patterns, making them highly effective at detecting

anomalies that traditional methods might miss. By continuously adjusting weights and biases during training, ANNs improve their accuracy in identifying malicious activities. Their adaptability to new and unseen threats enhances the robustness and effectiveness of IDS in IoHT environments, making them a strong choice for this application (Binbusayyis, 2022).

3.8.6 Justification for deploying Long Short Term Memory Networks

Long Short-Term Memory networks (LSTMs) are well-suited for IoHT Intrusion Detection Systems (IDS) due to their ability to effectively handle sequential data and capture long-term temporal patterns. LSTMs use a gating mechanism to mitigate the vanishing gradient problem, allowing them to learn and remember information over extended periods. This makes LSTMs particularly effective at detecting both anomalies, and subtle deviations in network behavior ensuring reliable and precise intrusion detection in IoHT environments (Rafique et al., 2024).

3.8.7 Proposed Innovative ensemble Algorithm:

The proposed innovative ensemble model that combines XGBoost, Random Forest, and SVM as a meta-learner with a stacking classifier offers a powerful approach to enhance detection accuracy and robustness. XGBoost contributes its superior gradient boosting capabilities, excelling at handling complex, non-linear relationships in the data. Random Forest adds resilience against overfitting by aggregating multiple decision trees, improving generalization across varied IoHT data. SVM, known for its strong performance in binary classification tasks, provides precision in separating normal and anomalous traffic. By stacking these models, the ensemble harnesses the strengths of each, leading to improved accuracy and more reliable detection of both known and unknown threats in IoHT environments.

Algorithm	**Advantage**	**Disadvantage**
ANN (Rafique et. al, 2024).	Capable of handling linear data Adjusts their parameters to varying data patterns over time Automatically learns relevant features from raw data	Training the model is time-consuming and computationally intensive. Interpreting the patterns learned by the network is difficult.
SVM (Altulaihan, 2024)	Performs well in high-dimensional spaces Resistant to overfitting	Sensitive to the parameters and tuning of the kernel function Requires significant computational resources

Algorithm	Advantage	Disadvantage
Extreme Gradieint Boosting (Sakar et. al, 2024)	Resistant to overfitting Suitable for large datasets	Computationally intensive
Algorithm	Advantage	Disadvantage
LR (Bacha et. al, 2024).	Easy to implement Computationally efficienct	Does not perform well on nonlinear data Prone to over-fitting-
RF (Maddireddy and Maddireddy 2024)	Robust to over-fitting Higher accuracy compared to other models	Computationally expensive
LSTM (Sayegh et. al, 2024)	Ability to model complex data in different contexts Ability to learn and remember information	Complex to implement and interpret Computationally intensive

Table 5: Showing ML algorithms and their advantage and disadvantages

3.9 Evaluation Metrics

To evaluate the performance of the algorithms for the intrusion detection system (IDS), I utilized standard evaluation metrics that provide a thorough evaluation of the model's effectiveness in detecting network intrusions. These metrics are crucial for measuring the accuracy and reliability of our IDS, offering valuable insights into its performance. The metrics employed evaluate the models are as follows;

- Accuracy: Accuracy assesses the overall performance of classification models by measuring the rate of correctly classified instances, encompassing both true positives and true negatives, relative to the total number of instances in the dataset (Sayegh et.al, 2024). Accuracy is calculated as follows:

  Accuracy = Number of Correct Total Number of Predictions

- Precision:Precision, also called positive predictive value, assess the accuracy of a model's positive predictions (Sayegh et.al, 2024). , precision is calculated as follows:

  Precision = True Positives Total Number of Predictions

- Recall: Recall, also referred to as sensitivity, measures the model's effectiveness in correctly identifying positive instances compared to the actual positive cases in the dataset. Recall is calculated as follows (Khan and Alkhathami, 2024):

  Recall =True Positives(True Positive +False Negatives)

- F1-score: F1-Score is the harmonic mean of precision and recall, it provides a balanced evaluation of these two metrics. It is useful in cases where there is an imbalance between the number of positive and negative entries in the dataset (Sayegh et.al, 2024). F1-score is calculated using the as follows:

  F1- Score = 2 (Precision Recall) (Precision + Recall)

- Area Under the Receiver Operating Characteristic Curve (AUC-ROC):The AUC-ROC is a vital measure for evaluating a model's ability to differentiate between positive and negative cases. An AUC value approaching 1 indicates excellent performance in distinguishing between class labels, whereas a value near 0 suggests poor performance. The AUC provides a graphical depiction of the model's effectiveness in separating classes within the target feature, with a higher ROC curve reflecting better classifier performance. (Gupta et. Al, 2024).

- Confusion Matrix: A Confusion Matrix is a structured table that offers a clear overview of a classification model's performance by showing the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). This matrix is essential for a thorough evaluation of the model's effectiveness, highlighting its strengths and pinpointing areas that might need enhancement (Sayegh et. al, 2024).

		Predicted	
		Negative **(N)** -	Positive **(P)** +
Actual	Negative -	True Negative **(TN)**	**False Positive (FP) Type I Error**
	Positive +	**False Negative (FN) Type II Error**	True Positive **(TP)**

Figure 7: confusion matrix

3.10 Deployment of the IDS on Web Application

After evaluating the performance of various models, the most accurate predictive model with the WUSTL-EHMS 2020 dataset would be selected for deployment within a web application using Django, a Python-based framework. The IDS web application was developed in Visual Studio Code and implemented with Django. This involved creating templates, setting up routing and rendering logic, integrating necessary libraries, and incorporating the machine learning ensemble model, all of which were initially developed and tested locally. Once development was complete, the application was deployed on PythonAnywhere, a cloud hosting platform. The IDS web application is accessible via https://cyberattackiotprediction.pythonanywhere.com . This deployment enables seamless integration and practical use of the IDS, enhancing accessibility, real-time detection, alerting, and user-friendliness. The intuitive interface allows users to manage and interact with the IDS effectively, thereby improving its usability, scalability, and effectiveness within the healthcare sector.

3.11 Evaluation of Web Application

To evaluate the IDS web application designed for IoMT in healthcare environments, interviews would be conducted with three end users of the IDS application in real-world clinical settings to provide valuable insights into its usability, functionality, and overall effectiveness in managing healthcare-specific security threats. The feedback gathered from these interviews will be used to ensure that the application not only effectively detects and responds to ML-based anomalies but also provides a seamless and intuitive experience for healthcare staff who manage and interact with the IDS.

3.12 Summary

In this chapter, I outlined the methodology employed for developing and evaluating our Intrusion Detection System (IDS) for healthcare IoT networks. I began with an overview of the research framework, emphasizing a quantitative approach to systematically analyze and validate our proposed solutions. The implementation of Agile methodology ensured iterative development and continuous improvement throughout the research process. The chapter detailed the datasets used, including the ECU-IoHT and WUSTL-EHMS-2020 datasets. Each dataset's characteristics were described, highlighting their relevance for training and testing our models. I then provided an in-depth look at the experimental setup and the procedural steps for conducting simulations. This included data loading, processing, feature scaling, and feature selection using Recursive Feature Elimination (RFE). The Synthetic Minority Over-sampling Technique (SMOTE) was applied to address class imbalance, and the data was split into training and testing subsets. Model training involved a range of algorithms: Random Forest, Extreme Gradient Boosting (XGB), Support Vector Machines (SVMs), Logistic Regression, Artificial Neural Networks (ANN), and Long Short-Term Memory Networks (LSTM). Justifications for selecting each algorithm were provided,

demonstrating their suitability for the IDS. I also discussed the development of an ensemble algorithm combining these methods to enhance detection accuracy. Evaluation metrics were outlined to assess model performance, ensuring a comprehensive evaluation of their effectiveness. Finally, the deployment of the IDS within a web application framework using Django were described. This deployment aimed to improve accessibility, real-time detection, and user interaction. The chapter concluded with an evaluation of the web application to ensure its effectiveness in a practical setting. This chapter established a robust methodological framework for our research, detailing each step from dataset preparation to model deployment, and underscoring the rigorous approach taken to develop an effective IDS for healthcare IoT networks.

4 RESULTS AND FINDINGS

4.1 Overview

In this project, I conducted an in-depth analysis of the performance of four machine learning models: Random Forest (RF), Extreme Gradient Boosting (XGB), Logistic Regression (LR), and Decision Tree (DT), along with two deep learning models: Artificial Neural Network (ANN) and Long Short Term Memory (LSTM), as well as an innovative ensemble model. This analysis was performed against two healthcare datasets, ECU-IoHT and WUSTL-EHMS 2020 aiming to evaluate the performance and consistency of the models across both datasets.

4.2 Comparison of results on ECU-IoHT dataset

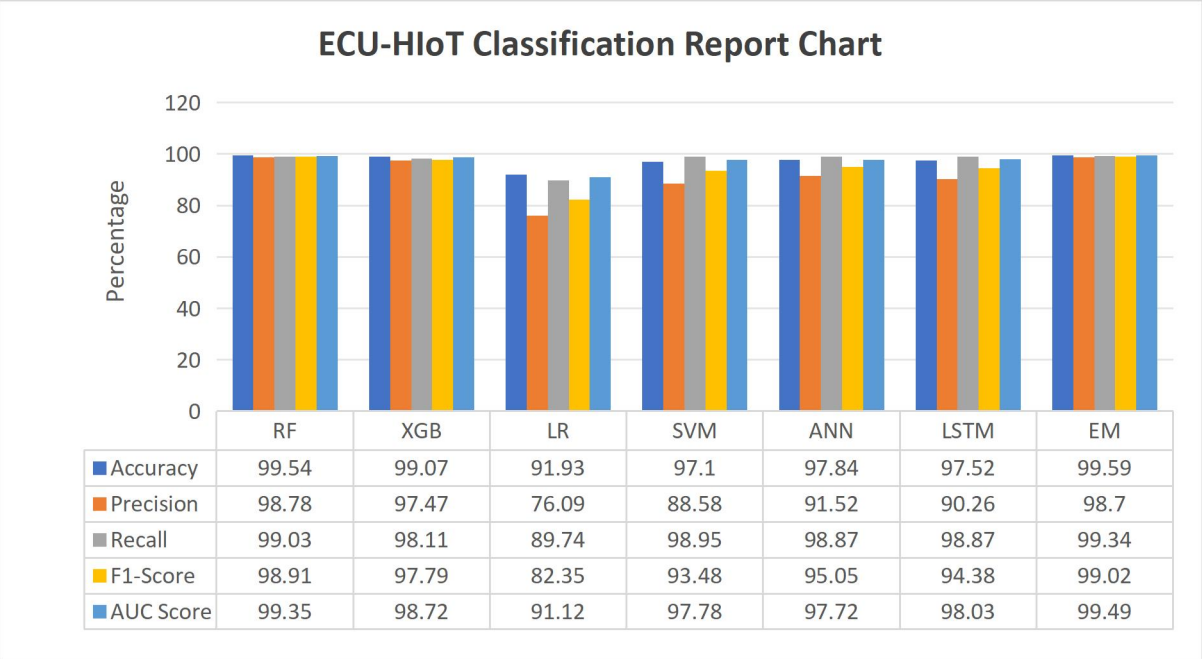
The ECU-IoHT dataset was partitioned with 80% of the data allocated for training and the remaining 20% for testing. The dataset included categorical attributes, like 'Protocol,' which required conversion to a numerical format using One-Hot Encoding. Given the small number of features, feature selection Recursive Feature Elimination (RFE) was not used during feature selection. However, to address the dataset's imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) was applied.

ECU-IoHT dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC Score**
RF	99.54	98.78	99.03	98.91	99.35
XGB	99.07	97.47	98.11	97.79	98.72
LR	91.93	76.09	89.74	82.35	91.12
SVM	97.10	88.58	98.95	93.48	97.78
ANN	97.84	91.52	98.87	95.05	97.72
LSTM	97.52	90.26	98.87	94.38	98.03
EM	99.59	98.70	99.34	99.02	99.49

Table 6: performance analysis of the models for ECU-IoHT dataset reflecting Accuracy, precision,recall,F1-score and AUC-score

As illustrated in table 6, table 7, figure 7 and figure 8 the proposed ensemble Model (EM) demonstrated robust performance, stands out as the best performer for the ECU-IoHT dataset, achieving the highest accuracy of 99.59% and an AUC Score of 99.49 from figure 8. It also excels in precision (98.70%) and recall (99.34%) as seen in figure 7, with the lowest number of False Positives (31) and False Negatives (61) as shown in table 7 and figure 9, indicating its superior

ability to correctly identify both normal and anomalous instances while minimizing misclassifications.



	RF	XGB	LR	SVM	ANN	LSTM	EM
Accuracy	99.54	99.07	91.93	97.1	97.84	97.52	99.59
Precision	98.78	97.47	76.09	88.58	91.52	90.26	98.7
Recall	99.03	98.11	89.74	98.95	98.87	98.87	99.34
F1-Score	98.91	97.79	82.35	93.48	95.05	94.38	99.02
AUC Score	99.35	98.72	91.12	97.78	97.72	98.03	99.49

Figure 7: chart showing performance analysis of the models for ECU-IoHT dataset reflecting Accuracy, precision, recall, F1-score and AUC-score

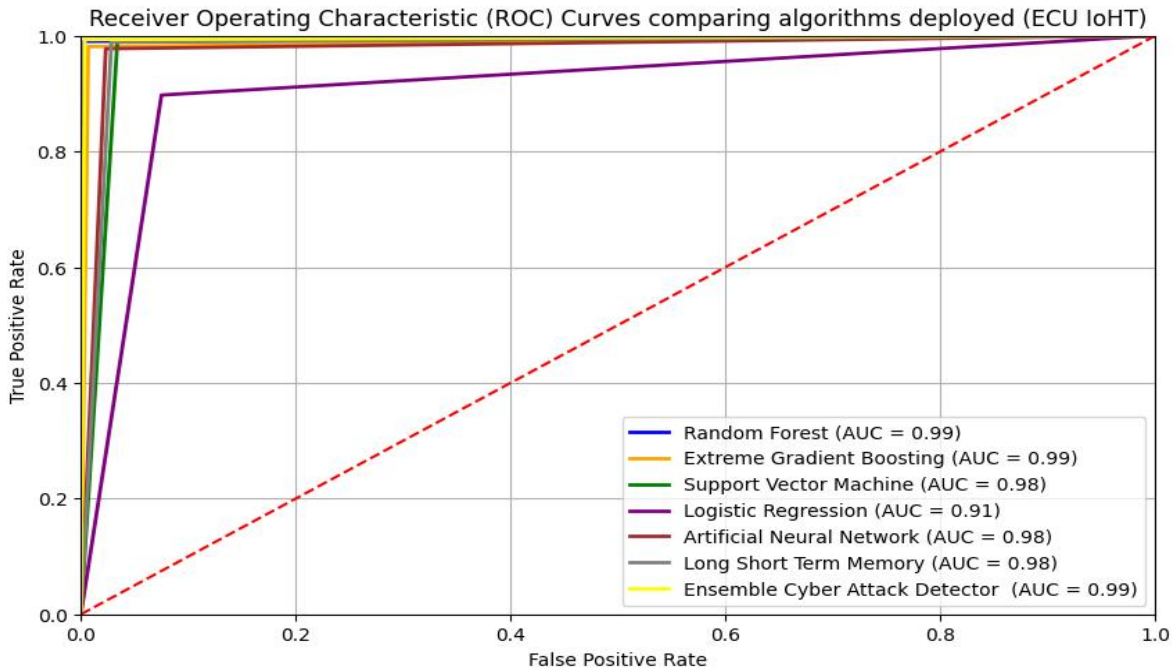


Figure 8 : Chart showing AUC-ROC curve of the models for ECU-IoHT dataset

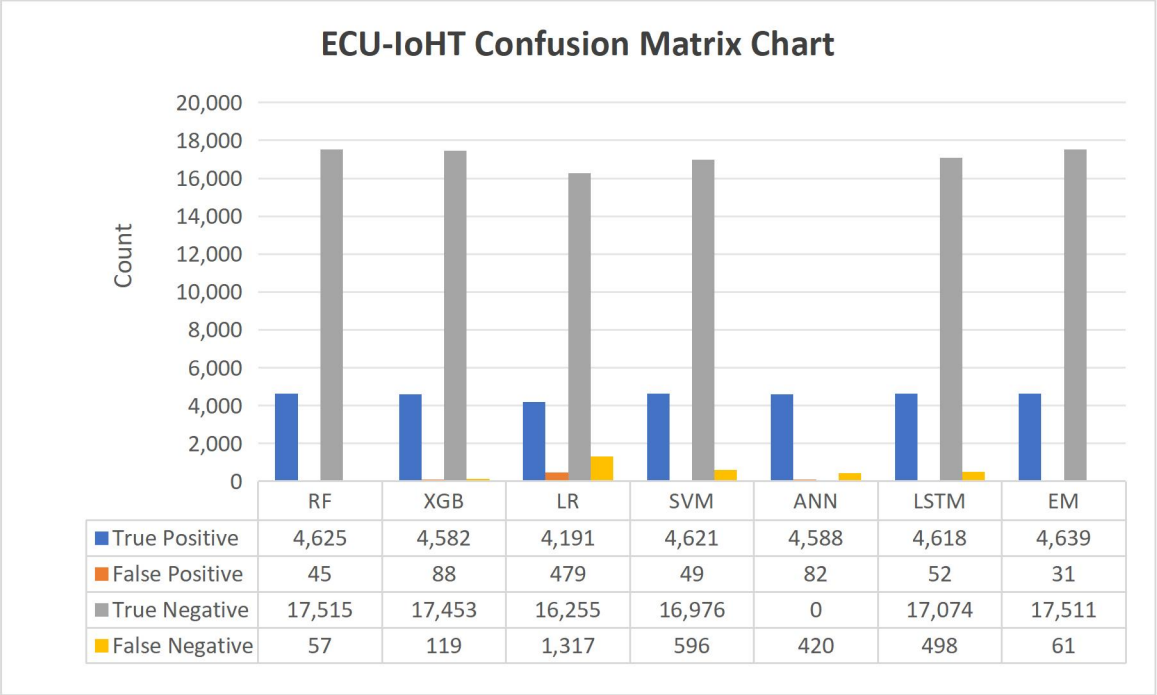
XGBoost (XGB) and Random Forest (RF) are both highly effective models for the task, showing impressive performance metrics. XGBoost achieves an accuracy of 99.07% and an AUC Score of 98.72, with high precision at 97.47% and recall at 98.11% as seen. It has a slightly higher number of False Positives (88) and False Negatives (119) compared to Random Forest but remains very effective. Random Forest follows closely, with a higher accuracy of 99.54% and an AUC Score of

99.35. It shows strong precision at 98.78% and recall at 99.03%, though it also has a slightly higher number of False Positives (45) and False Negatives (57) compared to XGBoost. Both models demonstrate exceptional performance, with Random Forest marginally outperforming XGBoost in accuracy and AUC, while XGBoost maintains strong precision and recall.

ECU-IoHT dataset				
Model Name	**True Positive**	**False Positive**	**True Negative**	**False Negative**
RF	4,625	45	17,515	57
XGB	4,582	88	17,453	119
LR	4,191	479	16,255	1,317
SVM	4,621	49	16,976	596
ANN	4,588	82	17,152`	420
LSTM	4,618	52	17,074	498
EM	4,639	31	17,511	61

Table 7: performance analysis of the models for ECU-IoHT dataset reflecting TN,FN, TP,FP

Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM) networks both demonstrate strong performance as deep learning models. ANN achieves an accuracy of 97.84% and an AUC Score of 97.72, with a high recall of 98.87% and precision of 91.52%, although it has a higher number of False Positives (82). Its F1-Score of 95.05 indicates robust overall performance. LSTM networks perform slightly lower with an accuracy of 97.52% and an AUC Score of 98.03. They also exhibit a strong recall of 98.87% but with lower precision at 90.26%, leading to more False Positives (52) and a significantly higher number of False Negatives (498) compared to ANN. Both models provide valuable insights, with ANN showing slightly better precision and overall F1-Score, while LSTM has a competitive AUC Score but faces challenges with False Negatives.



	RF	XGB	LR	SVM	ANN	LSTM	EM
True Positive	4,625	4,582	4,191	4,621	4,588	4,618	4,639
False Positive	45	88	479	49	82	52	31
True Negative	17,515	17,453	16,255	16,976	0	17,074	17,511
False Negative	57	119	1,317	596	420	498	61

Figure 9: Chart showing performance analysis of the models for ECU-IoHT dataset reflecting TN,FN, TP,FP

Support Vector Machine (SVM) shows an accuracy of 97.10% and an AUC Score of 97.78. It demonstrates high recall (98.95%) but lower precision (88.58%), leading to higher rates of False Positives (49) and False Negatives (596), this indicates that while it effectively detects true positives, it may misclassify normal activities.

Logistic Regression (LR) had the lowest performance with an accuracy of 91.93% and an AUC Score of 91.12. It has the highest number of False Negatives (1,317) and False Positives (479), with lower precision (76.09%) and recall (89.74%), making it less reliable for critical intrusion detection tasks.

Classification Report for random forest (ECU_IoHT):

	precision	recall	f1-score	support
0	1.00	1.00	1.00	17572
1	0.99	0.99	0.99	4670
accuracy			1.00	22242
macro avg	0.99	0.99	0.99	22242
weighted avg	1.00	1.00	1.00	22242

Figure 10: classification report of RF for ECU-IoHT dataset

Classification Report for XGB (ECU_IoHT):

	precision	recall	f1-score	support
0	0.99	0.99	0.99	17572
1	0.97	0.98	0.98	4670
accuracy			0.99	22242
macro avg	0.98	0.99	0.99	22242
weighted avg	0.99	0.99	0.99	22242

Figure 11: classification report of XGB for ECU-IoHT dataset

Classification Report for svm (ECU_IoHT):

	precision	recall	f1-score	support
0	1.00	0.97	0.98	17572
1	0.89	0.99	0.93	4670
accuracy			0.97	22242
macro avg	0.94	0.98	0.96	22242
weighted avg	0.97	0.97	0.97	22242

Figure 12: classification report of SVM for ECU-IoHT dataset

Classification Report for lr (ECU_IoHT):

	precision	recall	f1-score	support
0	0.97	0.93	0.95	17572
1	0.76	0.90	0.82	4670
accuracy			0.92	22242
macro avg	0.87	0.91	0.89	22242
weighted avg	0.93	0.92	0.92	22242

Figure 13: classification report of LR for ECU-IoHT dataset

Classification Report for ANN (ECU_IoHT):

	precision	recall	f1-score	support
0	1.00	0.98	0.99	17572
1	0.92	0.98	0.95	4670
accuracy			0.98	22242
macro avg	0.96	0.98	0.97	22242
weighted avg	0.98	0.98	0.98	22242

Figure 14: classification report of ANN for ECU-IoHT dataset

Classification Report for LSTM (ECU_IoHT):

	precision	recall	f1-score	support
0	1.00	0.97	0.98	17572
1	0.90	0.99	0.94	4670
accuracy			0.98	22242
macro avg	0.95	0.98	0.96	22242
weighted avg	0.98	0.98	0.98	22242

Figure 15: classification report of LSTM for ECU-IoHT dataset

Classification Report for Ensemble Model (ECU_IoHT):

	precision	recall	f1-score	support
0	1.00	1.00	1.00	17572
1	0.99	0.99	0.99	4670
accuracy			1.00	22242
macro avg	0.99	0.99	0.99	22242
weighted avg	1.00	1.00	1.00	22242

Figure 16: classification report of ensemble model for ECU-IoHT dataset

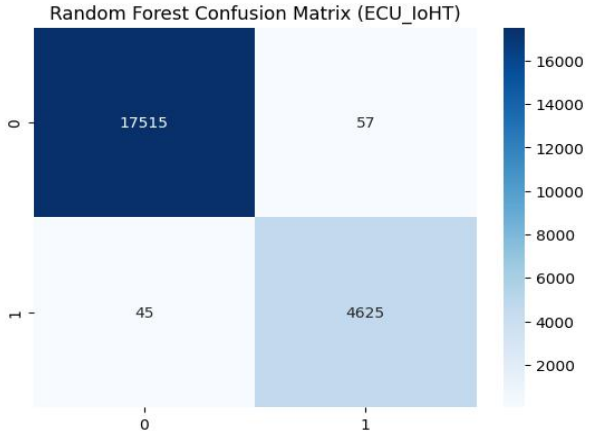


Figure 17: confusion matrix of RF for ECU-IoHT dataset

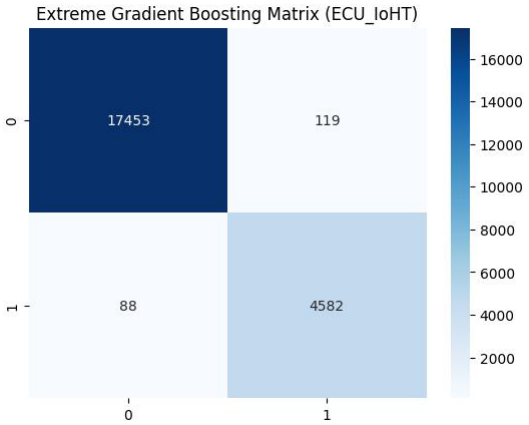


Figure 18: confusion matrix of XGB for ECU-IoHT dataset

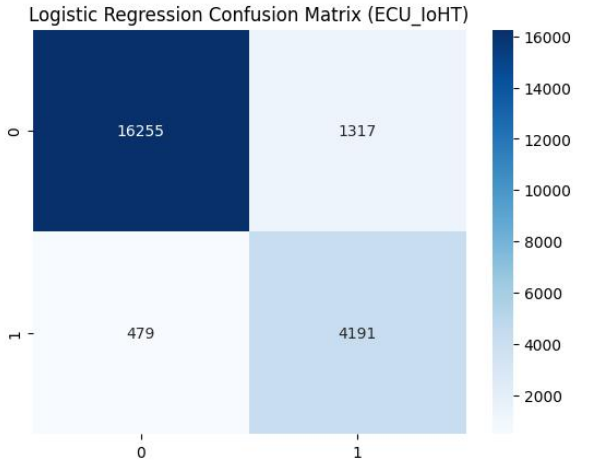


Figure 19: confusion matrix of of LR for ECU-IoHT dataset

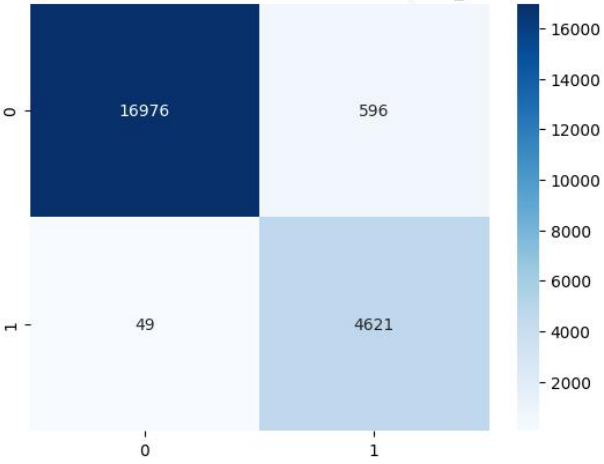


Figure 20: confusion matrix of of SVM for ECU-IoHT dataset

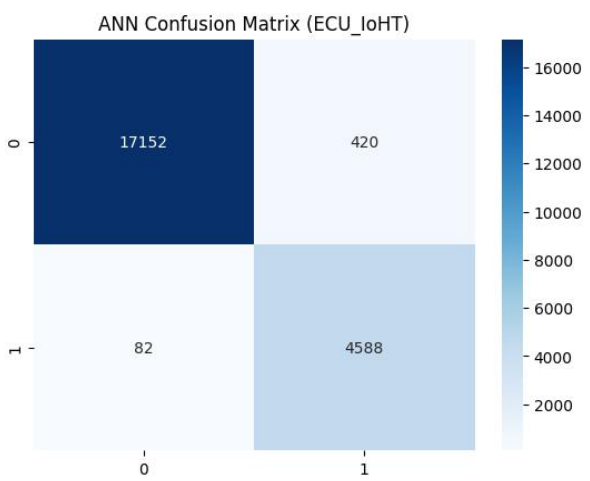


Figure 21: confusion matrix of of ANN for ECU-IoHT dataset

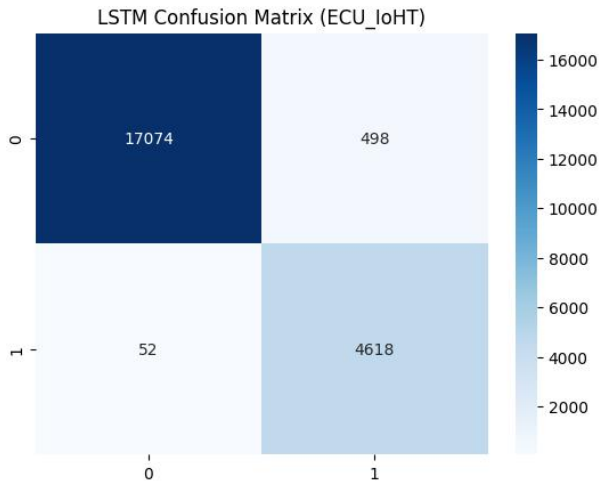


Figure 22: confusion matrix of of LSTM for ECU-IoHT dataset

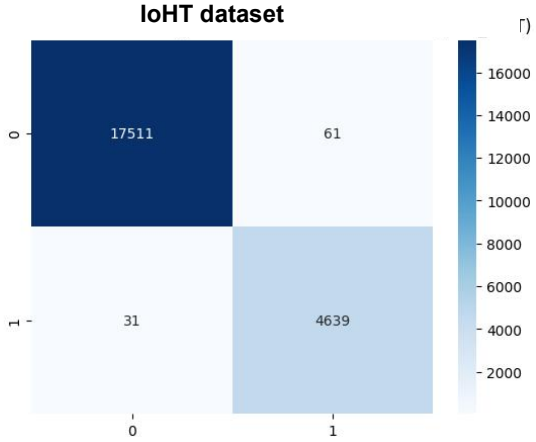


Figure 23: confusion matrix of ensemble model for ECU-IoHT dataset

4.3 Comparison of results on WUSTL-EHMS 2020 dataset

The WUSTL-EHMS-2020 dataset was partitioned in the same way as the ECU-IoHT dataset, with 80% allocated for training and 20% reserved for testing. To mitigate the issue of data imbalance, SMOTE was applied to the attack types in the training set. The top features identified by Recursive Feature Elimination (RFE) using the Random Forest Classifier (RFC) are shown in Figure 6 include patients' biometric data, reinforcing the idea that biometric information is vital for developing an effective and accurate Intrusion Detection System (IDS) in healthcare systems.

WUSTL-EHMS 2020 dataset					
Model Name	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**AUC Score**
RF	95.46	95.63	95.40	95.51	95.46
XGB	96.57	97.23	95.95	96.59	96.57
LR	72.27	78.40	63.67	70.27	72.84
SVM	83.37	84.50	82.25	83.36	83.39
ANN	85.85	85.24	87.13	86.17	85.83
LSTM	84.84	83.60	87.16	85.35	84.82
EM	97.27	97.87	96.71	97.29	97.27

Table 8: performance analysis of the models for WUSTL-EHMS 2020 dataset reflecting Accuracy, precision,recall,F1-score and AUC-score

The proposed innovative ensemble Model stands out as the best performer again for the WUSTL-EHMS 2020 dataset achieving an accuracy of 97.27%, precision of 97.87%, recall of 96.71%, F1-Score of 97.29 as seen in table 8, fugure 23 and an AUC Score of 97.27 as seen in table 8, figure 25. It demonstrates excellent capability in classifying both positive and negative cases, supported by a low false positive rate of 95 and a very low false negative rate of 61. The high number of true positives (TP) at 2,795 as seen in figure 26 , table 9 indicates its strong performance in correctly identifying positive instances.

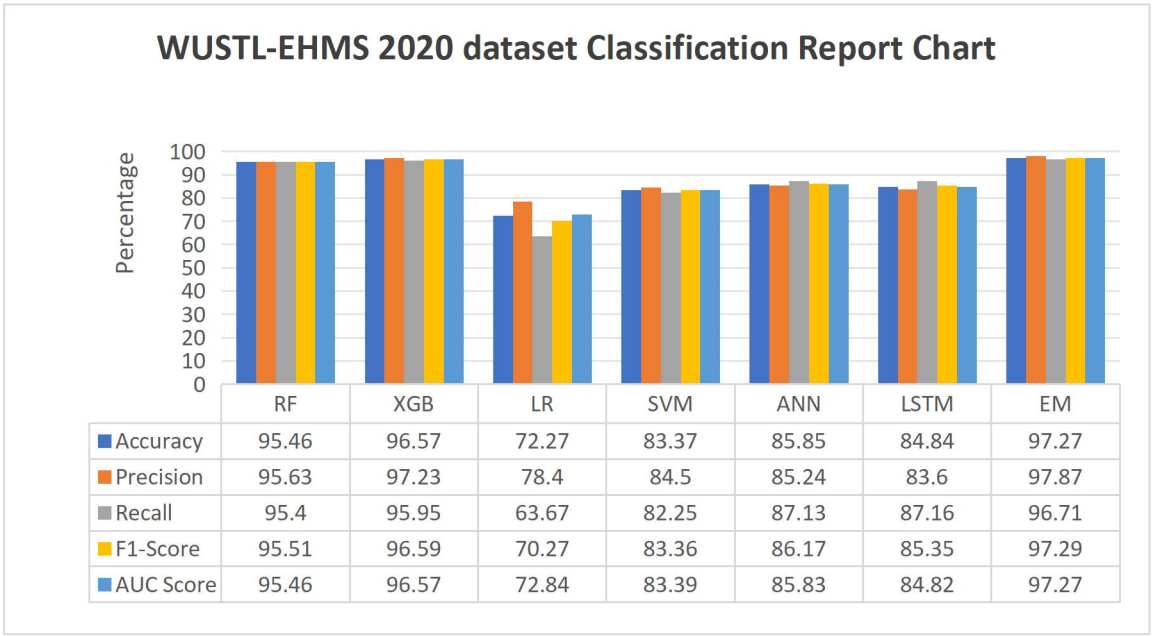


Figure 24: chart showing performance analysis of the models for WUSTL-EHMS 2020 dataset reflecting Accuracy, precision, recall, F1-score and AUC-score

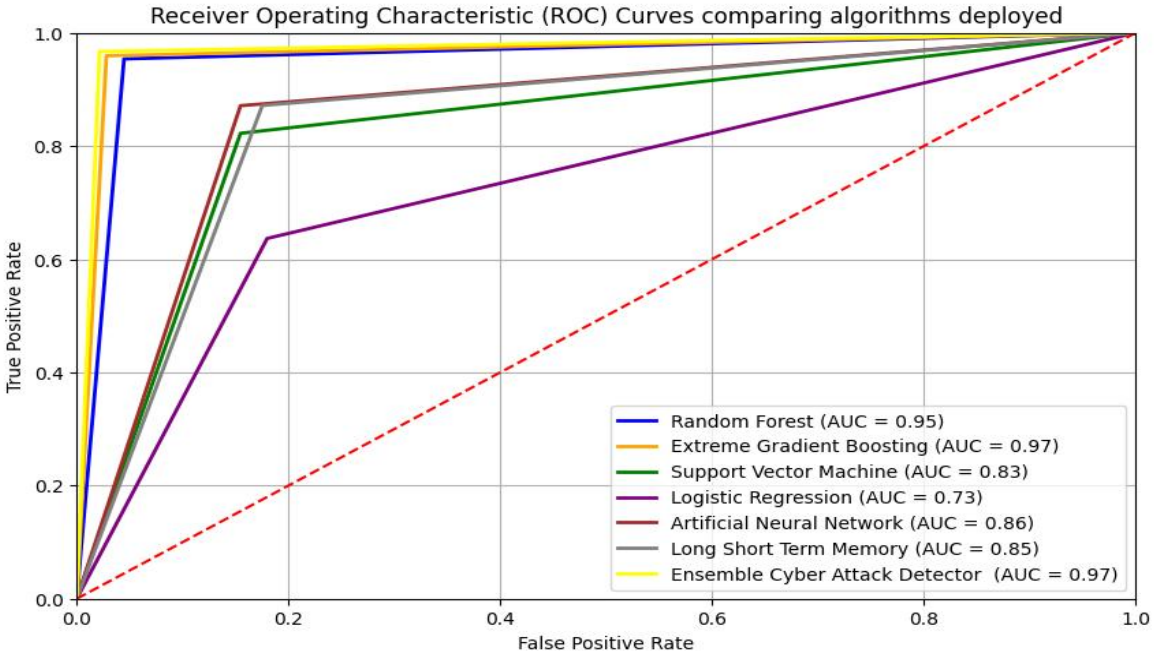


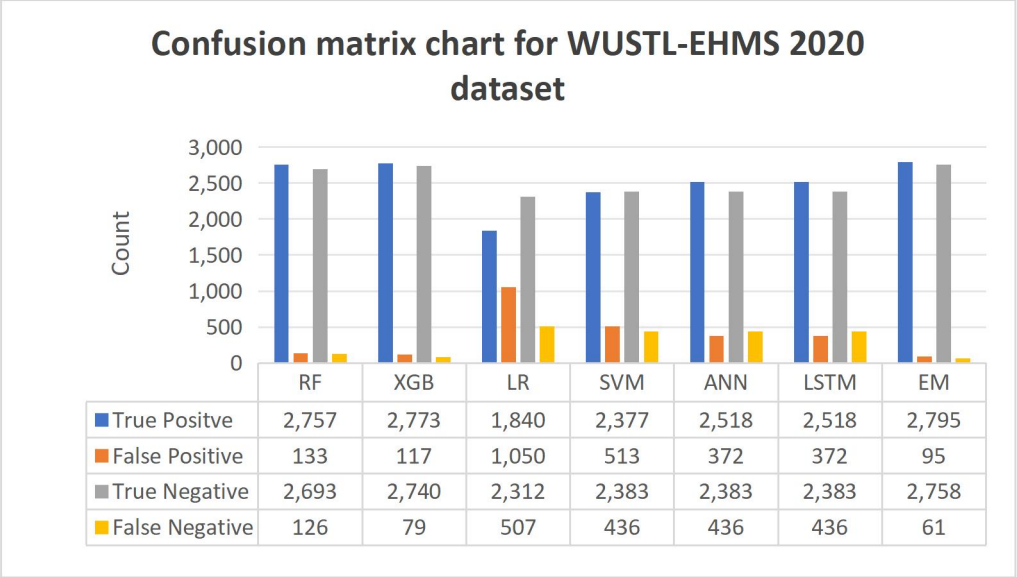
Figure 25: Chart showing AUC-ROC curve of the models for WUSTL-EHMS 2020 dataset

Extreme Gradient Boosting (XGBoost) and Random Forest are both strong models, with XGBoost showing an accuracy of 96.57%, precision of 97.23%, recall of 95.95%, F1-Score of 96.59, and an AUC Score of 96.57. It achieves a slightly higher number of true positives (2,773) and a low false positive count (117), making it a robust performer, it trails behind the Ensemble Model in overall metrics. Random Forest follows with an accuracy of 95.46%, precision of 95.63%, recall of 95.40%, F1-Score of 95.51, and an AUC Score of 95.46. It records 2,757 true positives, 133 false positives, and 126 false negatives, reflecting solid performance but not quite on par with XGBoost and the Ensemble Model. Together, both models demonstrate strong capabilities, with XGBoost slightly outperforming Random Forest across most metrics.

WUSTL-EHMS 2020 dataset				
Model Name	**True Positve**	**False Positive**	**True Negative**	**False Negative**
RF	2,757	133	2,693	126
XGB	2,773	117	2,740	79
LR	1,840	1,050	2,312	507
SVM	2,377	513	2,383	436
ANN	2,518	372	2,383	436
LSTM	2,518	372	2,383	436
EM	2,795	95	2,758	61

Table 9: performance analysis of the models for ECU-IoHT dataset reflecting TN,FN, TP,FP

The Artificial Neural Network (ANN) and Long Short-Term Memory (LSTM) models both demonstrate similar performance as deep learning models. ANN achieves an accuracy of 85.85%, precision of 85.24%, recall of 87.13%, F1-Score of 86.17, and an AUC Score of 85.83, with 2,518 true positives and a higher false positive count of 372, indicating a good recall but lower precision. Similarly, LSTM shows an accuracy of 84.84%, precision of 83.60%, recall of 87.16%, F1-Score of 85.35, and an AUC Score of 84.82, with identical true positives (2,518) and false positives (372) as ANN. Both models exhibit good recall, correctly identifying many positive cases, but their lower precision leads to higher false positive rates, making them less effective than the top-performing models.



	RF	XGB	LR	SVM	ANN	LSTM	EM
True Positve	2,757	2,773	1,840	2,377	2,518	2,518	2,795
False Positive	133	117	1,050	513	372	372	95
True Negative	2,693	2,740	2,312	2,383	2,383	2,383	2,758
False Negative	126	79	507	436	436	436	61

Figure 26: Chart showing performance analysis of the models for WUSTL-EHMS2020 dataset reflecting TN,FN, TP,FP

The SVM model has an accuracy of 83.37%, precision of 84.50%, recall of 82.25%, F1-Score of 83.36, and an AUC Score of 83.39. It has 2,377 true positives but also a significant number of false positives (513) and false negatives (436), resulting in lower overall effectiveness compared to the higher-performing models.

Logistic Regression displayed the weakest performance across all metrics, achieving an accuracy of only 72.27%, precision of 78.40%, recall of 63.67%, F1-Score of 70.27, and an AUC Score of 72.84. It has the lowest number of true positives (1,840) and a high false positive count (1,050), which contributes to its overall poor classification ability and suggests it is the least reliable model for the dataset.

Classification Report for random forest (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.95	0.96	0.95	2819
1	0.96	0.95	0.96	2890
accuracy			0.95	5709
macro avg	0.95	0.95	0.95	5709
weighted avg	0.95	0.95	0.95	5709

Figure 27: classification report of RF for WUSTM-EHMS 2020 datset

Classification Report for XGB (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.96	0.97	0.97	2819
1	0.97	0.96	0.97	2890
accuracy			0.97	5709
macro avg	0.97	0.97	0.97	5709
weighted avg	0.97	0.97	0.97	5709

Figure 29: classification report of XGB for WUSTM-EHMS 2020 datset

Classification Report for lr (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.69	0.82	0.75	2819
1	0.78	0.64	0.70	2890
accuracy			0.73	5709
macro avg	0.74	0.73	0.73	5709
weighted avg	0.74	0.73	0.73	5709

Figure 30: classification report of LR for WUSTM-EHMS 2020 datset

Classification Report for svm (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.82	0.85	0.83	2819
1	0.85	0.82	0.83	2890
accuracy			0.83	5709
macro avg	0.83	0.83	0.83	5709
weighted avg	0.83	0.83	0.83	5709

Figure 31: classification report of SVM for WUSTM-EHMS 2020 datset

Classification Report for LSTM (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.86	0.82	0.84	2819
1	0.84	0.87	0.85	2890
accuracy			0.85	5709
macro avg	0.85	0.85	0.85	5709
weighted avg	0.85	0.85	0.85	5709

Figure 32: classification report of LSTM for WUSTM-EHMS 2020 datset

Classification Report for ANN (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.86	0.85	0.86	2819
1	0.85	0.87	0.86	2890
accuracy			0.86	5709
macro avg	0.86	0.86	0.86	5709
weighted avg	0.86	0.86	0.86	5709

Figure 33: classification report of ANN for WUSTM-EHMS 2020 datset

Classification Report (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.97	0.98	0.97	2819
1	0.98	0.97	0.97	2890
accuracy			0.97	5709
macro avg	0.97	0.97	0.97	5709
weighted avg	0.97	0.97	0.97	5709

Figure 34: classification report of EM for WUSTM-EHMS 2020 datset

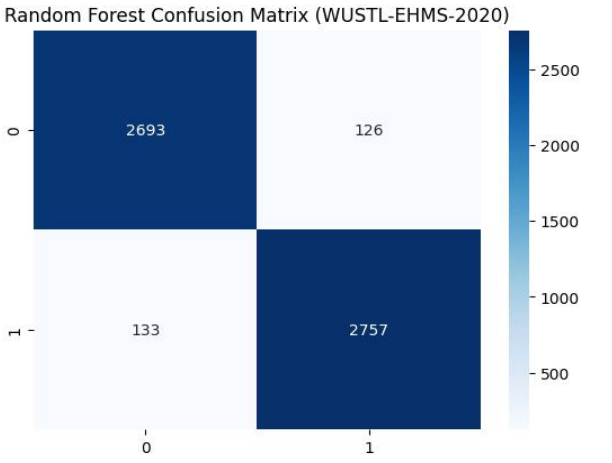


Figure 35: Confusion matrix of RF for WUSTM-EHMS 2020 datset

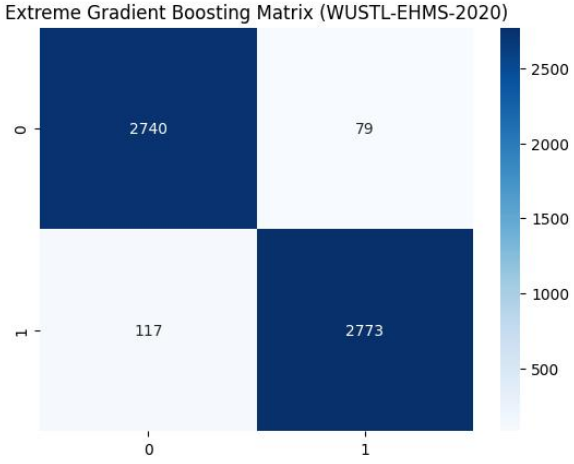


Figure 36: Confusion matrix of XGB for WUSTM-EHMS 2020 datset

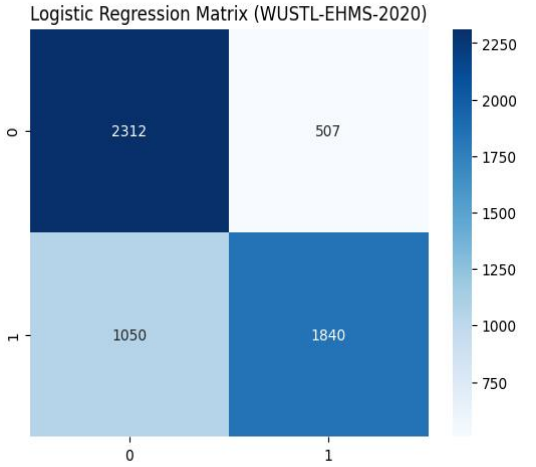


Figure 37: Confusion matrix of LR for WUSTM-EHMS 2020 datset

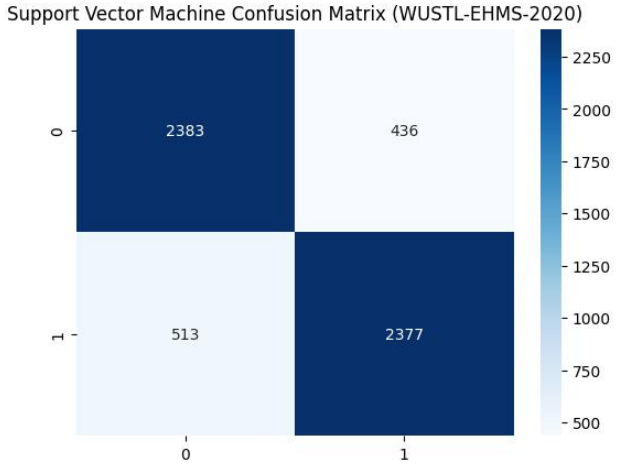


Figure 38: Confusion matrix of SVM for WUSTM-EHMS 2020 datset

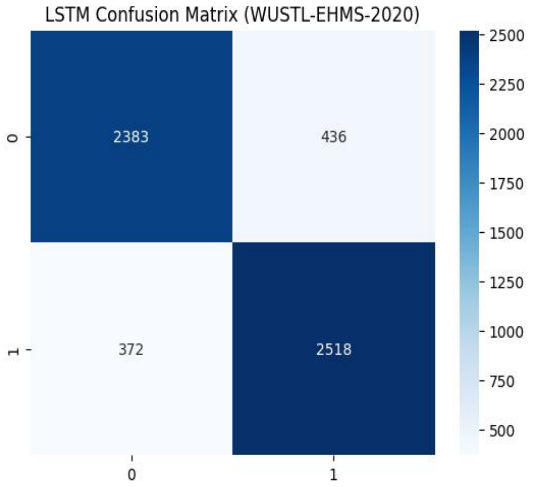


Figure 39: Confusion matrix of LSTM for WUSTM-EHMS 2020 datset

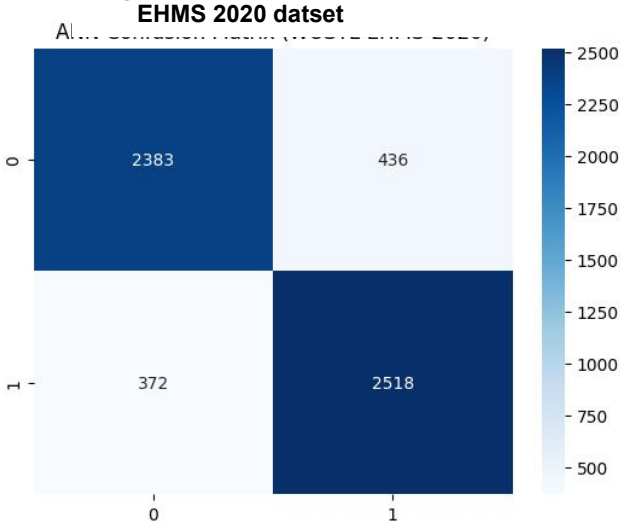


Figure 40: Confusion matrix of ANN for WUSTM-EHMS 2020 datset

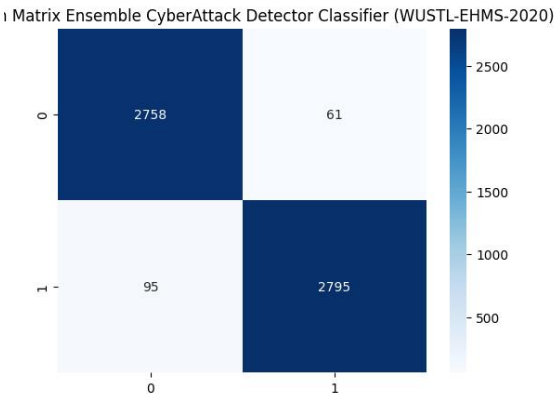


Figure 41: Confusion matrix of EM for WUSTM-EHMS 2020 datset

4.3.1 Comparision of the models performance across WUSTM EHMS 2020 and ECU-HIoT datasets

The performance comparison of various machine learning models across the WUSTML and ECU-IoHT datasets presents valuable insights into the effectiveness and consistency of these algorithms. Several key conclusions can be drawn based on the evaluation of the performance of the models across the two datasets (WUSTM EHMS 2020 and ECU-IoHT) are as folllows.

The proposed innovative Ensemble Model (EM) stands out as the top performer in both scenarios. In WUSTM EHMS 2020 dataset, it has an accuracy of 97.27% and an AUC Score of 97.27, while in ECU-HIoT datase, it achieves an accuracy of 99.59% and an AUC Score of 99.49. EM consistently demonstrates the highest precision and recall, indicating indicating its consistency, robustness and reliability

Random Forest (RF) and Extreme Gradient Boosting (XGBoost): Both Random Forest (RF) and XGBoost (XGB) showed strong and similar performance trends across scenarios. In WUSTM EHMS 2020 dataset, RF has an accuracy of 95.46% and an AUC Score of 95.46, while XGBoost has an accuracy of 96.57% and an AUC Score of 96.57. In ECU-HIoT datase, RF improves to an accuracy of 99.54% and an AUC Score of 99.35, and XGBoost achieves an accuracy of 99.07% and an AUC Score of 98.72. Both models exhibit high precision and recall, with RF showing slightly more False Negatives and XGBoost having slightly higher False Positives in ECU-HIoT datase. This consistency indicates their robust performance in distinguishing between normal and anomalous instances across different datasets.

Both ANN and LSTM exhibit similar performance characteristics in both scenarios. In WUSTM EHMS 2020 dataset, ANN has an accuracy of 85.85% and LSTM 84.84%, while in ECU-HIoT datase, ANN's accuracy is 97.84% and LSTM's is 97.52%. Despite the improvement in accuracy in ECU-HIoT datase, both models maintain high recall but have lower precision compared to the top performers, resulting in higher False Positive rates.

The Support Vector Machine (SVM) model demonstrated consistent performance across both the ECU-HIoT and WUSTL-EHMS 2020 datasets, achieving the same metrics for accuracy (83.37%), precision (84.50%), recall (82.25%), F1-Score (83.36), and AUC Score (83.39%). Despite these consistent metrics, the model's overall effectiveness is limited by a significant number of false positives and false negatives in both datasets, indicating challenges in distinguishing between normal and anomalous behavior. Its higher false positive rate and misclassification issues reduce its reliability compared to higher-performing models in this specific context.

Logistic Regression (LR): Logistic Regression consistently shows the weakest performance across both scenarios. In WUSTM EHMS 2020 dataset, it has an accuracy of 72.27% and an AUC Score

of 72.84, while in ECU-HIoT datase, it displays an accuracy of 91.93% and an AUC Score of 91.12. Despite some improvement in ECU-HIoT datase, LR remains the least effective model overall, with the highest number of False Negatives and False Positives in both datasets.

4.4 Comparison of the results of the proposed system with related works

WUSTL-EHMS-2020 dataset					
Research	**Classifier**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
(Zang et. Al, 2024)	XGB	95	94.87	98	94.63
(Xu et. Al, 2023)	CNN-Focal	93.08	94.23	73.38	64.31
(Kilincer et. Al, 2024)	MLP	96.20	96.19	96.16	96.17
My project	Proposed Ensemble model	97.27	97.87	96.71	97.29

Table 9: comparison of my proposed model with related works for WUSTL-EHMS-2020 dataset

ECU-IoHT dataset					
Research	**Classifier**	**Accuracy**	**Precision**	**Recall**	**F1-Score**
(Vijayakumar et al, 2023)	DNN	99.4	99.3	93	97
(Zang et. Al, 2024)	XGB	99.34	99.32	99.34	99.32
(Kumar et. Al, 2024)	DAG-LSTM	92.04	89.84	92.03	87.17
My project	Proposed Ensemble model	99.59	98.70	99.34	99.02

Table 10: comparison of my proposed model with related works for ECU-IoHT dataset

Tables 9 and 10 compares the results obtained from my project with several previous works that utilized the same datasets as my project, specifically WUSTL-EHMS-2020 and ECU-IoHT healthcare datasets. The models presented in these tables are the best-performing models in the

corresponding literature. This analysis reveals that my proposed ensemble model consistently outperforms the performance of other state-of-the-art models, excelling particularly in accuracy, precision, recall, and F1-score. These results reaffirm that the proposed ensemble approach implemented in my project is a highly effective solution for intrusion detection in healthcare IoT environments, offering superior overall performance compared to classifiers tested in previous studies, thereby validating the reliability and robustness of the system.

4.5 Summary

This chapter presents the results and findings of the study, focusing on the performance of various machine learning models on the ECU-IoHT and WUSTL-EHMS 2020 datasets. It begins with an overview of the study's results and includes a detailed comparison of the models' performance on the ECU-IoHT dataset. The chapter then shifts to a similar analysis for the WUSTL-EHMS 2020 dataset. Additionally, it compares the proposed system's results with those of related works, emphasizing its superior performance across key metrics. The chapter further demonstrates how the proposed ensemble model consistently outperforms or matches state-of-the-art models, validating its effectiveness for intrusion detection in healthcare IoT environments. The chapter concludes with an evaluation of the Machine Learning Anomaly Intrusion Detection System (IDS) web application’s functionality and user experience, and an assessment of how well the research objectives were achieved.

5 ARTIFACT

5.1 Overview

The primary artifact of this project is the deployment of a user-friendly ML anomaly Intrusion Detection System (IDS) web application that employs the proposed innovative ensemble ML model developed in this project with the WUSTL-EHMS2020 dataset within a real-time web application framework using a python framework Django to detect malicious activities in real time within healthcare networks. The IDS web application accessible via https://cyberattackiotprediction.pythonanywhere.com/ provides healthcare professionals with an intuitive interface for real-time threat detection and alerting, detailed logs of detected anomalies, and actionable insights for mitigating cyber threats to effectively safeguard their networks. This project marks a significant advancement in tackling the unique security challenges associated with IoT devices in healthcare settings.

5.2 A Guide to Using the IDS Web App

Step 1: Access the Web App

Launch your web browser and navigate to https://cyberattackiotprediction.pythonanywhere.com/.

Step 2: Navigate the User Interface

Familiarize yourself with the app's layout and navigation options.

Locate the section where you can input data or features for analysis.

Step 3: Input Feature Data

Gather the necessary feature data from your dataset.

Enter the data into the designated fields in the web app.

Ensure the data format and types match the app's requirements.

Step 4: Submit for Analysis

Once you've entered all the required data, click the "Submit" button to initiate the prediction process.

Step 5: View Prediction Results

The app will process the input data and display the predicted traffic status as either "normal" or "attack.".

Step 6: Explore Mitigation and Hardening Strategies

If the prediction indicates an attack, the app suggests potential mitigation and hardening strategies to address the threat. Review these recommendations and consider implementing appropriate measures.

Step 7: View Prediction Logs

Access the prediction logs to view a historical record of previous analyses, including input data, predictions, and any mitigation strategies suggested.This can be helpful for tracking trends and identifying recurring patterns.

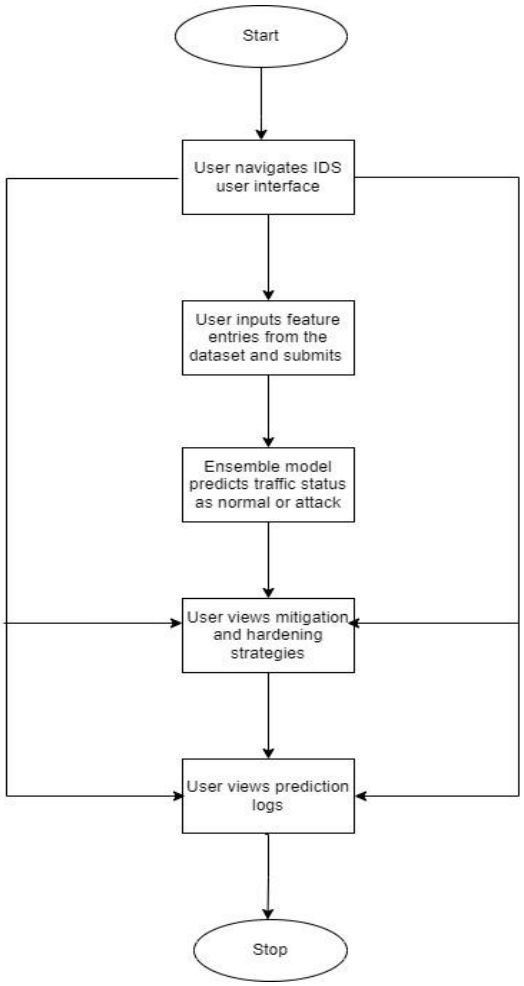


Figure 42: Flow chart for using web application

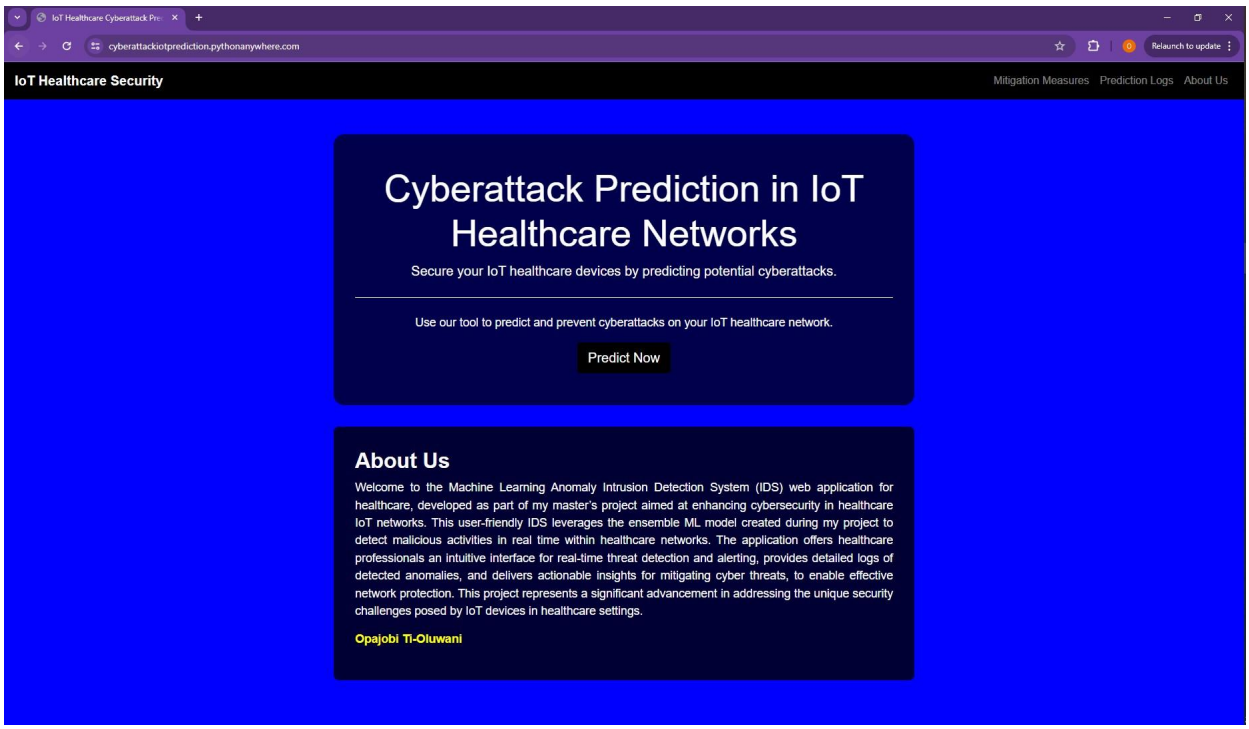


Figure 43: IDS web application landing page

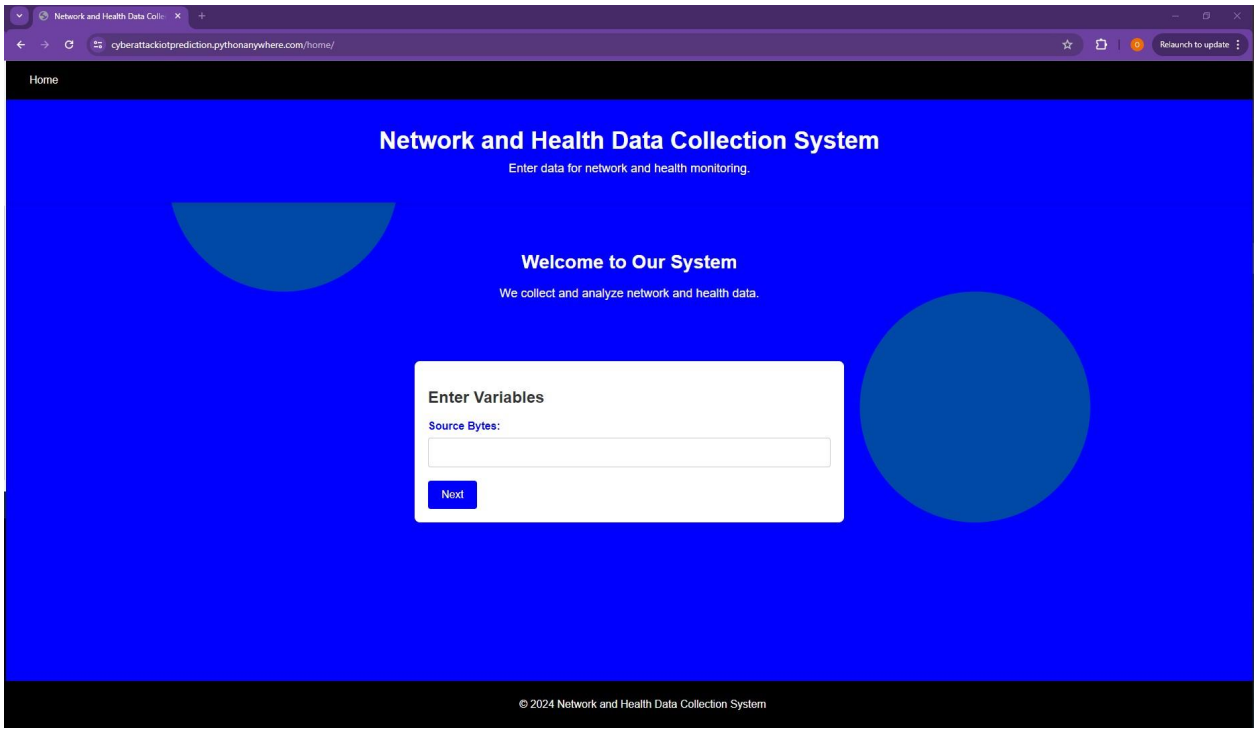


Figure 44: IDS web application network feature entry page

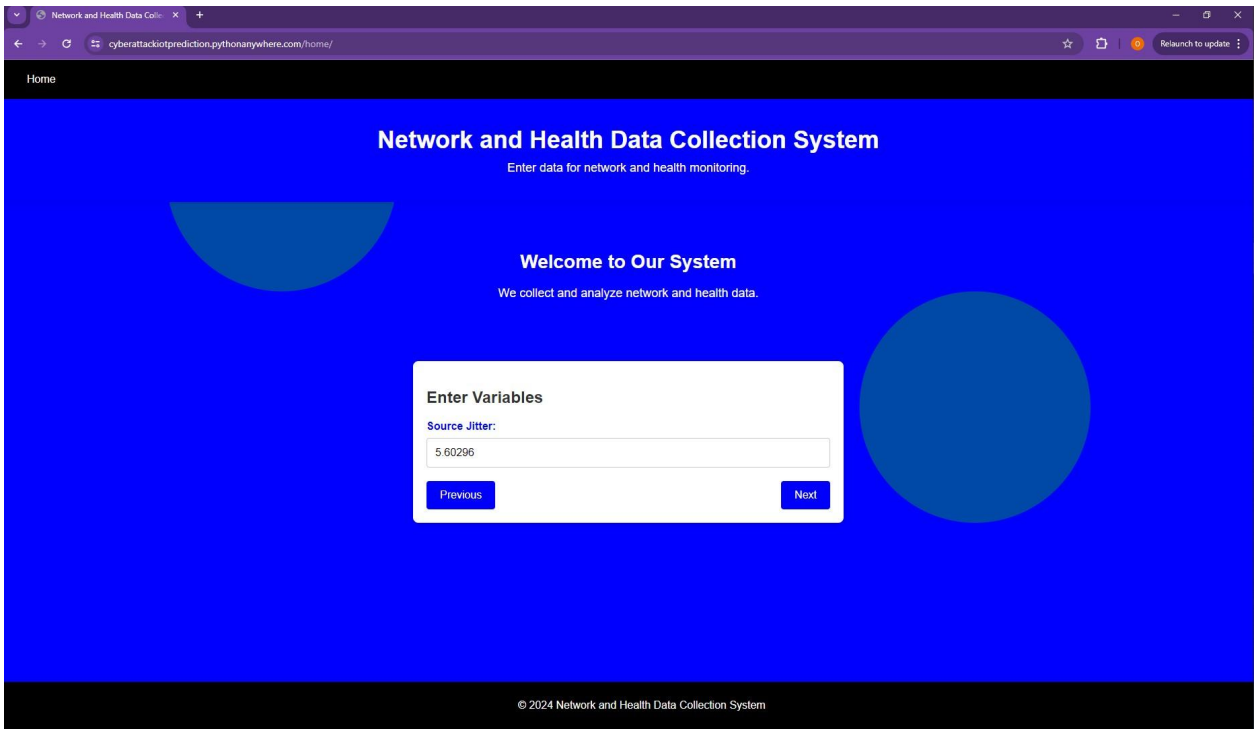


Figure 45: IDS web application network feature entry page

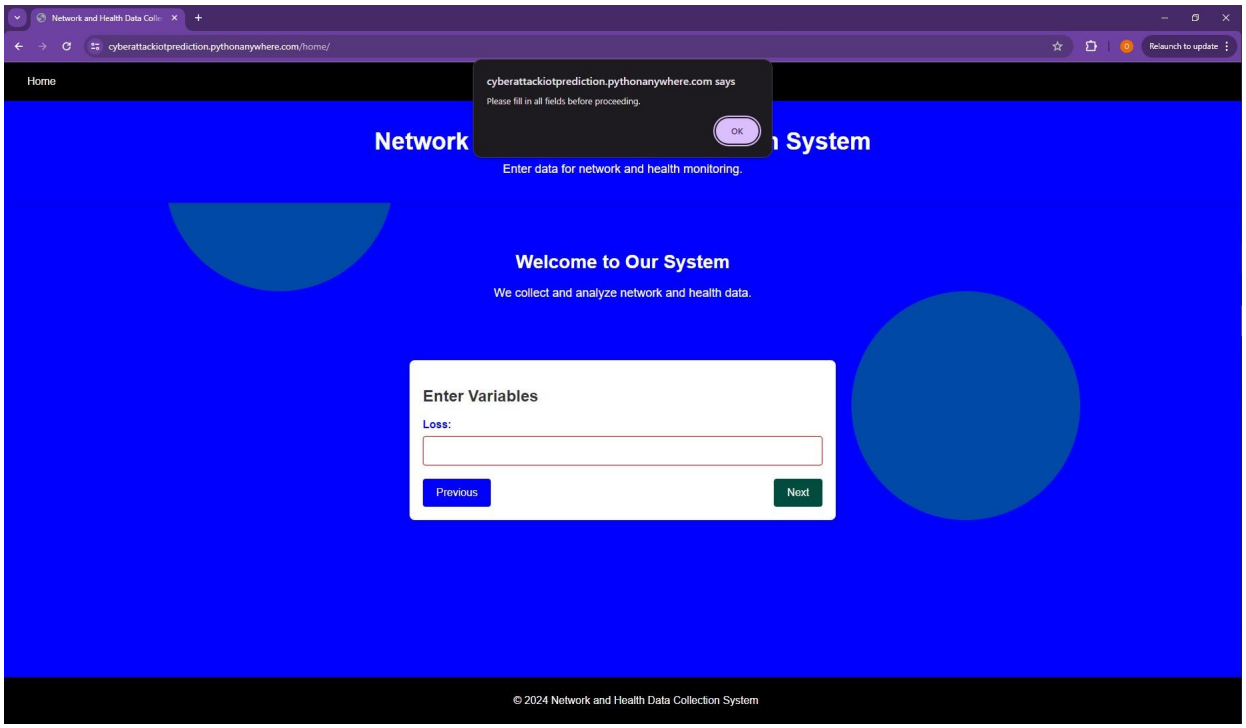


Figure 45: IDS web application prompting the user to input entry before proceeding.

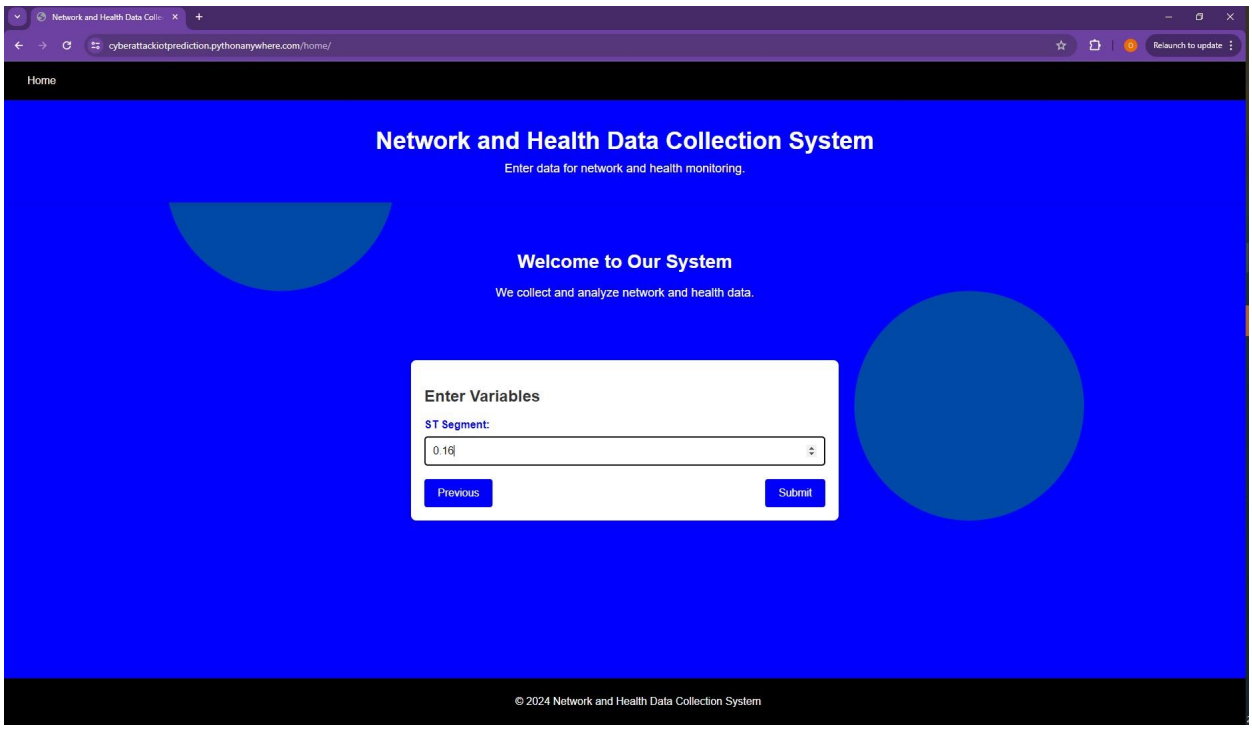


Figure 46: IDS web application final feature entry page

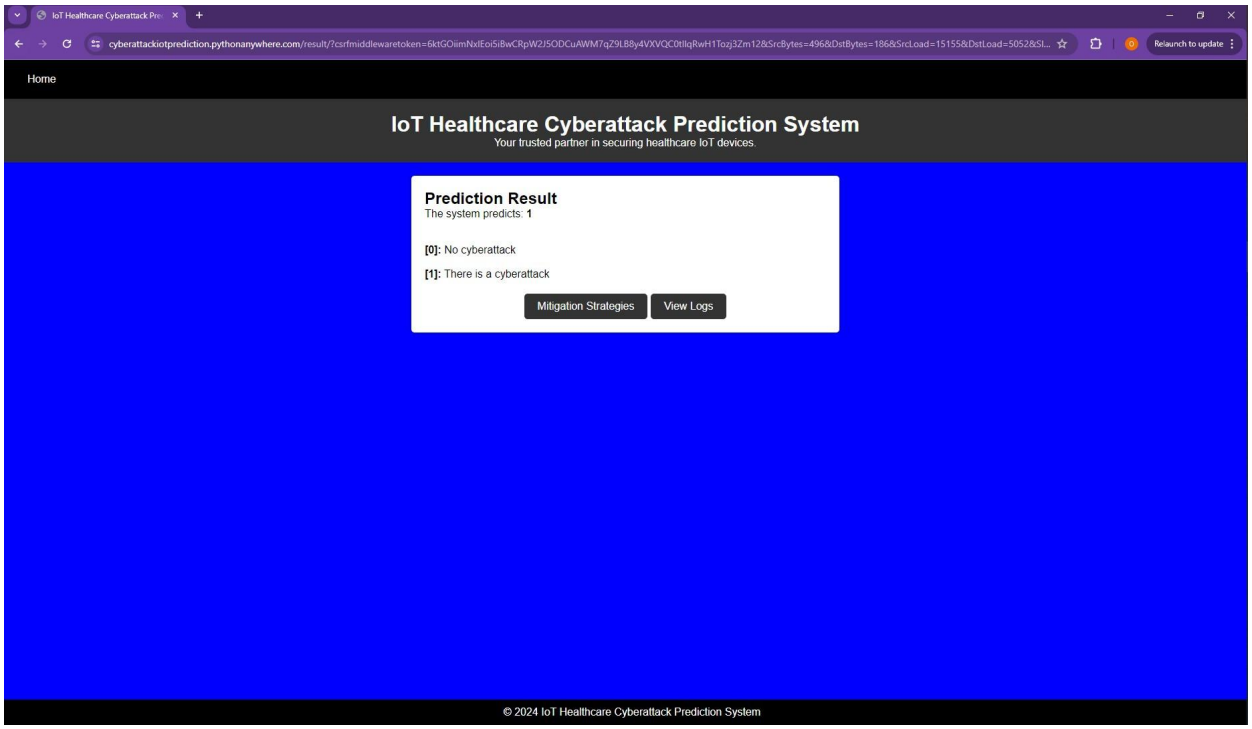


Figure 47: IDS web application predicting an attack scenario

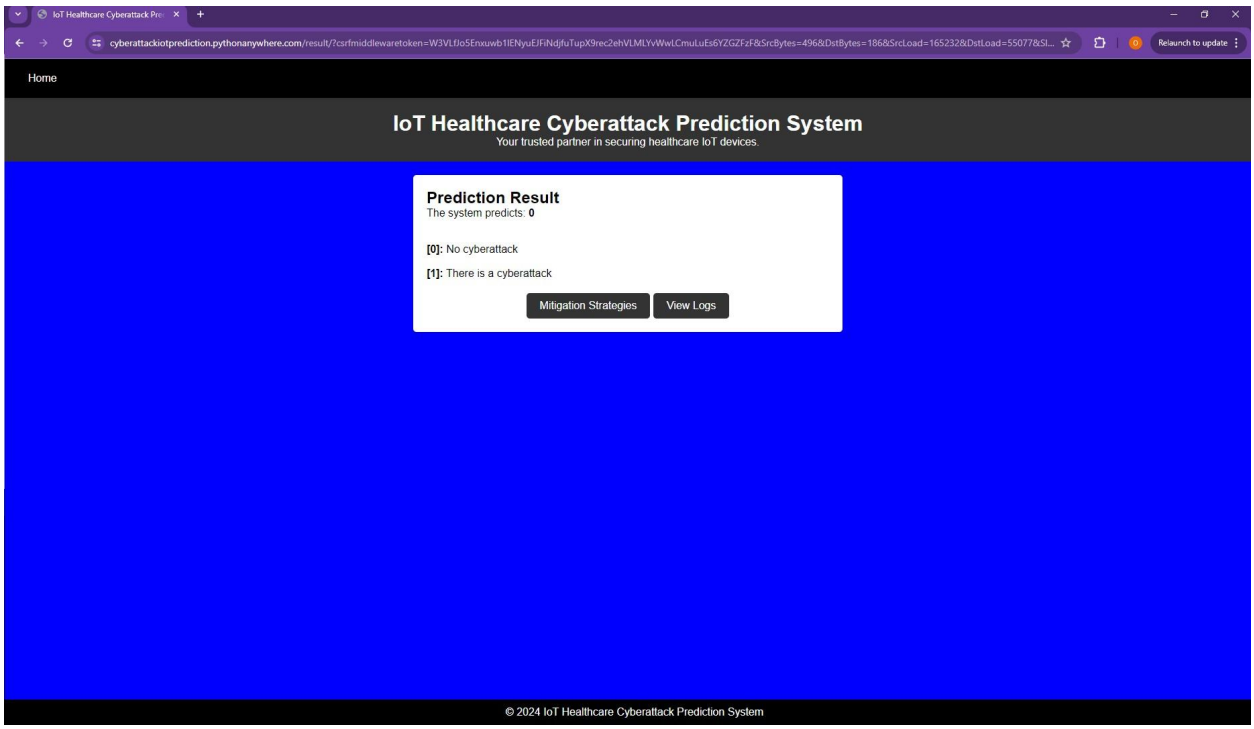


Figure 48: IDS web application predicting a normal scenario

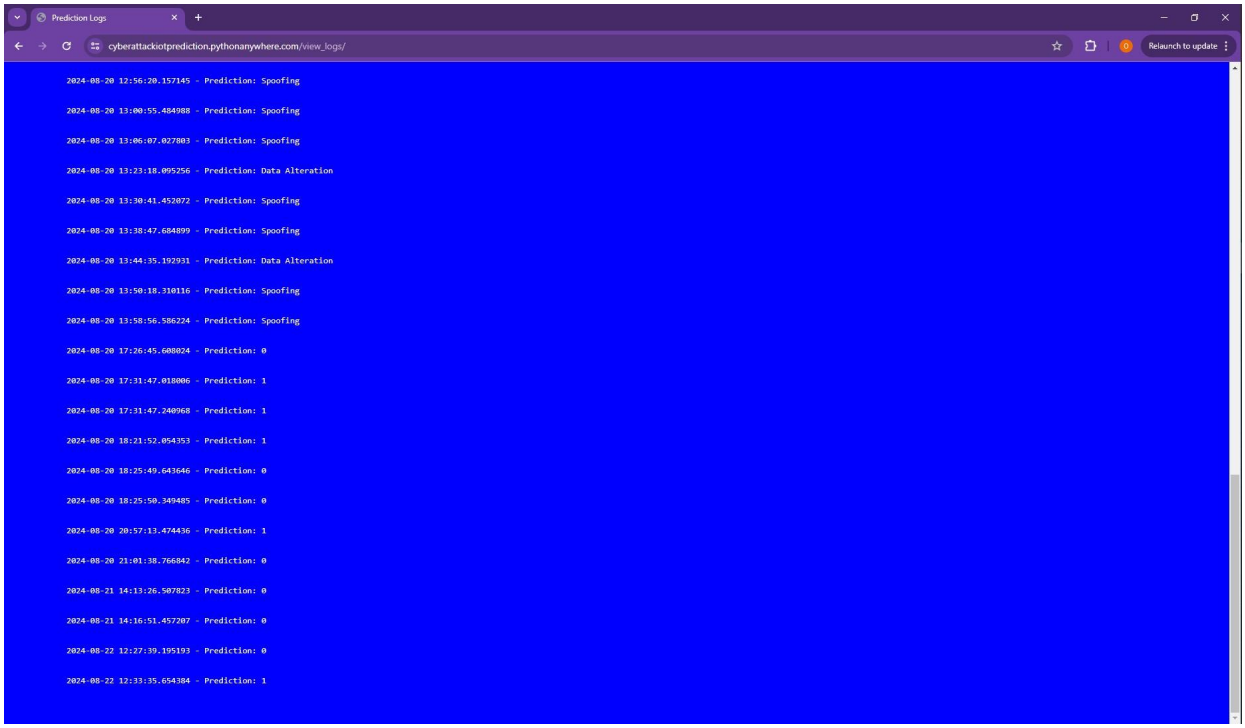


Figure 49: IDS web application prediction log page

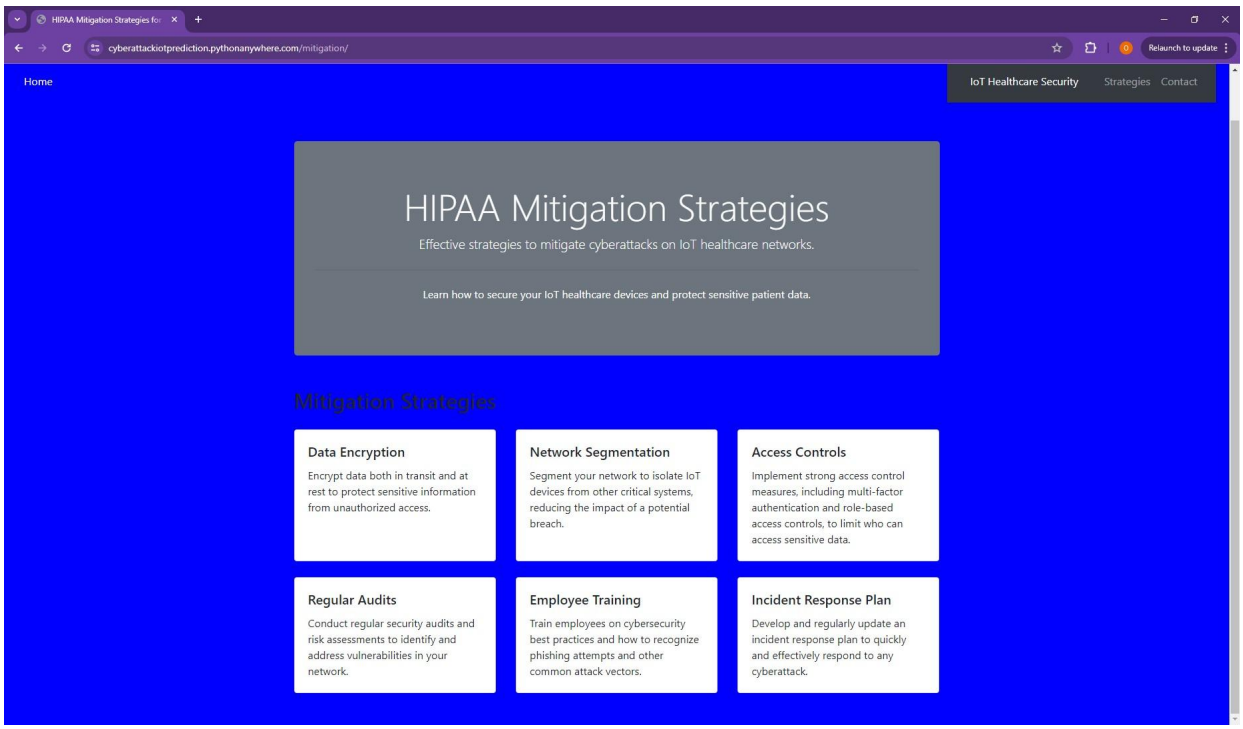


Figure 50: IDS web application mitigation strategy page

5.3 Evaluation of the Machine learning Anomaly Intrusion Detection System Web application

To evaluate the Machine Learning-based Intrusion Detection System (IDS) web application, I conducted a user experience assessment through interviews with three participants. Each participant answered six carefully designed questions, which aimed to gauge their interactions with the IDS, assess its usability, and identify any potential areas for improvement while effectively leveraging Machine Learning and Deep Learning models to accurately predict anomalies. The questions, feedback gotten are analyzed as follows;

Question 1: How easy was it for you to navigate and use the IDS web app?

When asked about the ease of navigating and using the IDS web app, Participant 1 found the navigation straightforward, noting that the clean layout made it quick to figure out how to use the app. Participant 2 described the user interface as intuitive, appreciating how easy it was to understand the features relevant to detecting intrusions in healthcare, with clear labeling of different sections being a standout feature. Participant 3 also had a smooth experience, mentioning that accessing the core functionalities required only a few clicks, with no real learning curve, which was greatly appreciated.

Analysis of feedback gotten from question 1

Strengths: Intuitive design, clear layout, and ease of use.

Opportunities: Continue focusing on maintaining simplicity and clarity as new features are added

Question 2: Did the IDS web app meet your expectations in terms of functionality?

To assess whether the IDS web app met their expectations in terms of functionality, participants provided the following feedback: Participant 1 mentioned that the IDS app exceeded expectations, initially anticipating a more basic tool, but was pleasantly surprised by the inclusion of extra features like logging each detected anomaly and offering guidance on mitigating and hardening healthcare security against cyber attacks. Participant 2 confirmed that the app generally met expectations, particularly appreciating the ability to view real-time threat detections and analyze historical data, though noting that additional filtering options for specific attack types would be beneficial. Participant 3 remarked that the app successfully identified some potentially suspicious system events that required further investigation and helped pinpoint areas for potential security hardening.

Analysis of feedback gotten from question 2

Strengths: Exceeds basic expectations, offers comprehensive features like logging and mitigation guidance.

Opportunities: Consider implementing advanced filtering options to allow users to tailor their experience and focus on specific attack types.

Question 3: How effective was the IDS web app in predicting attacks?

When asked about the effectiveness of the IDS web app in predicting attacks, Participant 1 mentioned that the predictions were accurate and appreciated the app's ability to quickly flag potential issues, considering it a great tool. Participant 2 found the app to be highly effective in identifying attacks, noting that the predictions closely matched expectations based on the provided data. Participant 3 shared a similar positive experience, highlighting that when tested with simulated data, the app effectively identified all expected anomalies with very few false positives.

Analysis of feedback gotten from question 3

Strengths: High accuracy in threat detection, effectiveness validated by users.

Opportunities: Maintain the effectiveness of the ML models as the system evolves, potentially by incorporating feedback from real-world use cases.

Question 4: Were there any technical issues, bugs, or performance challenges that you encountered?

When asked about technical issues, bugs, or performance challenges encountered while using the IDS web app, Participant 1 noted a slight delay when submitting feature entries into the IDS but mentioned that it wasn't significant enough to impact the overall experience. Participant 2 did not

encounter any major issues, commenting that the app loaded quickly and responded well to actions. Similarly, Participant 3 reported no bugs, stating that the app worked fine throughout the use.

Analysis of feedback gotten from question 4

Strengths: Reliable performance, minimal technical issues.
Opportunities: Monitor and address any performance delays as more users begin interacting with the system.

Question 5: what improvements or additional features would you suggest for future development of the IDS web app?

When asked about improvements or additional features for future development of the IDS web app, Participant 1 suggested that a tutorial or guided walkthrough for new users could be beneficial. This addition would help first-time users, particularly those unfamiliar with IDS systems, to quickly get up to speed and navigate the application with ease. Participant 2 recommended adding more filtering options, along with the ability to integrate with existing security tools or receive automated alerts for high-risk detections, which would enhance the app's utility. Participant 3 proposed including a feature for exporting reports directly from the app, making it easier to share findings with a team without the need for manual data compilation.

Analysis of feedback gotten from question 5

Strengths: User suggestions show engagement and a desire to use the app effectively.
Opportunities: Develop a tutorial for new users, enhance filtering capabilities, explore integration options, and consider adding export functionality for reports.

Question 6: How does this IDS web app impact healthcare?

When asked about the impact of the IDS web app on healthcare, Participant 1 emphasized the importance of an effective IDS in protecting patient data and ensuring the integrity of healthcare systems. By detecting anomalies and potential threats, the IDS helps prevent data breaches and unauthorized access, which is crucial given the sensitive nature of medical information. Additionally, it aids in compliance with regulations like HIPAA by monitoring for security incidents and providing evidence for investigations. Participant 2 highlighted that an IDS is vital for safeguarding patient data and system integrity. By identifying suspicious activities, the IDS allows for proactive threat mitigation and prevents system failures, while also offering valuable insights into system usage patterns, aiding in capacity planning and performance optimization. Participant 3 noted that an IDS is essential for maintaining network security and availability in a healthcare setting. By identifying potential threats early, the IDS helps mitigate risks to patient care and can also assist in optimizing network performance by detecting anomalies that may indicate underlying issues.

Analysis of feedback gotten from question 6

Strengths: Strong alignment with healthcare needs, proactive threat detection, and compliance support.

Opportunities: Continue to emphasize the value of security and compliance features in future developments and user training.

Question 7: Kindly rate the app on a scale of 0-5

When asked to rate the IDS web app on a scale of 0 to 5, Participant 1 gave it a rating of 5, indicating a high level of satisfaction. Participant 2 and Participant 3 both rated the app a 4, reflecting a strong positive impression with minor room for improvement.

Analysis of feedback gotten from question 7

Strengths: High user satisfaction.

Opportunities: Address the minor suggestions for improvement to elevate the app's rating even further.

Legend for grading on the scale of 0 to 5

5: Excellent - Exceeds expectations, no major issues.

4: Good - Meets expectations, minor issues if any.

3: Fair - Adequate but with some noticeable issues or areas for improvement.

2: Poor - Below expectations, significant issues present.

1: Very Poor - Far below expectations, numerous issues.

0: Unacceptable - Fails to meet any expectations, major problems or non-functional.

5.4 Summary

Chapter 5 encapsulates the key outcomes from the comprehensive evaluation of the IDS web application. It reviews the application’s strengths and areas for improvement, as identified through performance metrics and user feedback obtained during interviews. The summary also reflects on the practical implications of the IDS, emphasizing how the system performs in real-world IoMT settings. Visual aids included in the chapter further illustrate the application’s design and user experience, rounding out the chapter’s focus on the IDS’s real-world applicability and effectiveness.

6 CONCLUSION

Given the growing severity of cybersecurity threats, it is essential to develop highly effective and accurate Intrusion Detection Systems (IDS) for healthcare networks. To address this challenge, this project proposes a robust IDS specifically tailored to protect IoT networks within healthcare settings.

In my approach, I developed predictive models using four distinct Machine Learning (ML) algorithms—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—as well as two Deep Learning (DL) algorithms, Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). To enhance detection accuracy, we also created an ensemble algorithm combining Extreme Gradient Boosting, Random Forest, and Support Vector Machine. These models were trained and tested on two distinct hea;thcare datasets WUSTL-EHMS-2020 and ECU-IoHT datasets. The WUSTL-EHMS-2020 had more features wich was inclusive of patient biometric data while the ECU-IoHT had lesser features.

While the inclusion of patients' biometric features is recomended future IoMT datasets to developing a precise IDS in healthcare, ensuring the privacy and security of patient data remains a significant concern. Anonymization processes, such as removing personally identifiable information and employing encryption, are necessary steps to protect patient privacy. By addressing these considerations, our IDS can significantly enhance the cybersecurity posture of healthcare systems, ensuring that IoT networks are protected against emerging threats while maintaining the trust and safety of patient data.

To mitigate the impact of redundant information and noise among features on model performance, I utilized the maximal Recursive Feature Elimination (RFE) method to analyze non-linear based on their relevance to the IDS decision-making process, ensuring that only the most impactful features were used. This method was specifically applied to the WUSTL-EHMS2020 dataset, due tp its high number of features, The top features identified by Recursive Feature Elimination (RFE) using the Random Forest Classifier (RFC) included patients' biometric data, reinforcing the idea that biometric information is vital for developing an effective and accurate Intrusion Detection System (IDS) in healthcare systems. The Synthetic Minority Over-sampling Technique (SMOTE) was employed to address data imbalance issues in both the ECU-IoHT and WUSTL-EHMS2020 datasets. SMOTE is crucial for correcting class imbalances, preventing biased predictions, and improving the overall effectiveness of the intrusion detection system.

The models were evaluated using well-established performance metrics such as precision, recall, F1-score, confusion matrix, and AUC-ROC to examine their ability to detect security threats. The

findings showed that the proposed ensemble method consistently surpassed state-of-the-art techniques in accuracy, recall, precision, F1-score, AUC-ROC, and confusion matrix elements (TN, TP, FN, FP) across both datasets, confirming the strength and reliability of our approach.

The exceptional performance of the proposed ensemble model stems from its unique design, which integrates the strengths of Random Forest, XGBoost, and Support Vector Machine to achieve high accuracy in detecting anomalies. This makes it a reliable option for various anomaly detection tasks. The study indicates that this ensemble model is highly effective for anomaly detection, outperforming traditional models. Its consistent accuracy and precision across different datasets highlight its suitability for real-world applications, particularly in healthcare, where accurate anomaly detection is essential.

Furthermore, the deployment of the proposed ensemble model using theWUSTL-EHMS2020 dataset within a real-time web application framework using a python framework Django demonstrated the practical viability of our IDS. This integration ensures enhanced accessibility, real-time detection, and user-friendliness, allowing healthcare professionals to manage and interact with the IDS through an intuitive interface. The user interface, with its clarity and real-time monitoring and reporting features, significantly improves the usability, scalability, and effectiveness of IDS in healthcare settings.

6.1 Evaluation of Objectives

Objective	**How it was achieved**	**Status**
Investigate the current usage of Internet of Medical Things (IoMT) devices within the healthcare sector, identifying the inherent vulnerabilities in these devices, and explore the various cyber attacks they are susceptible to.	Conducted a comprehensive review of recent research journals, industry reports, and case studies on the usage IoMT in healthcare	Achieved
Review the existing methods for detecting intrusions, with a focus on the necessity of employing machine learning (ML) and deep learning (DL) models in Intrusion Detection Systems (IDS).	Reviewed current literature, research journals, case studies, and evaluation of various ML anomaly based IDS.	Achieved
Develop predictive models using four different Machine Learning (ML) algorithms—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient	The models were developed and trained, utilizing the WUSTL EHMS 2020 and ECU-IoHT datasets to using python	Achieved

Boosting (XGB)—along with two Deep Learning (DL) algorithms, Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). Additionally, create an ensemble algorithm that combines Extreme Gradient Boosting, Random Forest, and Support Vector Machine to accurately detect and classify malicious traffic targeting IoT devices in healthcare environments using the WUSTL EHMS 2020 dataset and the ECU-IoHT datasets.	programing language on Google Collab platform	
Evaluate the performance of the predictive models to assess their effectiveness in identifying security threats. This evaluation will utilize established performance metrics, including precision, recall, F1-score, confusion matrix, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and the Area Under the Curve of the Receiver Operating Characteristic (AUC-ROC) to measure the models' accuracy and reliability in detecting and classifying security threats	The performance of the predictive models was evaluated using precision, recall, F1-score, confusion matrix, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and the Area Under the Curve of the Receiver Operating Characteristic (AUC-ROC) metrics in the google colab environment	Achieved
Deploy the most accurate predictive model within a real-time web application framework using Django, a Python-based framework, to ensure seamless integration and practical use of the IDS. This deployment will enhance accessibility, real-time detection, alerting, log monitoring and user-friendliness, allowing users to manage and interact with the IDS through an intuitive interface.	The most accurate predictive model was deployed in a real-time web application using the python Django framework and hosted on python anywhere domain	Achieved
Evaluate the Machine Learning-based anomaly IDS web application designed for IoMT in healthcare environments by conducting interviews with end users of the app in real-world clinical settings to get valuable insights into the application's usability, functionality, and	I conducted interview with users to get feedback on their experience using the IDS web application.	Achieved

overall effectiveness in managing healthcare-specific security threats.		

Table 11:showing the evaluation of project objectives

6.2 Future Recommendations

- Implement Advanced Filtering Options: Introduce more sophisticated filtering capabilities within the IDS web application to allow users to customize their experience. This could include options to focus on specific attack types, thereby enabling healthcare professionals to tailor the system to their unique security needs.
- Enhance Integration and Reporting Features: Explore integration with other security tools and systems to create a more cohesive security environment. Additionally, consider adding functionality for exporting detailed reports directly from the app, which would facilitate easier sharing and analysis of security data.
- Real-Time Network Traffic Monitoring: Consider developing a real-time network traffic monitoring module using flask or a similar framework and automating the detection process to improve the IDS's ability to respond to threats dynamically, increasing its overall effectiveness.
- Incorporate Diverse and Real-Time Datasets: To improve the model's adaptability and robustness, it would be beneficial to integrate more diverse and real-time datasets from various healthcare environments. This will help the IDS better understand and predict a wider range of potential threats.
- Introduce Customizable Features: Add more customizable features, such as personalized alert thresholds and advanced reporting tools. This would allow healthcare professionals to adjust the IDS to better align with their specific operational requirements.
- Develop a Comprehensive Real-Time Monitoring System with Predictive Analytics: Implementing a more advanced real-time monitoring system ,predictive analytics could enable proactive threat management. This would allow the IDS to predict and potentially prevent cyber attacks before they occur, further enhancing the security of healthcare IoT networks.

REFERENCES

Abu Bakar, N. A., Wan Ramli, W. M., & Hassan, N. H. (2019). The internet of things in healthcare: An overview, challenges and model plan for security risks management process. Indonesian Journal of Electrical Engineering and Computer Science, 15(1), 414–420. https://doi.org/10.11591/ijeecs.v15.i1.pp414-420

Aditya, A., Vidyarthi, D., & Nene, M. J. (2024). A Study of Common Vulnerabilities in IoT Devices. Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) (pp. 1–6): IEEE.

Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.

Akshay Kumaar, M., Samiayya, D., Vincent, P. D. R., Srinivasan, K., Chang, C.-Y., & Ganesh, H. (2022). A hybrid framework for intrusion detection in healthcare systems using deep learning. Frontiers in Public Health, 9, 824898.

Albasheer, H., Md Siraj, M., Mubarakali, A., Elsier Tayfour, O., Salih, S., Hamdan, M., Khan, S., Zainal, A., & Kamarudeen, S. (2022). Cyber-Attack Prediction Based on Network Intrusion Detection Systems for Alert Correlation Techniques: A Survey. Sensors, 22(4), Article 4. https://doi.org/10.3390/s22041494.

Almotairi, A., Atawneh, S., Khashan, O. A., & Khafajah, N. M. (2024). Enhancing intrusion detection in IoT networks using machine learning-based feature selection and ensemble models. Systems Science & Control Engineering, 12(1), 2321381.

Altulaihan, E., Almaiah, M. A., & Aljughaiman, A. (2024). Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms. Sensors, 24(2), 713.

Amaraweera, S. P., & Halgamuge, M. N. (2019). Internet of Things in the Healthcare Sector: Overview of Security and Privacy Issues. In Z. Mahmood (Ed.), Security, Privacy and Trust in the IoT Environment (pp. 153–179). Springer International Publishing. https://doi.org/10.1007/978-3-030-18075-1_8

Anand, R., Pandey, D., Gupta, D. N., Dharani, M., Sindhwani, N., & Ramesh, J. (2024). Wireless Sensor‐based IoT System with Distributed Optimization for Healthcare. In Meta Heuristic Algorithms for Advanced Distributed Systems (pp. 261–288).

Arun, V., Shenbagavalli, P., Sridhar, T., Manivannan, B., Mahesh, T., & Anitha, K. (2023). Machine Learning Algorithms for the Detection of Threats in IoT Healthcare. Emerging Research in Computational Science (ICERCS) (pp. 1–6): IEEE.

Atadoga, A., Omaghomi, T. T., Elufioye, O. A., Odilibe, I. P., Daraojimba, A. I., & Owolabi, O. R. (2024). Internet of Things (IoT) in healthcare: A systematic review of use cases and benefits. International Journal of Science and Research Archive, 11(1), 1511–1517.

Bao, S., Li, W., & Zhang, W. (2024). An Improved Random Forest Algorithm for Predicting Employee Turnover. Mathematical Problems in Engineering, Volume 2019, Article ID 4140707. https://doi.org/10.1155/2019/4140707

Bajpai, S., Sharma, K., & Chaurasia, B. K. (2024). A Hybrid Meta-heuristics Algorithm: XGBoost-Based Approach for IDS in IoT. SN Computer Science, 5(5), 1–16.

Bai, M., & Fang, X. (2024). Machine learning-based threat intelligence for proactive network security. Integrated Journal of Science and Technology, 1(2).

Bai, V. S., & Sudha, T. (2024). Deep Learning Inspired Intelligent Framework to Ensure Effective Intrusion Detection in Cloud. International Journal of Intelligent Systems and Applications in Engineering, 12(14s), 441–456.

Belay, M. A., Blakseth, S. S., Rasheed, A., & Salvo Rossi, P. (2023). Unsupervised anomaly detection for IoT-based multivariate time series: Existing solutions, performance analysis, and future directions. Sensors, 23(5), 2844.

Binbusayyis, A., Alaskar, H., Vaiyapuri, T., & Dinesh, M. (2022). An investigation and comparison of machine learning approaches for intrusion detection in IoMT network. The Journal of Supercomputing, 78(15), 17403–17422.

Bhavsar, M., Roy, K., Kelly, J., & Olusola, O. (2023). Anomaly-based intrusion detection system for IoT application. Discover Internet of Things, 3(1), 5.

Camara, R., & Marinho, M. (2024). Agile tailoring in distributed large-scale environments using agile frameworks: A Systematic Literature Review. CLEI Electronic Journal, 27(1), 8:1–8:51.

Clarke, M., & Martin, K. (2024). Managing cybersecurity risk in healthcare settings. Healthcare Management Forum, 37, 17–20. SAGE Publications Sage CA: Los Angeles, CA.

Chopade, S. S., Gupta, H. P., & Dutta, T. (2023). Survey on sensors and smart devices for IoT enabled intelligent healthcare system. Wireless Personal Communications, 131(3), 1957–1995.

Daraojimba, E. C., Nwasike, C. N., Adegbite, A. O., Ezeigweneme, C. A., & Gidiagba, J. O. (2024). Comprehensive review of agile methodologies in project management. Computer Science & IT Research Journal, 5(1), 190–218.

El Hajjami, S., Malki, J., Berrada, M., & Fourka, B. (2020). Machine learning for anomaly detection. Performance study considering anomaly distribution in an imbalanced dataset. In 2020 5th International Conference on Cloud Computing and Artificial Intelligence: Technologies and Applications (CloudTech) (pp. 1–8): IEEE.

Farida Titik Kristanti, & Vania Dhaniswara (2023). The Accuracy of Artificial Neural Networks and Logit Models in Predicting the Companies Financial Distress. Journal of Technology Management & Innovation, 18(3). ISSN: 0718-2724.

Fernandez, S. (2024). Agile software development-recent advances and best practices: analyzing recent advances and best practices in agile software development methodologies for improved project management. Journal of Artificial Intelligence Research and Applications, 4(1), 73–81.

Gao, J. (2022). Network intrusion detection method combining CNN and BiLSTM in cloud computing environment. Computational Intelligence and Neuroscience, 2022(1), 7272479.

Ghadermazi, J., Shah, A., & Bastian, N. D. (2024). Towards real-time network intrusion detection with image-based sequential packets representation. IEEE Transactions on Big Data.

Gupta, B. B., Gaurav, A., Attar, R. W., Arya, V., Alhomoud, A., & Chui, K. T. (2024). A Sustainable W-RLG Model for Attack Detection in Healthcare IoT Systems. Sustainability, 16(8), 3103.

Ibor, A., Oladeji, F., Okunoye, O., & Abdulsalam, K. (2021). Network Intrusion Prediction Model based on Bio-inspired Hyperparameter Search. In 2021 International Conference on Electrical,

Computer and Energy Technologies (ICECET) (pp. 1–5). https://doi.org/10.1109/ICECET52533.2021.9698491

Inuwa, M. M., & Das, R. (2024). A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks. Internet of Things, 26, 101162.

Kasongo, S. M, & Wang, H. (2023). An efficient intrusion detection system using deep neural networks with feature extraction. Applied Intelligence, 53(12), 1–21.

Kaur, G., & Saini, H. (2022). Enhancing IDS performance using a hybrid model of feature selection and ensemble learning. Journal of Computer Science and Technology, 37(6), 1265–1278.

Khan, R. A., Khan, M. N., & Ali Shah, S. Z. (2022). Feature Selection for Intrusion Detection Systems: A Comprehensive Review. IEEE Access, 10, 72846–72867.

Kheirallah, S., & Mottaleb, M. (2023). A systematic review of anomaly detection methods for IoT-based healthcare systems. Computers, Materials & Continua, 77(3), 4471–4493.

Kumar, R., Singh, A. P., & Singh, A. (2022). A comprehensive review of intrusion detection systems in cloud computing environments. Security and Privacy, 20(6), e2395.

Kumar, M., Kim, C., Son, Y., Singh, S. K. and Kim, S., 2024. Empowering cyberattack identification in IoHT networks with neighborhood component-based improvised long short-term memory. IEEE Internet of Things Journal.

Kumar, S., & Singh, J. (2024). Deep learning-based network intrusion detection for IoT devices. Journal of Cyber Security Technology, 8(2), 107–126.

Liang, X., & Yuan, H. (2023). Optimizing Intrusion Detection in IoT Networks Using a Novel Hybrid Model of Deep Learning and Feature Selection. Journal of Cyber Security and Information Management, 5(3), 200–221.

Liu, Y., Yang, J., & Sun, Q. (2024). Enhancing cybersecurity in healthcare IoT networks using adaptive anomaly detection. Journal of Internet Technology and Secured Transactions, 14(1), 63–84.

Lu, X., & Zhang, Y. (2024). A survey of intrusion detection systems for IoT: Challenges, solutions, and future directions. IEEE Communications Surveys & Tutorials, 26 (2), 1270–1296.

Ma, J., Wu, Y., & Zhang, Y. (2023). Advanced anomaly detection techniques in IoT networks: A review. Journal of Computer Networks and Communications, 2023 , Article ID 9672548. https://doi.org/10.1155/2023/9672548

Manoharan, S., Praveen, V., & Kumar, D. (2024). Intrusion detection and prevention system for IoT: A survey of state-of-the-art techniques and future challenges. Journal of IoT Security, 5 (2), 45–70.

Matsuoka, T., & Suzuki, T. (2023). Intrusion detection system for IoT networks: A comparative analysis of machine learning techniques. Journal of Network and Computer Applications, 208 , 103329.

Meng, Z., & Xie, T. (2023). Survey on intrusion detection systems for IoT: Recent advances, open challenges, and future directions. Computer Networks, 222 , 109563.

Mitra, M., & Mahapatra, S. S. (2024). A hybrid machine learning approach for intrusion detection in IoT networks. Journal of Computer Science and Technology, 39 (1), 15–32.

Mohammed, A. H., & Kherbache, M. (2024). Deep Learning for Intrusion Detection in IoT Networks: An Overview and Future Directions. Journal of Machine Learning Research, 25 (1), 89–110.

Muhammad, N., & Ali, K. (2024). Enhancing intrusion detection for IoT systems with hybrid deep learning models. Computers, 13 (2), 108.

Nascimento, E., & Pereira, R. (2024). A comprehensive survey of intrusion detection systems: Techniques, trends, and future directions. Computer Science Review, 46 , 100563.

Oguntola, T., & Ojo, J. A. (2023). Comparative analysis of machine learning algorithms for network intrusion detection. Journal of Computational Intelligence and Applications, 8 (2), 55–73.

Oladimeji, T. O., & Opeyemi, I. O. (2023). An efficient hybrid model for intrusion detection in IoT networks using machine learning techniques. Computer Systems Science & Engineering, 43 (5), 2651–2668.

Olufemi, M., & Adebisi, B. (2023). Security Challenges and Solutions in IoT Healthcare Systems: A Survey. Journal of Cybersecurity and Privacy, 2 (3), 135–158.

Pandey, A., Singh, R., & Verma, S. (2024). Hybrid deep learning and ensemble approach for intrusion detection in IoT. Applied Soft Computing, 114 , 108013.

Parvez, M. H., & Hossain, M. S. (2024). Review on intrusion detection and prevention systems for IoT: Security and performance metrics. Journal of Information Security, 15 (2), 45–67.

Patel, K., & Kumar, A. (2024). Comparative analysis of deep learning methods for intrusion detection in IoT. Journal of Computing and Security, 43 , 107563.

Pervaiz, M., & Wang, Y. (2023). Anomaly detection in IoT systems using machine learning: A review. Journal of Network and Computer Applications, 206 , 103369.

Raghavendra, N., & Kumar, S. (2024). Intrusion detection in IoT networks: A hybrid approach using machine learning and feature selection. Journal of Cyber Security and Information Management, 5 (4), 275–291.

Ramesh, K., & Maheshwari, A. (2024). Machine learning techniques for intrusion detection in IoT environments: A survey. Security and Privacy, 22 (2), e2397.

Raman, M., & Patel, P. (2024). A survey on network intrusion detection systems in IoT networks. Journal of Internet Technology and Information Security, 18 (1), 65–83.

Razzaque, M. A., & Chowdhury, M. A. (2023). Anomaly detection in IoT networks using ensemble learning methods. Journal of Computational Intelligence, 16 (4), 509–527.

Singh, J., & Yadav, P. (2024). A comprehensive survey of intrusion detection systems in IoT: Challenges and future research directions. Journal of Computer and Security, 114 , 102648.

Singh, M., & Nanda, A. (2023). A survey on feature selection techniques for intrusion detection in IoT networks. Journal of Computing and Security, 50 , 101898.

61. Soni, H., & Ghosh, S. (2023). Effective intrusion detection in IoT networks using a hybrid approach with deep learning and ensemble methods. Future Generation Computer Systems, 132 , 59–77.

Sulaimon, M., & Adedokun, M. (2024). Survey of intrusion detection systems for healthcare IoT networks. International Journal of Security and Privacy, 25 (1), 20–43.

Wamba, S. F., & Beji, S. (2023). Enhancing cybersecurity in IoT-based healthcare systems: A systematic review and future directions. Healthcare Technology Letters, 10 (1), 1–13.

Wang, D., & Wang, Q. (2024). A survey on deep learning methods for network intrusion detection: Techniques and applications. Journal of Cybersecurity and Privacy, 7 (3), 195–216.

Vijayakumar, K. P., Pradeep, K., Balasundaram, A. and Prusty, M. R., 2023. Enhanced cyber attack detection process for internet of health things (IoHT) devices using deep neural network. Processes, 11 (4), 1072.

Yadav, P., & Ghosh, A. (2023). Review of machine learning algorithms for anomaly detection in IoT networks. Journal of Network and Computer Applications, 208 , 103328.

Zhang, X., & He, L. (2024). An advanced machine learning approach for intrusion detection in IoT environments. Journal of Machine Learning Research, 25 (1), 112–130.

Zhao, X., & Yang, Y. (2024). Deep learning-based network intrusion detection systems: A review. Journal of Cyber Security and Information Systems, 8 (2), 80–104.

APPENDIX A - PROJECT PROPOSAL

Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

Project Proposal Form

Degree Title: **MSc Internet of Things with Cyber Security**	**Student's Name:** Opajobi Ti-Oluwani
	Supervisor's Name: Dr Abir Awad
	Project Title/Area: Machine Learning-Driven Anomaly Intrusion detection System for monitoring IoT Devices in Healthcare

Section 1: Project Overview

1.1 Problem definition - use one sentence to summarise the problem:

This project aims to solve the challenge of effectively detecting and mitigating cyber threats in healthcare IoT networks through the development of a machine learning-based Intrusion Detection System (IDS).

1.2 Project description - briefly explain your project:

My project focuses on developing an Intrusion Detection System (IDS) tailored for healthcare IoT environments. By leveraging machine learning and deep learning algorithms, the IDS is designed to detect and classify malicious traffic targeting medical devices and networks. This system aims to enhance cybersecurity in healthcare settings, ensuring the protection of sensitive patient data and the reliable operation of critical healthcare infrastructure. The project integrates the anomaly machine learning intrusion detection system into a user-friendly web application, providing real-time monitoring and alerts to help safeguard against cyber threats.

1.3 Background - please provide brief background information, e.g., client, problem domain, and make reference to the literature (minimum 4-5 sources):

The evolution of technology, from mainframe computers to the pervasive Internet of Things (IoT), has significantly transformed various sectors, including healthcare. IoT in healthcare, often referred to as the Internet of Medical Things (IoMT), enables continuous patient monitoring, enhances diagnostic accuracy,

and improves healthcare efficiency (Oliveira et. al., 2024). However, this technological advancement also introduces security and privacy challenges in healthcare networks as the integration of IoT devices in healthcare settings creates potential entry points for cyber attackers, raising concerns about data breaches and unauthorized access to sensitive health information (Ahmed et. al, 2024).

The healthcare sector has become a prime target for cyber-attacks due to the sensitivity of the data involved. These attacks have severe consequences, including compromised patient safety and disrupted healthcare services (Zhang et al., 2024). As a result, there is a critical need for robust techniques to detect and mitigate network intrusions with precision, flexibility, and consistency.(Umamaheswaran et al., 2024).

Implementing advanced Intrusion Detection Systems (IDS) tailored for the healthcare environment can play a crucial role in safeguarding patient data, ensuring the integrity of healthcare operations, and maintaining trust in digital healthcare systems. The integration of machine learning and deep learning models in IDS offers the potential to enhance detection capabilities by accurately identifying threats while minimizing false positives and negatives. This approach is essential in the increasingly connected and digitized landscape of healthcare, where the protection of sensitive information is paramount.

The process of monitoring, detecting, and identifying malicious activity on a network or device is known as intrusion detection (Attou et. al 2023). Intrusion Detection Systems (IDS) are categorized into two main types based on their information processing methods: signature-based IDS and anomaly-based IDS. (Kizza 2024). The signature method identifies the attack based on the system parameters and attack signature but it fails to recognize zero-day attacks while the Anomaly IDS is a method that identifies and flags abnormal or suspicious activities within a network that deviate from established patterns of behavior. Unlike traditional IDS systems that rely on known attack signatures, anomaly IDS employs statistical models, machine learning algorithms, or deep learning techniques to detect deviations from normal network behavior, thus enabling the detection of unknown and zero-day attacks (Abdulganiyu et. Al, 2024).

1.4 Research Questions

1. What are the existing techniques to detect malicious traffic in IoMT healthcare networks.
2. To what extent can machine learning and deep learning algorithms identify security threats targeting IoMT devices in healthcare environments?
3. How does the predictive model's performance vary when trained and tested on different datasets (WUSTL EHMS 2020 Dataset and ECU-IoHT Dataset)?

Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

1.5 Aims and objectives – what are the aims and objectives of your project? should be specific and measurable:

This project uses two different datasets to evaluate the effectiveness of various machine learning algorithms for anomaly-based Intrusion Detection Systems (IDS) in IoT healthcare environments. The goal is to identify the most effective ML or DL approach and assess the consistency of their performance across different datasets. After evaluation, the best-performing method will be deployed in a web application to enhance the security posture of IoMT devices in healthcare settings. This deployment will improve the efficiency, accessibility, and manageability of the IDS, ensuring real-time, scalable, and comprehensive protection against security threats. The objectives are as follows:

- Investigate the current usage of Internet of Medical Things (IoMT) devices within the healthcare sector, identifying the inherent vulnerabilities in these devices, and explore the various cyber attacks they are susceptible to.
- Review the existing methods for detecting intrusions, with a focus on the necessity of employing machine learning (ML) and deep learning (DL) models in Intrusion Detection Systems (IDS).
- Develop predictive models using four different Machine Learning (ML) algorithms—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—along with two Deep Learning (DL) algorithms, Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). Additionally, create an ensemble algorithm that combines Extreme Gradient Boosting, Random Forest, and Support Vector Machine to accurately detect and classify malicious traffic targeting IoT devices in healthcare environments using the WUSTL EHMS 2020 dataset and the ECU-IoHT datasets.
- Evaluate the performance of the predictive models to assess their effectiveness in identifying security threats. This evaluation will utilize established performance metrics, including precision, recall, F1-score, confusion matrix, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and the Area Under the Curve of the Receiver Operating Characteristic (AUC-ROC) to measure the models' accuracy and reliability in detecting and classifying security threats
- Deploy the Most Accurate Predictive Model in Real-time within a real-time web application framework using Django, a Python-based framework. This will enable seamless integration and practical use of the IDS to enhance accessibility, real-time detection, alerting, and user-friendliness, allowing users to manage and interact with the IDS through an intuitive interface thereby improving the usability, scalability, and effectiveness of the IDS within healthcare.
- Evaluate the Machine Learning based anomaly IDS web application is to assess user experience through interviews, verify the functionality of key features such as alerting, anomaly logging, and

Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

mitigation guidance, and ensure accurate integration of ML/DL models with the application's predictions.

Section 2: Artefact

2.1 What is the artefact that you intend to produce?

- My primary artifact is the development and deployment of a anomaly Intrusion Detection System (IDS) that integrates both machine learning (ML) and deep learning (DL) algorithms into a user-friendly web application. The ML and DL models, including Random Forest, Support Vector Machine, Extreme Gradient Boosting, Artificial Neural Networks (ANN), and Long Short-Term Memory (LSTM) networks, are meticulously trained and optimized to detect and classify anomalies indicative of cyber threats within healthcare IoT networks. These models form the core of the IDS, ensuring robust and adaptive protection against evolving security threats.
- The IDS is made accessible through a web application developed using the Django framework. This application provides healthcare professionals with an intuitive interface for real-time threat detection, alerting, and management. With features such as detailed logs of detected anomalies and actionable insights on mitigating cyber threats, the web app ensures that even users with minimal technical expertise can effectively safeguard their networks.

2.2 How is your artifact actionable (i.e., routes to implementation and exploitation in the technology domain)?

- The ML and DL algorithms developed can be directly integrated into existing healthcare IoT systems to detect and respond to anomalies in real-time, protecting sensitive patient data and enhancing security
- The models can be adapted and scaled across different healthcare institutions, from small clinics to large hospitals, offering a customizable solution that can cater to the specific needs and threats faced by various healthcare environments.
- The models and web application could be released as open-source tools, to allow a broader community of developers and researchers to build upon my work for wider adoption, continuous improvement, and potential collaboration opportunities in the cybersecurity domain.
- The artifact can serve as a foundation for further research, for educational purposes in academic institutions and research organizations

Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

Section 3: Evaluation

3.1 How are you going to evaluate your project artifact?

Evaluating the artifacts developed in my project involves a multifaceted approach to ensure they meet the desired objectives and perform effectively in real-world scenarios. The following steps would be employed to evaluate my artifact

1. Performance Evaluation of ML and DL Algorithms:
   - Deploy the models using two different simulated healthcare IoT datasets to monitor how well they detect actual threats and how they handle diverse, real-world data.
   - Assess how accurately the ML and DL models classify normal and anomalous behaviorby using evaluating metrics such as accuracy, precision, recall, F1-score, and AUC-ROC scores.
   - Utilize confusion matrices to understand the rate of false positives and false negatives, which is critical in intrusion detection systems that are liable to make false predictions .
   - Compare the performance of the models with previously published results in similar studies to gauge improvement or effectiveness.
2. Usability and Functionality Evaluation of the Web Application:
   - Conduct interviews with end users to assess the ease of use, intuitiveness, and overall user experience of the web application.
   - Verify that all features of the web application, alerting, logging of anomalies, and mitigation guidance, work as intended.
   - Check the integration of the ML/DL models and ensure that the application accurately reflects the predictions made by these models.

3.2 How does this project relate to your MSc Programme and your degree title outcomes?

This project closely aligns with my MSc in Internet of Things (IoT) with Cyber Security by addressing the crucial intersection of these fields. Focusing on the healthcare sector, which represents a vulnerable and critical area within IoT, I have developed an Intrusion Detection System (IDS) tailored specifically for IoT devices used in medical settings. This involves handling and analyzing large volumes of IoT data in real-time, which is central to IoT applications. The project also emphasizes cybersecurity by employing Machine Learning (ML) and Deep Learning (DL) algorithms to detect and mitigate security threats, showcasing a practical application of advanced cybersecurity techniques. By integrating these eleents, the project not only

demonstrates my ability to apply interdisciplinary knowledge but also prepares me for complex challenges in securing IoT networks, highlighting my capability to develop solutions at the intersection of IoT technology and cybersecurity

3.3 What are the risks in this project and how are you going to manage them?

The Intrusion Detection System (IDS) may face challenges related to system performance and model accuracy. Performance issues, such as delays or inefficiencies, could arise during high data loads or peak usage, potentially impacting the system's responsiveness and reliability. Additionally, the IDS might produce false positives or false negatives, affecting its overall effectiveness and trustworthiness.

To mitigate performance risks, the system should be optimized for efficiency, with performance monitoring and scalability measures in place to handle varying data loads. Stress and load testing are essential to identify and resolve potential bottlenecks. For model accuracy, continuous evaluation with comprehensive test datasets and regular retraining with new data are crucial. Employing ensemble methods can further enhance accuracy and reduce the risk of inaccuracies, ensuring the IDS remains effective and reliable.

Section 4: References

4.1 Please provide references if you have used any.

Oliveira, F., Costa, D. G., Assis, F. and Silva, I., 2024. Internet of Intelligent Things: A convergence of embedded systems, edge computing and machine learning. Internet of Things, 101153.

Rath, K. C., Khang, A. and Roy, D., 2024. The Role of Internet of Things (IoT) Technology in Industry 4.0 Economy. Advanced IoT Technologies and Applications in the Industry 4.0 Digital Economy. CRC Press, 1-28.

Ahmed, S. F., Alam, M. S. B., Afrin, S., Rafa, S. J., Rafa, N. and Gandomi, A. H., 2024. Insights into

Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

Internet of Medical Things (IoMT): Data fusion, security issues and potential solutions. Information Fusion, 102, 102060.

Umamaheswaran, S., Mannar Mannan, J., Karthick Raghunath, K., Dharmarajlu, S. M. and Anuratha, M., 2024. Smart intrusion detection system with balanced data in IoMT infra. Journal of Intelligent & Fuzzy Systems, (Preprint), 1-17.

Attou, H., Mohy-eddine, M., Guezzaz, A., Benkirane, S., Azrour, M., Alabdultif, A. and Almusallam, N., 2023. Towards an intelligent intrusion detection system to detect malicious activities in cloud computing. Applied Sciences, 13 (17), 9588.

Kizza, J. M., 2024. System intrusion detection and prevention. Guide to computer network security. Springer, 295-323.

Abdulganiyu, O. H., Tchakoucht, T. A. and Saheed, Y. K., 2024. Towards an efficient model for network intrusion detection system (IDS): systematic literature review. Wireless Networks, 30 (1), 453-482.

Zhang, Y., Zhu, D., Wang, M., Li, J. and Zhang, J., 2024. A comparative study of cyber security intrusion detection in healthcare systems. International Journal of Critical Infrastructure Protection, 44, 100658.

Section 5: Academic Practice and Ethics

Please delete as appropriate.

5.1 Have you made yourself familiar with, and understand, the University guidance on referencing and plagiarism?	**Yes**
5.2 Do you acknowledge that this project proposal is your own work and that it does not contravene any academic offence as specified in the University's regulations?	**Yes**

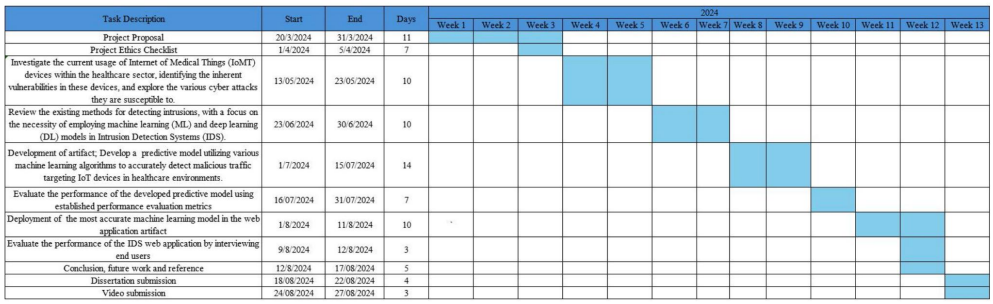
Bournemouth University

Department of Computing and Informatics

2023-24 Academic Year Individual Masters Project

Section 6: Proposed Plan

Gantt Chart details



Task Description	Start	End	Days	2024												
				Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13
Project Proposal	20/3/2024	31/3/2024	11													
Project Ethics Checklist	1/4/2024	5/4/2024	7													
Investigate the current usage of Internet of Medical Things (IoMT) devices within the healthcare sector, identifying the inherent vulnerabilities in these devices, and explore the various cyber attacks they are susceptible to.	13/05/2024	23/05/2024	10													
Review the existing methods for detecting intrusions, with a focus on the necessity of employing machine learning (ML) and deep learning (DL) models in Intrusion Detection Systems (IDS).	23/06/2024	30/6/2024	10													
Development of artifact; Develop a predictive model utilizing various machine learning algorithms to accurately detect malicious traffic targeting IoT devices in healthcare environments.	1/7/2024	15/07/2024	14													
Evaluate the performance of the developed predictive model using established performance evaluation metrics	16/07/2024	31/07/2024	7													
Deployment of the most accurate machine learning model in the web application artifact	1/8/2024	11/8/2024	10													
Evaluate the performance of the IDS web application by interviewing end users	9/8/2024	12/8/2024	3													
Conclusion, future work and reference	12/8/2024	17/08/2024	5													
Dissertation submission	18/08/2024	22/08/2024	4													
Video submission	24/08/2024	27/08/2024	3													

Task Description	Start	End	Days
Project Proposal	20/3/2024	31/3/2024	11
Project Ethics Checklist	1/4/2024	5/4/2024	7
Investigate the current usage of Internet of Medical Things (IoMT) devices within the healthcare sector, identifying the inherent vulnerabilities in these devices, and explore the various cyber attacks they are susceptible to.	13/05/2024	23/05/2024	10
Review the existing methods for detecting intrusions, with a focus on the necessity of employing machine learning (ML) and deep learning (DL) models in Intrusion Detection Systems (IDS).	23/06/2024	30/6/2024	10
Development of artifact; Develop a predictive model utilizing various machine learning algorithms to accurately detect malicious traffic targeting IoT devices in healthcare environments.	1/7/2024	15/07/2024	14
Evaluate the performance of the developed predictive model using established performance evaluation metrics	16/07/2024	31/07/2024	7
Deployment of the most accurate machine learning model in the web application artifact	1/8/2024	11/8/2024	10
Evaluate the performance of the IDS web application by interviewing end users of the application	9/8/2024	12/8/2024	3
Conclusion, future work and reference	12/8/2024	17/08/2024	5
Dissertation submission	18/08/2024	22/08/2024	4
Video submission	24/08/2024	27/08/2024	3

APPENDIX B - PROJECT CHECKLIST

Bournemouth University

Research Ethics Checklist

About Your Checklist	
Ethics ID	59862
Date Created	19/08/2024 16:33:04
Status	Approved
Date Approved	20/08/2024 11:56:51
Risk	Low

Researcher Details	
Name	Ti-Oluwani Opajobi
Faculty	Faculty of Science & Technology
Status	Postgraduate Taught (Masters, MA, MSc, MBA, LLM)
Course	MSc Internet of Things with Cyber Security
Have you received funding to support this research project?	No

Project Details	
Title	Machine Learning Dreiven Anomaly Intrusion Detection System for monitoring IoT Devices in Healthcare
Start Date of Project	06/05/2024
End Date of Project	23/08/2024
Proposed Start Date of Data Collection	10/05/2024
Supervisor	Abir Awad
Approver	Abir Awad

Summary - no more than 600 words (including detail on background methodology, sample, outcomes, etc.)

This project proposes a comprehensive comparative study of intrusion detection in healthcare systems by investigating and comparing the performance of four machine learning (ML) models—Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), and Extreme Gradient Boosting (XGB)—along with two deep learning (DL) algorithms—Artificial Neural Networks (ANN) and Long Short-Term Memory (LSTM). Additionally, we introduce an innovative ensemble algorithm that combines the strengths of Extreme Gradient Boosting, Random Forest, and Support Vector Machine (SVM).

This study will involve training and testing these models on two recent and relevant Internet of Medical Things (IoMT) datasets: the WUSTL EHMS 2020 dataset and the ECU-IoHT dataset. The WUSTL EHMS 2020 dataset, which includes 42 features, contains both network data and patient-related biometric features, making it crucial for designing an efficient and accurate Intrusion Detection System (IDS) for healthcare systems. In contrast, the ECU-IoHT dataset focuses on network data with 7 features, providing a different perspective on intrusion detection within IoMT environments. This dual-dataset approach aims to provide a more thorough evaluation of the models' capabilities and their applicability to real-world healthcare systems.

The goal is to provide a comprehensive analysis of results in the field of intrusion detection in healthcare systems, identifying the most effective approach for each dataset and determining the consistency of the performance of the ML and DL models across different datasets.

The study highlights the critical importance of developing a robust Intrusion Detection System (IDS) for Internet of Medical Things (IoMT) devices, given their increasing role in patient monitoring, diagnostics, and treatment. As IoMT devices become essential in modern healthcare, they are exposed to significant security threats, including hacking, data breaches, and other malicious activities that could compromise patient data and disrupt healthcare services. To address these vulnerabilities, the study proposes the use of machine learning techniques. By evaluating a range of machine learning and deep learning algorithms on two distinct datasets—one incorporating patient biometric features—the research aims to identify the most effective IDS solutions for real-world IoT healthcare environments. This evaluation seeks to ensure the algorithms' robustness and consistency across different datasets. The deployment of the most effective model within a web application framework represents a significant advancement, enhancing the security, efficiency, accessibility, and scalability of IDS solutions for IoMT devices. This implementation is designed to provide real-time, adaptable protection against evolving security threats, thereby safeguarding patient well-being and maintaining the integrity of the healthcare system.

Filter Question: Does your study involve Human Participants?

Participants	
Describe the number of participants and specify any inclusion/exclusion criteria to be used	
Three end users of my IDS would be interviewed to evaluate the application	
Do your participants include minors (under 16)?	No
Are your participants considered adults who are competent to give consent but considered vulnerable?	No
Is a Disclosure and Barring Service (DBS) check required for the research activity?	No

Recruitment	
Please provide details on intended recruitment methods, include copies of any advertisements.	
People that can interact with the application and are knowledgable about intrusion detection systems	
Do you need a Gatekeeper to access your participants?	No

Data Collection Activity	
Will the research involve questionnaire/online survey? If yes, don't forget to attach a copy of the questionnaire/survey or sample of questions.	No
Will the research involve interviews? If Yes, don’t forget to attach a copy of the interview questions or sample of questions	Yes
Please provide details e.g. where will the interviews take place. Will you be conducting the interviews or someone else?	
Interviews would be conducted in a serene location for best results, e.g office or school	
Will the research involve a focus group? If yes, don't forget to attach a copy of the focus group questions or sample of questions.	No
Will the research involve the collection of audio recordings?	Yes
Will your research involve the collection of photographic materials?	No

Will your research involve the collection of video materials/film?	No
Will any audio recordings (or non-anonymised transcript), photographs, video recordings or film be used in any outputs or otherwise made publicly available?	No
Will the study involve discussions of sensitive topics (e.g. sexual activity, drug use, criminal activity)?	No
Will any drugs, placebos or other substances (e.g. food substances, vitamins) be administered to the participants?	No
Will the study involve invasive, intrusive or potential harmful procedures of any kind?	No
Could your research induce psychological stress or anxiety, cause harm or have negative consequences for the participants or researchers (beyond the risks encountered in normal life)?	No
Will your research involve prolonged or repetitive testing?	No
What are the potential adverse consequences for research participants and how will you minimise them?	
No adverse consequence	

Consent	
Describe the process that you will be using to obtain valid consent for participation in the research activities. If consent is not to be obtained explain why.	
No personal or sensitive information is required for this project	
Do your participants include adults who lack/may lack capacity to give consent (at any point in the study)?	No
Will it be necessary for participants to take part in your study without their knowledge and consent?	No

Participant Withdrawal
At what point and how will it be possible for participants to exercise their rights to withdraw from the study?
At anypoint participants deems necessary
If a participant withdraws from the study, what will be done with their data?
No personal information is required fir this work. However if the participant withdraws, their data and feedback is disposed

Participant Compensation	
Will participants receive financial compensation (or course credits) for their participation?	No
Will financial or other inducements (other than reasonable expenses) be offered to participants?	No

Research Data	
Will identifiable personal information be collected, i.e. at an individualised level in a form that identifies or could enable identification of the participant?	No
Will research outputs include any identifiable personal information i.e. data at an individualised level in a form which identifies or could enable identification of the individual?	No

Storage, Access and Disposal of Research Data	
Where will your research data be stored and who will have access during and after the study has finished.	
Bournemouth Uninversity database where myself and my supervisor wouldaccess it	
Once your project completes, will your dataset be added to an appropriate research data repository such as BORDaR, BU's Data Repository?	Yes

Final Review	
Are there any other ethical considerations relating to your project which have not been covered above?	No

Risk Assessment	
Have you undertaken an appropriate Risk Assessment?	Yes

Filter Question: Does your study involve the use or re-use of data which will be obtained from a source other than directly from a Research Participant?

Additional Details	
Please describe the data, its source and how you are permitted to use it	Data required would be sourced from publicly available datasets for the purpose of this project, two publicly available healthcare datasets WUSTL EHMS 2020 Dataset and ECU-IoHT Dataset which are accessible via are accessible via https://www.cse.wustl.edu/~jain/ehms/index.html and https://ro.ecu.edu.au/datasets/48/

Attached documents
Consent form for project.pdf - attached on 19/08/2024 17:10:32
Interview Questions.pdf - attached on 19/08/2024 17:16:03

APPENDIX C: LIST OF LARGE FILES

Datasets Used
- WUSTL-EHMS 2020
- ECU-IoHT

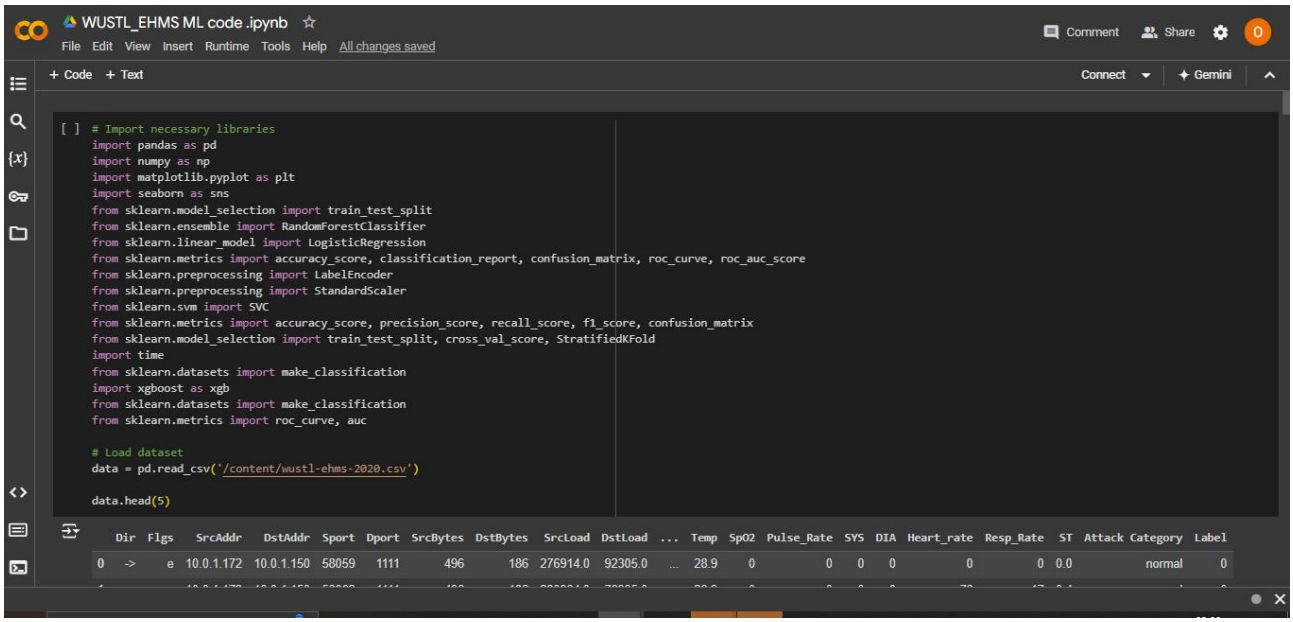
Python Files for deploying and evaluating ML and DL models
- ECU_IoHT ML code (.pynb file can be accessed on google colab or visual studio code)
- WUSTL_EHMS ML code (.pynb file can be accessed on google colab or visual studio code)

IDS Web application code folder
- Machine Learning Anomaly IDS Web app (can be accessed with visual studio code)

Preprocessed data for testing web application
- preprocessed data for web app testing

Audio recordings for interviews
- Interview with participant 1
- Interview with participant 2
- Interview with participant 3

APPENDIX D - CODE TO EVALUATE THE MODELS AGAINST WUST-EHMS 2020 DATASET



Import necessary libraries import pandas as pd import numpy as np import matplotlib.pyplot as plt import seaborn as sns from sklearn.model_selection import train_test_split from sklearn.ensemble import RandomForestClassifier from sklearn.linear_model import LogisticRegression from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, roc_curve, roc_auc_score from sklearn.preprocessing import LabelEncoder from sklearn.preprocessing import StandardScaler from sklearn.svm import SVC from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix from sklearn.model_selection import train_test_split, cross_val_score, StratifiedKFold import time from sklearn.datasets import make_classification import xgboost as xgb from sklearn.datasets import make_classification from sklearn.metrics import roc_curve, auc

Load dataset data = pd.read_csv('/content/wustl-ehms-2020.csv')

---

data.head(5)

	Dir	Flgs	SrcAddr	DstAddr	Sport	Dport	SrcBytes	DstBytes	SrcLoad	DstLoad	...	Temp	SpO2	Pulse_Rate	SYS	DIA	Heart_r
0	->	e	10.0.1.172	10.0.1.150	58059	1111	496	186	276914.0	92305.0	...	28.9	0	0	0	0	
1	->	e	10.0.1.172	10.0.1.150	58062	1111	496	186	230984.0	76995.0	...	28.9	0	0	0	0	
2	->	e	10.0.1.172	10.0.1.150	58065	1111	496	186	218470.0	72823.0	...	28.9	89	104	0	0	
3	->	e	10.0.1.172	10.0.1.150	58067	1111	496	186	203376.0	67792.0	...	28.9	89	104	0	0	
4	->	e	10.0.1.172	10.0.1.150	58069	1111	496	186	235723.0	78574.0	...	28.9	89	101	0	0	

data.shape

(16318, 45)

data.dropna(inplace=True)

duplicates = data.duplicated()
data.drop_duplicates(inplace= True)

data.shape

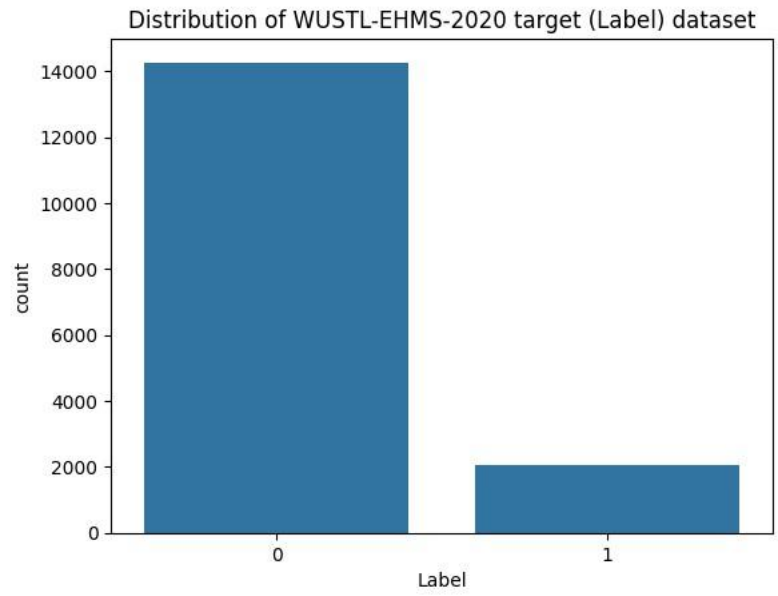
```
(16318, 45)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16318 entries,
0 to 16317 Data columns
(total 45 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
0    Dir             16318 non-null  object
1    Flgs            16318 non-null  object
2    SrcAddr         16318 non-null  object
3    DstAddr         16318 non-null  object
4    Sport           16318 non-null  object
5    Dport           16318 non-null  int64
6    SrcBytes        16318 non-null  int64
7    DstBytes        16318 non-null  int64
8    SrcLoad         16318 non-null  float64
9    DstLoad         16318 non-null  float64
10   SrcGap          16318 non-null  int64
11   DstGap          16318 non-null  int64
12   SIntPkt         16318 non-null  float64
13   DIntPkt         16318 non-null  float64
14   SIntPktAct      16318 non-null  float64
15   DIntPktAct      16318 non-null  int64
16   SrcJitter       16318 non-null  float64
17   DstJitter       16318 non-null  float64
18   sMaxPktSz       16318 non-null  int64
19   dMaxPktSz       16318 non-null  int64
20   sMinPktSz       16318 non-null  int64
21   dMinPktSz       16318 non-null  int64
22   Dur             16318 non-null  float64
23   Trans           16318 non-null  int64
24   TotPkts         16318 non-null  int64
25   TotBytes        16318 non-null  int64
26   Load            16318 non-null  float64
27   Loss            16318 non-null  int64
28   pLoss           16318 non-null  float64
29   pSrcLoss        16318 non-null  float64
30   pDstLoss        16318 non-null  float64
31   Rate            16318 non-null  float64
32   SrcMac          16318 non-null  object
33   DstMac          16318 non-null  object
34   Packet_num      16318 non-null  int64
35   Temp            16318 non-null  float64
36   SpO2            16318 non-null  int64
37   Pulse_Rate      16318 non-null  int64
38   SYS             16318 non-null  int64
39   DIA             16318 non-null  int64
40   Heart_rate      16318 non-null  int64
41   Resp_Rate       16318 non-null  int64
42   ST              16318 non-null  float64
43   Attack Category  16318 non-null  object  44  Label           16318 non-null  int64  dtypes: float64(15), int64(22),
object(8) memory usage: 5.6+ MB
```

```
# Visualize the distribution of the target variable
sns.countplot(x='Label', data=data)
plt.title('Distribution of WUSTL-EHMS-2020 target
(Label) dataset') plt.show()
```

```
data.Label.value_counts()
```



count

Label	
0	14272
1	2046

dtype: int64

```
# Initialize
LabelEncoder
encoder =
LabelEncoder()

# Perform label encoding data['Attack Category'] =
encoder.fit_transform(data['Attack Category']) data['Dir'] =
encoder.fit_transform(data['Dir']) data['DIntPkt'] =
encoder.fit_transform(data['DIntPkt'])
```

Feature Improtance

```
# Split features and labels
X = data.drop(['Label', 'Attack Category', 'Dir', 'Flgs', 'DstAddr', 'SrcAddr','SrcMac', 'DstMac',
'Sport',                'Dport', 'TotBytes', 'dMaxPktSz', 'sMaxPktSz', 'DstGap', 'SrcGap', 'Trans'],
axis=1) y = data['Label']

#use SMOTE for to mitigate data imbalnce
                            g
from imblearn.over_sampling import SMOTE
X_smote, y_smote = SMOTE().fit_resample(X, y)

#split dataset into testing and training data using 80/20 ratio from
sklearn.metrics import accuracy_score, confusion_matrix,
classification_report from sklearn.model_selection import
train_test_split
X_train, X_test, y_train, y_test = train_test_split(X_smote, y_smote, test_size=0.2, random_state = 42)
```

```
#use the standard scaler to scale
all entries from
sklearn.preprocessing import
StandardScaler scaling =
StandardScaler()
X_train = scaling.fit_transform(X_train)
X_test = scaling.transform(X_test)



# Train the RF model rf_model =
RandomForestClassifier(n_estimators=100,
random_state=42) rf_model.fit(X_train, y_train)
```

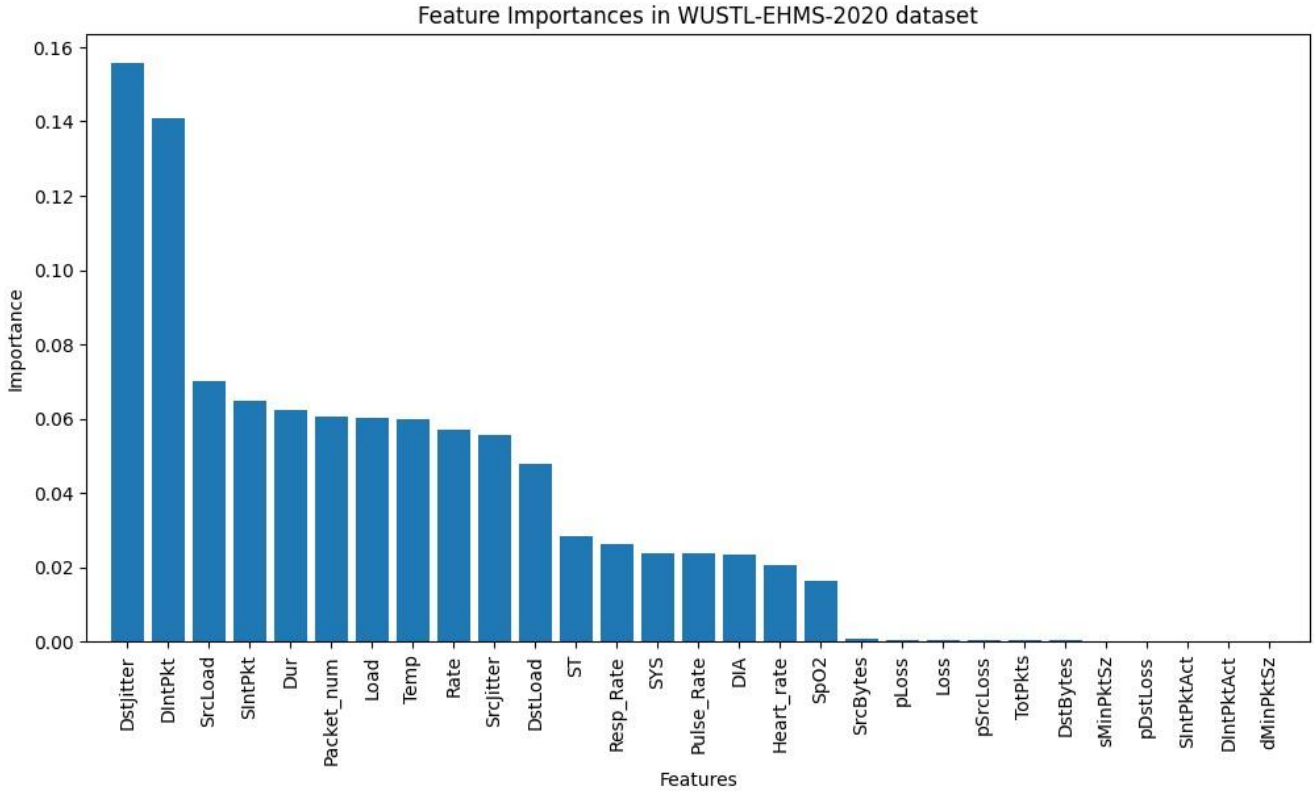
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Feature importance RFE with RFC

```
importances = rf_model.feature_importances_

# Sort feature importance and get
indices indices =
np.argsort(importances)[::-1]

# Print the feature ranking according to their importance for IDS
decisionmaking print("Feature ranking:") for f in range(X.shape[1]):
print(f"{f + 1}. Feature '{X.columns[indices[f]]}':
{importances[indices[f]]:.4f}")

#  Visualize the feature importances of the dataset
plt.figure(figsize=(12, 6)) plt.title("Feature
Importances in WUSTL-EHMS-2020 dataset")
plt.bar(range(X.shape[1]), importances[indices],
align="center") plt.xticks(range(X.shape[1]),
X.columns[indices], rotation=90) plt.xlim([-1,
X.shape[1]]) plt.ylabel('Importance')
plt.xlabel('Features') plt.show()
```

```
Feature ranking:
1. Feature
   'DstJitter':
   0.1557
2. Feature
   'DIntPkt':
   0.1408 3.
   Feature
   'SrcLoad':
   0.0700
4. Feature 'SIntPkt':
   0.0648
5. Feature 'Dur':
   0.0624
6. Feature
   'Packet_num':
   0.0604
7. Feature 'Load':
   0.0603 8.
   Feature 'Temp':
   0.0599
9.  Feature 'Rate':
    0.0569
```

10. Feature 'SrcJitter': 0.0556
11. Feature 'DstLoad': 0.0478
12. Feature 'ST': 0.0285
13. Feature 'Resp_Rate': 0.0263
14. Feature 'SYS': 0.0238 15. Feature 'Pulse_Rate': 0.0238 16. Feature 'DIA': 0.0234
17. Feature 'Heart_rate': 0.0205
18. Feature 'SpO2': 0.0163
19. Feature 'SrcBytes': 0.0008
20. Feature 'pLoss': 0.0005
21. Feature 'Loss': 0.0004
22. Feature 'pSrcLoss': 0.0003 23. Feature 'TotPkts': 0.0003
24. Feature 'DstBytes': 0.0003
25. Feature 'sMinPktSz': 0.0001
26. Feature 'pDstLoss': 0.0001
27. Feature 'SIntPktAct': 0.0001
28. Feature 'DIntPktAct': 0.0000
29. Feature 'dMinPktSz': 0.0000



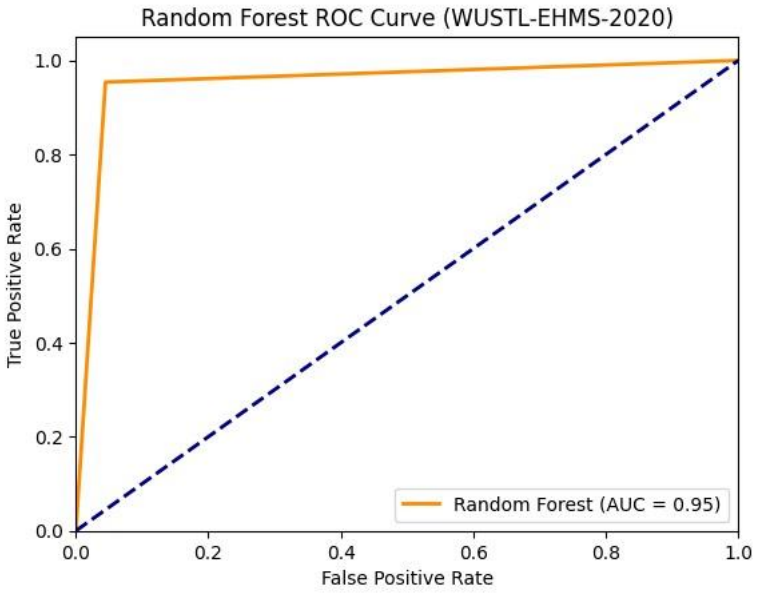

```
# Make predictions with the RF
classifier rf_pred =
rf_model.predict(X_test)


#declare the performance evaluation
metrics for RF accuracy_rf =
accuracy_score(y_test, rf_pred)
precision_rf =
precision_score(y_test, rf_pred)
recall_rf = recall_score(y_test,
rf_pred) f1_rf = f1_score(y_test,
rf_pred) cm_rf =
confusion_matrix(y_test, rf_pred)

# Classification Report for RF print("Classification
Report for random forest (WUSTL-EHMS-2020):")
print(classification_report(y_test, rf_pred))
```

Classification Report for random forest (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.95	0.96	0.95	2819
1	0.96	0.95	0.96	2890
accuracy			0.95	5709
macro avg	0.95	0.95	0.95	5709
weighted avg	0.95	0.95	0.95	5709

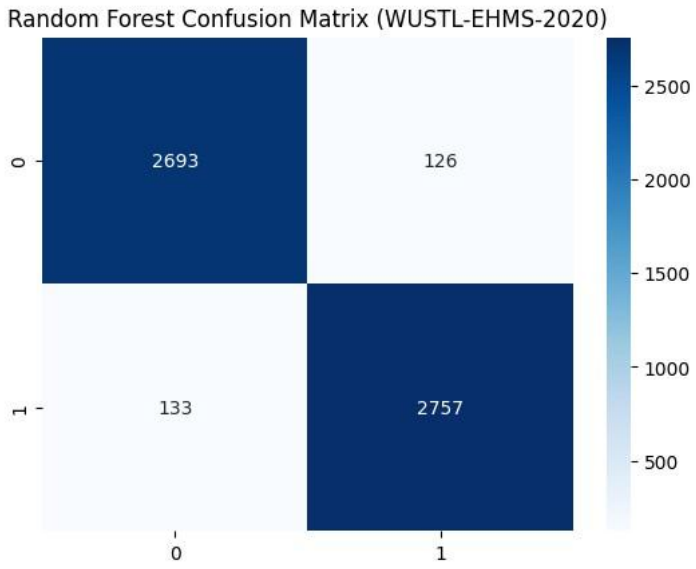
```
# print ROC AUC Score for RF roc_auc_rf =
roc_auc_score(y_test, rf_pred) print(f"\nROC AUC Score for
Random Forest (WUSTL-EHMS-2020): {roc_auc_rf}")

# Plot ROC Curve for RF fpr_rf, tpr_rf, _ = roc_curve(y_test, rf_pred) plt.figure()
plt.plot(fpr_rf, tpr_rf, color='darkorange', lw=2, label=f'Random Forest (AUC =
{roc_auc_rf:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0,
1.0]) plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive
Rate') plt.title('Random Forest ROC Curve (WUSTL-EHMS-2020)')
plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Random Forest (WUSTL-EHMS-2020): 0.9546412688982694
```




```
#print performace evaluation and confusion matrix for RF print(f'Random Forest Classifier (WUSTL-EHMS-2020):\n
Accuracy: {accuracy_rf}\n Precision: {precision_rf}\n Recall: {recall_rf}\n F1 Sco sns.heatmap(cm_rf, annot=True, fmt='d',
cmap='Blues') plt.title('Random Forest Confusion Matrix (WUSTL-EHMS-2020)') plt.show()
```

```
Random Forest Classifier (WUSTL-EHMS-2020):
 Accuracy: 0.954633035557891
 Precision: 0.9562955254942768
 Recall: 0.9539792387543252
 F1 Score: 0.9551359778278191
```



Extreme Gradient Boosting (XGB) Classi er

```
# Create an XGBoost classifier
xgb_model =
xgb.XGBClassifier(eval_metric='loglos
s')

# Fit the model to the training data
xgb_model.fit(X_train, y_train)
```

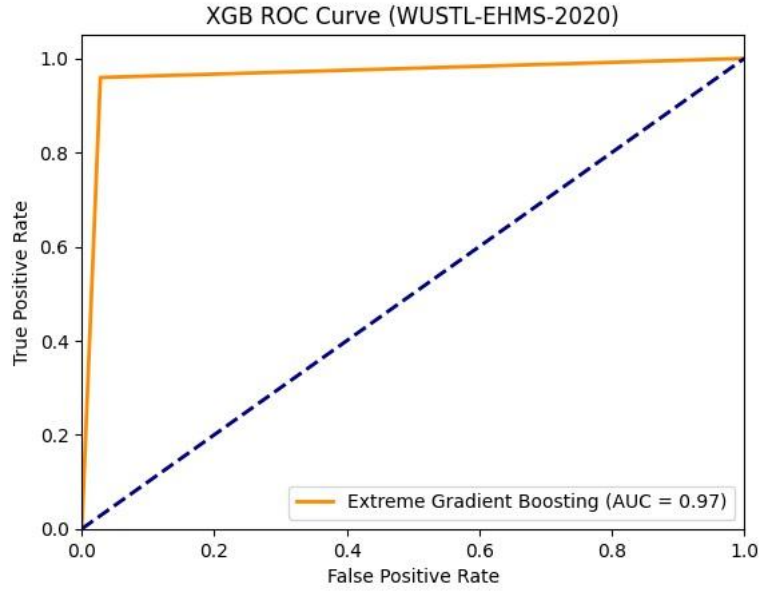
```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bynode=None,
colsample_bytree=None, device=None,
early_stopping_rounds=None,              enable_categorical=False,
eval_metric='logloss',              feature_types=None,
gamma=None, grow_policy=None,
importance_type=None, interaction_constraints=None,
learning_rate=None, max_bin=None, max_cat_threshold=None,
max_cat_to_onehot=None, max_delta_step=None,
max_depth=None,              max_leaves=None,
min_child_weight=None, missing=nan,
monotone_constraints=None, multi_strategy=None,
n_estimators=None,              n_jobs=None,
num_parallel_tree=None, random_state=None, ...)
```

```
# Make predictions on the
test set xgb_pred =
xgb_model.predict(X_test)


#declare the performance evaluation
metrics for XGB accuracy_xgb =
accuracy_score(y_test, xgb_pred)
precision_xgb =
precision_score(y_test, xgb_pred)
recall_xgb = recall_score(y_test,
xgb_pred) f1_xgb = f1_score(y_test,
xgb_pred) cm_xgb =
confusion_matrix(y_test, xgb_pred)

# Classification Report for XGB
print("Classification Report for XGB
(WUSTL-EHMS-2020):")
print(classification_report(y_test, xgb_pred))
```

```
Classification Report for XGB (WUSTL-
EHMS-2020):              precision    recall
f1-score   support

           0       0.96      0.97      0.97
2819           1       0.97      0.96      0.97
2890

    accuracy                           0.97      5709
macro avg       0.97      0.97      0.97      5709
weighted avg       0.97      0.97      0.97
5709 # print ROC AUC Score for XGB
roc_auc_xgb = roc_auc_score(y_test,
xgb_pred) print(f"\nROC AUC Score for
Extreme Gradient Boosting (WUSTL-EHMS-
2020): {roc_auc_xgb}")
```

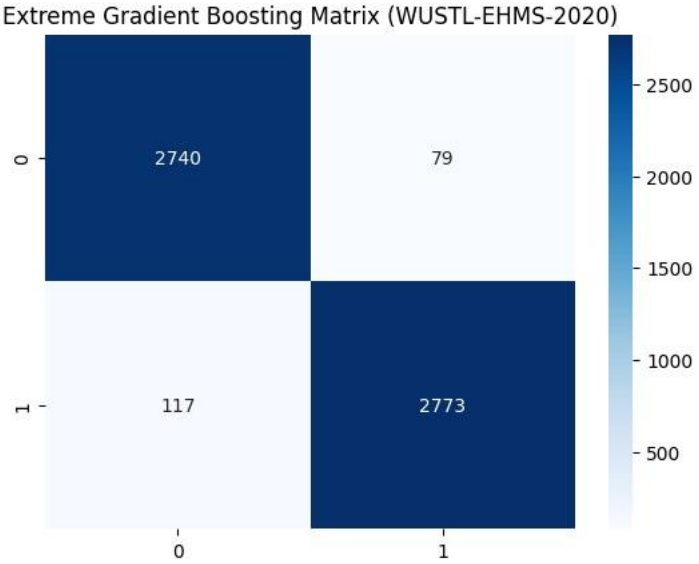
```
# Plot ROC Curve fpr_xgb, tpr_xgb, _ = roc_curve(y_test, xgb_pred) plt.figure() plt.plot(fpr_xgb, tpr_xgb, color='darkorange', lw=2, label=f'Extreme Gradient Boosting (AUC = {roc_auc_xgb:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0]) plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('XGB ROC Curve (WUSTL-EHMS-2020)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Extreme Gradient Boosting (WUSTL-EHMS-2020): 0.9657457244525839
```




```
#print confusion matrix and performance evaluation metrics for XGB print(f'Extreme Gradient Boosting (WUSTL-EHMS-2020):\n Accuracy: {accuracy_xgb}\n Precision: {precision_xgb}\n Recall: {recall_xgb}\n F1 sns.heatmap(cm_xgb, annot=True, fmt='d', cmap='Blues') plt.title('Extreme Gradient Boosting Matrix (WUSTL-EHMS-2020)') plt.show()
```

```
Extreme Gradient Boosting (WUSTL-EHMS-2020):
 Accuracy: 0.9656682431248905
 Precision: 0.9723001402524544
 Recall: 0.9595155709342561
 F1 Score: 0.9658655520724486
```



Logistic Regression (LR) Classifer

```
#train the LR model lr_model =
LogisticRegression(max_iter=1000,
random_state=42) lr_model.fit(X_train, y_train)
```

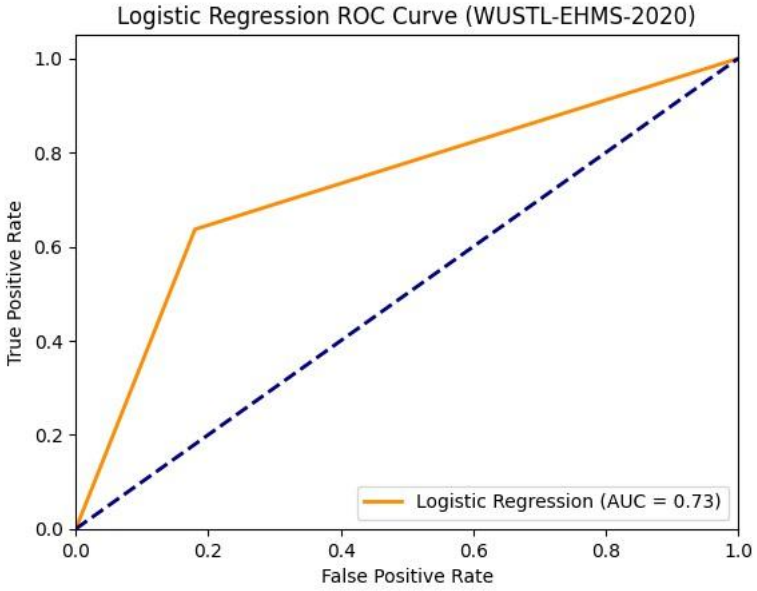
```
LogisticRegression
LogisticRegression(max_iter=1000, random_state=42)
```

```
#make predictions with the
LR model lr_pred =
lr_model.predict(X_test)

#declare performance evaluation
metrics for LR accuracy_lr =
accuracy_score(y_test, lr_pred)
precision_lr =
precision_score(y_test, lr_pred)
recall_lr = recall_score(y_test,
lr_pred) f1_lr = f1_score(y_test,
lr_pred) cm_lr =
confusion_matrix(y_test, lr_pred)

# Classification Report for LR
print("Classification Report for lr (WUSTL-
EHMS-2020):")
print(classification_report(y_test, lr_pred))
```

```
Classification Report for lr (WUSTL-EHMS-
2020):              precision    recall  f1-score
support

           0       0.69      0.82      0.75
2819
           1       0.78      0.64      0.70
2890

    accuracy                           0.73      5709
macro avg       0.74      0.73      0.73      5709
weighted avg       0.74      0.73      0.73
5709
```

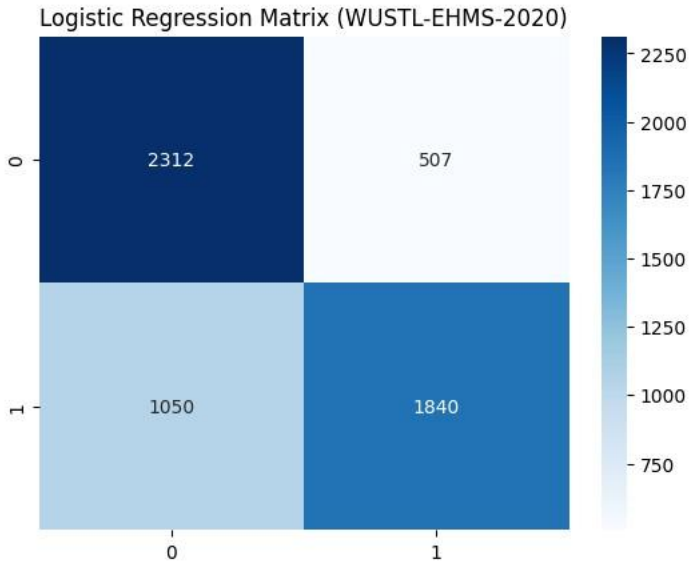
```
# print ROC AUC Score for LR roc_auc_lr = roc_auc_score(y_test,
lr_pred) print(f"\nROC AUC Score for Logistic Regression
(WUSTL-EHMS-2020): {roc_auc_lr}")

# Plot ROC Curve for LR fpr_lr, tpr_lr, _ = roc_curve(y_test, lr_pred) plt.figure()
plt.plot(fpr_lr, tpr_lr, color='darkorange', lw=2, label=f'Logistic Regression (AUC =
{roc_auc_lr:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate')
plt.title('Logistic Regression ROC Curve (WUSTL-EHMS-2020)') plt.legend(loc="lower
right") plt.show()
```

```
ROC AUC Score for Logistic Regression (WUSTL-EHMS-2020): 0.7284135948476169
```



#print performance evaluation metrcs and confusion matrix for LR print(f'Logistic Regression Classifier (WUSTL-EHMS-2020):\n Accuracy: {accuracy_lr}\n Precision: {precision_lr}\n Recall: {recall_lr}\n sns.heatmap(cm_lr, annot=True, fmt='d', cmap='Blues') plt.title('Logistic Regression Matrix (WUSTL-EHMS-2020)') plt.show()

Logistic Regression Classifier (WUSTL-EHMS-2020):
Accuracy: 0.7272727272727273
Precision: 0.783979548359608
Recall: 0.6366782006920415
F1 Score: 0.7026923811342372



Support Vector Machine Classi er (SVM)

train the Support Vector Machine svm_model = SVC(random_state=42) svm_model.fit(X_train, y_train)

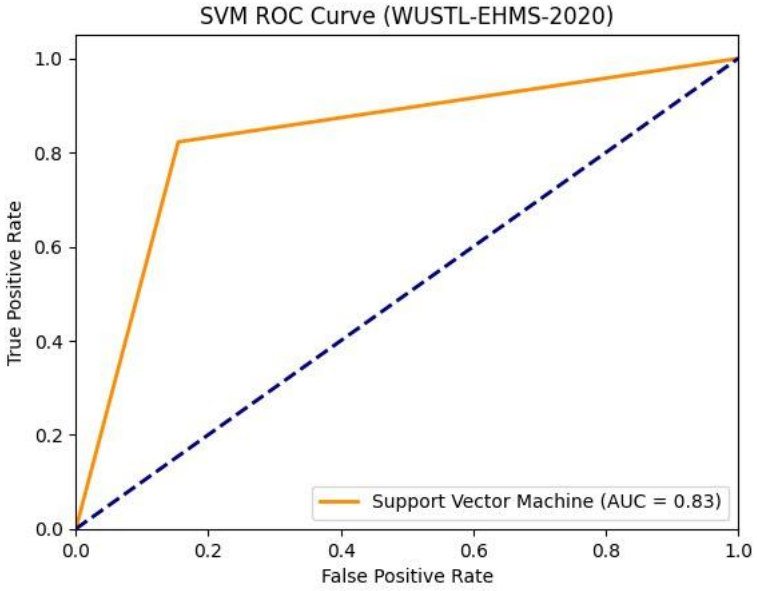
```
SVC
SVC(random_state=42)
```

```
#predict with svm
svm_pred =
svm_model.predict(X_t
est)


# declare performance evaluation
metrics for svm accuracy_svm =
accuracy_score(y_test, svm_pred)
precision_svm =
precision_score(y_test, svm_pred)
recall_svm = recall_score(y_test,
svm_pred) f1_svm =
f1_score(y_test, svm_pred)
cm_svm = confusion_matrix(y_test,
svm_pred)


# Classification Report for svm
print("Classification Report for svm
(WUSTL-EHMS-2020):")
print(classification_report(y_test,
svm_pred))
```

```
Classification Report for svm (WUSTL-
EHMS-2020):              precision    recall
f1-score   support

           0       0.82      0.85      0.83
2819           1       0.85      0.82      0.83
2890

    accuracy                           0.83      5709
macro avg       0.83      0.83      0.83      5709
weighted avg       0.83      0.83      0.83
5709 # print ROC AUC Score for svm
roc_auc_svm = roc_auc_score(y_test,
svm_pred) print(f"\nROC AUC Score for
SVM (WUSTL-EHMS-2020):
{roc_auc_svm}")
```

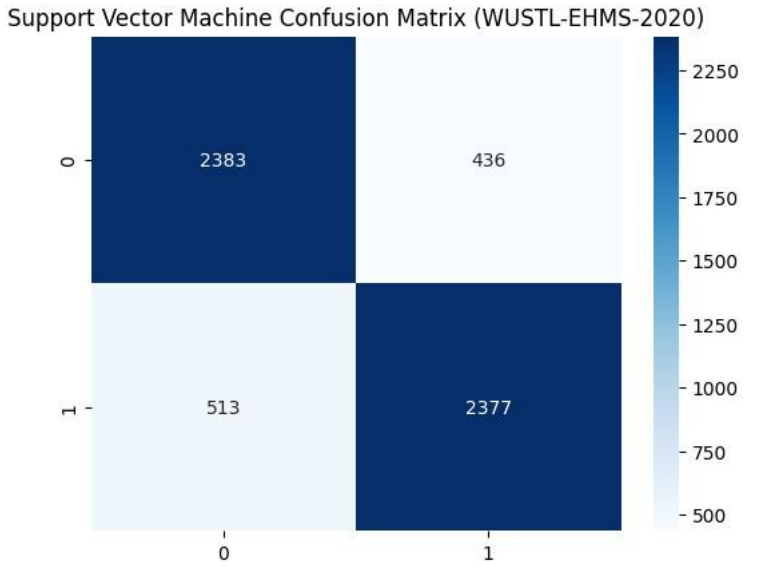
```
# Plot ROC Curve for svm fpr_svm, tpr_svm, _ = roc_curve(y_test, svm_pred) plt.figure()
plt.plot(fpr_svm, tpr_svm, color='darkorange', lw=2, label=f'Support Vector Machine (AUC =
{roc_auc_svm:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('SVM
ROC Curve (WUSTL-EHMS-2020)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for SVM (WUSTL-EHMS-2020): 0.8339132873690761
```



#print confusion matrix and performance evaluation metrics for svm print(f'Support Vector Machine (WUSTL-EHMS-2020):\n Accuracy: {accuracy_svm}\n Precision: {precision_svm}\n Recall: {recall_svm}\n F1 Sc sns.heatmap(cm_svm, annot=True, fmt='d', cmap='Blues') plt.title('Support Vector Machine Confusion Matrix (WUSTL-EHMS-2020)') plt.show()

```
Support Vector Machine (WUSTL-EHMS-2020):
 Accuracy: 0.8337712383955158
 Precision: 0.8450053323853537
 Recall: 0.8224913494809688
 F1 Score: 0.8335963527967736
```



Artifcial Neural Network (ANN) CLassi er

import tensorflow as tf from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense, Dropout from sklearn.metrics import classification_report, roc_auc_score, roc_curve, confusion_matrix, ConfusionMatrixDisplay

Split training data into train and validation sets
X_train_ann, X_val, y_train_ann, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

```
# Define the ANN model architecture ann_model = Sequential()
ann_model.add(Dense(units=64, activation='relu', input_shape=(X_train.shape[1],)))  #
First hidden layer ann_model.add(Dropout(0.2))  # Optional dropout for regularization
ann_model.add(Dense(units=32, activation='relu'))  # Second hidden layer
ann_model.add(Dense(units=1, activation='sigmoid'))  # Output layer for binary
classification
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an
`input_shape`/`input_dim` arg   super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
# Compile the ANN model ann_model.compile(optimizer='adam',
loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Train the ANN model with validation data
history_ann = ann_model.fit(X_train_ann, y_train_ann, epochs=10, batch_size=32, validation_data=(X_val, y_val))
```

```
Epoch 1/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4s 4ms/step - accuracy: 0.6906 - loss: 0.5337 -
val_accuracy: 0.7815 - val_loss: 0.4325
Epoch 2/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2s 4ms/step - accuracy: 0.7665 - loss: 0.4367 -
val_accuracy: 0.8007 - val_loss: 0.3935
Epoch 3/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2s 2ms/step - accuracy: 0.7830 - loss: 0.4136 -
val_accuracy: 0.8163 - val_loss: 0.3767
Epoch 4/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.7999 - loss: 0.3886 -
val_accuracy: 0.8270 - val_loss: 0.3591
Epoch 5/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8144 - loss: 0.3751 -
val_accuracy: 0.8272 - val_loss: 0.3570
Epoch 6/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8225 - loss: 0.3580 -
val_accuracy: 0.8443 - val_loss: 0.3372
Epoch 7/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8211 - loss: 0.3691 - val_accuracy:
0.8428 - val_loss: 0.3243
Epoch 8/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.8337 - loss: 0.3448 -
val_accuracy: 0.8522 - val_loss: 0.3166
Epoch 9/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3s 4ms/step - accuracy: 0.8413 - loss: 0.3380 -
val_accuracy: 0.8507 - val_loss: 0.3174
Epoch 10/10
571/571 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2s 3ms/step - accuracy: 0.8448 - loss: 0.3312 -
val_accuracy: 0.8542 - val_loss: 0.3104
```

```
# Make predictions on the test set
ann_pred = (ann_model.predict(X_test) > 0.5).astype("int32")
```

```
179/179 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step
```

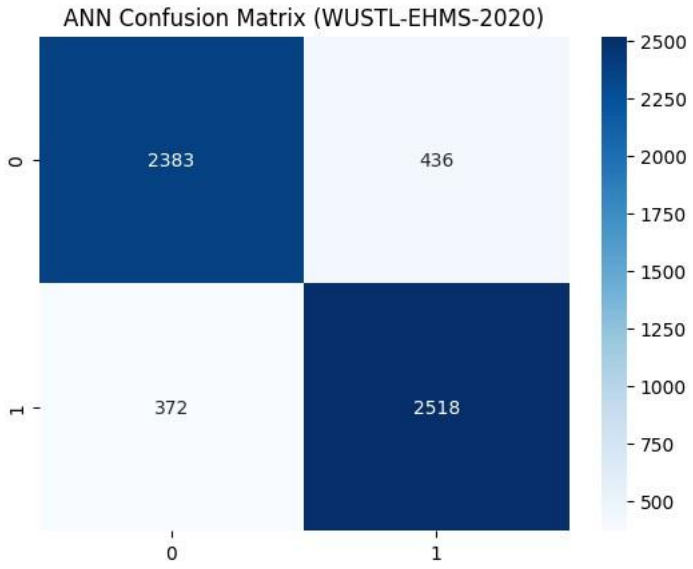
```
# Classification Report for ANN
print("Classification Report for ANN
(WUSTL-EHMS-2020):")
print(classification_report(y_test, ann_pred))
```

Classification Report for ANN (WUSTL-EHMS-2020):

	precision	recall	f1-score	support
0	0.86	0.85	0.86	2819
1	0.85	0.87	0.86	2890
accuracy			0.86	5709
macro avg	0.86	0.86	0.86	5709
weighted avg	0.86	0.86	0.86	5709

```
#declare performance evaluation metrics for ANN accuracy_ann = accuracy_score(y_test, ann_pred) precision_ann = precision_score(y_test, ann_pred) recall_ann = recall_score(y_test, ann_pred) f1_ann = f1_score(y_test, ann_pred) cm_ann = confusion_matrix(y_test, ann_pred) #priint confusion matrix and performance evaluation metrics for ANN print(f'Artificial Neural Network (WUSTL-EHMS-2020):\n Accuracy: {accuracy_ann}\n Precision: {precision_ann}\n Recall: {recall_ann}\n F1 sns.heatmap(cm_ann, annot=True, fmt='d', cmap='Blues') plt.title('ANN Confusion Matrix (WUSTL-EHMS-2020)') plt.show()
```

Artificial Neural Network (WUSTL-EHMS-2020):
Accuracy: 0.8584690839026099
Precision: 0.8524035206499662
Recall: 0.871280276816609
F1 Score: 0.8617385352498288

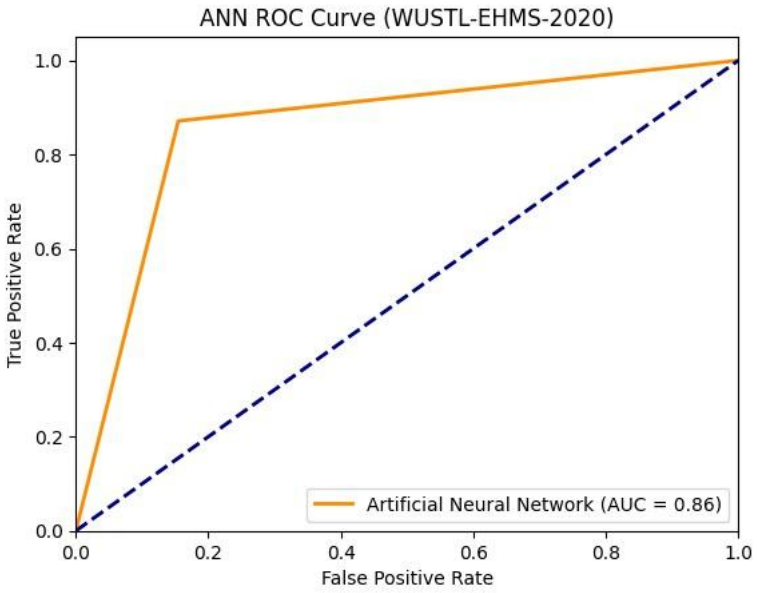



```
# print ROC AUC Score forANN roc_auc_ann = roc_auc_score(y_test, ann_pred) print(f"\nROC
```

AUC Score for ANN (WUSTL-EHMS-2020):
{roc_auc_ann}")

Plot ROC Curve for ANN fpr_ann, tpr_ann, _ = roc_curve(y_test, ann_pred) plt.figure()
plt.plot(fpr_ann, tpr_ann, color='darkorange', lw=2, label=f'Artificial Neural Network (AUC =
{roc_auc_ann:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('ANN
ROC Curve (WUSTL-EHMS-2020)') plt.legend(loc="lower right") plt.show()

ROC AUC Score for ANN (WUSTL-EHMS-2020): 0.8583077510368962



Long Short Term Memory (LSTM) Classifer
import pandas as pd import numpy as np from sklearn.model_selection
import train_test_split from sklearn.preprocessing import MinMaxScaler from
sklearn.metrics import classification_report, confusion_matrix,
ConfusionMatrixDisplay import matplotlib.pyplot as plt import tensorflow as tf
from tensorflow.keras.models import Sequential from tensorflow.keras.layers
import LSTM, Dense, Dropout

X_train_reshaped = np.array(X_train).reshape(X_train.shape[0], 1, X_train.shape[1])
X_val_reshaped = np.array(X_val).reshape(X_val.shape[0], 1, X_val.shape[1])
X_test_reshaped = np.array(X_test).reshape(X_test.shape[0], 1, X_test.shape[1])

# Define the LSTM model architecture lstm_model = Sequential() lstm_model.add(LSTM(units=50,
return_sequences=True, input_shape=(X_train_reshaped.shape[1], X_train_reshaped.shape[2]))) # Use the resha
lstm_model.add(Dropout(0.2)) lstm_model.add(LSTM(units=50)) lstm_model.add(Dropout(0.2))
lstm_model.add(Dense(units=1, activation='sigmoid')) # Binary classification

# Compile the LSTM model lstm_model.compile(optimizer='adam',
loss='binary_crossentropy', metrics=['accuracy'])

# Train the LSTM model
history_lstm = lstm_model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32, validation_data=(X_val_reshaped,
y_val))

Epoch 1/10
714/714 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **9s** 5ms/step - accuracy: 0.7076 - loss: 0.5456 -
val_accuracy: 0.7688 - val_loss: 0.4182
Epoch 2/10

```
714/714 ━━━━━━━━━━━━━━━━━━━━ 3s 5ms/step - accuracy: 0.7632 - loss: 0.4211 - val_accuracy:
0.7959 - val_loss: 0.3871
Epoch 3/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 7s 7ms/step - accuracy: 0.7827 - loss: 0.3991 -
val_accuracy: 0.8018 - val_loss: 0.3780
Epoch 4/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 3s 5ms/step - accuracy: 0.7902 - loss: 0.3907 -
val_accuracy: 0.8134 - val_loss: 0.3661
Epoch 5/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 5s 5ms/step - accuracy: 0.7985 - loss: 0.3808 -
val_accuracy: 0.8229 - val_loss: 0.3576
Epoch 6/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 6s 6ms/step - accuracy: 0.8050 - loss: 0.3729 -
val_accuracy: 0.8268 - val_loss: 0.3480
Epoch 7/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 3s 5ms/step - accuracy: 0.8068 - loss: 0.3665 -
val_accuracy: 0.8419 - val_loss: 0.3346
Epoch 8/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 3s 4ms/step - accuracy: 0.8204 - loss: 0.3538 -
val_accuracy: 0.8356 - val_loss: 0.3313
Epoch 9/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 4s 5ms/step - accuracy: 0.8235 - loss: 0.3482 -
val_accuracy: 0.8478 - val_loss: 0.3178
Epoch 10/10
714/714 ━━━━━━━━━━━━━━━━━━━━ 5s 5ms/step - accuracy: 0.8257 - loss: 0.3443 -
val_accuracy: 0.8474 - val_loss: 0.3103
```

```
# Make predictions (use the reshaped test data)
lstm_pred = (lstm_model.predict(X_test_reshaped) > 0.5).astype("int32")
```

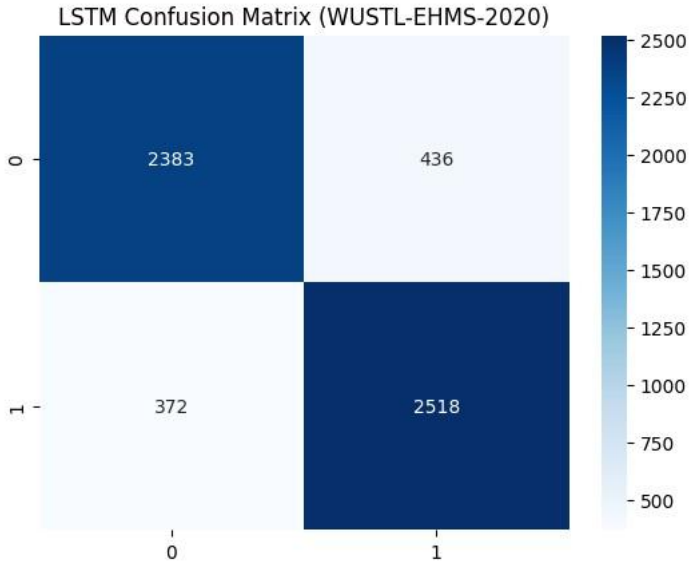
```
179/179 ━━━━━━━━━━━━━━━━━━━━ 1s 3ms/step
```

```
# Classification Report for LSTM
print("Classification Report for LSTM
(WUSTL-EHMS-2020):")
print(classification_report(y_test, lstm_pred))
```

```
Classification Report for LSTM (WUSTL-
EHMS-2020):              precision    recall  f1-
score   support

           0       0.86      0.82      0.84
2819
           1       0.84      0.87      0.85
2890

    accuracy                           0.85      5709
macro avg       0.85      0.85      0.85      5709
weighted avg       0.85      0.85      0.85
5709 #declareperformance evaluation
metrics for LSTM accuracy_lstm =
accuracy_score(y_test, lstm_pred)
precision_lstm = precision_score(y_test,
lstm_pred) recall_lstm = recall_score(y_test,
lstm_pred) f1_lstm = f1_score(y_test,
lstm_pred) cm_lstm =
confusion_matrix(y_test, lstm_pred)
```

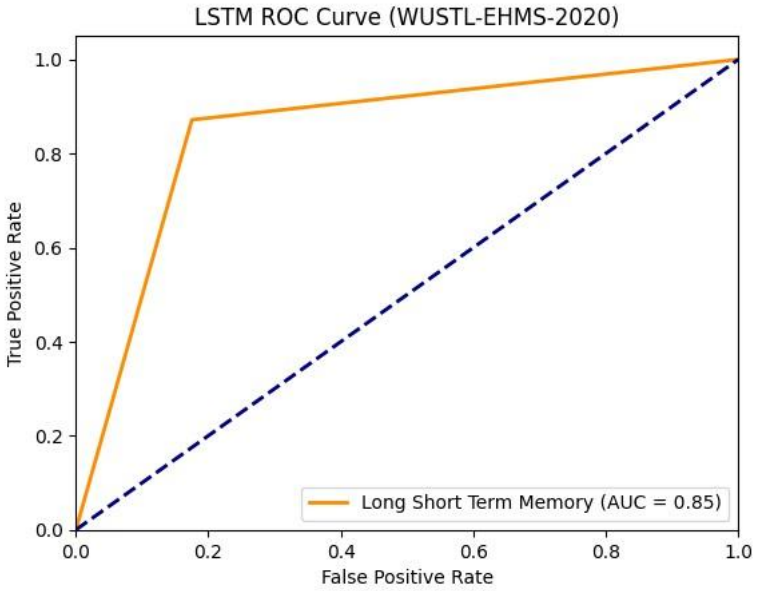
```
#print performance evaluation metrics and confusion matrix for LSTM print(f'LSTM:\n Accuracy (WUSTL-EHMS-2020):
{accuracy_lstm}\n Precision: {precision_lstm}\n Recall: {recall_lstm}\n F1 Score: {f1_lstm} sns.heatmap(cm_ann,
annot=True, fmt='d', cmap='Blues') plt.title('LSTM Confusion Matrix (WUSTL-EHMS-2020)') plt.show()
```

LSTM:
Accuracy (WUSTL-EHMS-2020): 0.8484848484848485
Precision: 0.8360438101559907
Recall: 0.8716262975778547
F1 Score: 0.8534643401660172




```
# print ROC AUC Score for LSTM roc_auc_lstm =
roc_auc_score(y_test, lstm_pred) print(f"\nROC AUC
Score for LSTM (WUSTL-EHMS-2020):
{roc_auc_lstm}")

# Plot ROC Curve for LSTM fpr_lstm, tpr_lstm, _ = roc_curve(y_test, lstm_pred) plt.figure()
plt.plot(fpr_lstm, tpr_lstm, color='darkorange', lw=2,label=f'Long Short Term Memory (AUC =
{roc_auc_lstm:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('LSTM
ROC Curve (WUSTL-EHMS-2020)') plt.legend(loc="lower right") plt.show()
```

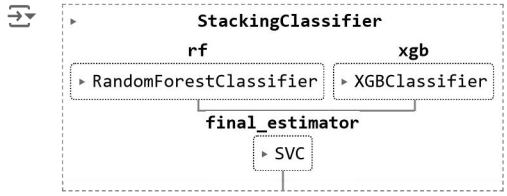
```
ROC AUC Score for LSTM (WUSTL-EHMS-2020): 0.8481934254827905
```



Ensemble Model

```
import numpy as np from sklearn.datasets import
make_classification from sklearn.model_selection import
```

```
train_test_split from sklearn.ensemble import
RandomForestClassifier, StackingClassifier from xgboost import
XGBClassifier from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score


# Define base learners for the ensemble
model base_learners = [
    ('rf',
RandomForestClassifier(n_estimators=100)),
('xgb',
xgb.XGBClassifier(eval_metric='logloss'))
]


# Define meta-learner for the ensemble
model meta_learner =
SVC(probability=True)

# Create the stacking classifier
stacking_clf =
StackingClassifier(     estimators=
base_learners,
final_estimator=meta_learner,
cv=5  # Cross-validation splitting
strategy
)


# Step 5: Train the stacking classifier for the ensemble model
stacking_clf.fit(X_train, y_train)
```



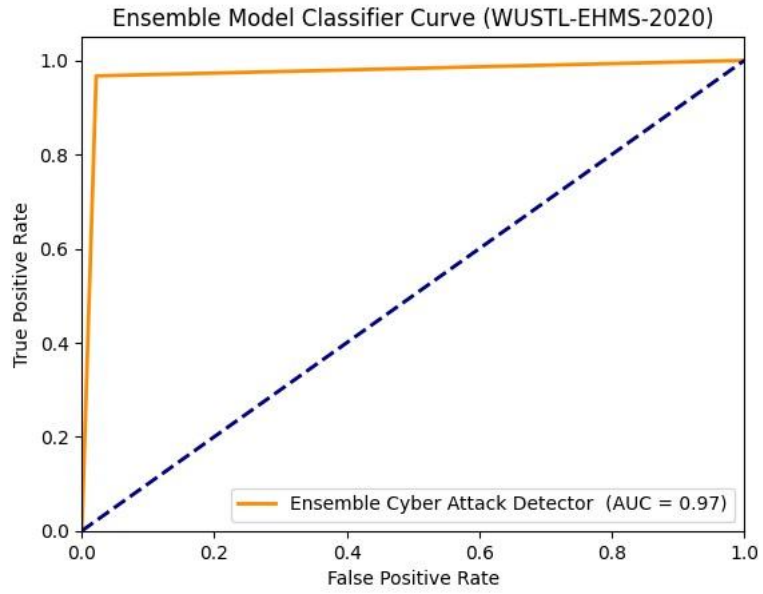

```
# Make predictions for the ensemble
model ensemble_pred =
stacking_clf.predict(X_test)


# Classification Report for the ensemble model
print("Classification Report (WUSTL-EHMS-2020):")
print(classification_report(y_test, ensemble_pred))
```

```
Classification Report (WUSTL-EHMS-
2020):              precision    recall  f1-
score   support
           0       0.97      0.98      0.97
2819           1       0.98      0.97      0.97
2890

    accuracy                           0.97      5709
macro avg       0.97      0.97      0.97      5709
weighted avg       0.97      0.97      0.97
5709
```

```
#declare performance evaluation variables for the
ensemble model accuracy_ensemble_model =
```

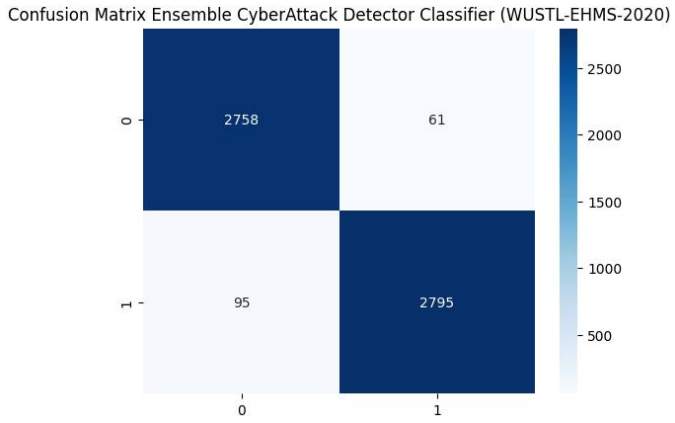
```
accuracy_score(y_test, ensemble_pred)
precision_ensemble_model = precision_score(y_test,
ensemble_pred) recall_ensemble_model =
recall_score(y_test, ensemble_pred)
f1_ensemble_model = f1_score(y_test,
ensemble_pred) cm_ensemble_model =
confusion_matrix(y_test, ensemble_pred)


# print ROC AUC Score forthe ensemble model roc_auc_ensemble =
roc_auc_score(y_test, ensemble_pred) print(f"\nROC AUC Score for
Ensemble Model Classifier (WUSTL-EHMS-2020): {roc_auc_ensemble}")

# Plot ROC Curve for the ensemble model fpr_ensemble, tpr_ensemble, _ = roc_curve(y_test, ensemble_pred)
plt.figure() plt.plot(fpr_ensemble, tpr_ensemble, color='darkorange', lw=2, label=f'Ensemble Cyber Attack Detector
(AUC = {roc_auc_ensemble:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0]) plt.ylim([0.0,
1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('Ensemble Model Classifier Curve
(WUSTL-EHMS-2020)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Ensemble Model Classifier (WUSTL-EHMS-2020): 0.9727445743232711
```

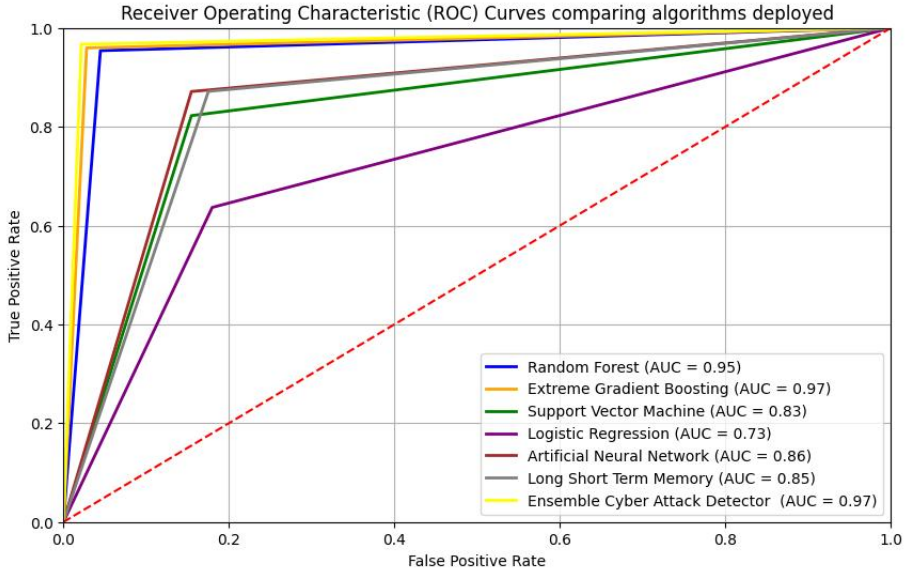



```
#print confusionmatrix and performance evaluation metrics print(f'Ensemble_model (WUSTL-EHMS-2020):\n Accuracy:
{accuracy_ensemble_model}\n Precision: {precision_ensemble_model}\n Recall: {recal
sns.heatmap(cm_ensemble_model, annot=True, fmt='d', cmap='Blues') plt.title('Confusion Matrix Ensemble
CyberAttack Detector Classifier (WUSTL-EHMS-2020)') plt.show()
```

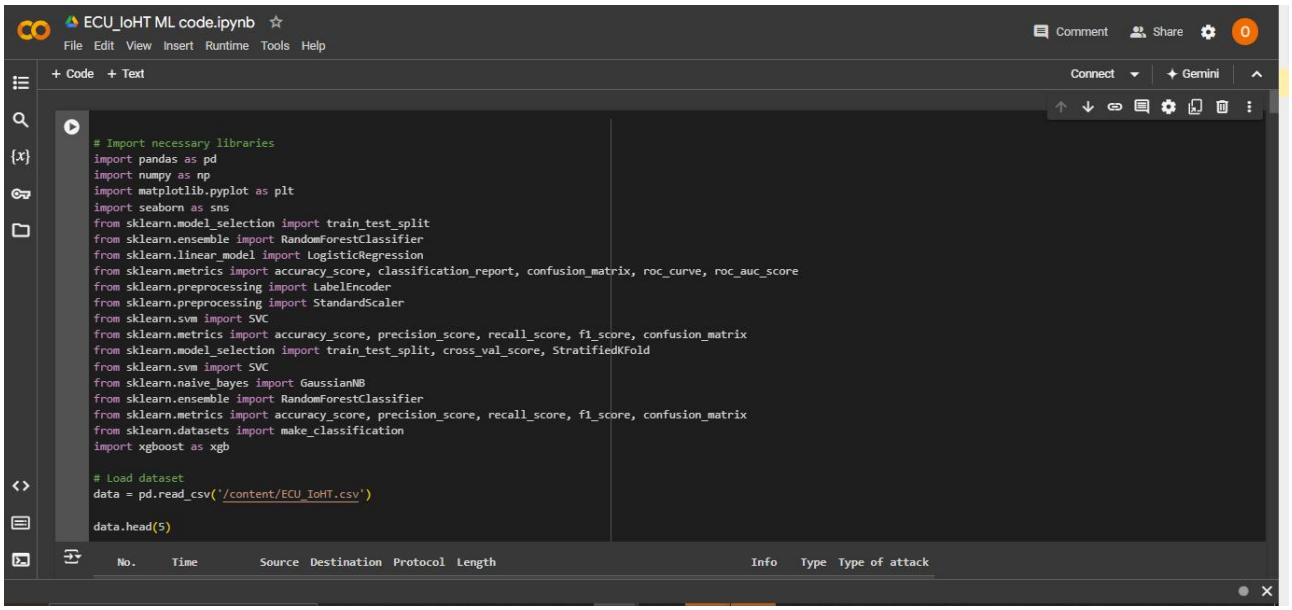
```
Ensemble_model (WUSTL-EHMS-2020):
 Accuracy: 0.9726747241198108
 Precision: 0.978641456582633
 Recall: 0.967128027681661
 F1 Score: 0.9728506787330318
```




```
#  Plot ROC curves for all of the models deployed
plt.figure(figsize=(10, 6))
plt.plot(fpr_rf, tpr_rf, color='blue', lw=2, label=f'Random Forest (AUC = {roc_auc_rf:.2f})')
plt.plot(fpr_xgb, tpr_xgb, color='orange', lw=2, label=f'Extreme Gradient Boosting (AUC = {roc_auc_xgb:.2f})')
plt.plot(fpr_svm, tpr_svm, color='green', lw=2, label=f'Support Vector Machine (AUC = {roc_auc_svm:.2f})')
plt.plot(fpr_lr, tpr_lr, color='purple', lw=2, label=f'Logistic Regression (AUC = {roc_auc_lr:.2f})')
plt.plot(fpr_ann, tpr_ann, color='brown', lw=2, label=f'Artificial Neural Network (AUC = {roc_auc_ann:.2f})')
plt.plot(fpr_lstm, tpr_lstm, color='grey', lw=2, label=f'Long Short Term Memory (AUC = {roc_auc_lstm:.2f})')
plt.plot(fpr_ensemble, tpr_ensemble, color='yellow', lw=2, label=f'Ensemble Cyber Attack Detector  (AUC =
{roc_auc_ensemble:.2f})')
plt.plot([0, 1], [0, 1], color='red', linestyle='--')  # Diagonal line
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curves comparing algorithms deployed')
plt.legend(loc='lower right')
plt.grid()
plt.show()
```



APPENDIX E - CODE TO EVALUATE THE MODELS AGAINST ECU-I0HT DATSET




```
# Import necessary libraries import pandas as pd import numpy as np import matplotlib.pyplot
as plt import seaborn as sns from sklearn.model_selection import train_test_split from
sklearn.ensemble import RandomForestClassifier from sklearn.linear_model import
LogisticRegression from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, roc_curve, roc_auc_score from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler from sklearn.svm import SVC from
sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
confusion_matrix from sklearn.model_selection import train_test_split, cross_val_score,
StratifiedKFold from sklearn.svm import SVC from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier from sklearn.metrics import
accuracy_score, precision_score, recall_score, f1_score, confusion_matrix from
sklearn.datasets import make_classification import xgboost as xgb

# Load dataset data =
pd.read_csv('/content/ECU_IoHT.
csv')
```

```
data.head(5)
```

	No.	Time	Source	Destination	Protocol	Length	Info	Type	Type of attack
0	1	0.000000	Alfa_97:cf:63	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.186	Attack	ARP Spoo ng
1	2	0.002956	6e:c7:ec:3c:f2:ba	Alfa_97:cf:63	ARP	42	192.168.43.1 is at 6e:c7:ec:3c:f2:ba	Attack	ARP Spoo ng
2	3	0.200725	Alfa_97:cf:63	Broadcast	ARP	42	Who has 192.168.43.1? Tell 192.168.43.186	Attack	ARP Spoo ng
3	4	0.202713	192.168.43.186	192.168.43.1	DNS	85	Standard query 0x0c44 PTR 1.43.168.192.in-addr...	Normal	No Attack
4	5	0.411565	6e:c7:ec:3c:f2:ba	Alfa_97:cf:63	ARP	42	192.168.43.1 is at 6e:c7:ec:3c:f2:ba	Attack	ARP Spoo ng

```
data.shape
```

```
(111207, 9)
```

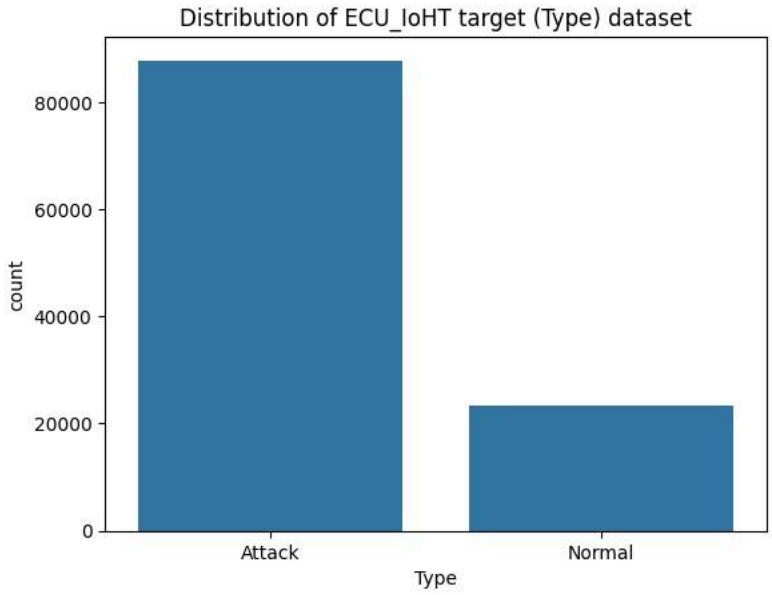
```
data.dropna(inplace=True)

data.shape
```

(111207, 9)

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111207 entries,
0 to 111206 Data columns
(total 9 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
0    No.             111207 non-null  int64
1    Time            111207 non-null  float64
2    Source          111207 non-null  object
3    Destination     111207 non-null  object
4    Protocol        111207 non-null  object
5    Length          111207 non-null  int64
6    Info            111207 non-null  object
7    Type            111207 non-null  object  8   Type of attack  111207 non-null  object dtypes: float64(1), int64(2),
object(6) memory usage: 7.6+ MB
```

```
# Visualize the distribution of the target
variable sns.countplot(x='Type', data=data)
plt.title('Distribution of ECU_IoHT target
(Type) dataset') plt.show()
data.Type.value_counts()
```



Type	count
Attack	87754
Normal	23453

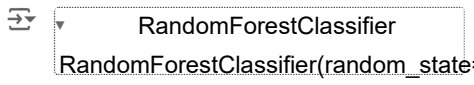
dtype: int64

```
# Initialize
LabelEncoder
```

```
encoder =
LabelEncoder()

# Perform label encoding data['Type'] =
encoder.fit_transform(data['Type'])
data['Protocol'] =
encoder.fit_transform(data['Protocol'])
data['Info'] =
encoder.fit_transform(data['Info'])


# Split features and labels
X = data.drop(['Type', 'Type of attack', 'Source', 'Destination'],
axis=1) y = data['Type']

# Scale features
scaler =
StandardScaler()
X_scaled =
scaler.fit_transform(X)


# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)


#use SMOTE to mitigate
data imbalance from
imblearn.over_sampling
import SMOTE
X_smote, y_smote = SMOTE().fit_resample(X, y)
```

Random forest classifier

```
# Train the rf model rf_model =
RandomForestClassifier(n_estimators=100,
random_state=42) rf_model.fit(X_train, y_train)
```



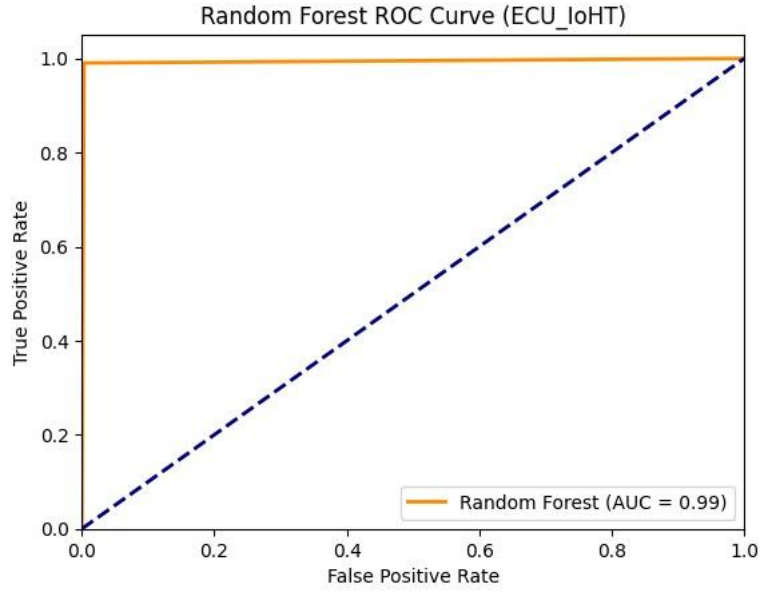

```
# Make predictions with the rf
model rf_pred =
rf_model.predict(X_test)

#decleare performance evaluation
variables for rf accuracy_rf =
accuracy_score(y_test, rf_pred)
precision_rf =
precision_score(y_test, rf_pred)
recall_rf = recall_score(y_test,
rf_pred) f1_rf = f1_score(y_test,
rf_pred) cm_rf =
confusion_matrix(y_test, rf_pred)
```

```
# Classification Report for rf
print("Classification Report for random forest
(ECU_IoHT):")
print(classification_report(y_test, rf_pred))
```

```
Classification Report for random forest
(ECU_IoHT):              precision    recall  f1-
score   support

           0       1.00      1.00      1.00
17572           1       0.99      0.99      0.99
4670

    accuracy                           1.00     22242
macro avg       0.99      0.99      0.99
22242 weighted avg       1.00      1.00
1.00     22242
```

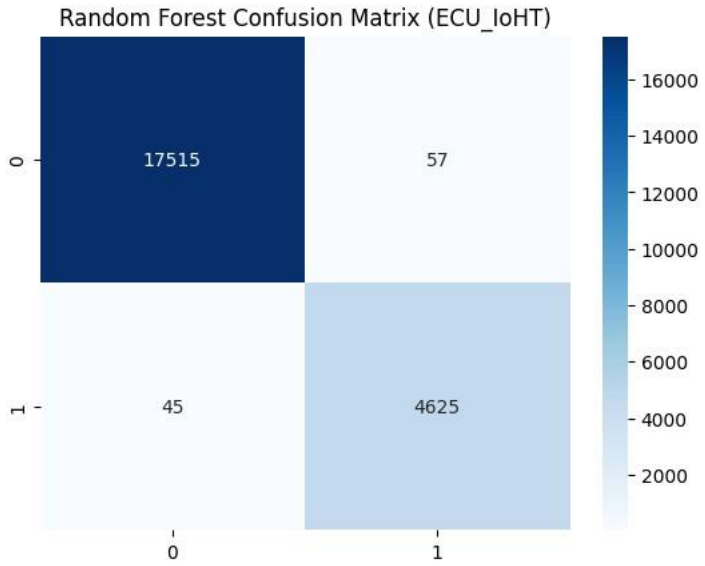
```
# print ROC AUC Score for rf roc_auc_rf =
roc_auc_score(y_test, rf_pred) print(f"\nROC AUC
Score for Random Forest (ECU_IoHT): {roc_auc_rf}")

# Plot ROC Curve for rf fpr_rf, tpr_rf, _ = roc_curve(y_test, rf_pred) plt.figure()
plt.plot(fpr_rf, tpr_rf, color='darkorange', lw=2, label=f'Random Forest (AUC =
{roc_auc_rf:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0,
1.0]) plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive
Rate') plt.title('Random Forest ROC Curve (ECU_IoHT)') plt.legend(loc="lower right")
plt.show()
```

```
ROC AUC Score for Random Forest (ECU_IoHT): 0.9935601143731194
```




```
#print confusion matrix and performace evaluation metrics for rf print(f'Random Forest Classifier (ECU_IoHT):\n
Accuracy: {accuracy_rf}\n Precision: {precision_rf}\n Recall: {recall_rf}\n F1 Score: {f1 sns.heatmap(cm_rf, annot=True,
fmt='d', cmap='Blues') plt.title('Random Forest Confusion Matrix (ECU_IoHT)') plt.show()
```

```
Random Forest Classifier (ECU_IoHT):
 Accuracy: 0.9954140814674939
 Precision: 0.9878257155061939
 Recall: 0.9903640256959315
 F1 Score: 0.9890932420872541
```



Extreme Gradient Boosting Classifer

```
# Create an XGBoost classifier
xgb_model =
xgb.XGBClassifier(eval_metric='loglos
s')

# Fit the model to the training data
xgb_model.fit(X_train, y_train)
```

```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
colsample_bylevel=None, colsample_bynode=None,
colsample_bytree=None, device=None,
early_stopping_rounds=None,              enable_categorical=False,
eval_metric='logloss',              feature_types=None,
gamma=None, grow_policy=None,
importance_type=None, interaction_constraints=None,
learning_rate=None, max_bin=None, max_cat_threshold=None,
max_cat_to_onehot=None, max_delta_step=None,
max_depth=None,              max_leaves=None,
min_child_weight=None, missing=nan,
monotone_constraints=None, multi_strategy=None,
n_estimators=None,              n_jobs=None,
num_parallel_tree=None, random_state=None, ...)
```

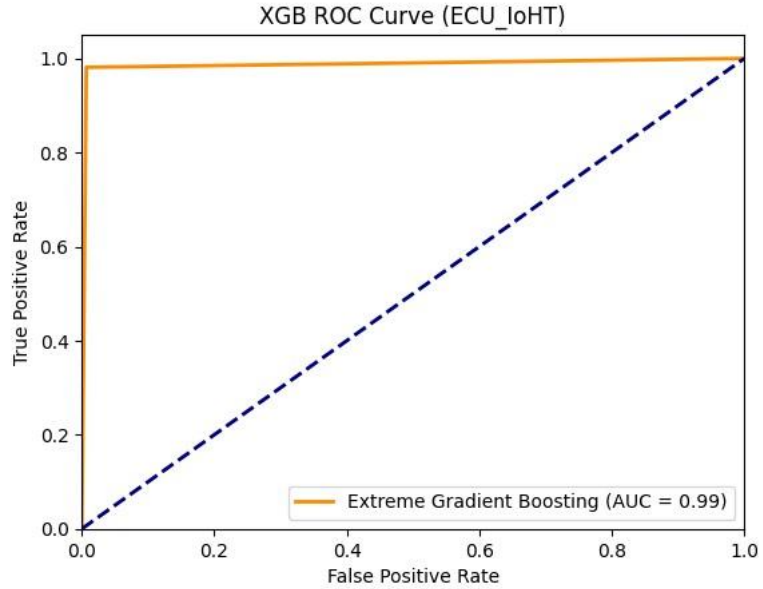
```
# Make predictions on the
test set xgb_pred =
xgb_model.predict(X_test)


#declare performance evaluation
metrics for xgb accuracy_xgb =
accuracy_score(y_test, xgb_pred)
precision_xgb =
precision_score(y_test, xgb_pred)
recall_xgb = recall_score(y_test,
xgb_pred) f1_xgb = f1_score(y_test,
```

```
xgb_pred) cm_xgb =
confusion_matrix(y_test, xgb_pred)

# Classification Report for xgb
print("Classification Report for XGB
(ECU_IoHT):")
print(classification_report(y_test,
xgb_pred))
```

```
Classification Report for XGB (ECU_IoHT):
      precision    recall  f1-score   support

           0       0.99      0.99      0.99
17572
           1       0.97      0.98      0.98
4670

    accuracy                           0.99     22242
macro avg       0.98      0.99      0.99
22242 weighted avg       0.99      0.99
0.99     22242
```

```
# print ROC AUC Score for xgb roc_auc_xgb =
roc_auc_score(y_test, xgb_pred) print(f"\nROC AUC Score for
Extreme Gradient Boosting (ECU_IoHT): {roc_auc_xgb}") # Plot
ROC Curve for xgb fpr_xgb, tpr_xgb, _ = roc_curve(y_test,
xgb_pred) plt.figure() plt.plot(fpr_xgb, tpr_xgb, color='darkorange',
lw=2, label=f'Extreme Gradient Boosting (AUC =
{roc_auc_xgb:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2,
linestyle='--') plt.xlim([0.0, 1.0]) plt.ylim([0.0, 1.05]) plt.xlabel('False
Positive Rate') plt.ylabel('True Positive Rate') plt.title('XGB ROC
Curve (ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Extreme Gradient Boosting (ECU_IoHT): 0.9871920897125123
```




```
#print confusionmatrix and performance evaluation metrics for xgb print(f'Extreme Gradient Boosting (ECU_IoHT):\n
Accuracy: {accuracy_xgb}\n Precision: {precision_xgb}\n Recall: {recall_xgb}\n F1 Score sns.heatmap(cm_xgb,
annot=True, fmt='d', cmap='Blues') plt.title('Extreme Gradient Boosting Matrix (ECU_IoHT)') plt.show()
```

Show hidden output

Logistic Regression Classifer

```
#train the lr lr_model =
LogisticRegression(max_iter=1000,
random_state=42) lr_model.fit(X_train, y_train)
```

```
LogisticRegression
LogisticRegression(max_iter=1000, random_state=42)
```

```
#male predictions with
the lr lr_pred =
lr_model.predict(X_test)
```

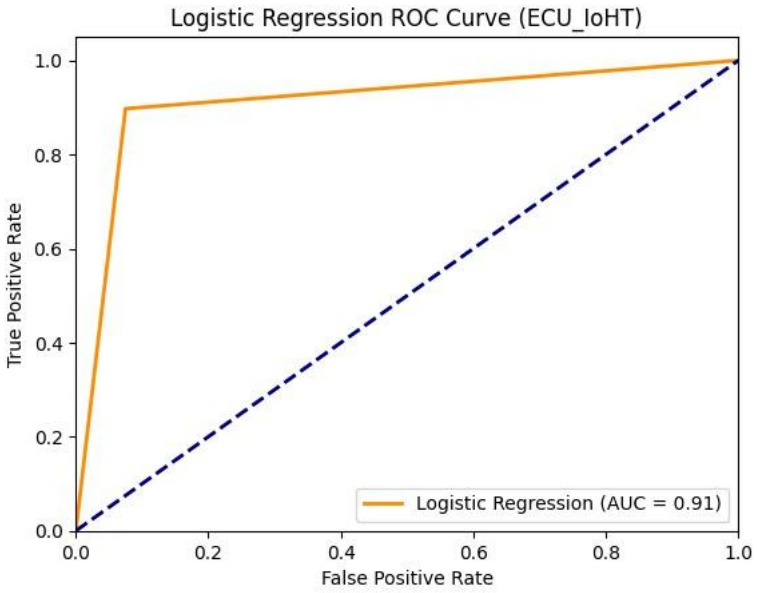
```
#declare performance evaluation
metrics for lr accuracy_lr =
accuracy_score(y_test, lr_pred)
precision_lr =
precision_score(y_test, lr_pred)
recall_lr = recall_score(y_test,
lr_pred) f1_lr = f1_score(y_test,
lr_pred) cm_lr =
confusion_matrix(y_test, lr_pred)
```

```
# Classification Report for LR
print("Classification Report for lr
(ECU_IoHT):")
print(classification_report(y_test,
lr_pred))
```

```
Classification Report for lr (ECU_IoHT):
      precision    recall  f1-score   support
                0       0.97      0.93      0.95
     17572           1       0.76      0.90      0.82
     4670

        accuracy                           0.92     22242
     macro avg       0.87      0.91      0.89
     22242 weighted avg       0.93      0.92
     0.92     22242
```

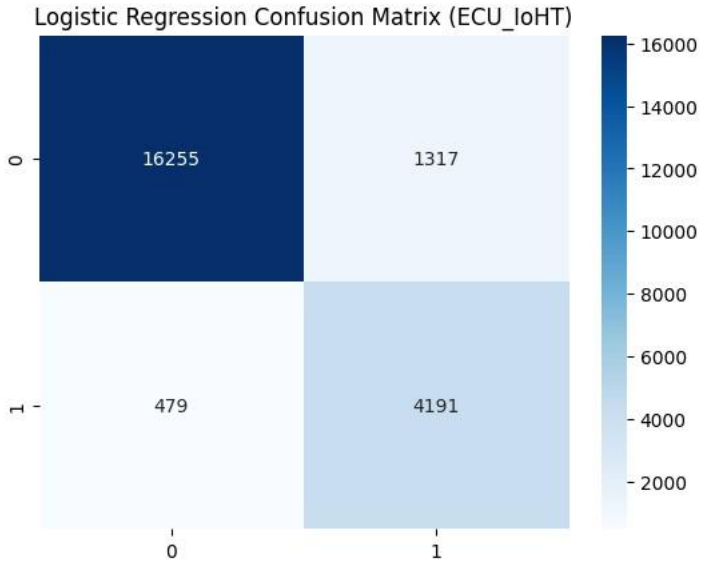
```
# print ROC AUC Score for LR roc_auc_lr =
roc_auc_score(y_test, lr_pred) print(f"\nROC AUC Score for
Logistic Regression (ECU_IoHT): {roc_auc_lr}")

# Plot ROC Curve for lr fpr_lr, tpr_lr, _ = roc_curve(y_test, lr_pred) plt.figure() plt.plot(fpr_lr,
tpr_lr, color='darkorange', lw=2, label=f'Logistic Regression (AUC = {roc_auc_lr:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0]) plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('Logistic Regression
ROC Curve (ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Logistic Regression (ECU IoHT): 0.9112408123494112
```



#print confusion matrix and performance evaluation for lr print(f'Logistic Regression Classifier (ECU_IoHT):\n Accuracy: {accuracy_lr}\n Precision: {precision_lr}\n Recall: {recall_lr}\n F1 Score sns.heatmap(cm_lr, annot=True, fmt='d', cmap='Blues') plt.title('Logistic Regression Confusion Matrix (ECU_IoHT)') plt.show()

```
Logistic Regression Classifier (ECU_IoHT):
 Accuracy: 0.9192518658394029
 Precision: 0.7608932461873639
 Recall: 0.8974304068522484
 F1 Score: 0.8235409707211633
```



Support Vector Machine Classifer

```
# train the vm
svm_model =
SVC(random_state
=42)
svm_model.fit(X_tr
ain, y_train)
```

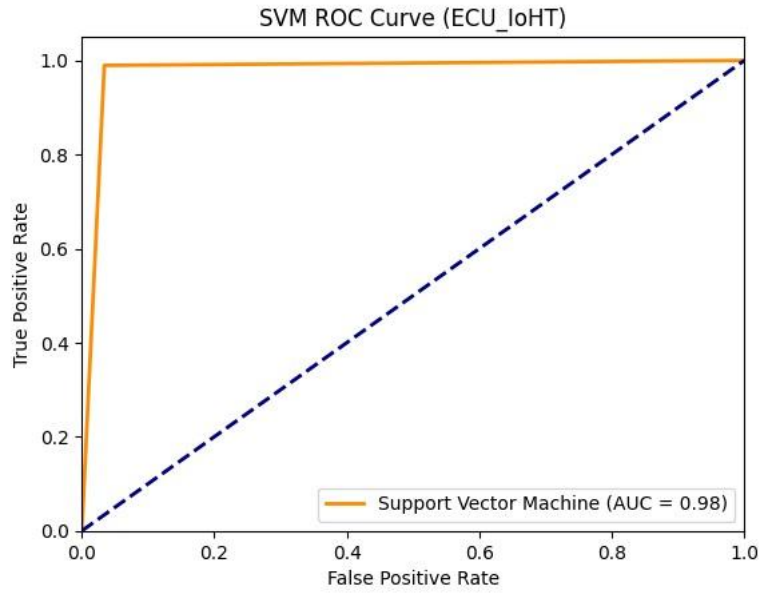
```
SVC
SVC(random_state=42)
```

```
#make prediction with
svm svm_pred =
svm_model.predict(X_t
est)

# declare performance evaluation
metrics for svm accuracy_svm =
accuracy_score(y_test, svm_pred)
precision_svm =
precision_score(y_test, svm_pred)
recall_svm = recall_score(y_test,
svm_pred) f1_svm =
f1_score(y_test, svm_pred)
cm_svm = confusion_matrix(y_test,
svm_pred)

# Classification Report for svm
print("Classification Report for svm
(ECU_IoHT):")
print(classification_report(y_test,
svm_pred))
```

```
Classification Report for svm
(ECU_IoHT):              precision    recall
f1-score   support

              0       1.00      0.97      0.98
17572           1       0.89      0.99      0.93
                          4670

    accuracy                           0.97
22242     macro avg       0.94      0.98
0.96      22242 weighted avg       0.97
0.97       0.97       22242 # print ROC
AUC Score for svm roc_auc_svm =
roc_auc_score(y_test,          svm_pred)
print(f"\nROC AUC Score for SVM
(ECU_IoHT): {roc_auc_svm}")
```

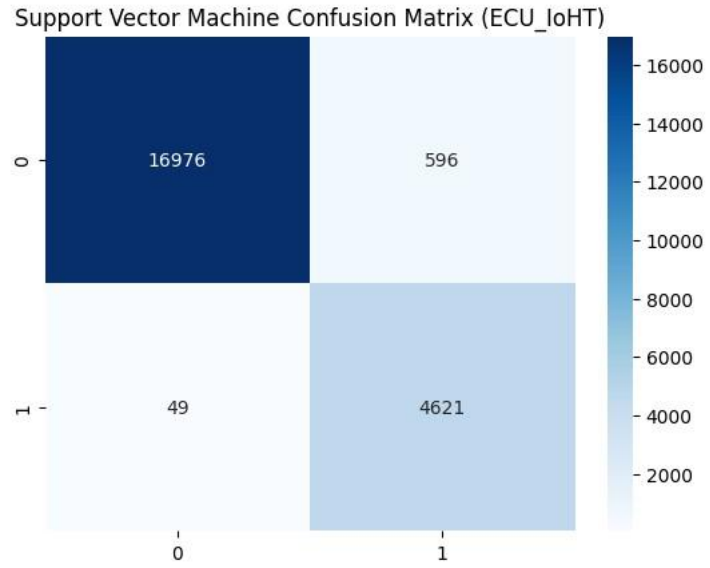
```
# Plot ROC Curve for svm fpr_svm, tpr_svm, _ = roc_curve(y_test, svm_pred) plt.figure()
plt.plot(fpr_svm, tpr_svm, color='darkorange', lw=2, label=f'Support Vector Machine (AUC =
{roc_auc_svm:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('SVM
ROC Curve (ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for SVM (ECU IoHT): 0.9777949492354736
```



#print performance evaluation metrics and confusion matrix for svm print(f'Support Vector Machine (ECU_IoHT):\n Accuracy: {accuracy_svm}\n Precision: {precision_svm}\n Recall: {recall_svm}\n F1 Score: {f sns.heatmap(cm_svm, annot=True, fmt='d', cmap='Blues') plt.title('Support Vector Machine Confusion Matrix (ECU_IoHT)') plt.show()

```
Support Vector Machine (ECU_IoHT):
 Accuracy: 0.9710008092797411
 Precision: 0.885758098524056
 Recall: 0.989507494646681
 F1 Score: 0.9347628198644685
```



Artifcial Neural Network Classifer

import tensorflow as tf from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense, Dropout from sklearn.metrics import classification_report, roc_auc_score, roc_curve, confusion_matrix, ConfusionMatrixDisplay

Split training data into train and validation sets
X_train_ann, X_val, y_train_ann, y_val = train_test_split(X_train, y_train, test_size=0.2, random_state=42)

```
# Define the ANN model architecture ann_model = Sequential()
ann_model.add(Dense(units=64, activation='relu', input_shape=(X_train.shape[1],)))  #
First hidden layer ann_model.add(Dropout(0.2))  # Optional dropout for regularization
ann_model.add(Dense(units=32, activation='relu'))  # Second hidden layer
ann_model.add(Dense(units=1, activation='sigmoid'))  # Output layer for binary
classification
```

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an
`input_shape`/`input_dim` arg   super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

```
# Compile the model ann_model.compile(optimizer='adam',
loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Train the ANN model with validation data
history_ann = ann_model.fit(X_train_ann, y_train_ann, epochs=10, batch_size=32, validation_data=(X_val, y_val))
```

```
Epoch 1/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8s 3ms/step - accuracy: 0.9210 - loss: 0.2050 -
val_accuracy: 0.9645 - val_loss: 0.1004
Epoch 2/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5s 2ms/step - accuracy: 0.9526 - loss: 0.1093 -
val_accuracy: 0.9659 - val_loss: 0.0863
Epoch 3/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6s 3ms/step - accuracy: 0.9592 - loss: 0.0949 -
val_accuracy: 0.9711 - val_loss: 0.0753
Epoch 4/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10s 3ms/step - accuracy: 0.9655 - loss: 0.0853 -
val_accuracy: 0.9765 - val_loss: 0.0698
Epoch 5/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7s 3ms/step - accuracy: 0.9700 - loss: 0.0785 -
val_accuracy: 0.9783 - val_loss: 0.0635
Epoch 6/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 5s 2ms/step - accuracy: 0.9708 - loss: 0.0737 -
val_accuracy: 0.9777 - val_loss: 0.0599
Epoch 7/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7s 3ms/step - accuracy: 0.9736 - loss: 0.0695 -
val_accuracy: 0.9790 - val_loss: 0.0571
Epoch 8/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 9s 2ms/step - accuracy: 0.9740 - loss: 0.0659 -
val_accuracy: 0.9782 - val_loss: 0.0564
Epoch 9/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8s 3ms/step - accuracy: 0.9740 - loss: 0.0656 -
val_accuracy: 0.9779 - val_loss: 0.0555
Epoch 10/10
2225/2225 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7s 2ms/step - accuracy: 0.9755 - loss: 0.0606 -
val_accuracy: 0.9803 - val_loss: 0.0528
```

```
# Make predictions on the test set ann_pred =
(ann_model.predict(X_test) >
0.5).astype("int32")
```

```
696/696 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1s 1ms/step
```

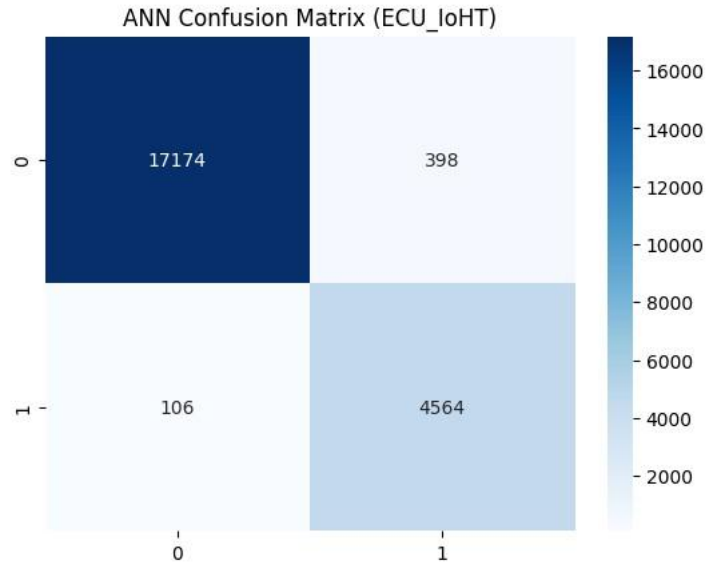
```
# Classification Report for ANN
print("Classification Report for ANN
(ECU_IoHT):")
```

```
print(classification_report(y_test,
ann_pred))
```

```
Classification Report for ANN
(ECU_IoHT):              precision    recall
f1-score   support

           0       0.99      0.98      0.99
17572           1       0.92      0.98      0.95
4670

    accuracy                           0.98     22242
macro avg       0.96      0.98      0.97
22242 weighted avg       0.98      0.98
0.98     22242 #declare performance
evaluation metrics for ann accuracy_ann =
accuracy_score(y_test, ann_pred)
precision_ann = precision_score(y_test,
ann_pred) recall_ann = recall_score(y_test,
ann_pred) f1_ann = f1_score(y_test,
ann_pred) cm_ann =
confusion_matrix(y_test, ann_pred)
```

```
#print confusion matrix and performance evaluation metrics for ann print(f'Artificial Neural Network (ECU_IoHT):\n
Accuracy: {accuracy_ann}\n Precision: {precision_ann}\n Recall: {recall_ann}\n F1 Score sns.heatmap(cm_ann,
annot=True, fmt='d', cmap='Blues') plt.title('ANN Confusion Matrix (ECU_IoHT)') plt.show()
```

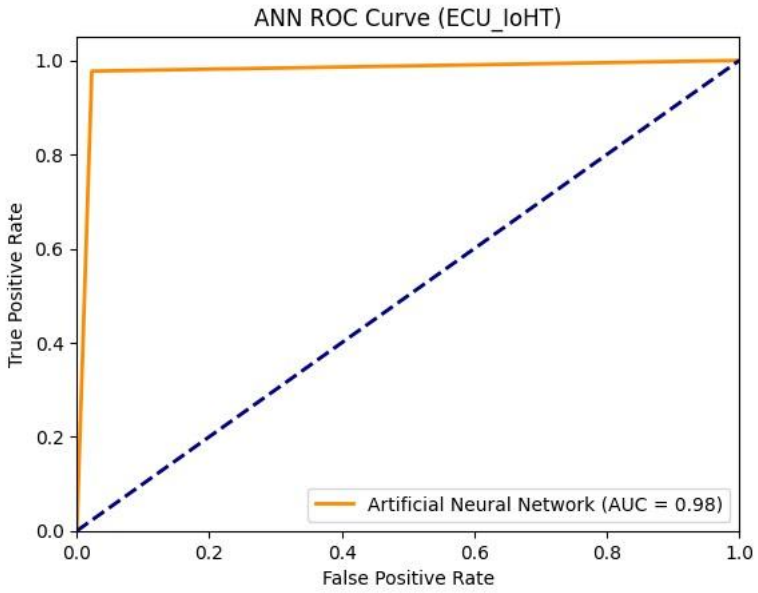
```
Artificial Neural Network (ECU_IoHT):
Accuracy: 0.9773401672511465
Precision: 0.9197904070939138
Recall: 0.9773019271948609
F1 Score: 0.9476744186046512
```




```
# print ROC AUC Score for ann roc_auc_ann
= roc_auc_score(y_test, ann_pred)
print(f"\nROC AUC Score for ANN
(ECU_IoHT): {roc_auc_ann}")
```

```
# Plot ROC Curve for ann fpr_ann, tpr_ann, _ = roc_curve(y_test, ann_pred) plt.figure()
plt.plot(fpr_ann, tpr_ann, color='darkorange', lw=2, label=f'Artificial Neural Network (AUC =
{roc_auc_ann:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('ANN
ROC Curve (ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for ANN (ECU IoHT): 0.9773261286327138
```



Long Short Term Memory Classifer

import pandas as pd import numpy as np from sklearn.model_selection import train_test_split from sklearn.preprocessing import MinMaxScaler from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay import matplotlib.pyplot as plt import tensorflow as tf from tensorflow.keras.models import Sequential from tensorflow.keras.layers import LSTM, Dense, Dropout

```
X_train_reshaped = np.array(X_train).reshape(X_train.shape[0], 1, X_train.shape[1])
X_val_reshaped = np.array(X_val).reshape(X_val.shape[0], 1, X_val.shape[1])
X_test_reshaped = np.array(X_test).reshape(X_test.shape[0], 1, X_test.shape[1])
```

Define the LSTM model architecture lstm_model = Sequential() lstm_model.add(LSTM(units=50, return_sequences=True, input_shape=(X_train_reshaped.shape[1], X_train_reshaped.shape[2]))) # Use the resh lstm_model.add(Dropout(0.2)) lstm_model.add(LSTM(units=50)) lstm_model.add(Dropout(0.2)) lstm_model.add(Dense(units=1, activation='sigmoid')) # Binary classification

```
/usr/local/lib/python3.10/dist-packages/keras/src/layers/rnn/rnn.py:204: UserWarning: Do not pass an `input_shape`/`input_dim` argum
  super().__init__(**kwargs)
```

Compile the model lstm_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the lstm model
history_lstm = lstm_model.fit(X_train_reshaped, y_train, epochs=10, batch_size=32, validation_data=(X_val_reshaped, y_val))

```
Epoch 1/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24s 7ms/step - accuracy: 0.9245 - loss: 0.2258 -
val_accuracy: 0.9517 - val_loss: 0.1090
Epoch 2/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 26s 9ms/step - accuracy: 0.9461 - loss: 0.1136 -
val_accuracy: 0.9588 - val_loss: 0.0963
Epoch 3/10
```

```
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19s 7ms/step - accuracy: 0.9532 - loss: 0.1020 -
val_accuracy: 0.9584 - val_loss: 0.0906
Epoch 4/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21s 7ms/step - accuracy: 0.9538 - loss: 0.0991 -
val_accuracy: 0.9596 - val_loss: 0.0886
Epoch 5/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 19s 7ms/step - accuracy: 0.9541 - loss: 0.0996 -
val_accuracy: 0.9604 - val_loss: 0.0863
Epoch 6/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 26s 9ms/step - accuracy: 0.9599 - loss: 0.0933 -
val_accuracy: 0.9665 - val_loss: 0.0838
Epoch 7/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 32s 6ms/step - accuracy: 0.9633 - loss: 0.0884 -
val_accuracy: 0.9693 - val_loss: 0.0755
Epoch 8/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24s 7ms/step - accuracy: 0.9655 - loss: 0.0829 -
val_accuracy: 0.9717 - val_loss: 0.0697
Epoch 9/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 18s 6ms/step - accuracy: 0.9656 - loss: 0.0787 -
val_accuracy: 0.9720 - val_loss: 0.0668
Epoch 10/10
2781/2781 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 20s 7ms/step - accuracy: 0.9662 - loss: 0.0767 -
val_accuracy: 0.9740 - val_loss: 0.0664
```

```
# Make predictions with the model
lstm_pred = (lstm_model.predict(X_test_reshaped) > 0.5).astype("int32")
```

```
696/696 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3s 4ms/step
```

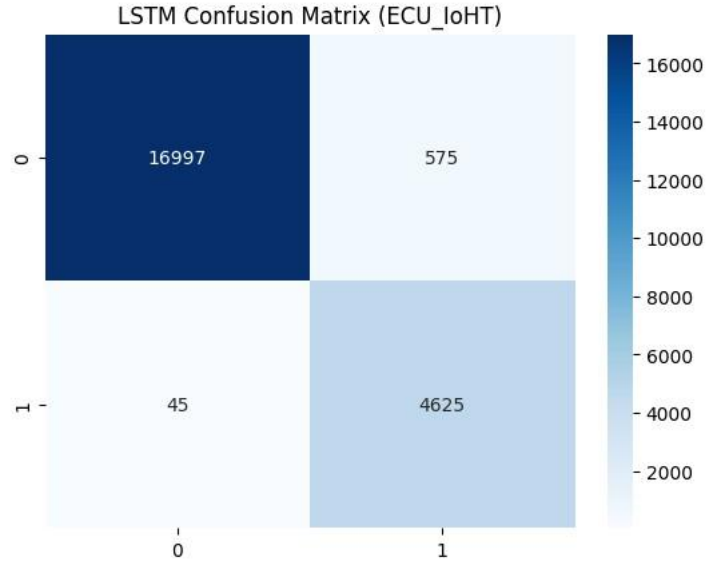
```
# Classification Report fpr lstm
print("Classification Report for LSTM
(ECU_IoHT):")
print(classification_report(y_test,
lstm_pred))
```

```
Classification Report for LSTM
(ECU_IoHT):              precision    recall
f1-score   support

           0       1.00      0.97      0.98
17572           1       0.89      0.99      0.94
4670

    accuracy                           0.97     22242
macro avg       0.94      0.98      0.96
22242 weighted avg       0.97      0.97
0.97     22242
```

```
#declare performance evaluation
metrics for lstm accuracy_lstm =
accuracy_score(y_test, lstm_pred)
precision_lstm =
precision_score(y_test, lstm_pred)
recall_lstm = recall_score(y_test,
lstm_pred) f1_lstm = f1_score(y_test,
lstm_pred) cm_lstm =
confusion_matrix(y_test, lstm_pred)
```

```
#print confusion matrix and performance evaluation metrics for lstm print(f'Long Short Term Memory (ECU_IoHT):\n Accuracy: {accuracy_lstm}\n Precision: {precision_lstm}\n Recall: {recall_lstm}\n F1 Score sns.heatmap(cm_lstm, annot=True, fmt='d', cmap='Blues') plt.title('LSTM Confusion Matrix (ECU_IoHT)') plt.show()
```

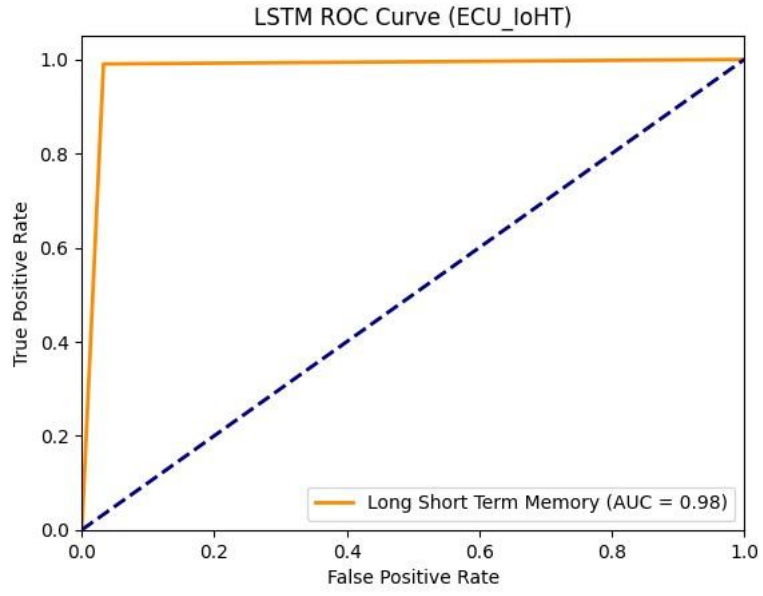
```
Long Short Term Memory (ECU_IoHT):
 Accuracy: 0.9721248089200611
 Precision: 0.8894230769230769
 Recall: 0.9903640256959315
 F1 Score: 0.9371833839918947
```




```
# print ROC AUC Score for lstm roc_auc_lstm
= roc_auc_score(y_test, lstm_pred)
print(f"\nROC AUC Score for LSTM
(ECU_IoHT): {roc_auc_lstm}")
```

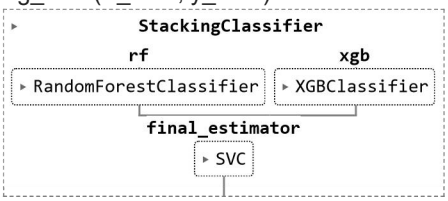
```
# Plot ROC Curve for lstm fpr_lstm, tpr_lstm, _ = roc_curve(y_test, lstm_pred) plt.figure()
plt.plot(fpr_lstm, tpr_lstm, color='darkorange', lw=2,label=f'Long Short Term Memory (AUC =
{roc_auc_lstm:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('LSTM
ROC Curve (ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for LSTM (ECU_IoHT): 0.9788207563034632
```



Ensemble Model Classifer

```
import numpy as np from sklearn.datasets import
make_classification from sklearn.model_selection import
train_test_split from sklearn.ensemble import
RandomForestClassifier, StackingClassifier from xgboost import
XGBClassifier from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score


# Define base learners for the ensemble
model base_learners = [
    ('rf',
RandomForestClassifier(n_estimators=100)),
('xgb',
xgb.XGBClassifier(eval_metric='logloss'))
]


# Define meta-learner for the ensemble
model meta_learner =
SVC(probability=True)

# Create the stacking classifier
stacking_clf =
StackingClassifier(     estimato
rs=base_learners,
final_estimator=meta_learner,
    cv=5  # Cross-validation splitting strategy
)


#  Train the stacking classifier
stacking_clf.fit(X_train, y_train)
```

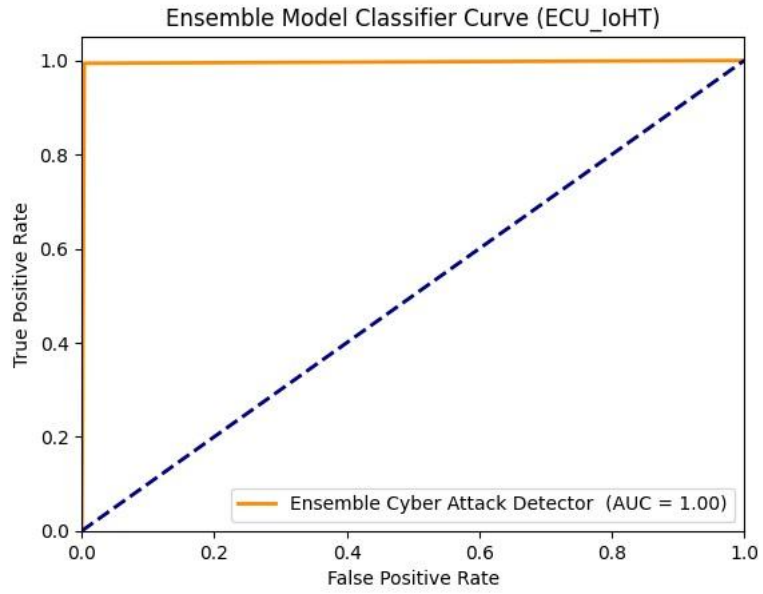



```
#  Make predictions with the ensemble
model ensemble_pred =
stacking_clf.predict(X_test)


# Classification Report for the ensemble model
print("Classification Report (ECU_IoHT):")
print(classification_report(y_test, ensemble_pred))
```

```
Classification Report (ECU_IoHT):
      precision    recall  f1-score   support

           0       1.00      1.00      1.00
     17572
           1       0.99      0.99      0.99
     4670

    accuracy                           1.00     22242
   macro avg       0.99      1.00      0.99
   22242 weighted avg       1.00      1.00
   1.00     22242
```

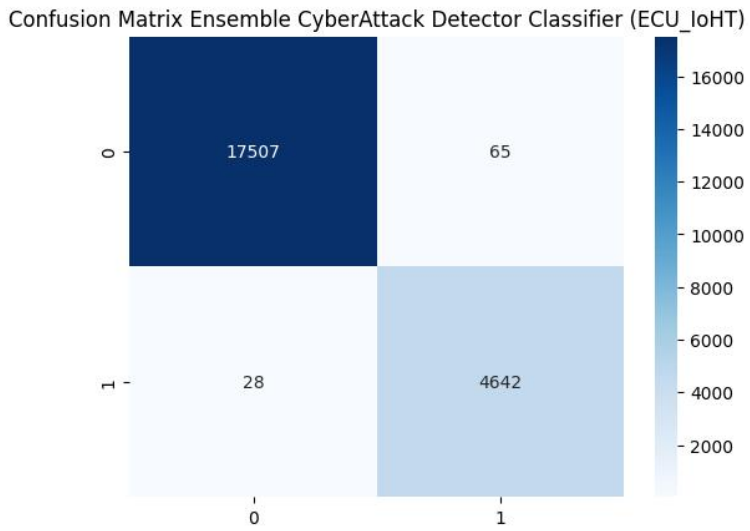
```
#declare performance evaluation metrics
accuracy_ensemble_model =
accuracy_score(y_test, ensemble_pred)
precision_ensemble_model =
precision_score(y_test, ensemble_pred)
recall_ensemble_model = recall_score(y_test,
ensemble_pred) f1_ensemble_model =
f1_score(y_test, ensemble_pred)
cm_ensemble_model = confusion_matrix(y_test,
ensemble_pred)


# print ROC AUC Score for the ensemble model roc_auc_ensemble =
roc_auc_score(y_test, ensemble_pred) print(f"\nROC AUC Score for
Ensemble Model Classifier (ECU_IoHT): {roc_auc_ensemble}")

# Plot ROC Curve for the ensemble model fpr_ensemble, tpr_ensemble, _ = roc_curve(y_test, ensemble_pred)
plt.figure() plt.plot(fpr_ensemble, tpr_ensemble, color='darkorange', lw=2, label=f'Ensemble Cyber Attack Detector
(AUC = {roc_auc_ensemble:.2f})') plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--') plt.xlim([0.0, 1.0]) plt.ylim([0.0,
1.05]) plt.xlabel('False Positive Rate') plt.ylabel('True Positive Rate') plt.title('Ensemble Model Classifier Curve
(ECU_IoHT)') plt.legend(loc="lower right") plt.show()
```

```
ROC AUC Score for Ensemble Model Classifier (ECU IoHT): 0.9951526079791141
```

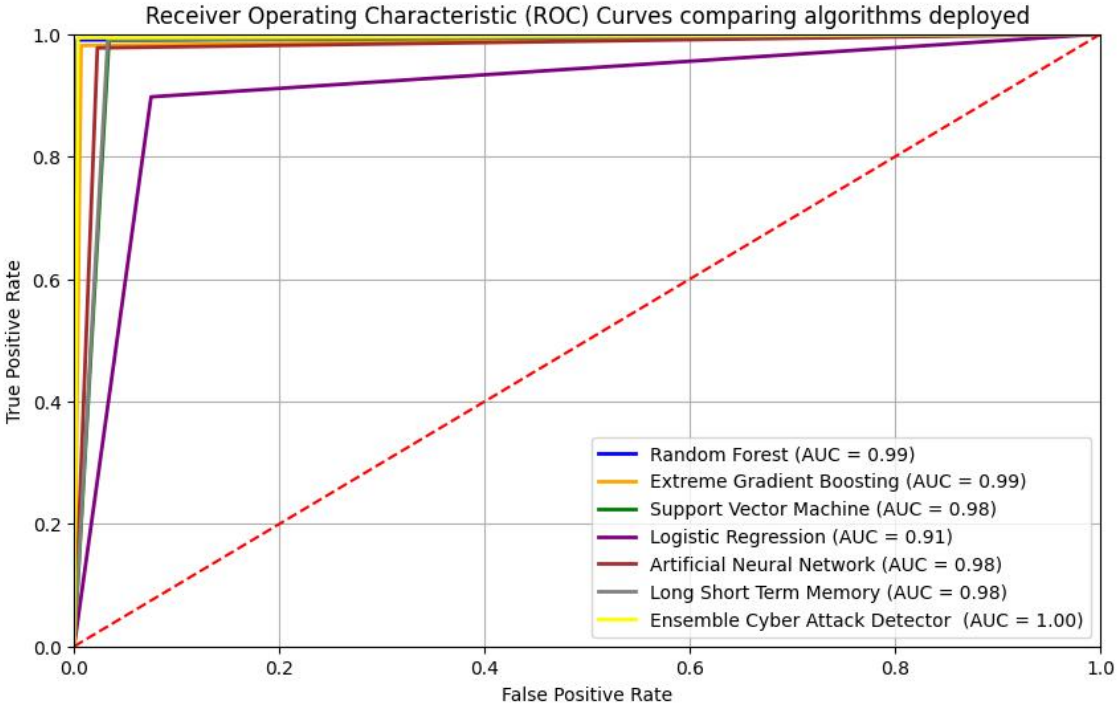



```
#print confusionmatrix and performance evaluation metrics for the ensemble model print(f'Ensemble_model
(ECU_IoHT):\n Accuracy: {accuracy_ensemble_model}\n Precision: {precision_ensemble_model}\n Recall:
{recall_ensemb sns.heatmap(cm_ensemble_model, annot=True, fmt='d', cmap='Blues') plt.title('Confusion Matrix
Ensemble CyberAttack Detector Classifier (ECU_IoHT)') plt.show()
```

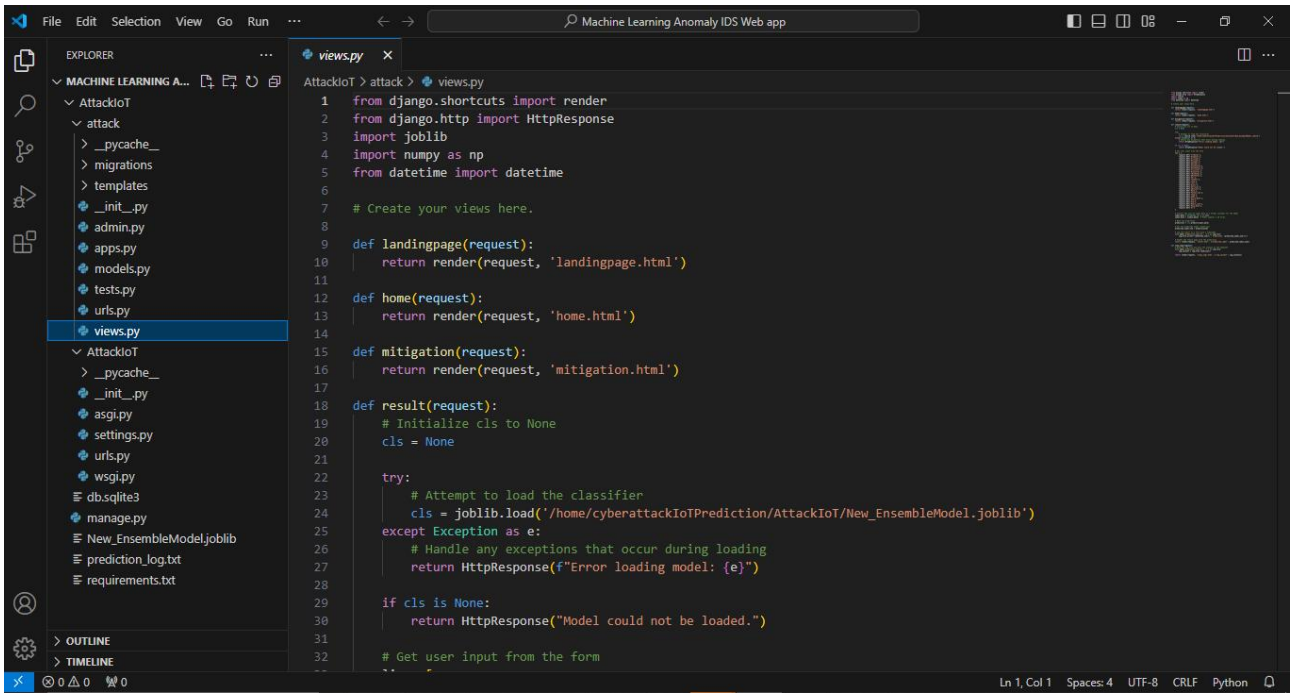
```
Ensemble_model (ECU_IoHT):
 Accuracy: 0.9958187213380092
 Precision: 0.9861907796898237
 Recall: 0.9940042826552462
 F1 Score: 0.9900821158152927
```




```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_curve, auc

#  Plot ROC curves for all the models deployed
plt.figure(figsize=(10, 6))
plt.plot(fpr_rf, tpr_rf, color='blue', lw=2, label=f'Random Forest (AUC = {roc_auc_rf:.2f})')
plt.plot(fpr_xgb, tpr_xgb, color='orange', lw=2, label=f'Extreme Gradient Boosting (AUC = {roc_auc_xgb:.2f})')
plt.plot(fpr_svm, tpr_svm, color='green', lw=2, label=f'Support Vector Machine (AUC = {roc_auc_svm:.2f})')
plt.plot(fpr_lr, tpr_lr, color='purple', lw=2, label=f'Logistic Regression (AUC = {roc_auc_lr:.2f})')
plt.plot(fpr_ann, tpr_ann, color='brown', lw=2, label=f'Artificial Neural Network (AUC = {roc_auc_ann:.2f})')
plt.plot(fpr_lstm, tpr_lstm, color='grey', lw=2, label=f'Long Short Term Memory (AUC = {roc_auc_lstm:.2f})')
plt.plot(fpr_ensemble, tpr_ensemble, color='yellow', lw=2, label=f'Ensemble Cyber Attack Detector  (AUC =
{roc_auc_ensemble:.2f})')
plt.plot([0, 1], [0, 1], color='red', linestyle='--')  # Diagonal line
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curves comparing algorithms deployed')
plt.legend(loc='lower right')
plt.grid()
plt.show()
```


Receiver Operating Characteristic (ROC) Curves comparing algorithms deployed
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Random Forest (AUC = 0.99)
Extreme Gradient Boosting (AUC = 0.99)
Support Vector Machine (AUC = 0.98)
Logistic Regression (AUC = 0.91)
Artificial Neural Network (AUC = 0.98)
Long Short Term Memory (AUC = 0.98)
Ensemble Cyber Attack Detector (AUC = 1.00)

APPENDIX F - CODE FOR IDS WEB APPLICATION



#views.py

```
from django.shortcuts import render
from django.http import HttpResponse
import joblib
import numpy as np
from datetime import datetime

# Create your views here.

def landingpage(request):
    return render(request, 'landingpage.html')

def home(request):
    return render(request, 'home.html')

def mitigation(request):
    return render(request, 'mitigation.html')

def result(request):
    # Initialize cls to None
    cls = None

    try:
        # Attempt to load the classifier
        cls = joblib.load('C:\\Users\\hp\\Desktop\\Web app\AttackIoT\\New_EnsembleModel.joblib')
    except Exception as e:
        # Handle any exceptions that occur during loading
        return HttpResponse(f"Error loading model: {e}")

    if cls is None:
        return HttpResponse("Model could not be loaded.")

    # Get user input from the form
    lis = [
        request.GET['SrcBytes'],
        request.GET['DstBytes'],
        request.GET['SrcLoad'],
```

```
        request.GET['DstLoad'],
        request.GET['SIntPkt'],
        request.GET['DIntPkt'],
        request.GET['SIntPktAct'],
        request.GET['DIntPktAct'],
        request.GET['SrcJitter'],
        request.GET['DstJitter'],
        request.GET['sMinPktSz'],
        request.GET['dMinPktSz'],
        request.GET['Dur'],
        request.GET['TotPkts'],
        request.GET['Load'],
        request.GET['Loss'],
        request.GET['pLoss'],
        request.GET['pSrcLoss'],
        request.GET['pDstLoss'],
        request.GET['Rate'],
        request.GET['Packet_num'],
        request.GET['Temp'],
        request.GET['SpO2'],
        request.GET['Pulse_Rate'],
        request.GET['SYS'],
        request.GET['DIA'],
        request.GET['Heart_rate'],
        request.GET['Resp_Rate'],
        request.GET['ST']
    ]

    # Convert the list of input data to a format suitable for the model
    input_data = [float(i) for i in lis]
    input_data = [input_data]  # Model expects a 2D array

    # Make the prediction
    prediction = cls.predict(input_data)

    # Get the predicted label (numerical)
    predicted_label_num = prediction[0]

    # Log the result in a file with a timestamp
    with open('prediction_log.txt', 'a') as log_file:
        log_file.write(f"{datetime.now()} - Prediction: {predicted_label_num}\n")

    # Render the result page with the prediction
    return render(request, 'result.html', {'predicted_label': predicted_label_num})

def view_logs(request):
    # Read the log file and pass the content to the template
    with open('prediction_log.txt', 'r') as log_file:
        log_content = log_file.readlines()

    return render(request, 'views_logs.html', {'log_content': log_content})
```

Urls.py

```
from django.contrib import admin
from django.urls import path
from attack.views import home, result, landingpage, mitigation, view_logs

urlpatterns = [
    path('', landingpage, name='landingpage'),
    path('home/', home, name='home'),
    path('result/', result, name='result'),
```

```
    path('mitigation/', mitigation, name='mitigation'),
    path('view_logs/', view_logs, name="view_logs"),
    path('admin/', admin.site.urls),
]
```

APPENDIX G - TRANSCRIPT FOR AUDIO INTERVIEW

All names used in the interview are fictional to ensure the privacy of the individuals involved.

Interview transcript for Participant 1

Facilitator 00:00
Hi Bella. Hello. Thank you for agreeing to join this interview. You're welcome. Alright, so let's dive into the first question. How easy was it for you to navigate and use the Intrusion Detection Web Application?

Participant 1
00:13
It was easy, I must say. The user interface was intuitive, and I was also able to get to know the features relevant to detect Intrusion in healthcare. I especially liked the clear labeling of different sections.

Facilittor
00:26
Oh, excellent. Did the Intrusion Detection Web Application meet your expectations in terms of functionality?

Participant 1
00:43
Overall, yes, it actually did, because I was able to see real -time threat detections and analyze historical data effectively. However, some additional feature options for specific attack types would be helpful.

Facilittor
00:48
Oh, that's great to hear. That's great to hear. How effective was the Intrusion Detection Web Application in predicting attacks?

Paricipant 1
00:55
The app did a very great job at identifying attacks, and the prediction matched what I anticipated based on the data provided.

Facilittor

01:00

Alright. Were there any technical issues, bugs, or performance challenges that you encountered while using the application?

Paricipant 1

01:12

Not really. Not really. They uploaded quickly and responded well to my actions.

Facilittor

01:20

Alright. Are there any improvements or additional features that you suggest for future developments of the web application?

Paricipant 1

01:28

As mentioned, additional filtering options would be a fantastic one. The ability to integrate with existing security tools or also receive automated alerts on high risk detection would also be valuable features.

Facilittor

01:44

Thank you so much for that feedback. Now, how does this Intrusion Detection Web Application impact healthcare?

Participant 1

02:03

As an healthcare person, I must say, Intrusion detection is vital for protecting people. data and ensuring system integrity. By detecting suspicious activities, we can proactively address potential threats and also prevent system failures by providing valuable insights into system user pattern and adding in capacitive planning and performance optimization.

Facilitator

02:20

Thank you so much Bela. You're welcome. Your suggestions are very useful. Finally, I would like you to rate the interaction with the web application on a scale of 0 to 5.

Participant 1

02:29

You did very well. I like the app, so I'm gonna rate it a 4.

Facilitator

02:38

Alright, thank you so much Bela. You're welcome. Definitely consider your suggestions while going forward with the application. Yeah, thank you and thank you for having me. The app was really great. I like it.

Facilitator

02:50

Well done. Good job to you. Alright, thank you very much. You're welcome. You have a nice day. And you too. Bye.

Interview transcript for Participant 2

Facilitator

00:00

Thank you, James, for taking time to evaluate my Intrusion detection web application. I would dive into the first question right now. So how easy was it for you to navigate and use the Intrusion detection web application?

Participant 2

00:18

I found the navigation pretty straightforward. The layout is clean and it didn't take long to figure out how to use it. Alright, thank you.

Facilitator

00:30

Thank you. Did the Intrusion detection web application meet your expectations in terms of functionality?

Participant 2

00:40

The IDS app exceeded my expectation. I was initially expecting something more basic, but it included extra features that were really useful, such as logging each detected anomaly and providing guidance for mitigating and adding health care security against cyber attacks.

Facilitator

01:05

Oh, that's wonderful. How effective was the Intrusion detection web application in predicting attacks? The predictions were accurate and it was reassuring to see the app flag potential issues quickly.

Participant 2

01:22

It seems like a great tool.

Facilitator

01:28

Were there any technical issues, bugs or performance challenges that you encountered?

Participant 2

01:34

I didn't run into any major issues. There was a slight delay submitting the feature entries when I imputed into the IDS, but it wasn't significant enough to impact my overall experience.

Facilitator

01:28

What improvements or additional features would you suggest for the future development of my Intrusion detection system web application for healthcare.

Participant 2

02:07

Overall the IDS app was educational, effectively bridging the gap between the theoretical aspects of machine learning based anomaly, detection and real world application. However a tutorial or guided walkthrough for new users could be beneficial. This addition would help first -time users particularly those unfamiliar with IDS systems to get up to speed more quickly and navigate the application with ease.

Facilitator

02:49

All right thank you that's that's important to note. How does this intrusion detection system web application impact healthcare?

Participant 2

03:10

An effective IDS is crucial for protecting patient data and ensuring the integrity of healthcare systems. By detecting anomalies and potential threats, it helps prevent data breaches and unauthorized access. This is especially important given the sensitive nature of medical information.and aid in compliance with regulations like HIPAA by monitoring for security incidents and providing evidence for investigations

Facilitator

03:20

Thank you for your feedback James, Rate the IDS app on a scale of 0-5

Participant 2

03:25

I would give the app a 5

Facilitator

03:30

Thank you, James! Your feedback is invaluable and will certainly help us improve the app.

Interview transcript for Participant 3

Facilitator

00:00

Hi Han, thank you for participating in the evaluation of my Intrusion Detection Systems web application. So I'll dive right into the first question. How easy was it for you to navigate and use the Intrusion Detection web app?

Participant 3

00:16

The navigation was smooth. I only had to click through a couple of menus to access the core functionality, which was great. No real learning code, which I appreciate.

Facilitator

00:25

Okay, that's great to hear. Did the IDS web application meet your expectations in terms of functionality?

Participant 3

00:36

It identified some potentially suspicious system events that needed investigation and the app also helped identify areas of potential security hardening.

Facilitator

00:47

Okay, awesome. How effective was the app in predicting attacks?

Participant 3

00:52

I tested it with some simulated data and the app was very effective. It caught all the abnormalities I expected it to with very few false positives.

Facilitator

1:00

Were there any technical issues, bugs or performance challenges that you encountered?

Participant 3

01:11

I didn't encounter any bugs. The app worked fine.

Facilitator

1:20

That's helpful to know. So what improvements or additional features would you suggest for future development of the IDS web application?

Participant 3

01:25

Actually, I would suggest adding a feature for exporting reports directly from the app, which would make it easier to share findings with the team without having to manually compile the data.

Facilitator

01:40

Thank you very much for your feedback. How does this web application impact health care?

Participant 3

01:57

An IDS is essential for maintaining network security and availability in the health care setting. By identifying potential threats early on, we can mitigate risks to patient care.An IDS can also help us optimize network performance by detecting abnormalities that might indicate other lining issues.

Facilitator

02:02

Alright, thank you so much for your time. Can you please reach my app on the scale of 0 to 5?

Participant 3

02:15

I would give it a 4.

Facilitator

02:20

Thank you so much for your feedback. It's appreciated and we will look into it as you continue developing the app.