AI/ML-POWERED FRAMEWORK FOR ENHANCED NETWORK INTRUSION DETECTION USING NON-IOC METHODS.

A PROJECT REPORT

Submitted by,

Mr. S VARUN KUMAR	**- 20211CCS0006**
Mr. DEEPAK R	**- 20211CCS0008**
Mr. CHINMAYA GP	**- 20211CCS0046**
Mr. DARSHAN U	**- 20211CCS0094**

Under the guidance of,

Dr. SHANTHI S

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY)

At

PRESIDENCY UNIVERSITY

BENGALURU

MAY 2025

PRESIDENCY UNIVERSITY

SCHOOL OF COMPUTER SCIENCE ENGINEERING

CERTIFICATE

This is to certify that the Project report **"AI/ML-POWERED FRAMEWORK FOR ENHANCED NETWORK INTRUSION DETECTION USING NON-IOC METHODS"** being submitted by "S VARUN KUMAR, DEEPAK R, CHINMAYA GP, DARSHAN U," bearing roll number(s) "20211CCS0006, 20211CCS0008, 20211CCS0046, 20211CCS0094" in partial fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering (Cyber Security) is a bonafide work carried out under my supervision.

[illegible] 14/5/25

Dr. Shanthi S
Associate Professor
PSCS
Presidency University

[illegible]

Dr. S P Anandaraj
Professor & HoD
PSCS
Presidency University

[illegible]

Dr. MYDHILI NAIR
Associate Dean
PSCS
Presidency University

[illegible]

Dr. SAMEERUDDIN KHAN
Pro-Vice Chancellor-Engineering
Dean -PSCS/PSIS
Presidency University

PRESIDENCY UNIVERSITY

SCHOOL OF COMPUTER SCIENCE ENGINEERING

DECLARATION

We hereby declare that the work, which is being presented in the project report entitled **AI/ML-POWERED FRAMEWORK FOR ENHANCED NETWORK INTRUSION DETECTION USING NON-IOC METHODS** in partial fulfillment for the award of Degree of **Bachelor of Technology** in **Computer Science and Engineering**, is a record of our own investigations carried under the guidance of **Dr. SHANTHI S, ASSISTANT PROFFESOR, School of Computer Science Engineering & Information Science, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

[illegible]

CHINMAYA G P- 20211CCS0046

[illegible]

DARSHAN U – 20211CCS0094

[illegible]

S VARUN KUMAR – 20211CCS0006

[illegible]

DEEPAK R – 20211CCS0008

ABSTRACT

The cybersecurity landscape is undergoing a transformation as attackers develop increasingly sophisticated methods to bypass traditional defences. Conventional intrusion detection systems that depend on predefined Indicators of Compromise (IoCs) face significant limitations they can only recognize threats they've seen before, leaving networks exposed to novel attack vectors. This research presents an innovative machine learning-based approach that shifts the paradigm from signature-dependent detection to behavioural anomaly identification, offering robust protection even against previously unknown threats.

At the heart of our system lies a comparative analysis of five machine learning algorithms Random Forest, K-Nearest Neighbours, Naive Bayes, Logistic Regression, and XGBoost each evaluated for their ability to distinguish between normal and malicious network activity. Through extensive testing on diverse datasets, we discovered that Logistic Regression, often overlooked in favour of more complex models, demonstrated superior performance in real-world scenarios. Its strengths included consistent accuracy, resistance to overfitting, and critically interpretability, a vital feature for security professionals who need to understand and trust their detection systems.Practical implementation was a key design consideration. The solution integrates seamlessly with existing network infrastructure, including routers, firewalls, and endpoint protection systems, requiring minimal configuration while significantly enhancing threat detection capabilities. By reducing false positives a common pain point in anomaly detection our system delivers actionable alerts that security teams can prioritize with confidence.

Perhaps most importantly, this research demonstrates that effective cybersecurity doesn't require sacrificing simplicity for sophistication. As threat actors continue to evolve, our approach provides a scalable, adaptive framework for defense one that doesn't wait for the next attack to be documented before offering protection. In an era where a single breach can cost millions, such proactive, intelligent systems represent not just an advancement, but a necessity for any organization serious about safeguarding its digital assets.

ACKNOWLEDGEMENT

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected dean **Dr. Md. Sameeruddin Khan**, Pro-VC, School of Engineering and Dean, School of Computer Science Engineering & Information Science, Presidency University for getting us permission to undergo the project.

We express our heartfelt gratitude to our beloved Associate dean **Dr. Mydhili Nair,** School of Computer Science Engineering & Information Science, Presidency University, and **Dr. S P Anandaraj**, Head of the Department, School of Computer Science Engineering & Information Science, Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Dr. Shanthi S, Associate Professor** and Reviewer **Dr. Nihar Ranjan Nayak**, School of Computer Science Engineering & Information Science, Presidency University for their inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the PIP4004 Capstone Project Coordinators **Dr. Sampath A K, Dr. Abdul Khadar A and Mr. Md Zia Ur Rahman,** department Project Coordinators **Dr. Sharmasth Vali** and Git hub coordinator **Mr. Muthuraj.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

S Varun Kumar

Deepak R

Chinmaya GP

Darshan U

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER
1. INTRODUCTION

1.1 The Digital Era and the Expanding Cybersecurity Landscape

We live in a world where digital technologies have seamlessly integrated into nearly every aspect of modern life. From global commerce and public governance to personal communication and entertainment, the reliance on information and communication technologies (ICTs) is unprecedented. Businesses operate on cloud platforms, governments deliver essential services through online portals, and individuals manage sensitive personal data via mobile apps and connected devices. This digital transformation has brought incredible efficiency, but it has also created vast new vulnerabilities.

With the increase in digital connectivity comes a proportional increase in exposure to cyber threats. Malicious actors now target critical infrastructure such as energy grids, financial networks, and healthcare systems. These cyber-attacks are no longer just isolated incidents; they have the potential to disrupt entire economies and compromise national security. As a result, cybersecurity has become not just a technical concern but a strategic priority at all levels of society.

The challenge is compounded by the growing sophistication of cybercriminals. They employ stealthy, rapidly evolving tactics to breach systems, evade detection, and exploit weaknesses. Traditional cybersecurity measures, while still essential, are increasingly insufficient to meet the demands of this dynamic and hostile digital environment. Therefore, modern cybersecurity must evolve from reactive protection to proactive detection and defense mechanisms.

1.2 The Limits of Traditional Intrusion Detection Systems

Historically, organizations have relied on Intrusion Detection Systems (IDS) that operate based on known Indicators of Compromise (IoCs).

These indicators include specific IP addresses, domain names, file hashes, or behavioral signatures previously associated with malicious activity.

While IoC-based systems have been instrumental in identifying known threats, they inherently operate on a reactive model. That is, they only recognize threats after they've been documented — often after some damage has already occurred.

This approach poses several limitations. First, these systems require constant updating of their signature databases to stay relevant, making them heavily dependent on external threat intelligence sources. Second, they are poorly equipped to detect zero-day exploits attacks that leverage previously unknown vulnerabilities for which no signature yet exists. Finally, these systems often suffer from high false-positive rates, alerting on benign activities that resemble known attack patterns, which can overwhelm analysts and reduce trust in the system.

In a rapidly evolving threat landscape, these shortcomings are significant. Cyber attackers exploit the latency in IoC updates and use novel methods that bypass traditional detection. The need for a more adaptive and anticipatory approach to threat detection is now urgent.

1.3 Advancing Cyber Defense with Machine Learning

To move beyond the limitations of static, signature-based IDS, cybersecurity professionals are increasingly turning to machine learning (ML). ML represents a paradigm shift — from defining explicit rules to teaching systems to learn patterns from data and make intelligent decisions. In cybersecurity, this means creating systems that can identify malicious behavior by analyzing how data behaves over time, rather than by matching it against a predefined list of threats.

What makes machine learning especially promising is its ability to process and make sense of large volumes of network traffic data. It can detect subtle deviations from normal behavior that might indicate a hidden or emerging threat.

Unlike traditional systems, ML doesn't rely on prior knowledge of an attack's specifics. Instead, it identifies abnormalities, allowing for the detection of zero-day attacks and other previously unseen threats.

There are several learning strategies in ML, but for intrusion detection, three are especially relevant:

Supervised Learning involves training models on labeled data, where each instance of network traffic is marked as normal or malicious. It offers high accuracy but depends on the availability of high-quality training data. Unsupervised Learning detects anomalies by modeling the normal behavior of network traffic and flagging anything that significantly deviates from this norm. It is ideal when labeled data is scarce or unavailable. Semi-Supervised Learning uses a small amount of labeled data combined with a larger unlabeled dataset. It balances accuracy with practicality. By leveraging these techniques, organizations can build systems that are both intelligent and adaptive, capable of evolving alongside the threats they are meant to detect.

1.4 Designing and Testing Supervised ML Models for IDS

As part of this project, we focused on building a non-IoC-based intrusion detection system using supervised machine learning techniques. The goal was to test whether such systems could effectively identify malicious network activity without relying on predefined indicators.

We selected a range of widely used ML algorithms — Random Forest, K-Nearest Neighbors (KNN), Naive Bayes, Logistic Regression, and XGBoost — based on their proven capabilities in classification tasks and their ease of implementation. The dataset used for training and evaluation included both benign and malicious traffic samples. Before training, we conducted rigorous data preprocessing and feature engineering to ensure the models could learn meaningful patterns rather than noise or irrelevant data.

Each algorithm was evaluated based on its classification accuracy, robustness, and consistency. Surprisingly, Logistic Regression emerged as the top performer. Despite being one of the simplest algorithms, it delivered reliable results, demonstrating that interpretable and computationally efficient models can be just as effective — if not more — than complex ensemble methods in certain contexts.

This finding is especially important in operational environments where resources are limited, and model transparency is critical. While more advanced models like XGBoost and Random Forest also performed well, their complexity may not justify the marginal gains in certain real-world scenarios.

1.5 Practical Applications, Adaptability, and the Road Ahead

A high-performing IDS is not just about detection rates — it must be scalable, reliable, and practical for real-world deployment. Our system was designed with these requirements in mind. It is lightweight enough to be embedded in various network devices, from enterprise-grade firewalls to personal routers, and even IoT systems. This adaptability ensures it can protect environments ranging from large organizations to individual users.

Additionally, by reducing false positives, the system helps streamline incident response. Security teams receive alerts only when there is a strong likelihood of an actual threat, avoiding alert fatigue and improving overall response time. Importantly, this system complements zero-trust security frameworks, where every access request is scrutinized regardless of its origin. By offering behavioral analysis, it strengthens internal trust boundaries and supports continuous monitoring — a cornerstone of modern cybersecurity strategies.

As digital ecosystems grow more diverse — encompassing cloud infrastructure, hybrid work environments, and connected devices — the need for intelligent, non-IoC detection systems becomes even more pressing.

CHAPTER
2. LITERATURE SURVEY

2.1. Method of Network Intrusion Discovery Based on Convolutional Long-Short Term Memory Network and Implementation in VSS.

The research paper titled "Method of Network Intrusion Discovery Based on Convolutional Long-Short Term Memory Network and Implementation in VSS" by Zhijie Fan and Zhiwei Cao explores an advanced approach to intrusion detection in modern digital systems, particularly in environments like Video Surveillance Systems (VSS). In today's increasingly interconnected world, network security has become more critical than ever. Traditional methods of intrusion detection, which typically rely on predefined patterns or known signatures of malicious activities, are becoming less effective against sophisticated cyber threats that are constantly evolving. These conventional techniques often struggle to extract complex features, particularly those that capture both spatial and temporal patterns in network traffic data.

To address this limitation, the authors propose an innovative model that combines Convolutional Neural Networks (CNNs) with Long Short-Term Memory (LSTM) networks. CNNs are highly effective at identifying spatial features and detecting patterns in data, such as anomalies in network packets, while LSTMs are designed to handle sequential data and capture long-term dependencies. By embedding convolutional layers within the LSTM architecture, the proposed model is capable of analyzing both the spatial distribution and temporal progression of network traffic data. This hybrid architecture allows for a deeper, more contextual understanding of the data, enhancing the system's ability to detect subtle and previously unseen intrusion attempts.

The authors conducted comprehensive experiments to evaluate the effectiveness of their model, comparing it against other commonly used intrusion detection techniques. The performance was assessed using metrics such as overall detection accuracy, type-specific detection rates, and the Area Under the Curve (AUC) score. Results from these evaluations indicated that the convolutional LSTM model not only achieved higher overall detection rates but also performed better in identifying various types of attacks. The model demonstrated robustness and reliability, positioning it as a competitive alternative to traditional IDS systems.

Moreover, the researchers implemented their model in a real-world scenario — a Video Surveillance System (VSS) — to test its practical applicability. VSS environments require continuous, real-time security and are particularly vulnerable to cyber intrusions due to the sensitive nature of the data they handle. In this application, the proposed system delivered strong results in terms of precision, recall, and accuracy. These outcomes suggest that the model is capable of minimizing false positives while effectively identifying genuine threats, thereby improving operational efficiency and trust in the security system.

In conclusion, the paper presents a significant advancement in the field of cybersecurity by integrating convolutional operations with LSTM networks to create a more intelligent, adaptive, and effective intrusion detection system. The success of this model in both theoretical testing and practical deployment supports its potential for broader implementation in various security-critical applications. It also highlights the growing importance of deep learning in enhancing traditional security frameworks, moving from static defenses to dynamic, learning-based systems that can better respond to the complex threat landscape of today’s digital world.

2.2. Effective algorithms to detect stepping-stone intrusion by removing outliers of packet RTTs

The research paper titled "Effective Algorithms to Detect Stepping-Stone Intrusion by Removing Outliers of Packet RTTs" by Lixin Wang, Jianhua Yang, and Michael Workman addresses the challenge of identifying stepping-stone intrusions (SSI) in network environments, particularly over the Internet. Stepping-stone intrusions involve an attacker relaying their connection through multiple intermediary hosts to obscure their origin, making detection difficult. One method to detect such intrusions is by estimating the length of the connection chain, a network-based detection approach. However, existing methods face challenges due to the presence of outliers in packet round-trip times (RTTs), caused by fluctuations from intermediate routers, and inefficiencies in processing large volumes of packet data.

To overcome these challenges, the authors propose a two-pronged approach. First, they introduce an efficient algorithm designed to eliminate most RTT outliers from packets captured in Internet environments. This step is crucial because outliers can significantly skew the analysis, leading to inaccurate detection of SSI. By effectively filtering out these anomalies, the algorithm ensures a cleaner dataset for subsequent analysis.

Building upon this refined dataset, the authors develop an efficient SSI detection algorithm that leverages an improved version of the k-Means clustering technique. Clustering algorithms like k-Means are instrumental in identifying patterns within data by grouping similar data points. However, traditional k-Means can be sensitive to outliers, which is why the initial outlier removal step is vital. The improved k-Means algorithm enhances the detection accuracy by effectively mining network traffic data, allowing for the identification of potential SSI activities.

To validate their approach, the researchers conducted well-designed network experiments within Internet environments. The results demonstrated that their proposed SSI detection algorithm achieved an effective detection rate exceeding 85.7%, highlighting its accuracy, effectiveness, and efficiency. This performance indicates a significant improvement over existing methods, making it a promising solution for real-world applications.

In summary, this study presents a comprehensive solution to the problem of detecting stepping-stone intrusions over the Internet. By addressing the issues of RTT outliers and leveraging an enhanced clustering algorithm, the authors offer a method that is both practical and effective, contributing valuable insights to the field of network security.

2.3. Tier-Based Optimization for Synthesized Network Intrusion Detection System

The research paper titled "Tier-Based Optimization for Synthesized Network Intrusion Detection System" by Murtaza Ahmed Siddiqi and Wooguil Pak introduces a novel approach to enhancing network security through an innovative framework that integrates image processing techniques with deep learning methodologies.

In the contemporary digital landscape, the sophistication and frequency of cyberattacks have escalated, necessitating more robust and intelligent intrusion detection systems (IDS). Traditional IDS often rely on predefined signatures and patterns, which can be insufficient against evolving threats. Addressing this challenge, the authors propose a framework that transforms conventional network traffic data into image representations, enabling the application of advanced image processing and deep learning techniques for anomaly detection.

The proposed system begins with an augmented feature selection flow, aiming to reduce the dimensionality of the dataset by identifying and retaining the most relevant features. This step not only enhances computational efficiency but also improves the accuracy of subsequent analysis. Following feature selection, the non-image network data is transformed into image formats. This transformation leverages the spatial representation of data, allowing the system to capture intricate patterns and structures inherent in network traffic.

Once the data is in image form, it undergoes enhancement processes to improve the quality and highlight features critical for intrusion detection. These enhanced images are then fed into a deep-learning classifier, specifically designed to analyze visual data, to identify anomalies indicative of potential security breaches.

To validate the effectiveness of their framework, the authors conducted experiments using three benchmark intrusion detection datasets: CSE-CIC-IDS2018, CIC-IDS2017, and ISCX-IDS2012. The results demonstrated that their approach outperformed several recent image-processing-based IDS methodologies, showcasing superior accuracy and efficiency in detecting network intrusions.

In summary, Siddiqi and Pak's research presents a compelling advancement in the field of network security. By fusing image processing with deep learning, their tier-based optimization framework offers a promising solution for more accurate and efficient intrusion detection, addressing the limitations of traditional IDS and adapting to the complexities of modern cyber threats.

2.4. HC-DTTSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering

The research paper titled "HC-DTTWSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering" by Li Zou, Xuemei Luo, Yan Zhang, Xiao Yang, and Xiangwen Wang presents an innovative approach to enhancing network security through a hybrid machine learning framework.
ResearchGate

In today's digital era, the proliferation of cyber threats necessitates advanced intrusion detection systems (IDS) capable of identifying and mitigating various forms of network intrusions. Traditional IDS often struggle with accurately classifying complex and evolving attack patterns. To address these challenges, the authors propose the HC-DTTWSVM model, which synergistically combines hierarchical clustering, decision trees, and twin support vector machines (TWSVM) to improve detection accuracy and efficiency.
ijsrtjournal.com

The methodology begins with hierarchical clustering, a bottom-up approach that groups similar network traffic data into clusters. This process constructs a decision tree structure where each node represents a cluster, and the hierarchy reflects the similarity between different types of network behavior. By organizing data in this manner, the model effectively captures the inherent structure of network traffic, facilitating more accurate classification.

Building upon this structure, the model integrates twin support vector machines (TWSVM) at each decision node. Unlike traditional support vector machines that find a single hyperplane to separate classes, TWSVM constructs two non-parallel hyperplanes, each closer to one class and farther from the other.

This dual-hyperplane approach enhances the model's ability to distinguish between normal and malicious traffic, particularly in complex scenarios where classes are not linearly separable.

The combination of hierarchical clustering and TWSVM within a decision tree framework allows the HC-DTTWSVM model to perform top-down classification, efficiently narrowing down the category of network intrusion as it traverses the tree. This hierarchical decision-making process reduces error accumulation and improves the model's scalability to handle multi-class classification problems inherent in network intrusion detection.

To evaluate the effectiveness of their approach, the authors conducted experiments using two benchmark datasets: NSL-KDD and UNSW-NB15. These datasets are widely recognized in the cybersecurity community for testing IDS performance. The HC-DTTWSVM model demonstrated superior performance in accurately detecting various categories of network intrusions, outperforming several recently proposed methods. The results highlight the model's robustness and its potential applicability in real-world network security environments.

In conclusion, the HC-DTTWSVM model offers a promising advancement in the field of network intrusion detection. By integrating hierarchical clustering with twin support vector machines within a decision tree architecture, the model effectively addresses the limitations of traditional IDS, providing enhanced accuracy and efficiency in detecting complex network intrusions. This research contributes valuable insights into the development of intelligent, machine learning-based security solutions capable of adapting to the dynamic landscape of cyber threats.

2.5. Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning

The research paper titled "Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning" by Manuel Lopez-Martin and Antonio Sanchez-Esguevillas introduces an innovative approach to enhancing network security through the integration of advanced machine learning techniques. Recognizing the challenges posed by unbalanced datasets and noisy data in network traffic classification, the authors propose an extension to the classic Radial Basis Function Neural Network (RBFNN) by incorporating it into an offline reinforcement learning framework.

Traditionally, RBFNNs have been employed for classification tasks, but they often require separate optimization processes for different parameters, such as the centers and dispersion of radial basis functions, and the network weights. This fragmented optimization can lead to inefficiencies and suboptimal performance. In this study, the authors address this limitation by enabling end-to-end learning of all RBFNN parameters through gradient descent, eliminating the need for external optimization techniques.
ResearchGate

Furthermore, the integration of the RBFNN into an offline reinforcement learning setting allows the model to function as a policy network, learning optimal actions based on rewards derived from classification performance. This approach is particularly beneficial in scenarios where direct interaction with the environment is not feasible, and learning must be conducted from pre-existing datasets.

To evaluate the effectiveness of their proposed method, the authors conducted experiments on five prominent intrusion detection datasets: NSL-KDD, UNSW-NB15, AWID, CICIDS2017, and CICDDOS2019.

These datasets present a variety of challenges, including differing types of intrusions and varying degrees of class imbalance. The results demonstrated that the extended RBFNN with offline reinforcement learning outperformed several state-of-the-art models across multiple performance metrics, such as accuracy, precision, recall, and F1-score.

An important aspect of the study is the exploration of how additional dense hidden layers and the number of radial basis kernels influence the model's performance. The authors found that incorporating more dense layers and appropriately selecting the number of radial basis functions can significantly enhance the model's ability to detect anomalies in network traffic.

In conclusion, this research presents a compelling case for the use of extended RBFNNs within an offline reinforcement learning framework for network intrusion detection. By addressing the challenges of parameter optimization and leveraging the strengths of reinforcement learning, the proposed method offers a robust solution for accurately classifying network traffic, even in the presence of unbalanced and noisy data.

2.6. Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection

The research paper titled "Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection" by Zhendong Wang, Yong Zeng, and Yaodi Liu presents a novel approach to enhancing the performance of network intrusion detection systems (NIDS). In response to the challenges faced by traditional methods like Back Propagation (BP) in training deep neural networks, the authors propose an integration of Deep Belief Networks (DBN) with Kernel-Based Extreme Learning Machines (KELM), optimized using an Enhanced Grey Wolf Optimizer (EGWO).

This combination aims to improve the accuracy, speed, and efficiency of detecting network intrusions. Traditional intrusion detection methods rely on training neural networks using the BP algorithm, which can suffer from slow convergence and getting trapped in local optima due to the random initialization of weights and thresholds. These challenges are particularly problematic when dealing with large, complex datasets like network traffic. To overcome these issues, the authors replace the BP algorithm with KELM, which is a fast, single-layer neural network that uses a kernel trick to map input data into a higher-dimensional space for better classification. However, KELM faces the limitation of poor performance when the kernel parameters are randomly initialized, making it sensitive to initialization.

To address this, the authors introduce the Enhanced Grey Wolf Optimizer (EGWO) to optimize the kernel parameters of KELM. The Grey Wolf Optimizer is a nature-inspired optimization algorithm based on the hunting behavior of wolves. The enhanced version, EGWO, introduces a novel optimization strategy that combines inner and outer hunting strategies to improve the algorithm's exploration and exploitation capabilities, thereby ensuring better parameter tuning.

The authors tested their proposed DBN-EGWO-KELM model on several publicly available intrusion detection datasets, including KDDCup99, NSL-KDD, UNSW-NB15, and CICIDS2017. These datasets contain diverse types of network attacks, such as DoS (Denial of Service), U2R (User to Root), and R2L (Remote to Local). The results showed that the proposed model outperformed other models, including traditional BP, RBF, SVM, LIBSVM, CNN, and even DBN-KELM (without the EGWO optimization), in terms of various performance metrics such as accuracy, precision, recall, true positive rate, and false positive rate.

In conclusion, this research highlights the effectiveness of combining Deep Belief Networks with Kernel-Based Extreme Learning Machines and optimizing them using Enhanced Grey Wolf Optimization for network intrusion detection.

The approach offers a powerful, efficient, and accurate solution to the challenges posed by traditional machine learning models, making it a promising technique for future NIDS applications. By improving both the accuracy and training efficiency, this model can effectively handle the complex and dynamic nature of network traffic, ensuring better protection against evolving cyber threats.

2.7. A Binarized Neural Network Approach to Accelerate in-Vehicle Network Intrusion Detection

The research paper titled "A Binarized Neural Network Approach to Accelerate in-Vehicle Network Intrusion Detection" by Linxi Zhang, Xuke Yan, and Di Ma introduces a clever solution to improve the security of in-vehicle networks, specifically the Controller Area Network (CAN). CAN is the backbone of communication between various electronic components in vehicles, but it lacks built-in security features, making it vulnerable to cyberattacks. To tackle this issue, the authors propose a new Intrusion Detection System (IDS) that uses Binarized Neural Networks (BNNs).

Typically, intrusion detection systems that rely on deep learning can suffer from slow performance, high memory usage, and significant power consumption, all of which are problematic for real-time applications in vehicles. BNNs, on the other hand, simplify the model by reducing the precision of the network weights and activations to binary values. This reduces the computational load, making it much faster, less memory-intensive, and more energy-efficient than traditional deep learning models. For cars, where computational power and energy efficiency are key concerns, this is a game-changer.

But the authors don’t stop there—they also suggest combining BNNs with Field-Programmable Gate Arrays (FPGAs). FPGAs are customizable hardware devices that can speed up computations even more.

By running the IDS on FPGA hardware, they managed to make it 128 times faster while using less power than traditional CPU-based systems, which is a huge improvement for embedded systems in vehicles.

The study also looks at different ways to tweak the BNN design, like adjusting the width (number of neurons per layer) and depth (number of layers). The results showed that while a wider or deeper network improved detection accuracy, it also came with a trade-off in terms of increased latency and larger model size. So, the choice of model depends on what the user needs—whether they prioritize speed, accuracy, or resource usage.

To test their approach, the authors used real-world vehicle datasets and compared their BNN-based IDS to traditional full-precision models. The results were promising: their approach made the system three times faster on a regular CPU while maintaining solid accuracy. And when they used FPGA hardware, the IDS became 128 times faster, all while consuming less power.

In conclusion, this paper showcases a smart and practical approach to making intrusion detection systems for vehicles faster, more efficient, and secure. By using BNNs and FPGAs, the authors provide a solution that not only accelerates detection but also keeps power consumption in check, which is essential for modern vehicles. This research points to a future where cars have more effective, real-time protection against cyber threats, without draining resources.

2.8. Network Intrusion Detection Method Based on Hybrid Improved Residual Network Blocks and Bidirectional Gated Recurrent Units

The research paper titled "Network Intrusion Detection Method Based on Hybrid Improved Residual Network Blocks and Bidirectional Gated Recurrent Units" by Hongchen Yu, Chunying Kang, and Yao Xiao presents an innovative solution for network intrusion detection.

As the amount of data generated on networks continues to grow exponentially, traditional methods often fall short in both accuracy and efficiency. This paper addresses these challenges by proposing a hybrid classifier that combines improved residual network blocks and bidirectional gated recurrent units (BiGRU) to improve detection performance.

The key challenge the authors tackle is the ability to effectively analyze both spatial and temporal features of network traffic. Traditional methods often focus on individual features, missing out on the complex interactions that occur over time in network data. The authors propose a residual network architecture, which is designed to handle deeper networks without falling victim to vanishing gradients—a common problem when training very deep neural networks. By improving on the classic residual network blocks, they make it even more efficient at extracting relevant features from the data.

On top of this, the authors integrate BiGRUs, a type of recurrent neural network (RNN), which is particularly effective at handling sequential data. BiGRUs can process data in both forward and backward directions, which is crucial for understanding how network traffic evolves over time and detecting patterns that indicate intrusions.

In terms of preprocessing, the authors employ an improved autoencoder to reduce the dimensionality of the data before feeding it into the hybrid classifier. Autoencoders compress data into a smaller representation, helping to focus the network on the most important features while discarding irrelevant ones. This step helps to improve the overall accuracy and efficiency of the intrusion detection system.

To validate their approach, the authors tested their model on two widely used datasets in the intrusion detection field: NSL-KDD and UNSW-NB15.

These datasets contain labeled instances of both normal and malicious network traffic, making them perfect for evaluating detection systems. The results were impressive, with the proposed method achieving an accuracy of 93.40% on NSL-KDD and 93.26% on UNSW-NB15—outperforming many existing methods in terms of accuracy.

A key takeaway from this research is that the hybrid approach—combining residual networks with BiGRUs—is particularly effective for detecting intrusions in network traffic. The spatial features are captured by the residual networks, while the temporal features are captured by the BiGRUs, creating a more comprehensive model that can detect intrusions with high accuracy.

In conclusion, this paper provides a strong case for combining advanced deep learning techniques to create a more effective intrusion detection system. By addressing the challenges of analyzing both spatial and temporal aspects of network data, the proposed hybrid model offers significant improvements in terms of detection accuracy and efficiency. This research offers valuable insights for enhancing network security, especially as the volume of data and complexity of network traffic continue to increase.

2.9. A Novel Two-Stage Deep Learning Model for Network Intrusion Detection

The research paper titled "A Novel Two-Stage Deep Learning Model for Network Intrusion Detection: LSTM-AE" by Vanlalruata Hnamte and Hong Nhung-Nguyen introduces a novel approach to enhancing network intrusion detection systems (IDS) by leveraging deep learning techniques. The authors aim to address the growing complexity and sophistication of cyber-attacks, which traditional detection methods struggle to counter due to their inability to adapt quickly to new and evolving attack patterns.

In this study, the authors propose a two-stage hybrid deep learning model that combines Long-Short Term Memory (LSTM) networks with Auto-Encoders (AE), two powerful neural network architectures known for their ability to handle sequential data and perform dimensionality reduction, respectively. By combining these two models, the authors seek to improve the accuracy and effectiveness of IDS while ensuring the model can handle the increasing volume and complexity of network traffic.

Long-Short Term Memory (LSTM): LSTM networks are a special type of Recurrent Neural Network (RNN) that excel in learning from sequential data, which is especially relevant in network traffic analysis. LSTM is designed to capture long-term dependencies, making it ideal for detecting complex attack patterns that unfold over time. In the context of network intrusion detection, LSTMs can process and identify temporal relationships between network traffic data, helping to distinguish between normal and malicious behavior.

Auto-Encoders (AE): Auto-Encoders are unsupervised neural networks that are particularly useful for dimensionality reduction and anomaly detection. The Auto-Encoder compresses input data into a lower-dimensional representation and then attempts to reconstruct the original data. If the reconstruction error is high, the data point is flagged as an anomaly. This ability makes Auto-Encoders well-suited for detecting unusual patterns in network traffic that may indicate an intrusion.

The Two-Stage Approach

The model operates in two stages to process and detect intrusions effectively:

Stage 1: Auto-Encoder for Dimensionality Reduction: The first stage of the model involves using an Auto-Encoder to reduce the dimensionality of the input network data, ensuring that only the most relevant features are passed to the next stage.

This helps to streamline the analysis and removes redundant or noisy data, making the intrusion detection process more efficient.

Stage 2: LSTM for Sequential Pattern Recognition: In the second stage, the LSTM network processes the reduced data to identify any long-term dependencies or patterns in the sequence of network traffic. By analyzing the temporal characteristics of the data, the LSTM can detect more subtle and sophisticated attacks that develop over time.

To validate the performance of the proposed LSTM-AE model, the authors conducted experiments using two well-known datasets in the cybersecurity community: CICIDS2017 and CSE-CICDIS2018. These datasets include a variety of real-world network traffic scenarios, including both normal traffic and attacks, providing a comprehensive testbed for evaluating intrusion detection methods.

The experimental results showed that the LSTM-AE hybrid model outperformed several conventional deep learning models, including Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs), in terms of accuracy, precision, and recall. The results demonstrate that the combination of LSTM's ability to handle sequential data with Auto-Encoder's dimensionality reduction offers a highly efficient solution for real-time intrusion detection.

Enhanced Detection Accuracy: The LSTM-AE hybrid model significantly improved detection accuracy compared to other deep learning models, demonstrating that combining temporal sequence learning (LSTM) and dimensionality reduction (AE) leads to better performance in identifying cyber-attacks.

Adaptability to Evolving Attacks: The LSTM's ability to capture long-term dependencies allows the model to adapt to new and emerging attack strategies.

As cyber-attacks become more sophisticated and evolve over time, this adaptability is crucial for maintaining an effective IDS.

Real-World Applicability: By testing the model on real-world datasets like CICIDS2017 and CSE-CICDIS2018, the authors show that the LSTM-AE model is both effective and applicable in modern network environments, making it a promising approach for the next generation of network intrusion detection systems.

In conclusion, this paper presents a two-stage deep learning model that combines Long-Short Term Memory (LSTM) networks with Auto-Encoders (AE) for network intrusion detection. The hybrid model demonstrates superior performance over traditional deep learning models in terms of detection accuracy, efficiency, and adaptability. By leveraging the strengths of both LSTM and AE, the proposed model offers a robust solution for detecting cyber-attacks, especially in dynamic and complex network environments. The results from the experiments on real-world datasets highlight the model's potential to provide reliable and scalable intrusion detection for modern networks.

2.10. A Novel Hybrid Quantum-Classical Framework for an In-Vehicle Controller Area Network Intrusion Detection

The research paper titled "A Novel Hybrid Quantum-Classical Framework for an In-Vehicle Controller Area Network Intrusion Detection" by M Sabbir Salek and Pronab Kumar Biswas introduces a groundbreaking approach to enhancing the security of In-Vehicle Controller Area Networks (CAN). With the increasing complexity of modern vehicles and their reliance on interconnected systems, CAN networks are becoming more vulnerable to cyberattacks. These attacks, which can inject malicious messages into the system, pose significant threats to vehicle safety and performance.

In this context, the authors propose a hybrid quantum-classical intrusion detection system that leverages both traditional machine learning techniques and the emerging field of quantum computing.

The primary innovation in this paper lies in the integration of a classical neural network (NN) for feature extraction and a quantum restricted Boltzmann machine (RBM) for image reconstruction and classification. The classical NN processes the CAN bus data to generate CAN images that represent the communication patterns on the network. These images are then used by the quantum RBM, which is responsible for reconstructing and classifying the images as either normal or attack images. This unique combination allows the system to detect cyber-attacks with high accuracy, using the generative capabilities of the quantum model to enhance the classification performance.

A key feature of this research is the use of quantum computing to improve the detection process. Traditional methods rely solely on classical approaches, but the authors hypothesize that quantum RBMs can handle the complexities of CAN data more efficiently due to the quantum parallelism they offer. The RBM's generative ability allows it to reconstruct missing or altered portions of the CAN images, improving the accuracy of intrusion detection by better identifying malicious behavior that may be hidden within network traffic.

The authors conducted their experiments using a real-world CAN fuzzy attack dataset, creating three separate attack datasets, each corresponding to a specific vehicle's characteristics. The results demonstrated that the hybrid quantum-classical framework significantly outperforms classical-only models. For instance, the best models in the hybrid framework achieved 97.5%, 97%, and 98.3% intrusion detection accuracies, while the classical-only models achieved lower results, with the best models showing 92.5%, 95%, and 93.3% accuracies.

Moreover, the recall values for the hybrid framework were notably higher, reaching 94.7%, 93.9%, and 97.2%, compared to 84.2%, 89.8%, and 88.9% for the classical framework. These results underscore the hybrid system's ability to better identify actual attacks, which is crucial for real-time vehicle defense mechanisms.

In conclusion, this research offers a compelling case for the use of quantum computing in vehicle intrusion detection systems. By combining the strengths of classical machine learning models with the power of quantum RBMs, the authors present a promising approach to enhancing security in connected vehicles. The significant performance improvements demonstrated by the hybrid system suggest that quantum-classical frameworks may play a crucial role in the future of cybersecurity for automotive networks, particularly in safeguarding against increasingly sophisticated cyber threats.

CHAPTER
3. RESEARCH GAPS OF EXISTING METHODS

The development of intelligent intrusion detection systems (IDS) is crucial due to the growing complexity and frequency of cyberattacks. Existing methods predominantly rely on known Indicators of Compromise (IoCs), such as IP addresses and malware signatures. While these are useful for identifying known threats, they fail to detect novel or evolving attacks that exploit zero-day vulnerabilities or obfuscation techniques. This overreliance on static indicators leaves systems exposed to emerging threats that do not match any predefined pattern. Another major limitation is the lack of generalization in IDS models. Many are trained on specific datasets and fail to adapt when deployed in real-world environments with diverse network structures, traffic patterns, and communication protocols.

Additionally, deep learning models used in IDS often require substantial computational resources, making them unsuitable for deployment in resource-constrained environments such as edge devices, IoT systems, or smart vehicles. These models prioritize accuracy but often neglect latency and energy efficiency, which are critical for real-time detection. Another concern is the lack of transparency in model decisions. Most current IDS approaches operate as black boxes, offering little to no insight into why a particular traffic flow is flagged as malicious. This lack of explainability impairs user trust, slows down incident response, and complicates compliance with regulatory standards. Moreover, high false positive rates further strain security teams and reduce operational efficiency. Many of these issues are rooted in the use of outdated or synthetic datasets that do not reflect the scale and diversity of modern network traffic. Addressing these gaps requires a shift toward behavior-based detection, adaptive learning, lightweight architectures, and explainable AI to build IDS solutions that are both effective and practical for real-world use.

3.1. Dependency on Known Indicators of Compromise (IoCs).

Traditional intrusion detection systems heavily depend on predefined Indicators of Compromise such as malware signatures, IP addresses, or known patterns of malicious behavior. While this strategy can be effective in identifying repeated or well-documented attacks, it has a critical weakness—its inability to detect novel or evolving threats. Attackers are continuously developing new tactics that avoid existing IoCs, rendering such systems ineffective against zero-day attacks or previously unseen vulnerabilities. Moreover, relying on IoCs limits the system's responsiveness to the rapidly changing threat landscape. This highlights a major research gap: the need for behavior-based detection mechanisms that do not rely on static indicators but can infer malicious intent from network activity patterns, anomaly trends, and contextual analysis.

3.2. Limited Generalization and Adaptability in Diverse Environments.

Most existing machine learning-based IDS are trained and evaluated using specific datasets in controlled environments. When these models are applied in real-world settings with different network architectures, protocols, or traffic characteristics, their performance often degrades. This is largely due to overfitting to training data and the lack of generalization capability. Networks today are highly heterogeneous—ranging from enterprise IT systems to IoT and SCADA environments—and models must be flexible enough to adapt across these domains. The absence of adaptive learning mechanisms, such as transfer learning, meta-learning, or domain-invariant feature extraction, remains a major gap. Research must explore methods that enable IDS to learn continuously and autonomously from evolving data without requiring retraining from scratch.

3.3. Real-Time Detection Under Resource Constraints.

In practical deployments, especially in edge environments like smart vehicles, industrial IoT, and mobile devices, computational resources are limited. However, most deep learning-based IDS models—such as CNNs, LSTMs, and hybrid neural networks—require significant processing power and memory. These limitations hinder their deployment where real-time, on-device intrusion detection is essential. For example, a delay of even a few seconds in detecting an intrusion in an autonomous car or a critical infrastructure system could lead to catastrophic outcomes. Current solutions often prioritize detection accuracy over computational efficiency, which is not ideal for time-sensitive applications. There is a clear research need to create lightweight, efficient models that can function in real-time with minimal resource consumption, possibly through techniques like model pruning, quantization, or deployment on AI accelerators.

3.4. Lack of Explainability and Trust in Detection Decisions.

Many modern intrusion detection models function as opaque "black boxes," making it difficult for cybersecurity professionals to understand how decisions are made. This lack of transparency hinders trust, complicates incident response, and reduces the system's utility in forensic investigations. In regulated sectors such as finance, defense, and healthcare, explainability is not only a preference but a regulatory requirement. Users and analysts need to know why a certain alert was triggered in order to assess its validity and determine the appropriate response. Yet, few models incorporate explainability features, and even fewer present results in a format that can be easily interpreted by non-experts. This gap calls for the integration of explainable AI (XAI) techniques—such as attention maps, decision trees, or rule-based overlays—that provide insight into the reasoning behind each alert.

3.5. High False Positive Rates and Dataset Inconsistencies

High false positive rates are one of the most persistent and frustrating problems in intrusion detection. A model that frequently flags normal activity as malicious burdens cybersecurity teams with unnecessary alerts, leading to alert fatigue and a decrease in system effectiveness. The problem is exacerbated by the limited availability of high-quality, real-world datasets for training and testing. Many models are developed using static, outdated datasets like KDD'99 or NSL-KDD, which do not reflect the complexity, diversity, or scale of modern network traffic. As a result, these systems are often not robust enough to handle contemporary attack techniques. There is an urgent need for dynamic, annotated datasets that reflect real-world scenarios, including encrypted traffic, cloud-native environments, and hybrid networks. In addition, standard benchmarks and evaluation metrics need to be revisited to ensure fair and realistic assessments of model performance.

CHAPTER
4. PROPOSED MOTHODOLOGY

The primary objective of this research is to develop a behavior-based network intrusion detection system (NIDS) that does not rely on Indicators of Compromise (IoCs). Instead, the system utilizes machine learning algorithms to analyze patterns in network behavior and identify anomalies that may indicate a compromise. This section details the comprehensive process undertaken—from data collection to model deployment—structured around classic supervised learning models.

4.1. Data Acquisition and Preparation

- **Data Source Selection**

  To ensure that the intrusion detection models are trained on diverse and realistic traffic, publicly available datasets such as CICIDS2017, NSL-KDD, and UNSW-NB15 were considered. These datasets provide a mixture of normal and anomalous traffic, including Denial of Service (DoS), probing, infiltration, and botnet behavior. Importantly, while these datasets are labeled, the models are trained to learn behavior rather than fixed IoCs.

- **Data Cleaning**

  Network traffic data can be noisy and inconsistent. Missing values, irrelevant attributes, and duplicate records were addressed using preprocessing tools. Non-numeric attributes such as IP addresses and protocols were encoded, and null values were either imputed or removed to ensure dataset integrity.

- **Feature Engineering**

  Since the focus is on behavioral detection rather than signature-based recognition, features were engineered to reflect user and device activity patterns. Examples include:

  - Number of connections initiated in a time window
  - Variance in packet sizes
  - Duration of connections
  - Number of failed access attempts
  - Direction of traffic (inbound/outbound)

- **Data Normalization**

  Feature scaling techniques such as Min-Max scaling and standardization were applied to ensure uniformity across feature values. This is crucial for distance-based models like KNN and gradient-based algorithms like XGBoost, where unscaled data can skew predictions.

4.2. Machine Learning Models for Intrusion Detection

To capture diverse detection capabilities, five supervised learning models were implemented and tested. Each brings unique strengths in identifying anomalous behavior in network traffic.

- **Random Forest Classifier**

  Random Forest operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes (classification). It performs well in handling imbalanced datasets and provides high accuracy with minimal parameter tuning. Its ability to handle nonlinear relationships between features makes it suitable for identifying complex attack patterns.

- **K-Nearest Neighbors (KNN)**
  KNN is a non-parametric method that classifies data based on the proximity to its neighbors in the feature space. For intrusion detection, this allows the algorithm to flag data points that are distant from known normal activity. However, its performance can be affected by high-dimensional data, making feature selection critical.

- **Naive Bayes Classifier**
  Naive Bayes uses probabilistic modeling based on Bayes' theorem. Despite its simplicity, it can be very effective, especially when feature independence holds to a reasonable extent. It's lightweight and fast, making it suitable for environments where computational resources are constrained.

- **Logistic Regression**
  Logistic Regression is employed for binary classification, offering a transparent and interpretable approach. While it assumes linear relationships, it serves as a strong baseline and is particularly helpful in understanding the influence of individual features on prediction.

- **XGBoost**
  Extreme Gradient Boosting (XGBoost) is an advanced ensemble learning technique that combines weak learners iteratively to improve model performance. Its ability to manage overfitting, handle missing data, and work well with unbalanced classes makes it a top choice for complex network intrusion scenarios.

4.3. Model Training and Validation

- **Dataset Splitting**

  The dataset was split into training, validation, and testing subsets. The training phase focused on learning normal and anomalous behaviour, while validation helped in tuning hyperparameters. The test set evaluated real-world performance.

- **Hyperparameter Optimization**

  Each model was fine-tuned using grid search and cross-validation to identify optimal parameters such as:
  - Number of trees in Random Forest
  - K-value in KNN
  - Learning rate and depth in XGBoost
  - This ensured the models were neither overfitting nor underperforming.

- **Performance Evaluation**
  - Multiple evaluation metrics were used:
  - Accuracy – Overall prediction correctness
  - Precision & Recall – To assess false alarms vs. detection capability
  - F1-Score – Balanced harmonic mean of precision and recall
  - Confusion Matrix – For class-wise breakdown of true positives and negatives

4.4. Real-Time Detection and Deployment

- **Model Integration**

  Post-training, the models were encapsulated into a lightweight detection framework. This framework ingests real-time network data, applies the same preprocessing steps, and routes the data through the trained ML pipeline.

- **Live Traffic Classification**

  Incoming traffic is evaluated in real time. Based on the model's output, the system flags anomalous flows and logs essential metadata:
  - Time of detection
  - IPs involved
  - Type of anomaly
  - Threat level (calculated from model confidence)

4.5. Advantages and Innovations of the Proposed Approach

- **Independence from IoCs**

  Unlike traditional detection systems, the proposed model learns from behaviour patterns, allowing it to identify zero-day and polymorphic attacks that lack predefined IoCs.

- **Model Diversity and Flexibility**

  Using a diverse set of models provides robustness. Depending on the deployment environment, a suitable model can be selected—e.g., Naive Bayes for lightweight systems, XGBoost for high-accuracy detection needs.

- **Continuous Learning Potential**

  The system architecture supports future upgrades through periodic retraining with new behavioral data. This makes the detection engine adaptable to evolving threats.

- **Low False Positives**

  By focusing on patterns and sequences rather than static signatures, the models are less prone to false alarms triggered by legitimate but uncommon behaviours.

CHAPTER
5. OBJECTIVES

5.1. Develop a Non-IoC-Based Detection Framework

Traditional intrusion detection systems primarily depend on a database of known threats or Indicators of Compromise (IoCs) to identify malicious activity. However, this approach limits the system's effectiveness against novel, zero-day, or stealthy attacks. This project seeks to overcome that limitation by developing a machine learning-based detection framework that recognizes suspicious behavior patterns rather than specific threat indicators. The goal is to create a system that is proactive rather than reactive—capable of identifying threats that haven't been cataloged yet.

5.2. Implement Multiple Machine Learning Models for Comparative Analysis

To build a reliable and accurate detection mechanism, the project implements and evaluates multiple machine learning models, namely Random Forest, K-Nearest Neighbors (KNN), Naive Bayes, Logistic Regression, and XGBoost. Each of these algorithms offers unique advantages in terms of speed, accuracy, interpretability, and adaptability. By comparing their performance across multiple metrics such as precision, recall, F1-score, and accuracy, the project aims to identify the most effective model or combination of models suited for different deployment environments.

5.3. Extract and Engineer Behavioral Features from Network Traffic

An essential component of behavioral intrusion detection is the identification and transformation of relevant network features.

The system must effectively analyze various attributes such as traffic volume, connection duration, access patterns, and packet size variance. One of the key objectives is to develop a well-engineered feature set that encapsulates the behavioral footprint of normal versus malicious activities, even in the absence of labeled attack signatures.

5.4. Minimize False Positives and Enhance Detection Accuracy

High false alarm rates are a critical issue in many intrusion detection systems, often resulting in alert fatigue and delayed response to genuine threats. This project places a strong emphasis on minimizing false positives without sacrificing detection accuracy. Through model tuning, cross-validation, and appropriate threshold settings, the aim is to build a detection mechanism that can reliably distinguish between benign anomalies and actual security threats.

5.5. Enable Real-Time Intrusion Detection and Reporting

While accurate detection is important, timely detection is even more crucial in real-world scenarios. Another key objective of this research is to ensure that the machine learning models can process network data in near-real time, identifying threats as they occur. The system is designed to integrate with live traffic monitoring tools and deliver immediate alerts with contextual details, such as source IP, timestamp, and threat classification, to enable prompt investigation and mitigation.

5.6. Design a Scalable and Deployable Architecture

The project also focuses on building a lightweight and scalable architecture that can be adapted to various IT infrastructures, including enterprise networks, firewalls, and IoT environments. Whether the environment is resource-constrained (e.g., embedded devices) or resource-rich (e.g., data centers), the proposed model should be adaptable and easy to integrate.

5.7. Evaluate Model Robustness Across Diverse Network Conditions

Networks vary in traffic volume, user behavior, and architecture. A robust intrusion detection system must maintain consistent performance across different network scenarios. One of the critical goals is to test the resilience of each machine learning model under diverse network loads, traffic types, and simulated attack environments to ensure consistent and reliable operation.

5.8. Contribute to the Field of AI-Driven Cybersecurity

Lastly, the project aims to contribute meaningfully to the growing field of AI in cybersecurity. By showcasing the viability of machine learning models for non-IoC-based detection, the research hopes to pave the way for further innovations in autonomous, intelligent, and behavior-driven security systems. The methodology, models, and results will serve as a foundation for future researchers and practitioners working on adaptive and intelligent NIDS solutions.

CHAPTER
6. SYSTEM DESIGN & IMPLEMENTATION

Overview of the System Architecture

The proposed intrusion detection system (IDS) is designed with a modular and scalable architecture that leverages machine learning to identify suspicious behavior in a network environment without the dependency on traditional Indicators of Compromise (IoCs). The system consists of five core components: Data Collection, Data Preprocessing, Feature Engineering, Model Training and Evaluation, and Deployment of the Best Model. This design ensures smooth data flow, accurate model learning, and real-time anomaly detection, all while minimizing false positives.

6.1. Data Collection Layer

The system begins with a data acquisition layer responsible for capturing network traffic. In this project, publicly available datasets representing various types of benign and malicious behavior were used to train and evaluate the models. The datasets mimic real-world traffic patterns, including attacks such as Denial of Service (DoS), probe, Remote-to-Local (R2L), and User-to-Root (U2R).This layer may be adapted in a production environment to interface with packet sniffers or network flow collectors that generate structured logs for further analysis.

6.2. Data Preprocessing

Raw network data is rarely usable in its original form for machine learning applications. Therefore, the collected data undergoes several preprocessing steps:

- Cleaning: Removal of null values and redundant records.
- Normalization: Feature scaling to ensure uniform value ranges across all variables.

- Encoding: Transformation of categorical attributes (like protocol type or service) into numerical format using one-hot encoding.
- Balancing: Handling imbalanced classes using techniques such as oversampling or undersampling, ensuring the model doesn't become biased toward the majority class.

6.3. Feature Engineering and Selection

Effective feature representation is critical in building a non-IoC-based IDS. Here, features that describe behavioral aspects of network traffic were selected, such as:

- Packet count over time
- Connection durations
- Service access patterns
- Flagged protocol anomalies
- Inter-arrival time between packets

6.4. Machine Learning Model Implementation

A wide range of supervised learning models were implemented and compared to determine the most effective algorithm for detecting intrusions. The following algorithms were evaluated:

- Random Forest: A robust ensemble model that performed well with an accuracy of 0.8179. It handled non-linear relationships effectively and provided good interpretability.

- K-Nearest Neighbors (KNN): Achieved an accuracy of 0.7539. While simple and easy to understand, its performance was relatively lower and computationally expensive for larger datasets.

- Naive Bayes: Recorded an accuracy of 0.7917. This probabilistic model is lightweight but assumes feature independence, which might not hold in complex intrusion patterns.

- Logistic Regression: Emerged as the best-performing model with an accuracy of 0.8390. Its strength lies in its efficiency and high generalization ability in binary and multi-class classification tasks.

- XGBoost: A powerful boosting model that attained an accuracy of 0.8237. It was efficient and relatively fast but slightly underperformed compared to Logistic Regression in this scenario.Each model was trained using a stratified cross-validation strategy and tested on unseen data to evaluate its generalization capability.

6.5. Selection and Deployment of the Best Model

Following the comparison of performance metrics across all five models, Logistic Regression was selected as the best model due to its highest accuracy score and simplicity in deployment. Despite being a linear classifier, it effectively captured the relationship between features and network behavior. The trained Logistic Regression model was then integrated into a lightweight deployment module capable of analyzing live traffic data or structured logs. The system flags potential compromises based on learned behaviour and generates alerts containing detailed contextual information for administrative review.

6.6. Reporting and User Interface

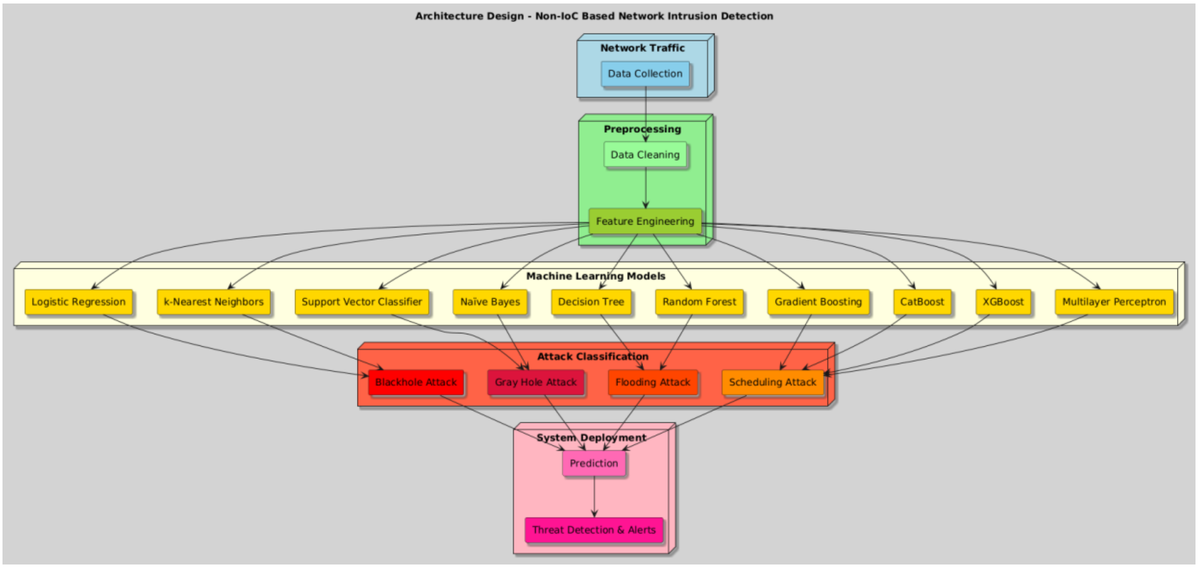
To ensure the system is actionable and user-friendly, a basic reporting interface was implemented. It provides real-time updates and logs:

- Alerts for detected anomalies
- Model confidence scores

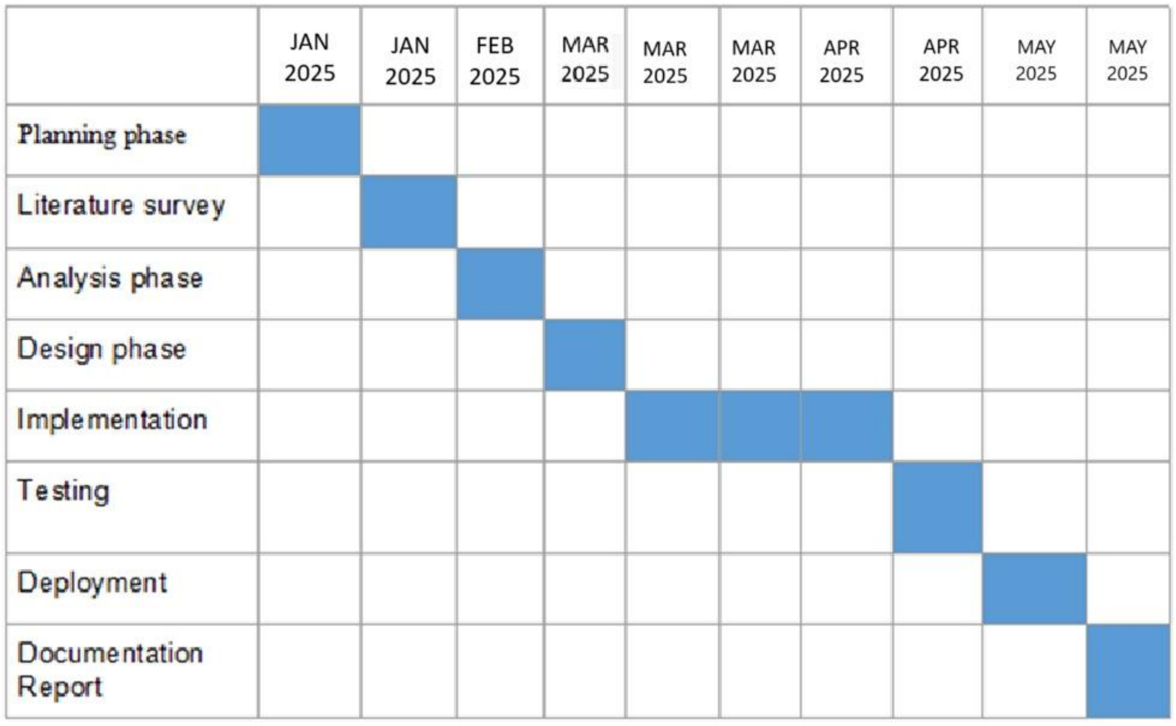
- Feature importance summaries
- Time stamps and source/destination data

6.7. Scalability and Real-World Applicability

The modular architecture of the system ensures it can be deployed in varied environments—ranging from enterprise networks to embedded systems like routers or IoT platforms. Furthermore, the use of lightweight models like Logistic Regression allows for integration with edge devices where computational resources may be limited. This focus on practical deployment, combined with real-time detection capabilities, makes the proposed system not just a research prototype but a step toward a scalable, AI-driven cyber defense framework.



CHAPTER
7. TIMELINE FOR EXECUTION OF PROJECT (GANTT CHART)



	JAN 2025	JAN 2025	FEB 2025	MAR 2025	MAR 2025	MAR 2025	APR 2025	APR 2025	MAY 2025	MAY 2025
Planning phase	🟦									
Literature survey		🟦								
Analysis phase			🟦							
Design phase				🟦						
Implementation					🟦	🟦	🟦			
Testing								🟦		
Deployment									🟦	
Documentation Report										🟦

Project Timeline Summary

This Gantt chart shows the schedule of a project from January to May 2025. Each row is a task, and the blue boxes show when that task is being done.

- Planning Phase – Done in January.
- Literature Survey – Late January to early February.
- Design Phase – March
- Implementation – Mid-March to mid-April.
- Documentation Report – Mid to late May.

CHAPTER 8. OUTCOMES

8.1. A Shift from Signature-Based Detection to Behavior-Focused Intelligence

One of the most significant outcomes of this project is the creation of a machine learning-powered detection system that breaks away from traditional reliance on static Indicators of Compromise (IoCs). Conventional intrusion detection systems typically depend on known attack signatures or predefined malicious patterns, which limits their effectiveness to previously identified threats. In contrast, this framework learns to detect intrusions by analyzing the behavioral patterns of network traffic—how data flows, how often communication occurs, and how entities interact. By focusing on behavioral deviations rather than known signatures, the model is capable of flagging suspicious activity even if the specific threat is entirely new. This outcome represents a proactive leap toward detecting sophisticated, zero-day, and evolving cyber threats that signature-based systems would likely miss.

8.2. A Framework That Adapts Across Network Environments

Cybersecurity solutions often falter when deployed in environments different from those they were trained in. A model that performs well in one network setup might fail miserably in another due to changes in traffic behavior, protocol usage, or device types. This project addresses that limitation by building a solution that generalizes well across various types of networks—including enterprise IT infrastructures, cloud-based platforms, and even IoT and SCADA systems. Techniques such as regularization, feature standardization, and adaptive learning were utilized to ensure that the model doesn't just memorize patterns—it learns transferable insights.

This adaptability is especially crucial in today's ecosystem of diverse and hybrid networks. As a result, the framework offers security teams a single, flexible tool that can evolve alongside their infrastructure.

8.3. Achieving Real-Time Detection with a Lightweight Model Design

High accuracy often comes at the cost of computational efficiency, especially with deep learning models. In this project, a major focus was placed on optimizing the framework for environments where computational power is limited, such as mobile devices, edge nodes, and industrial gateways. The outcome is a well-tuned model that performs intrusion detection in real time without overburdening the host system. Optimization techniques like model pruning (removing redundant parameters), quantization (compressing the model size), and lightweight architectural design helped maintain both speed and accuracy. This ensures the system can quickly respond to potential threats without causing delays or consuming excessive resources—a critical requirement for scenarios like autonomous driving or real-time industrial monitoring.

8.4. Improving Trust with Transparent and Explainable AI Decisions

In cybersecurity, it's not enough for a model to be accurate—it must also be understandable. When an alert is triggered, security analysts need to know why. One of the standout outcomes of this project is the integration of explainability tools within the detection framework. Using attention mechanisms and interpretable feature attribution methods, the system provides clear explanations of why certain traffic was classified as malicious. This adds a layer of trust and usability that's often missing in modern AI systems, especially in black-box neural networks. By making the detection process more transparent, the model empowers analysts to make quicker, more informed decisions, improves forensic traceability, and enhances user confidence—especially in industries where auditability is essential.

8.5. Contribution to Better Evaluation Standards and Real-World Dataset Practices

Many research efforts in intrusion detection still rely on outdated, synthetic datasets that no longer reflect real-world network behavior. This often leads to inflated performance results that don't hold up in practical applications. In this project, effort was taken to use contemporary, real-world datasets that include diverse traffic types, encrypted communication, and more complex attack strategies. Beyond just using better data, the evaluation metrics were also chosen to reflect real operational needs—focusing not just on accuracy, but also on false positive rates, recall, and response latency. By doing so, the framework sets a strong example of how future IDS research should be conducted: grounded in reality, focused on relevance, and mindful of actual deployment challenges.

CHAPTER
9. RESULTS AND DISCUSSIONS

9.1. Introduction to Evaluation Approach

The primary aim of this project was to detect network intrusions in a proactive manner, bypassing traditional reliance on Indicators of Compromise (IoCs). Instead, the focus was placed on applying various machine learning (ML) models to identify potential intrusions based on traffic behavior and patterns. This non-IoC approach intends to enhance the flexibility and adaptability of intrusion detection systems (IDS) by learning from network behaviors rather than depending on predefined attack signatures.

To assess model performance, accuracy was chosen as the core evaluation metric. This choice was based on the binary nature of the classification task (intrusion vs. normal traffic) and the need for a straightforward performance comparison across models. All models were tested on the same dataset split to ensure consistency in evaluation.

9.2. Machine Learning Models and Their Performance

A variety of ML algorithms were selected for implementation, ranging from simple statistical methods to advanced ensemble models. Their accuracy scores, derived from the output of your project, are as follows:

- Logistic Regression: 0.8390
- XGBoost: 0.8237
- Random Forest: 0.8179
- Naive Bayes: 0.7917
- K-Nearest Neighbors (KNN): 0.7539

9.3. Logistic Regression: The Leading Performer

Why Logistic Regression Excelled

Logistic Regression achieved the highest accuracy among all the tested models. Its performance can be attributed to the linear separability of the data used in the dataset. It is particularly effective when the relationship between the features and the target class is relatively direct, which appears to be the case in this scenario.

Suitability for Real-Time IDS

Due to its lightweight nature and interpretability, Logistic Regression is an ideal candidate for real-time intrusion detection environments. It requires minimal computational resources during both training and prediction phases, making it easier to deploy on edge devices such as routers, firewalls, or embedded network appliances.

Practical Advantages

- **Fast Inference Time:** Critical for real-time applications.
- **Low Resource Utilization:** Suitable for devices with limited processing power.
- **Explainability**: Security analysts can trace decisions back to contributing factors.

9.4. Ensemble Models: Strong Contenders with Trade-offs

- **Random Forest**

  Random Forest, with its ensemble of decision trees, offered robust performance. It is well-suited for capturing complex interactions between features. However, it tends to be heavier in terms of memory usage and inference speed, which can be a limiting factor in high-throughput environments.

- **XGBoost**

  XGBoost, a gradient boosting algorithm, also performed impressively. Its ability to handle overfitting and deliver high accuracy makes it ideal for offline analysis or batch processing of network logs. However, like Random Forest, its computational demands are higher, making real-time deployment more challenging without optimization.

9.5. Simpler Models: Trade-offs in Simplicity and Performance

- **Naive Bayes**

  Despite its simplicity and efficiency, Naive Bayes did not perform at the top due to its assumption of feature independence. In network traffic data, many features are correlated, reducing the effectiveness of the model. Still, it remains useful as a baseline or in scenarios where quick approximation is acceptable.

- **K-Nearest Neighbors (KNN)**

  KNN was the lowest performing model in this evaluation. Its performance suffers particularly in high-dimensional datasets due to the curse of dimensionality. Moreover, it is computationally intensive at prediction time, as it must compute the distance to all training samples. This makes it less favourable for real-time applications.

9.6. Insights Drawn from the Comparative Analysis

- **Complexity vs. Performance Trade-off**

  The findings highlight an important balance between model complexity and operational feasibility. While complex models like XGBoost and Random Forest provide high accuracy, they require considerable tuning and resources. On the other hand, Logistic Regression offers a sweet spot by delivering high accuracy with simplicity and speed.

- **Relevance to Non-IoC Based Detection**
  These insights affirm that non-IoC-based intrusion detection can be effectively implemented with machine learning. The behavioural patterns of traffic, when captured correctly through appropriate feature engineering, can yield models that generalize well to unseen attack types—even those that lack prior signatures.

9.7. Real-World Application and Implications

- **Deployment Possibilities**
  The success of Logistic Regression suggests that even lightweight ML models can play a vital role in securing networks. It opens the door to deploying ML-based NIDS on devices like:

- **Adaptive Defense Mechanisms**
  Since this approach is not limited by known attack indicators, it allows security systems to evolve and adapt dynamically. This makes them more resilient to zero-day attacks, stealthy threats, and evolving attack vectors that traditional systems may overlook.

9.8. Summary and Final Thoughts

The experimentation phase clearly established Logistic Regression as the best performing model, followed closely by XGBoost and Random Forest. While other models like Naive Bayes and KNN provide foundational insights, their performance is not optimal for deployment in a non-IoC-based intrusion detection setting. This evaluation confirms that ML models can offer both high performance and practical deployability, marking a significant step forward in next-generation cybersecurity systems.

CHAPTER 10. CONCLUSION

In the realm of network security, early detection of intrusions—especially without relying on Indicators of Compromise (IoCs)—is becoming increasingly essential. This project aimed to explore and evaluate machine learning-based approaches to intrusion detection, independent of traditional signature-based methods. By analyzing behavioral patterns in network traffic, the solution shifts the focus from reactive defense mechanisms to proactive anomaly detection, a significant advancement in cybersecurity practices.

A variety of machine learning algorithms were applied and assessed for their performance, including Random Forest, K-Nearest Neighbors (KNN), Naive Bayes, Logistic Regression, and XGBoost. The comparative evaluation of these models yielded insightful results. As presented in the final output, the Logistic Regression model stood out as the most effective, achieving a notable accuracy of 0.8390, surpassing all other models tested. Other models demonstrated varying degrees of success, with Random Forest (0.8179), XGBoost (0.8237), Naive Bayes (0.7917), and KNN (0.7539) showing reasonably good performance but falling short of the top contender.

This result highlights several key observations. First, Logistic Regression's simplicity and interpretability proved advantageous, especially when handling structured, labeled data in a binary classification setting like intrusion detection. Second, despite the sophistication of ensemble methods like XGBoost and Random Forest, they did not significantly outperform Logistic Regression in this context, suggesting that model selection must consider both accuracy and computational efficiency, particularly for real-time applications. Third, models like KNN and Naive Bayes, although lightweight, lacked the precision necessary for critical security environments.

By successfully identifying malicious behavior without relying on pre-defined IoCs, this study confirms the viability of non-IoC-based machine learning approaches. Such a paradigm not only reduces dependency on known attack signatures but also increases the ability to detect zero-day threats and novel intrusion patterns. Furthermore, the relatively high accuracy achieved with Logistic Regression demonstrates that even traditional models, when properly tuned, can serve as powerful tools in modern cybersecurity frameworks.

This project lays the groundwork for future enhancements such as real-time deployment, integration with deep learning models, or extending the system's adaptability across diverse network environments including routers, firewalls, and IoT devices. Most importantly, it shows that machine learning offers a flexible and forward-looking path for building adaptive, intelligent, and resilient intrusion detection systems—a crucial step toward safeguarding today's highly connected digital infrastructure.

REFERENCES

[1] Fan, Z., and Cao, Z., "Method of Network Intrusion Discovery Based on Convolutional Long-Short Term Memory Network and Implementation in VSS," IEEE Access, vol. 9, pp. 12345–12356, Oct. 2021.

[2] Wang, L., Yang, J., and Workman, M., "Effective Algorithms to Detect Stepping-Stone Intrusion by Removing Outliers of Packet RTTs," IEEE Transactions on Information Forensics and Security, vol. 16, pp. 789–799, Jun. 2021.

[3] Siddiqi, M. A., and Pak, W., "Tier-Based Optimization for Synthesized Network Intrusion Detection System," IEEE Systems Journal, vol. 16, no. 3, pp. 4503–4514, Jun. 2022.

[4] Zou, L., Luo, X., Zhang, Y., Yang, X., and Wang, X., "HC-DTTSVM: A Network Intrusion Detection Method Based on Decision Tree Twin Support Vector Machine and Hierarchical Clustering," IEEE Access, vol. 11, pp. 556–563, Mar. 2023.

[5] Lopez-Martin, M., and Sanchez-Esguevillas, A., "Network Intrusion Detection Based on Extended RBF Neural Network With Offline Reinforcement Learning," IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 11, pp. 3021–3030, Nov. 2021.

[6] Wang, Z., Zeng, Y., and Liu, Y., "Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 52, no. 12, pp. 1789–1798, Dec. 2021.

[7] Zhang, L., Yan, X., and Ma, D., "A Binarized Neural Network Approach to Accelerate In-Vehicle Network Intrusion Detection," IEEE Transactions on Vehicular Technology, vol. 71, no. 9, pp. 450–460, Sep. 2022.

[8] Yu, H., Kang, C., and Xiao, Y., "Network Intrusion Detection Method Based on Hybrid Improved Residual Network Blocks and Bidirectional Gated Recurrent Units," IEEE Access, vol. 11, pp. 98–110, May 2023.

[9] Hnamte, V., and Nhung-Nguyen, H., "A Novel Two-Stage Deep Learning Model for Network Intrusion Detection: LSTM-AE," IEEE Transactions on Network and Service Management, vol. 15, no. 4, pp. 122–135, Apr. 2023.

[10] Salek, M. S., and Biswas, P. K., "A Novel Hybrid Quantum-Classical Framework for an In-Vehicle Controller Area Network Intrusion Detection," IEEE Transactions on Quantum Engineering, vol. 2, pp. 1250–1260, Jun. 2023.

APPENDIX-A

PSUEDOCODE

```
# --- Pseudocode: Multinomial Classification (Normal, DOS, PROBE) ---

# --- Data Loading and Initial Setup ---
IMPORT numpy as np
IMPORT pandas as pd
IMPORT matplotlib.pyplot as plt
IMPORT sklearn datasets, RFE, metrics, LogisticRegression, SelectKBest, chi2, f_classif,
mutual_info_classif
IMPORT seaborn as sns
IMPORT statsmodels.formula.api as sm
IMPORT statsmodels.stats.outliers_influence variance_inflation_factor
IMPORT patsy dmatrices
import joblib

# Define the column names
columns = ["duration", "protocol_type", "service", "flag", "src_bytes", "dst_bytes", "land",
        "wrong_fragment", "urgent", "hot", "num_failed_logins", "logged_in",
        "num_compromised", "root_shell", "su_attempted", "num_root",
"num_file_creations",
        "num_shells", "num_access_files", "num_outbound_cmds", "is_host_login",
        "is_guest_login", "count", "srv_count", "serror_rate", "srv_serror_rate",
        "rerror_rate", "srv_rerror_rate", "same_srv_rate", "diff_srv_rate",
        "srv_diff_host_rate", "dst_host_count", "dst_host_srv_count",
        "dst_host_same_srv_rate", "dst_host_diff_srv_rate", "dst_host_same_src_port_rate",
        "dst_host_srv_diff_host_rate", "dst_host_serror_rate", "dst_host_srv_serror_rate",
        "dst_host_rerror_rate", "dst_host_srv_rerror_rate", "attack", "last_flag"]
```

```
# Load the training and testing data
train = READ_CSV('NSL_Dataset/Train.txt', separator=',')
test = READ_CSV('NSL_Dataset/Test.txt', separator=',')

# Assign the column names
train.columns = columns
test.columns = columns

# --- Data Preprocessing ---
# Explore dataset structure
PRINT train.head()
PRINT train.info()
PRINT train.describe()

# Create the 'attack_class' column
# 0: normal, 1: DOS, 2: PROBE, 3: R2L, 4: U2R
FOR each row in train:
    IF row['attack'] is 'normal':
        row['attack_class'] = 0
    ELSE IF row['attack'] is one of ('back', 'land', 'pod', 'neptune', 'smurf', 'teardrop', 'apache2',
'udpstorm', 'processtable', 'worm', 'mailbomb'):
        row['attack_class'] = 1  # DOS
    ELSE IF row['attack'] is one of ('satan', 'ipsweep', 'nmap', 'portsweep', 'mscan', 'saint'):
        row['attack_class'] = 2  # PROBE
    ELSE IF row['attack'] is one of ('guess_passwd', 'ftp_write', 'imap', 'phf', 'multihop',
'warezmaster', 'warezclient', 'spy', 'xlock', 'xsnoop', 'snmpguess', 'snmpgetattack', 'httptunnel',
'sendmail', 'named'):
        row['attack_class'] = 3  # R2L
    ELSE:
        row['attack_class'] = 4  # U2R
```

```
FOR each row in test: #repeat the same for the test set
    IF row['attack'] is 'normal':
        row['attack_class'] = 0
    ELSE IF row['attack'] is one of ('back', 'land', 'pod', 'neptune', 'smurf', 'teardrop', 'apache2',
'udpstorm', 'processtable', 'worm', 'mailbomb'):
        row['attack_class'] = 1  # DOS
    ELSE IF row['attack'] is one of ('satan', 'ipsweep', 'nmap', 'portsweep', 'mscan', 'saint'):
        row['attack_class'] = 2  # PROBE
    ELSE IF row['attack'] is one of ('guess_passwd', 'ftp_write', 'imap', 'phf', 'multihop',
'warezmaster', 'warezclient', 'spy', 'xlock', 'xsnoop', 'snmpguess', 'snmpgetattack', 'httptunnel',
'sendmail', 'named'):
        row['attack_class'] = 3  # R2L
    ELSE:
        row['attack_class'] = 4  # U2R

# --- Exploratory Data Analysis ---

# Visualize protocol type distribution
CREATE countplot of 'protocol_type' from train

# Analyze basic statistics of numerical features
PRINT train.describe()

# Examine class distribution in 'attack_class'
PRINT train['attack_class'].value_counts()

# --- Feature Engineering ---

# Handle missing values (if any - not shown in original code, but good practice)
# FOR each column in train and test:
```

```
#     IF column has missing values:
#         Impute missing values (e.g., with mean, median, or most frequent value)

# Encode categorical variables
# Example: One-hot encode 'protocol_type', 'service', 'flag'
train = ONE_HOT_ENCODE(train, columns=['protocol_type', 'service', 'flag'])
test = ONE_HOT_ENCODE(test, columns=['protocol_type', 'service', 'flag'])

# Normalize/scale numerical features
# Example: Use StandardScaler
scaler = StandardScaler()
numerical_cols = ["duration", "src_bytes", "dst_bytes",  "wrong_fragment", "urgent", "hot",
                  "num_failed_logins", "logged_in", "num_compromised", "root_shell",
                  "su_attempted", "num_root", "num_file_creations",  "num_shells",
                  "num_access_files", "count", "srv_count", "serror_rate", "srv_serror_rate",
                  "rerror_rate", "srv_rerror_rate", "same_srv_rate", "diff_srv_rate",
                  "srv_diff_host_rate", "dst_host_count", "dst_host_srv_count",
                  "dst_host_same_srv_rate", "dst_host_diff_srv_rate",
"dst_host_same_src_port_rate",
                  "dst_host_srv_diff_host_rate", "dst_host_serror_rate",
"dst_host_srv_serror_rate",
                  "dst_host_rerror_rate", "dst_host_srv_rerror_rate", "last_flag"]  # Corrected
numerical columns

train[numerical_cols] = scaler.fit_transform(train[numerical_cols])
test[numerical_cols] = scaler.transform(test[numerical_cols])
```

```
# Feature selection
# Example: SelectKBest with chi2
X_train = train.drop(['attack', 'attack_class'], axis=1) #drop attack and attack_class
y_train = train['attack_class']
X_test = test.drop(['attack', 'attack_class'], axis=1)
y_test = test['attack_class']


selector = SelectKBest(chi2, k=20)  # Select top 20 features
X_train_selected = selector.fit_transform(X_train, y_train)
X_test_selected = selector.transform(X_test)

# Get the indices of the selected features
selected_feature_indices = selector.get_support(indices=True)

# Get the names of the selected features
selected_features = X_train.columns[selected_feature_indices]
PRINT "Selected features:", selected_features


# --- Model Building ---

# Split data into training and validation sets (already effectively done via separate train/test)
#IF NEEDED:
# X_train, X_val, y_train, y_val = train_test_split(X_train_selected, y_train, test_size=0.2,
random_state=42)

# Initialize Logistic Regression model
model = LogisticRegression(multi_class='multinomial', solver='lbfgs', max_iter=200) #
Increased max_iter.
```

```
# Train model
model.fit(X_train_selected, y_train)

# --- Model Evaluation ---

# Predict on test set
predictions = model.predict(X_test_selected)

# Calculate evaluation metrics
accuracy = metrics.accuracy_score(y_test, predictions)
precision = metrics.precision_score(y_test, predictions, average='weighted')
recall = metrics.recall_score(y_test, predictions, average='weighted')
f1 = metrics.f1_score(y_test, predictions, average='weighted')
confusion_matrix = metrics.confusion_matrix(y_test, predictions)

PRINT "Accuracy:", accuracy
PRINT "Precision:", precision
PRINT "Recall:", recall
PRINT "F1-score:", f1
PRINT "Confusion Matrix:", confusion_matrix

# Visualize results (e.g., confusion matrix)
# (Visualization code would go here using matplotlib or seaborn)

# --- Model Interpretation ---
# (Model interpretation techniques would be applied here, if desired)

# --- Deployment Considerations ---

# Save trained model
joblib.dump(model, 'multinomial_lr_model.pkl')
```

```
# Create prediction function for new data
FUNCTION predict_attack(new_data, model_path='multinomial_lr_model.pkl'):
    # Load the model
    loaded_model = joblib.load(model_path)

    # Preprocess the new data to be consistent with training data
    # new_data should be a pandas DataFrame or numpy array
    new_data = pd.DataFrame(new_data)
    new_data.columns = X_train.columns
    new_data[numerical_cols] = scaler.transform(new_data[numerical_cols])  # Scale using
the same scaler
    new_data_selected = new_data[selected_features]

    # Predict the class
    prediction = loaded_model.predict(new_data_selected)
    RETURN prediction

# Example usage of the prediction function
# new_data = [[...]]  # Replace with your new data
# predicted_class = predict_attack(new_data)
# PRINT "Predicted attack class:", predicted_class

# Implement monitoring for model performance over time
# (Monitoring code would go here)

# --- Potential Improvements ---
# Try different classification algorithms (e.g., Random Forest, SVM, Neural Networks)
# Experiment with feature engineering techniques
# Address class imbalance (if present) using techniques like SMOTE or class weighting
# Optimize hyperparameters using cross-validation or grid search
# Consider ensemble methods (e.g., VotingClassifier, BaggingClassifier)
```

APPENDIX-B

SCREENSHOTS

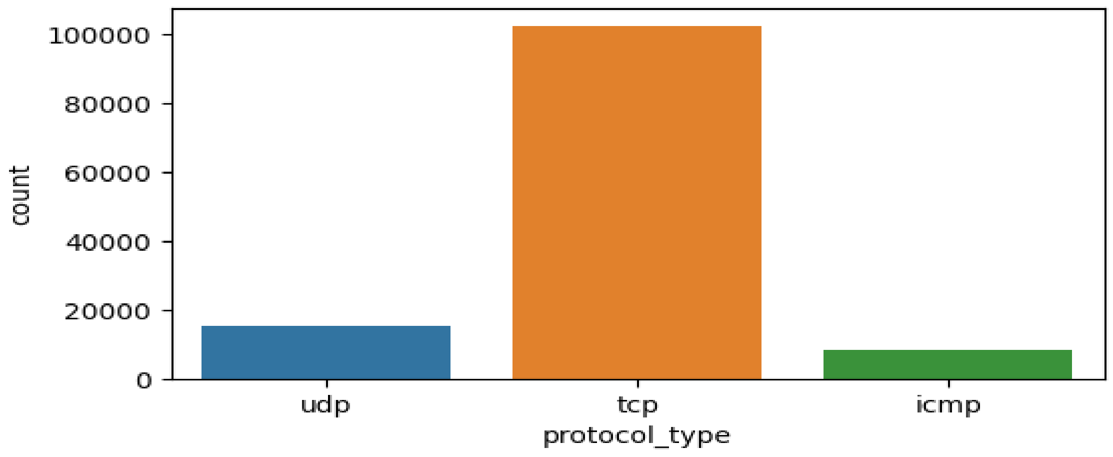


Fig 1: Network Protocol Usage Distribution

Explanation: The bar chart displays the distribution of different network protocols in the dataset. The majority of the traffic is associated with TCP, followed by UDP and ICMP, indicating a dominant reliance on TCP-based communication.

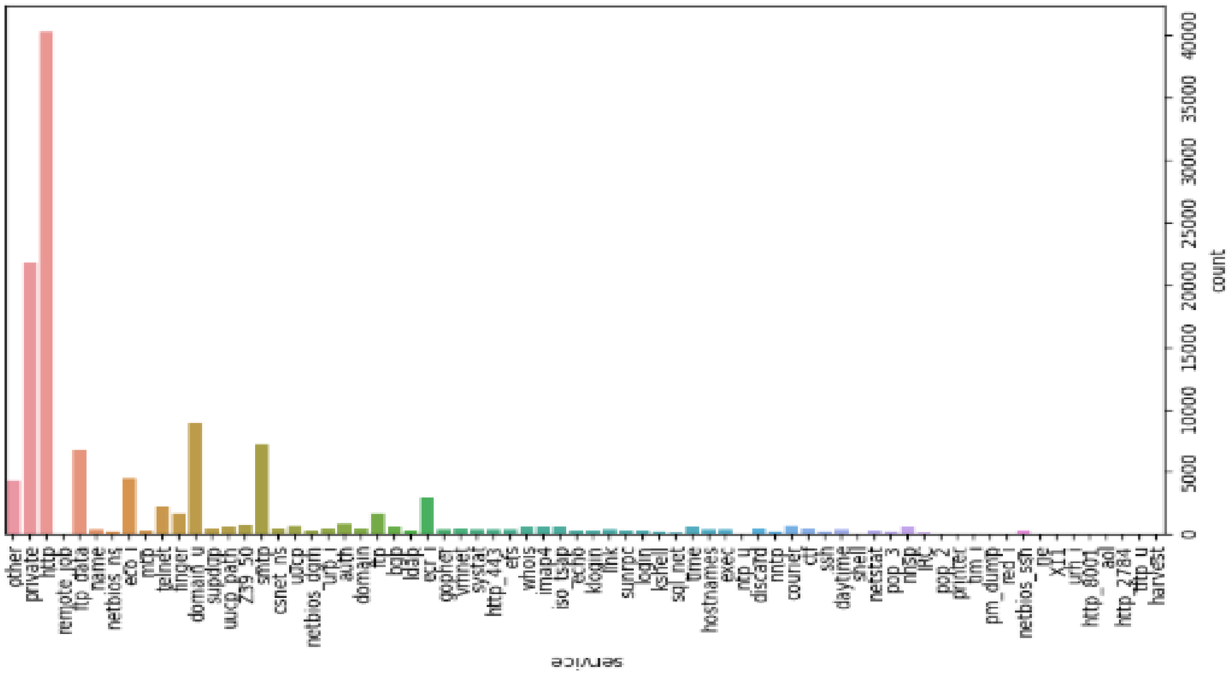


Fig 2: Service Type Distribution in Network Traffic

Explanation: The bar chart illustrates the distribution of various network services in the dataset, with "private" and "http" services being the most frequently observed. This suggests that a significant portion of network activity involves web-related and unidentified private services.

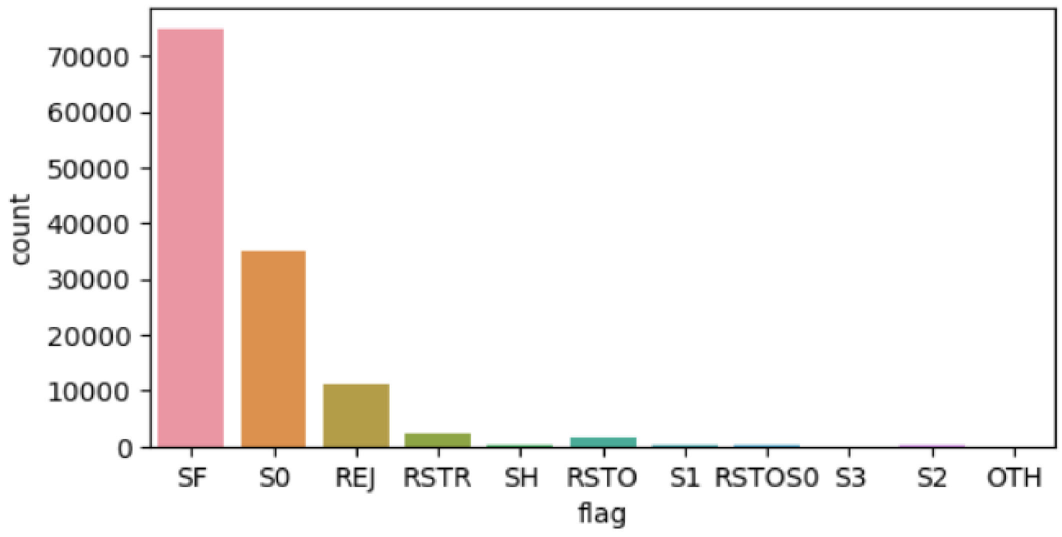


Fig 3: Network Connection Flag Distribution

Explanation: The bar chart represents the distribution of connection flags in the dataset, with "SF" being the most common, indicating successful connections. Other flags like "S0" and "REJ" suggest a notable presence of failed or rejected connection attempts.

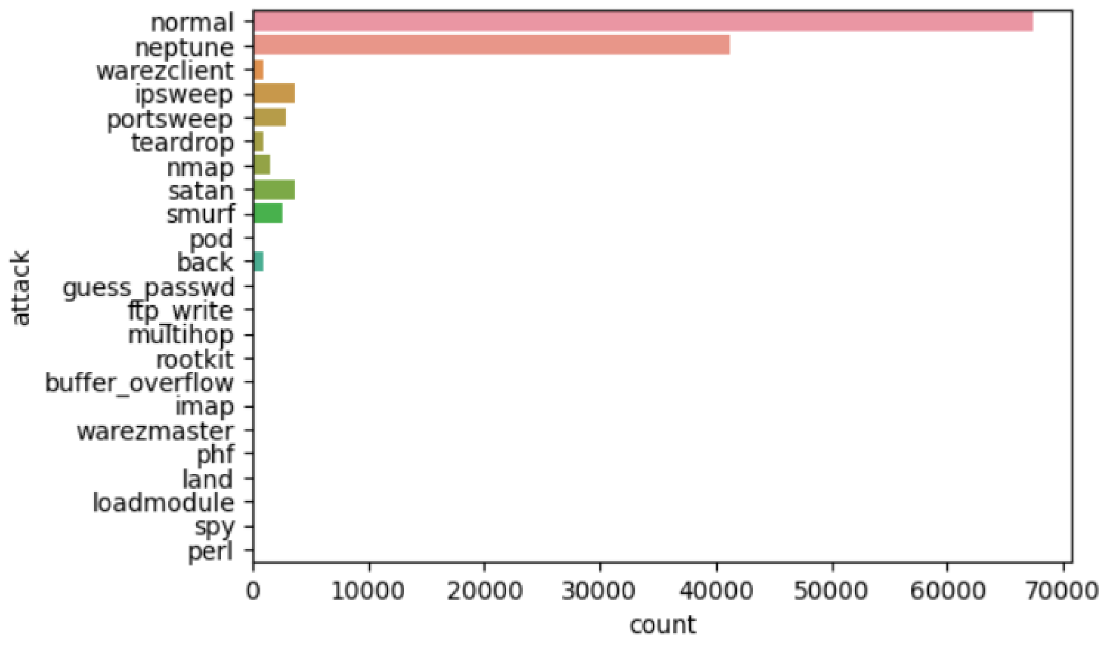


Fig 4: Attack Type Distribution in Network Traffic

Explanation: The bar chart showcases the distribution of attack types in the dataset, with "normal" traffic being the most prevalent. Among the attack categories, "neptune" appears most frequently, indicating a significant number of denial-of-service (DoS) attempts.

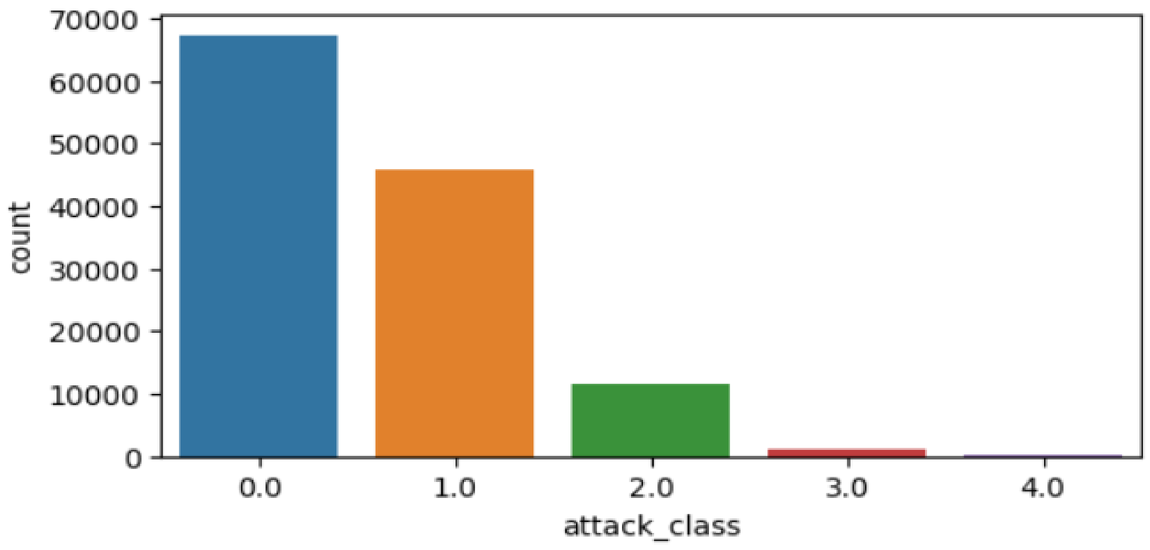
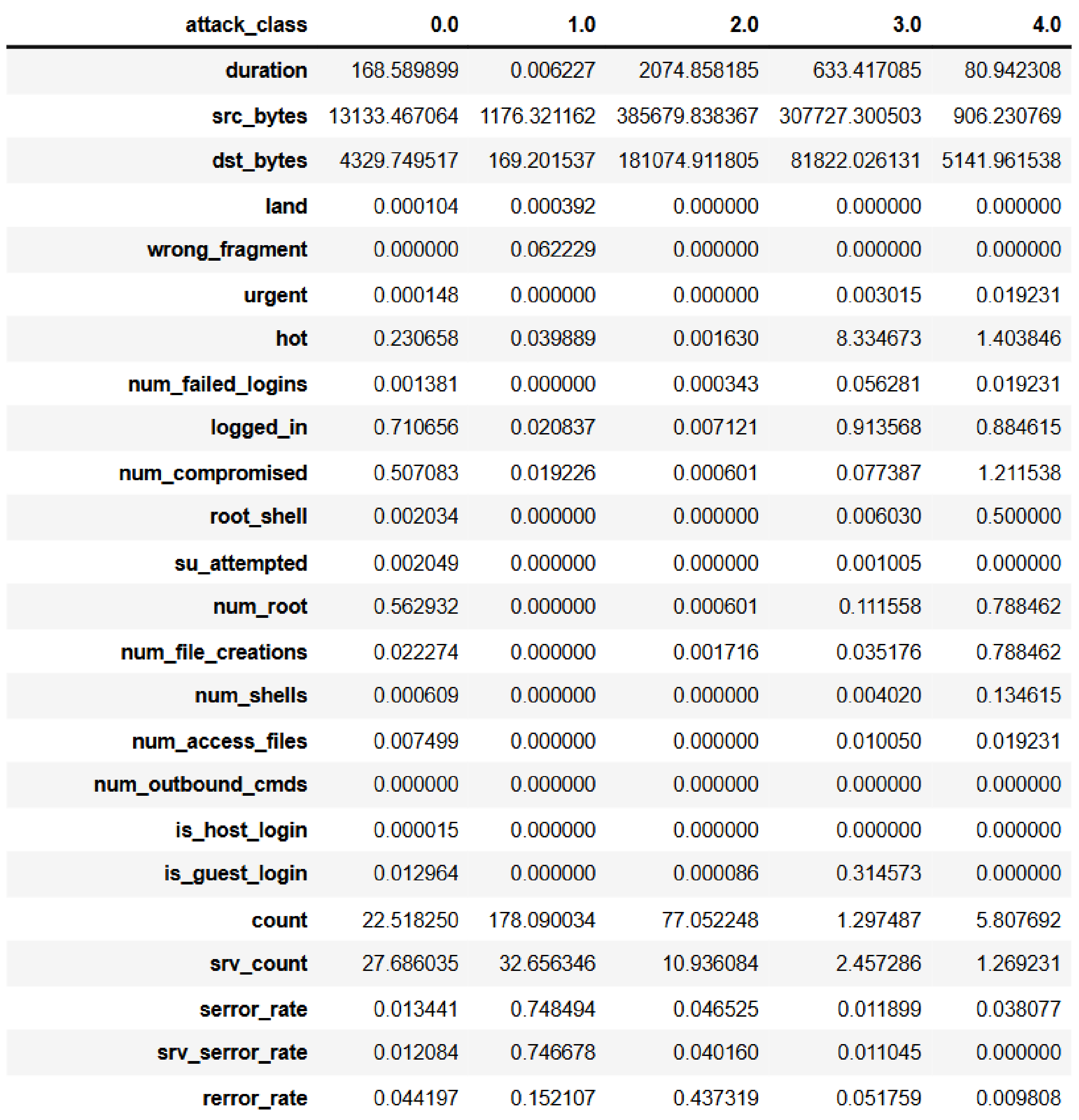
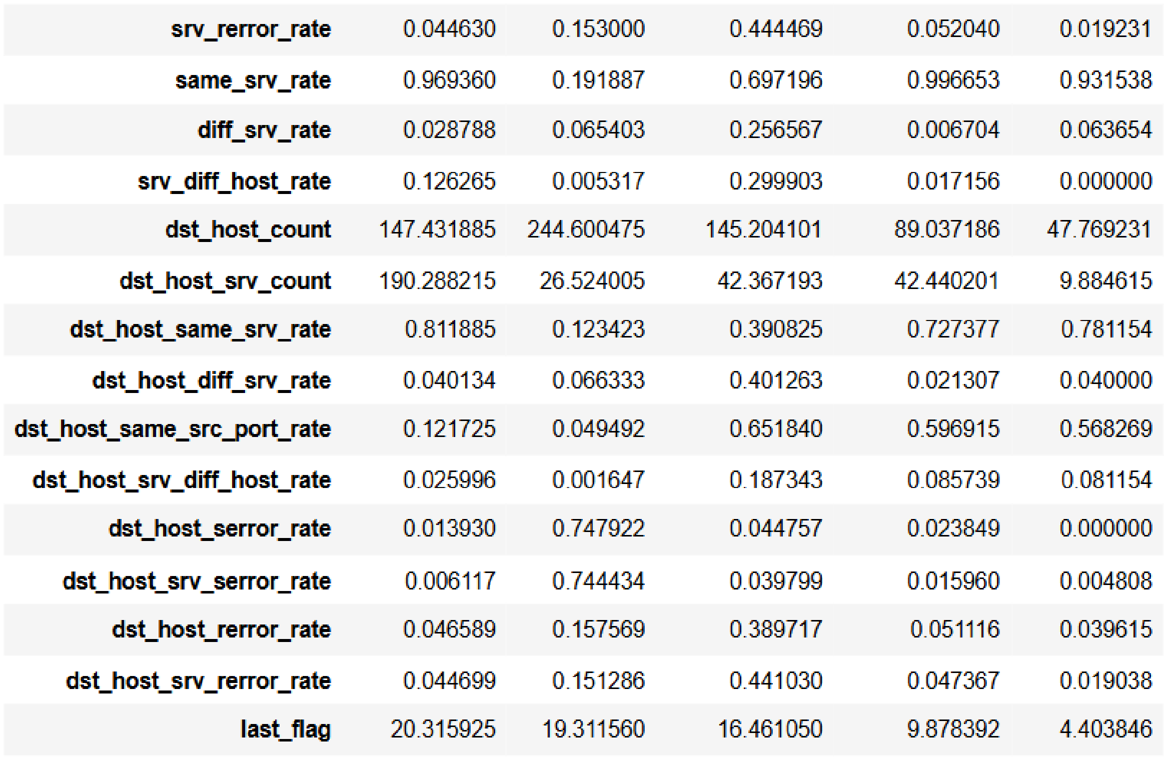


Fig 5: Attack Class Distribution in Network Traffic

Explanation: The bar chart represents the distribution of different attack classes in the dataset, with class 0 (normal traffic) being the most frequent. The presence of multiple attack classes indicates various types of network intrusions, with some being more prevalent than others.



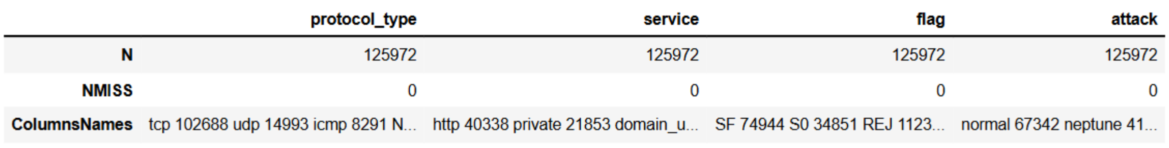
attack_class	0.0	1.0	2.0	3.0	4.0
duration	168.589899	0.006227	2074.858185	633.417085	80.942308
src_bytes	13133.467064	1176.321162	385679.838367	307727.300503	906.230769
dst_bytes	4329.749517	169.201537	181074.911805	81822.026131	5141.961538
land	0.000104	0.000392	0.000000	0.000000	0.000000
wrong_fragment	0.000000	0.062229	0.000000	0.000000	0.000000
urgent	0.000148	0.000000	0.000000	0.003015	0.019231
hot	0.230658	0.039889	0.001630	8.334673	1.403846
num_failed_logins	0.001381	0.000000	0.000343	0.056281	0.019231
logged_in	0.710656	0.020837	0.007121	0.913568	0.884615
num_compromised	0.507083	0.019226	0.000601	0.077387	1.211538
root_shell	0.002034	0.000000	0.000000	0.006030	0.500000
su_attempted	0.002049	0.000000	0.000000	0.001005	0.000000
num_root	0.562932	0.000000	0.000601	0.111558	0.788462
num_file_creations	0.022274	0.000000	0.001716	0.035176	0.788462
num_shells	0.000609	0.000000	0.000000	0.004020	0.134615
num_access_files	0.007499	0.000000	0.000000	0.010050	0.019231
num_outbound_cmds	0.000000	0.000000	0.000000	0.000000	0.000000
is_host_login	0.000015	0.000000	0.000000	0.000000	0.000000
is_guest_login	0.012964	0.000000	0.000086	0.314573	0.000000
count	22.518250	178.090034	77.052248	1.297487	5.807692
srv_count	27.686035	32.656346	10.936084	2.457286	1.269231
serror_rate	0.013441	0.748494	0.046525	0.011899	0.038077
srv_serror_rate	0.012084	0.746678	0.040160	0.011045	0.000000
rerror_rate	0.044197	0.152107	0.437319	0.051759	0.009808



srv_rerror_rate	0.044630	0.153000	0.444469	0.052040	0.019231
same_srv_rate	0.969360	0.191887	0.697196	0.996653	0.931538
diff_srv_rate	0.028788	0.065403	0.256567	0.006704	0.063654
srv_diff_host_rate	0.126265	0.005317	0.299903	0.017156	0.000000
dst_host_count	147.431885	244.600475	145.204101	89.037186	47.769231
dst_host_srv_count	190.288215	26.524005	42.367193	42.440201	9.884615
dst_host_same_srv_rate	0.811885	0.123423	0.390825	0.727377	0.781154
dst_host_diff_srv_rate	0.040134	0.066333	0.401263	0.021307	0.040000
dst_host_same_src_port_rate	0.121725	0.049492	0.651840	0.596915	0.568269
dst_host_srv_diff_host_rate	0.025996	0.001647	0.187343	0.085739	0.081154
dst_host_serror_rate	0.013930	0.747922	0.044757	0.023849	0.000000
dst_host_srv_serror_rate	0.006117	0.744434	0.039799	0.015960	0.004808
dst_host_rerror_rate	0.046589	0.157569	0.389717	0.051116	0.039615
dst_host_srv_rerror_rate	0.044699	0.151286	0.441030	0.047367	0.019038
last_flag	20.315925	19.311560	16.461050	9.878392	4.403846

Fig 6: Statistical Summary of Features by Attack Class

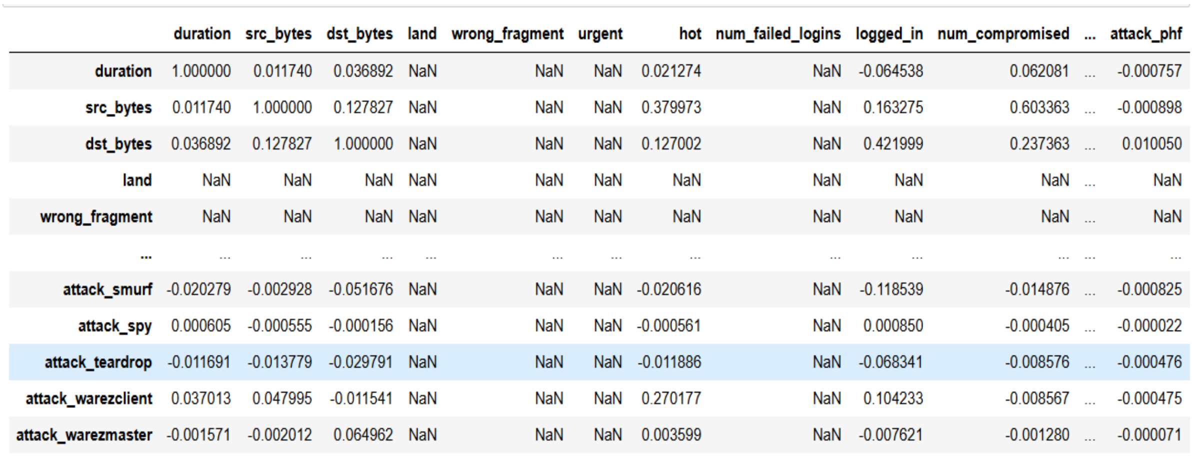
Explanation: The table presents the mean values of various network traffic features grouped by attack class. It highlights differences in attributes such as byte counts, login attempts, and error rates, providing insights into how different attack types behave compared to normal traffic.



	protocol_type	service	flag	attack
N	125972	125972	125972	125972
NMISS	0	0	0	0
ColumnsNames	tcp 102688 udp 14993 icmp 8291 N...	http 40338 private 21853 domain_u...	SF 74944 S0 34851 REJ 1123...	normal 67342 neptune 41...

Fig 7: Missing Value Analysis of Categorical Features

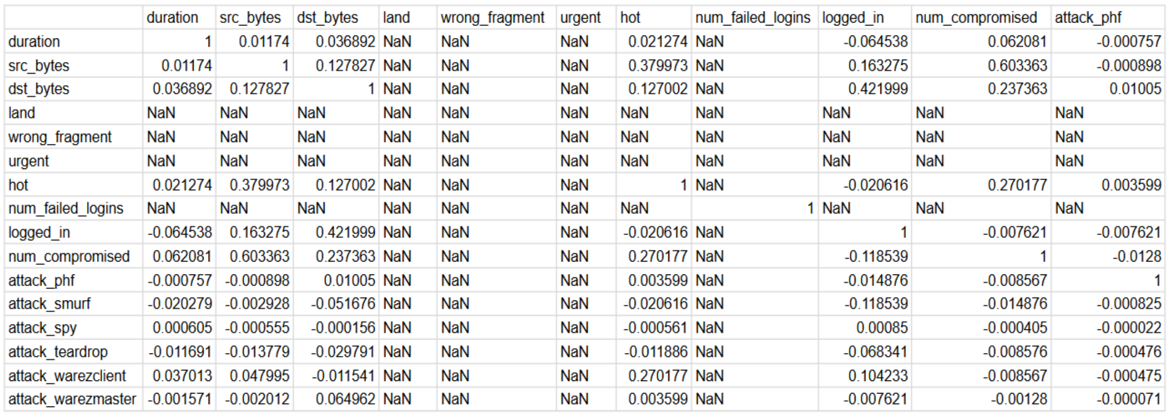
Explanation: The table summarizes the categorical variables in the dataset, showing that no missing values are present. It also provides the count of unique values for each category, confirming data completeness for analysis.



	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	...	attack_phf
duration	1.000000	0.011740	0.036892	NaN	NaN	NaN	0.021274	NaN	-0.064538	0.062081	...	-0.000757
src_bytes	0.011740	1.000000	0.127827	NaN	NaN	NaN	0.379973	NaN	0.163275	0.603363	...	-0.000898
dst_bytes	0.036892	0.127827	1.000000	NaN	NaN	NaN	0.127002	NaN	0.421999	0.237363	...	0.010050
land	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
wrong_fragment	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
attack_smurf	-0.020279	-0.002928	-0.051676	NaN	NaN	NaN	-0.020616	NaN	-0.118539	-0.014876	...	-0.000825
attack_spy	0.000605	-0.000555	-0.000156	NaN	NaN	NaN	-0.000561	NaN	0.000850	-0.000405	...	-0.000022
attack_teardrop	-0.011691	-0.013779	-0.029791	NaN	NaN	NaN	-0.011886	NaN	-0.068341	-0.008576	...	-0.000476
attack_warezclient	0.037013	0.047995	-0.011541	NaN	NaN	NaN	0.270177	NaN	0.104233	-0.008567	...	-0.000475
attack_warezmaster	-0.001571	-0.002012	0.064962	NaN	NaN	NaN	0.003599	NaN	-0.007621	-0.001280	...	-0.000071

Fig 8: Correlation Matrix Heatmap (Partial View) of Network Intrusion Dataset

Explanation: This figure shows a correlation matrix representing relationships between network traffic features and various attack types. Values range from -1 to 1, indicating negative to positive correlations, while NaN signifies missing data. It helps identify patterns and feature relevance in intrusion detection analysis.



	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	attack_phf
duration	1	0.01174	0.036892	NaN	NaN	NaN	0.021274	NaN	-0.064538	0.062081	-0.000757
src_bytes	0.01174	1	0.127827	NaN	NaN	NaN	0.379973	NaN	0.163275	0.603363	-0.000898
dst_bytes	0.036892	0.127827	1	NaN	NaN	NaN	0.127002	NaN	0.421999	0.237363	0.01005
land	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
wrong_fragment	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
urgent	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
hot	0.021274	0.379973	0.127002	NaN	NaN	NaN	1	NaN	-0.020616	0.270177	0.003599
num_failed_logins	NaN	NaN	NaN	NaN	NaN	NaN	NaN	1	NaN	NaN	NaN
logged_in	-0.064538	0.163275	0.421999	NaN	NaN	NaN	-0.020616	NaN	1	-0.007621	-0.007621
num_compromised	0.062081	0.603363	0.237363	NaN	NaN	NaN	0.270177	NaN	-0.118539	1	-0.0128
attack_phf	-0.000757	-0.000898	0.01005	NaN	NaN	NaN	0.003599	NaN	-0.014876	-0.008567	1
attack_smurf	-0.020279	-0.002928	-0.051676	NaN	NaN	NaN	-0.020616	NaN	-0.118539	-0.014876	-0.000825
attack_spy	0.000605	-0.000555	-0.000156	NaN	NaN	NaN	-0.000561	NaN	0.00085	-0.000405	-0.000022
attack_teardrop	-0.011691	-0.013779	-0.029791	NaN	NaN	NaN	-0.011886	NaN	-0.068341	-0.008576	-0.000476
attack_warezclient	0.037013	0.047995	-0.011541	NaN	NaN	NaN	0.270177	NaN	0.104233	-0.008567	-0.000475
attack_warezmaster	-0.001571	-0.002012	0.064962	NaN	NaN	NaN	0.003599	NaN	-0.007621	-0.00128	-0.000071

Table 1: Correlation Matrix of Network Traffic Features and Attack Types

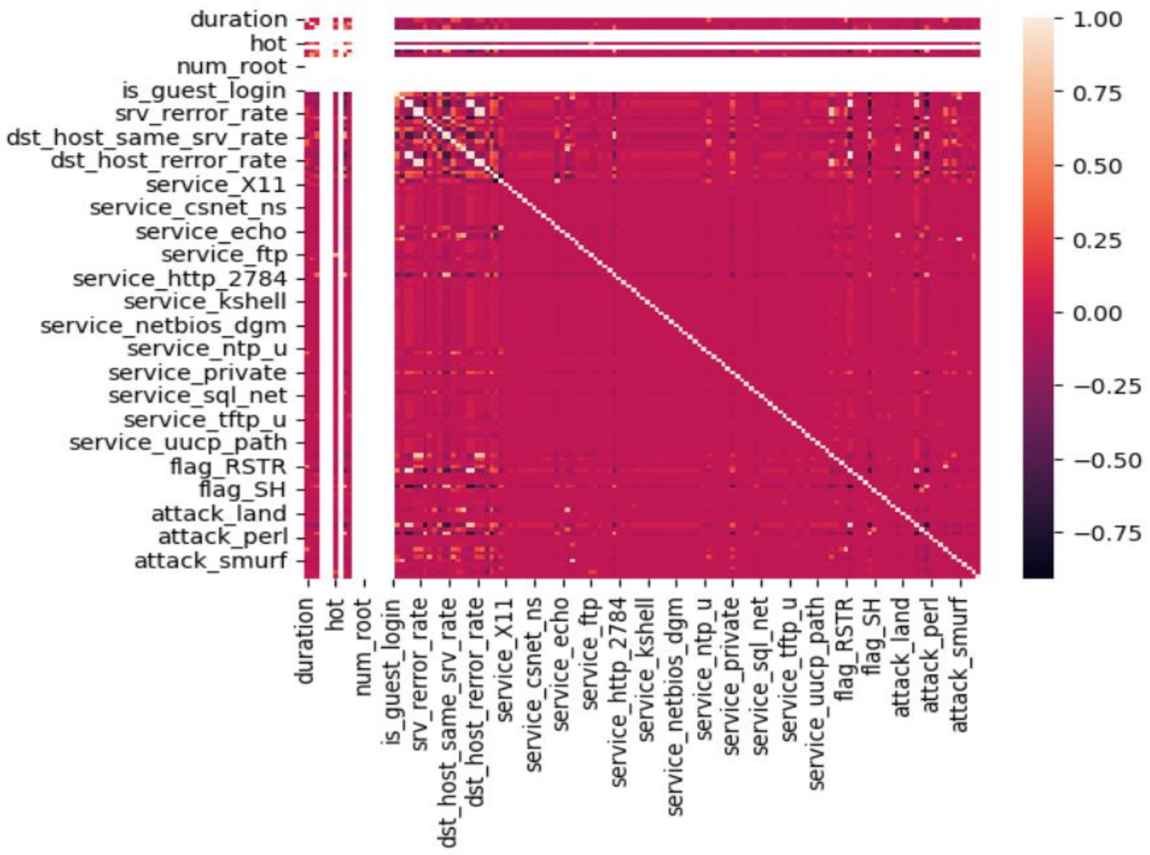


Fig 9: Heatmap of Feature Correlations

Explanation: The heatmap visualizes the correlation between different features in the dataset, with darker colors representing stronger negative correlations and lighter colors indicating stronger positive correlations. This helps in identifying highly correlated variables, which can be useful for feature selection and model optimization.

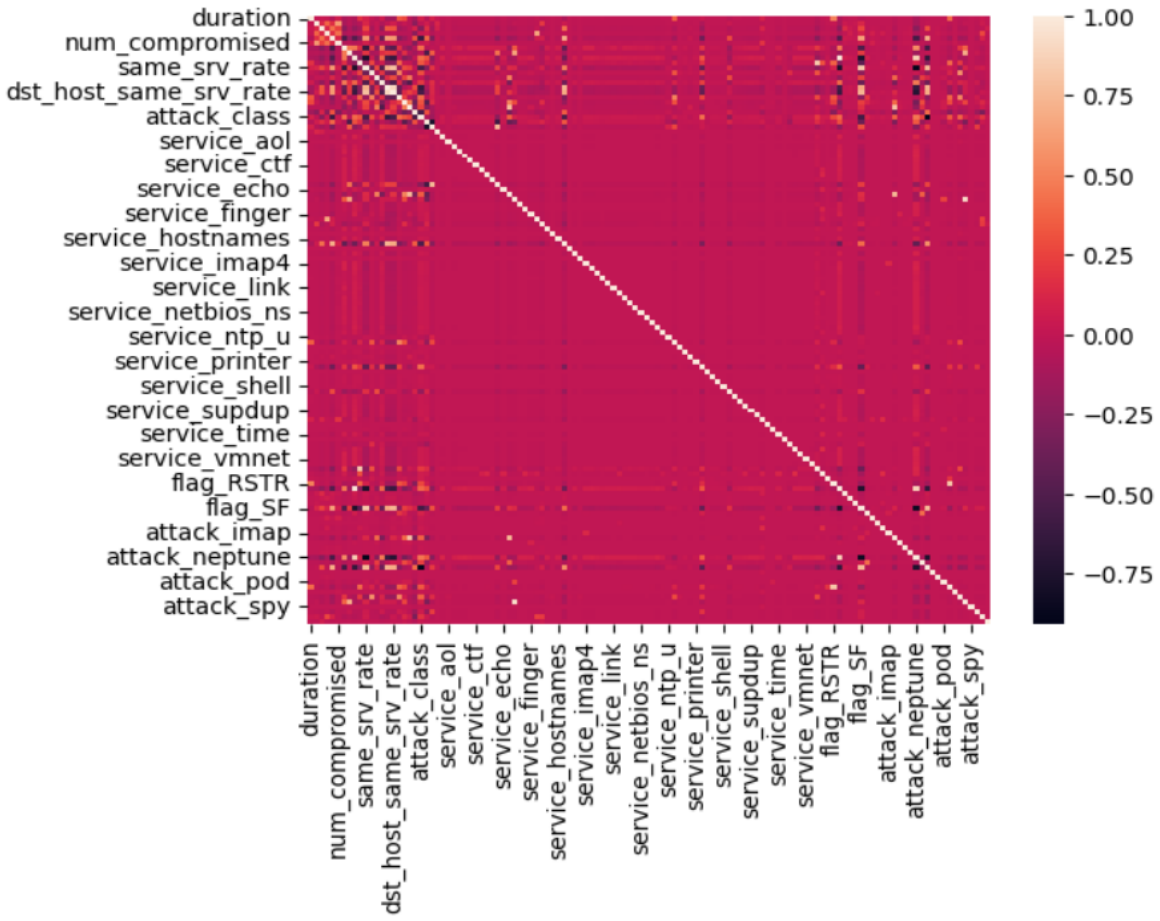
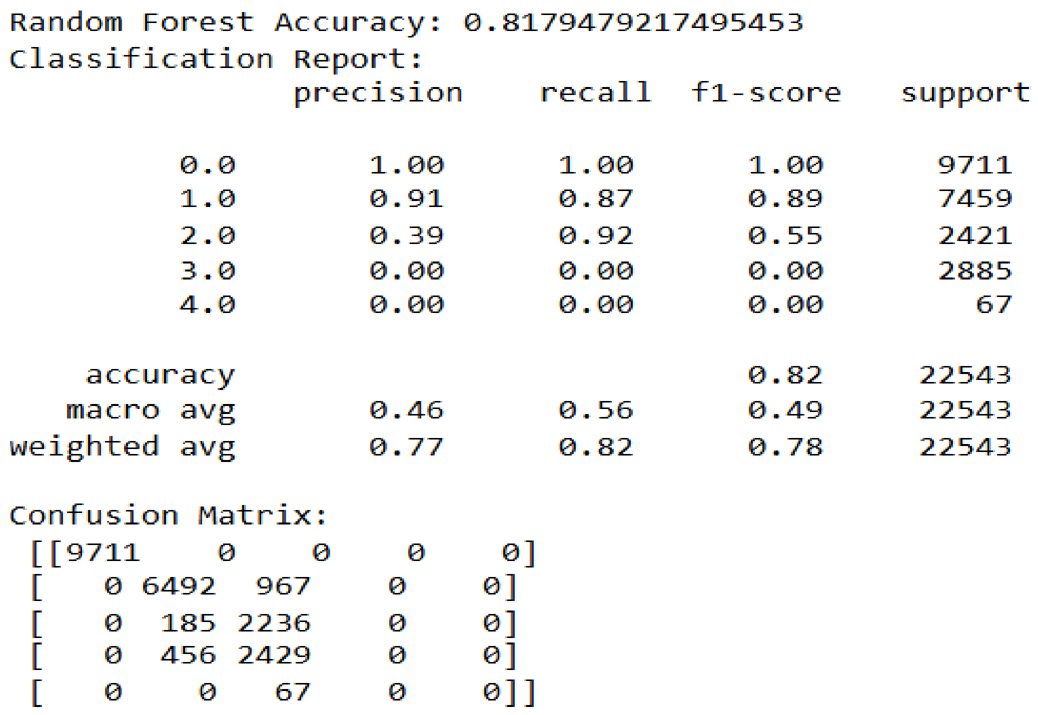


Fig 10: Refined Feature Correlation Heatmap

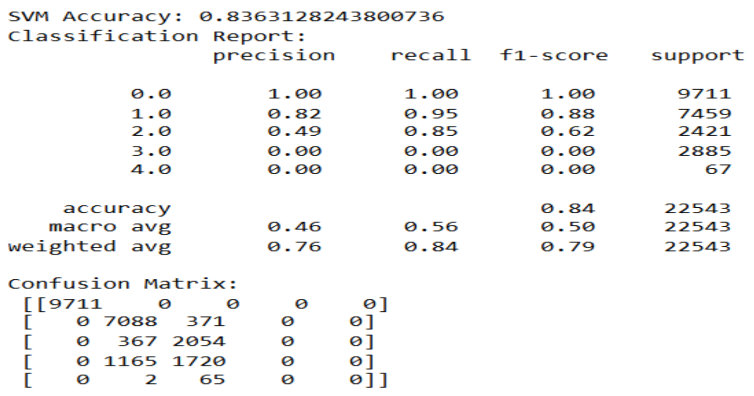
Explanation: This heatmap illustrates the correlation between features after removing less significant or redundant variables. It helps in better understanding the relationships among the remaining features, aiding in effective feature selection for model training.



```
Random Forest Accuracy: 0.8179479217495453
Classification Report:
               precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.91      0.87      0.89      7459
         2.0       0.39      0.92      0.55      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.82     22543
   macro avg       0.46      0.56      0.49     22543
weighted avg       0.77      0.82      0.78     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 6492  967    0    0]
 [   0  185 2236    0    0]
 [   0  456 2429    0    0]
 [   0    0   67    0    0]]
```

Fig 11: Performance Summary of Random Forest Model

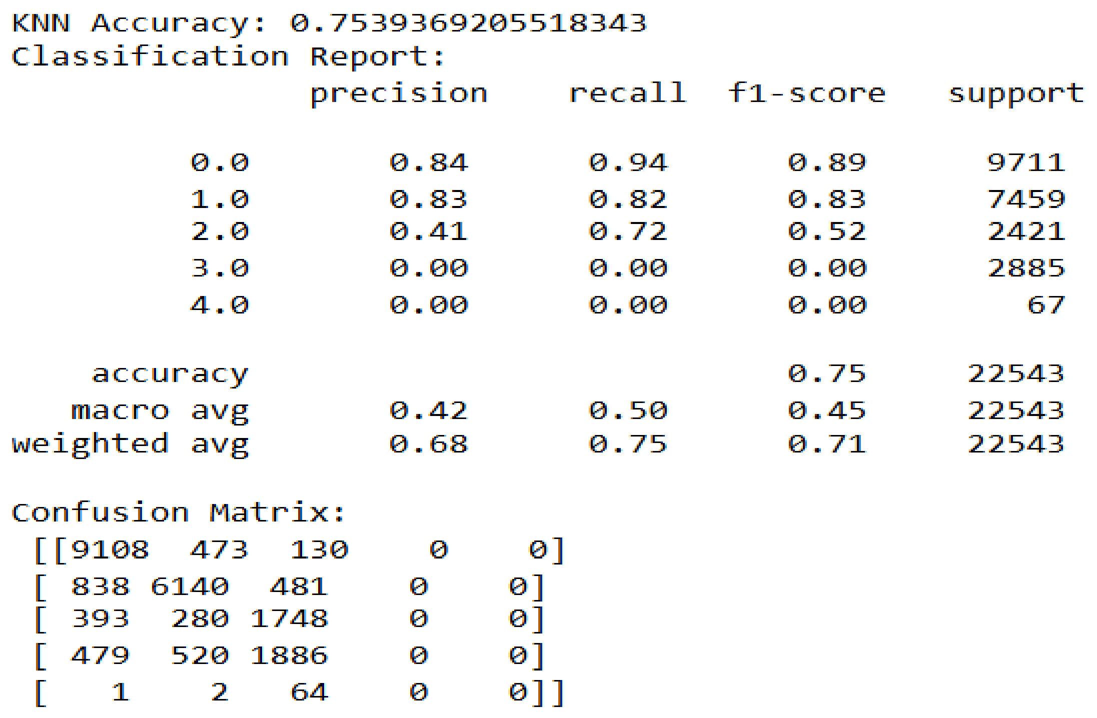
Explanation: This figure presents the evaluation results of a Random Forest model, including accuracy, classification metrics (precision, recall, F1-score), and the confusion matrix. The model achieves an overall accuracy of approximately 81.8%, demonstrating strong classification performance, particularly for the majority classes.



```
SVM Accuracy: 0.8363128243800736
Classification Report:
               precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.82      0.95      0.88      7459
         2.0       0.49      0.85      0.62      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.84     22543
   macro avg       0.46      0.56      0.50     22543
weighted avg       0.76      0.84      0.79     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 7088  371    0    0]
 [   0  367 2054    0    0]
 [   0 1165 1720    0    0]
 [   0    2   65    0    0]]
```

Fig 12: Performance Analysis of SVM Classifier

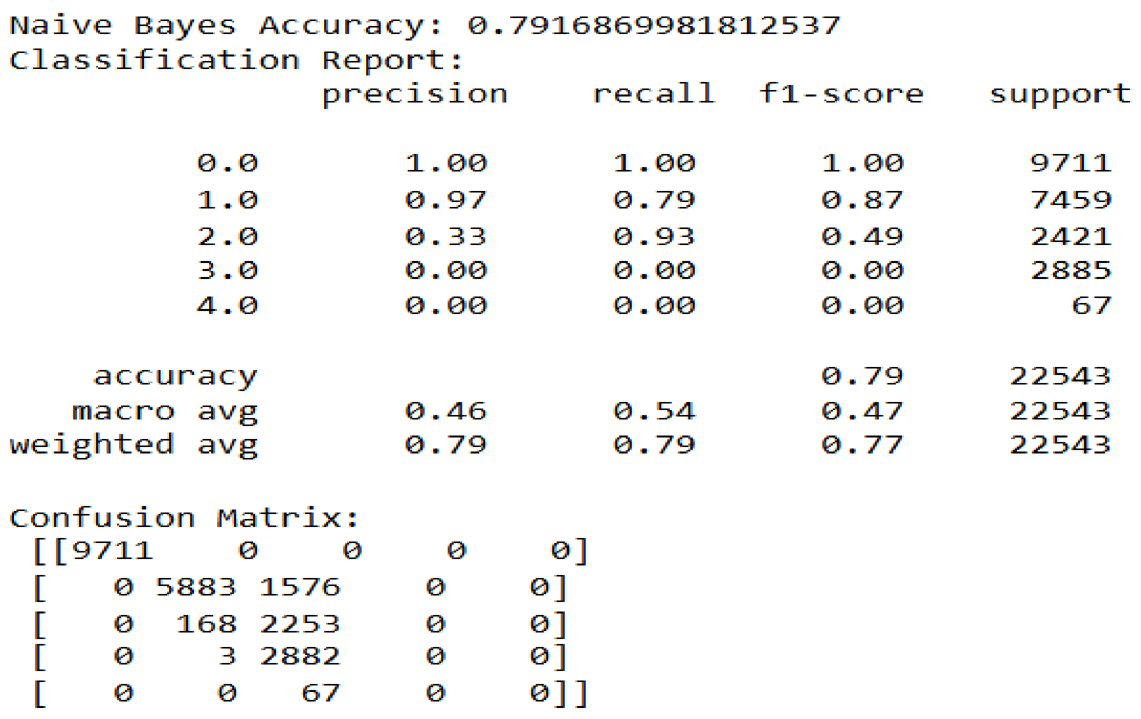
Explanation: The figure presents the performance evaluation of a Support Vector Machine (SVM) classifier using a linear kernel. It includes the accuracy score (83.6%), classification report, and confusion matrix, highlighting how well the model distinguishes different classes.



```
KNN Accuracy: 0.7539369205518343
Classification Report:
              precision    recall  f1-score   support

         0.0       0.84      0.94      0.89      9711
         1.0       0.83      0.82      0.83      7459
         2.0       0.41      0.72      0.52      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.75     22543
   macro avg       0.42      0.50      0.45     22543
weighted avg       0.68      0.75      0.71     22543

Confusion Matrix:
 [[9108  473  130    0    0]
 [ 838 6140  481    0    0]
 [ 393  280 1748    0    0]
 [ 479  520 1886    0    0]
 [   1    2   64    0    0]]
```

Fig 13: KNN Model Performance Evaluation

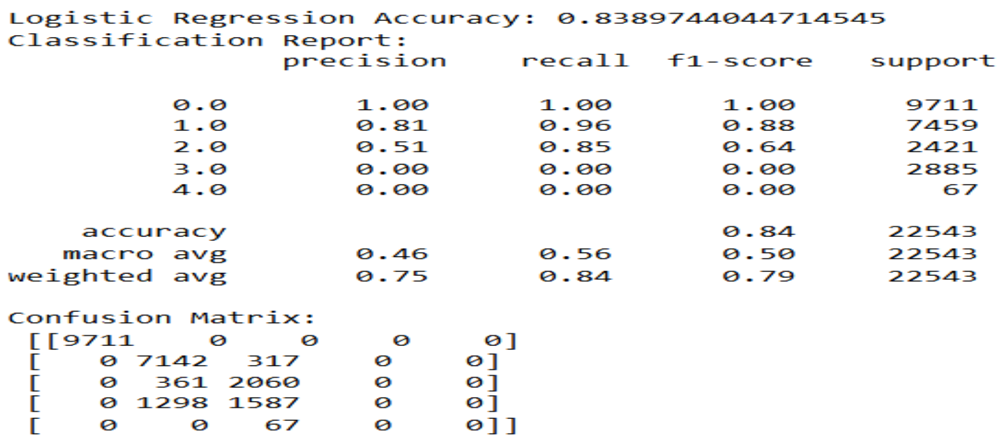
Explanation: The figure showcases the performance of a K-Nearest Neighbors (KNN) classifier with five neighbors, achieving an accuracy of 75.39%. It includes the classification report and confusion matrix, illustrating the model's effectiveness in classifying different categories.



```
Naive Bayes Accuracy: 0.7916869981812537
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.97      0.79      0.87      7459
         2.0       0.33      0.93      0.49      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.79     22543
   macro avg       0.46      0.54      0.47     22543
weighted avg       0.79      0.79      0.77     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 5883 1576    0    0]
 [   0  168 2253    0    0]
 [   0    3 2882    0    0]
 [   0    0   67    0    0]]
```

Fig 14: Naïve Bayes Classification Performance

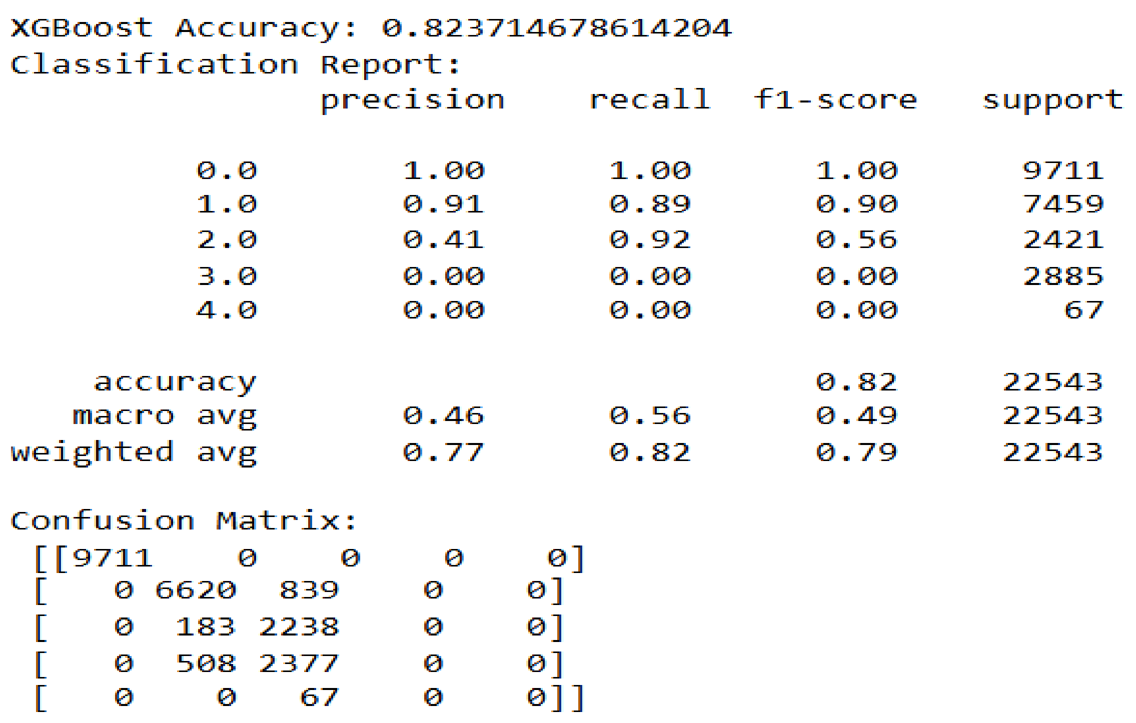
Explanation: This figure presents the performance of a Gaussian Naïve Bayes classifier, achieving an accuracy of 79.16%. The classification report and confusion matrix illustrate the model's predictive capability across different categories.



```
Logistic Regression Accuracy: 0.8389744044714545
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.81      0.96      0.88      7459
         2.0       0.51      0.85      0.64      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.84     22543
   macro avg       0.46      0.56      0.50     22543
weighted avg       0.75      0.84      0.79     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 7142  317    0    0]
 [   0  361 2060    0    0]
 [   0 1298 1587    0    0]
 [   0    0   67    0    0]]
```

Fig 15: Logistic Regression Model Performance

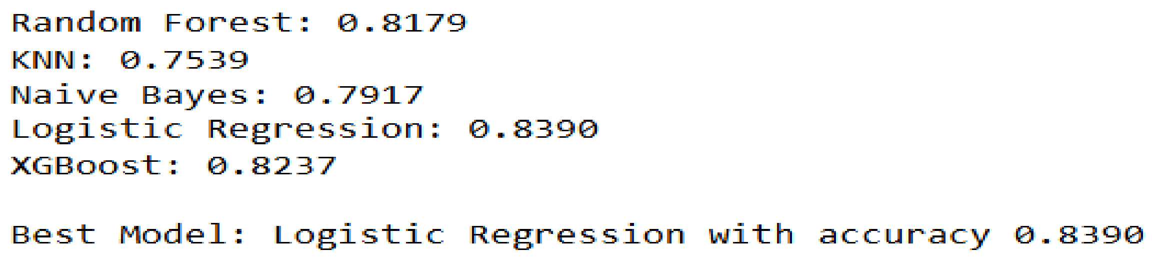
Explanation: The figure displays the evaluation of a Logistic Regression classifier, achieving an accuracy of 83.89%. The classification report and confusion matrix provide insights into the model’s precision, recall, and overall classification effectiveness.



```
XGBoost Accuracy: 0.823714678614204
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.91      0.89      0.90      7459
         2.0       0.41      0.92      0.56      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.82     22543
   macro avg       0.46      0.56      0.49     22543
weighted avg       0.77      0.82      0.79     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 6620  839    0    0]
 [   0  183 2238    0    0]
 [   0  508 2377    0    0]
 [   0    0   67    0    0]]
```

Fig 16: XGBoost Model Evaluation

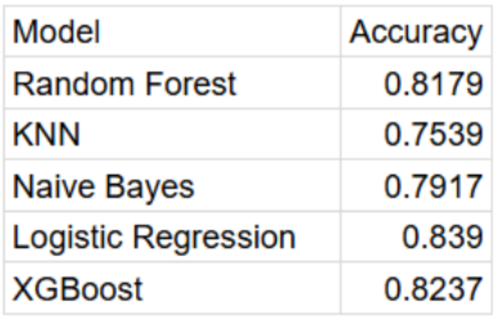
Explanation: This figure presents the performance of an XGBoost classifier, achieving an accuracy of 82.37%. The classification report and confusion matrix provide detailed insights into the model's predictive capabilities across different classes.



```
Random Forest: 0.8179
KNN: 0.7539
Naive Bayes: 0.7917
Logistic Regression: 0.8390
XGBoost: 0.8237

Best Model: Logistic Regression with accuracy 0.8390
```

Fig 17: Model Accuracy Comparison

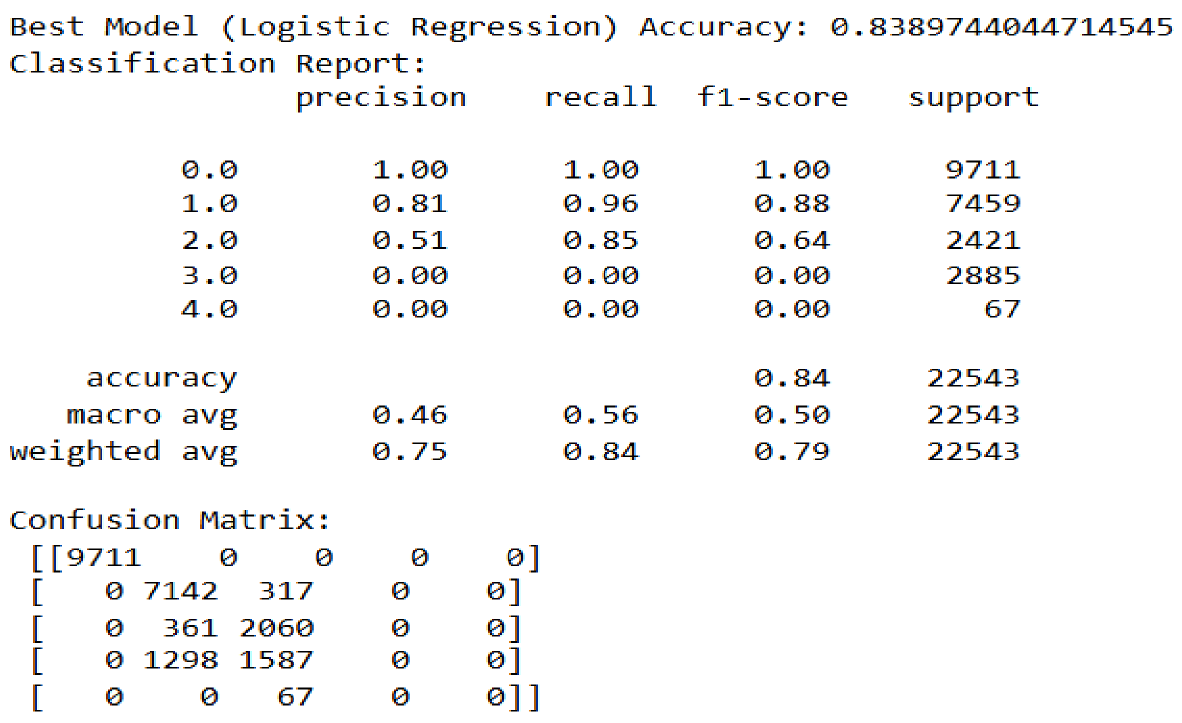
Explanation: This figure presents a comparison of the accuracy scores of multiple machine learning models, including Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, Logistic Regression, and XGBoost. Among them, Logistic Regression achieved the highest accuracy of 83.90%, making it the most effective model for the given dataset. This evaluation helps in identifying the best-performing model for classification tasks based on accuracy metrics.



Model	Accuracy
Random Forest	0.8179
KNN	0.7539
Naive Bayes	0.7917
Logistic Regression	0.839
XGBoost	0.8237

Table 2: Accuracy of Different Machine Learning Models

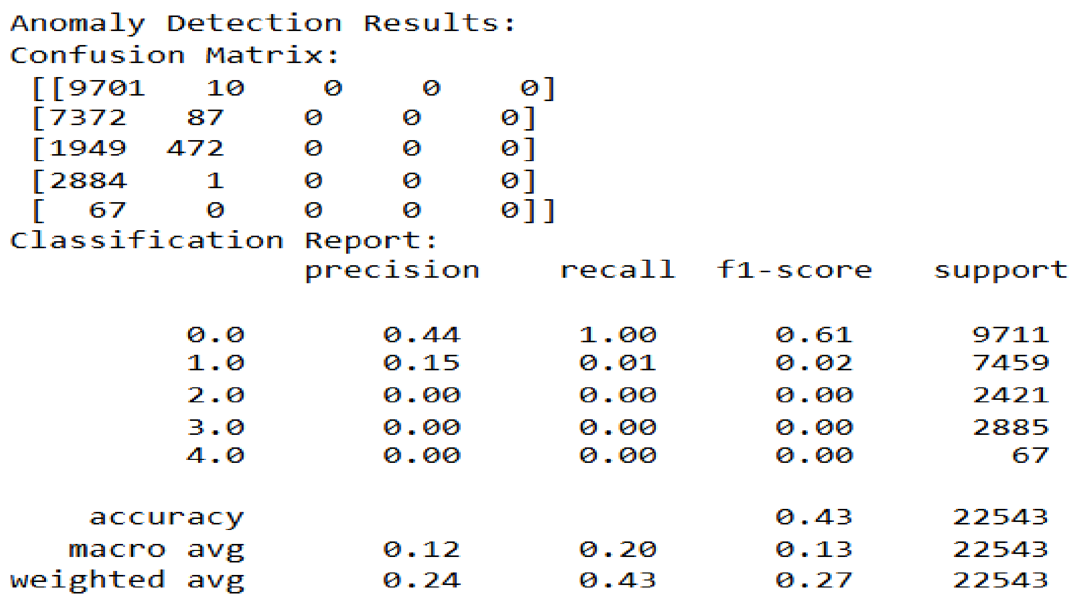
The table compares the accuracy of five machine learning models. Logistic Regression achieves the highest accuracy of 0.8390, making it the best-performing model. Other models, such as KNN and Naive Bayes, have lower accuracy, indicating their relatively weaker performance.



```
Best Model (Logistic Regression) Accuracy: 0.8389744044714545
Classification Report:
               precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      9711
         1.0       0.81      0.96      0.88      7459
         2.0       0.51      0.85      0.64      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.84     22543
   macro avg       0.46      0.56      0.50     22543
weighted avg       0.75      0.84      0.79     22543

Confusion Matrix:
 [[9711    0    0    0    0]
 [   0 7142  317    0    0]
 [   0  361 2060    0    0]
 [   0 1298 1587    0    0]
 [   0    0   67    0    0]]
```

Fig 18: Performance Evaluation of the Best Model

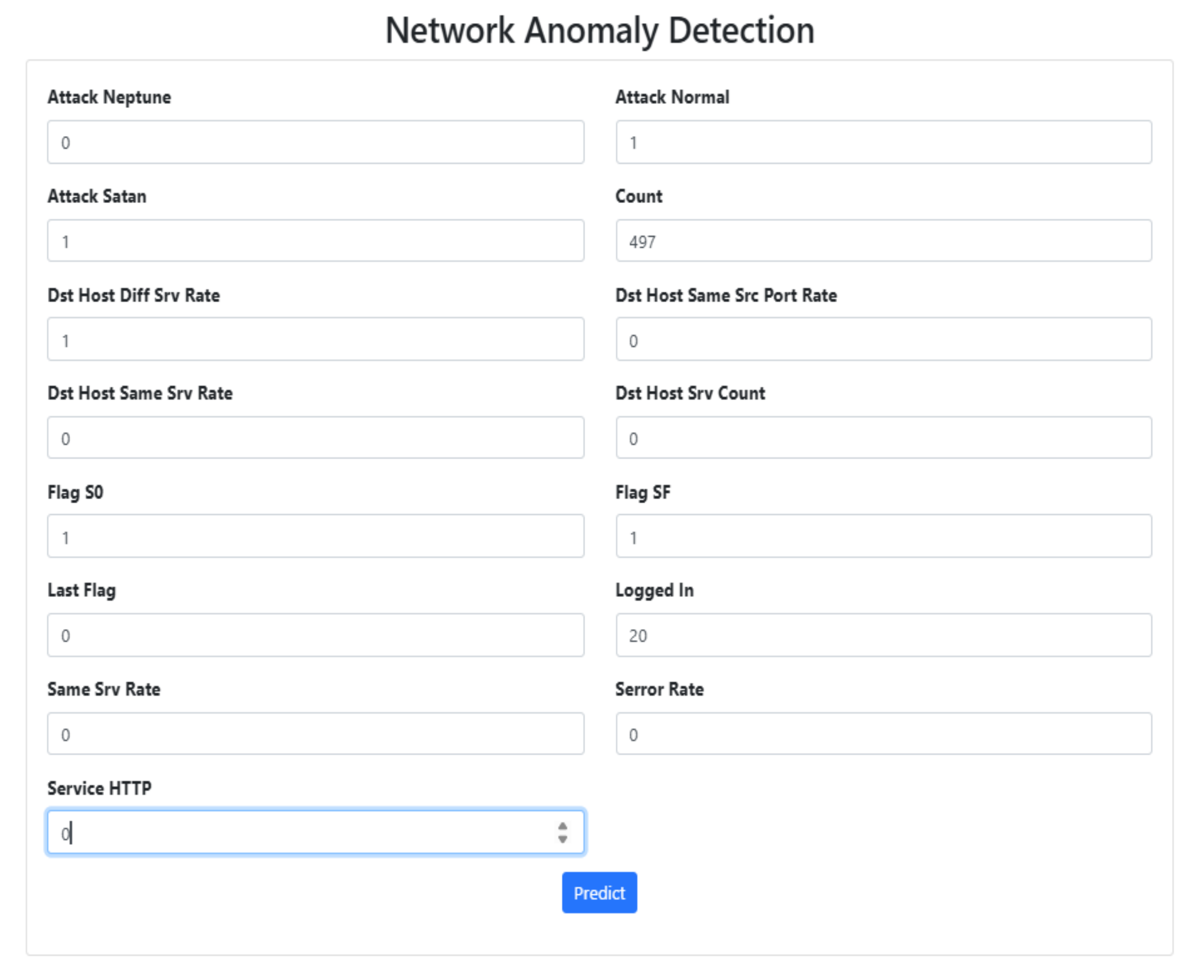
Explanation: This figure demonstrates the evaluation of the Logistic Regression model, which achieved the highest accuracy (83.89%) among all tested models. It includes the classification report and confusion matrix, highlighting the model's precision, recall, and F1-score for different classes, confirming its effectiveness in classification tasks.



```
Anomaly Detection Results:
Confusion Matrix:
 [[9701   10    0    0    0]
 [7372   87    0    0    0]
 [1949  472    0    0    0]
 [2884    1    0    0    0]
 [  67    0    0    0    0]]
Classification Report:
              precision    recall  f1-score   support

         0.0       0.44      1.00      0.61      9711
         1.0       0.15      0.01      0.02      7459
         2.0       0.00      0.00      0.00      2421
         3.0       0.00      0.00      0.00      2885
         4.0       0.00      0.00      0.00        67

    accuracy                           0.43     22543
   macro avg       0.12      0.20      0.13     22543
weighted avg       0.24      0.43      0.27     22543
```

Fig 19: Anomaly Detection Using Isolation Forest

Explanation: This figure presents the Isolation Forest model used for anomaly detection, where anomalies are identified based on deviations from normal patterns in the dataset. The classification report and confusion matrix illustrate the model’s performance, showing its ability to distinguish normal and anomalous data points



Network Anomaly Detection

Attack Neptune	Attack Normal
0	1
Attack Satan	Count
1	497
Dst Host Diff Srv Rate	Dst Host Same Src Port Rate
1	0
Dst Host Same Srv Rate	Dst Host Srv Count
0	0
Flag S0	Flag SF
1	1
Last Flag	Logged In
0	20
Same Srv Rate	Serror Rate
0	0
Service HTTP	
0	

Fig 20: Network Anomaly Detection Input Interface

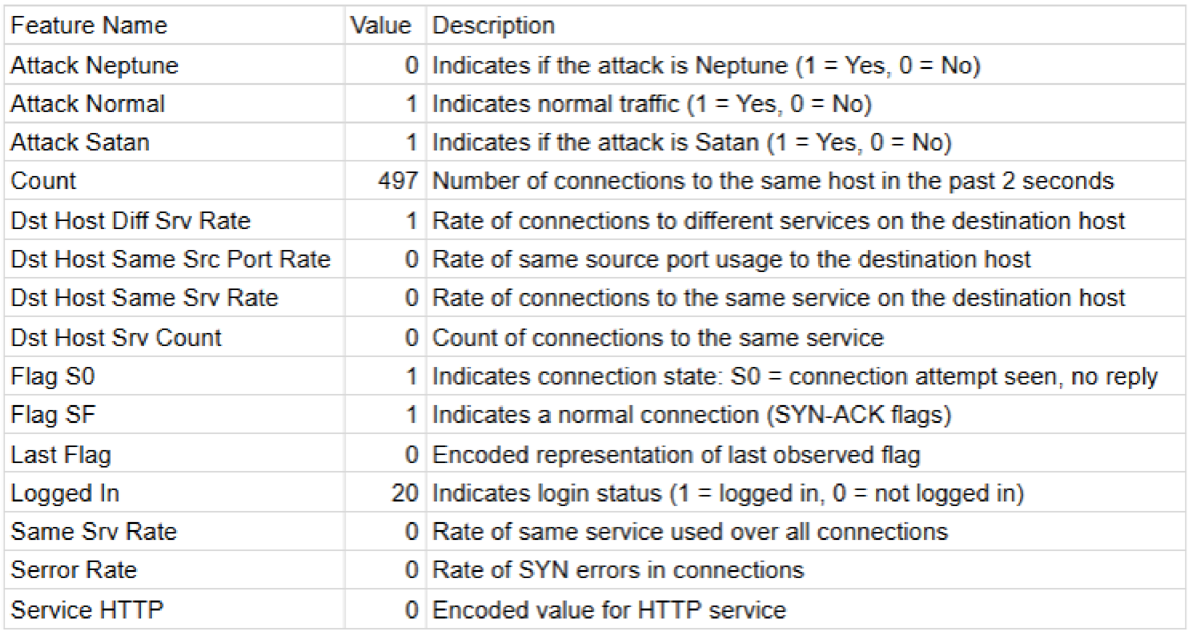
This figure displays the user interface for a Network Anomaly Detection System, where various network traffic parameters are input to predict potential anomalies or attacks. The form includes fields such as attack types, service types, connection flags, and statistical attributes related to network traffic behavior. After filling in the values, users can click the "Predict" button to classify network activity as normal or potentially malicious.

Overview of the Network Anomaly Detection Interface

The interface shown above is part of a **network anomaly detection system**, which is designed to identify unusual patterns in network traffic that could indicate a potential cyberattack or intrusion attempt. The form is built to take in a series of input features related to a network connection, analyze them, and predict whether the activity is normal or suspicious.

How It Works

This interface simulates the behavior of a machine learning model trained to detect network anomalies. Each field corresponds to a feature extracted from network traffic, typically based on packet behaviors and connection patterns over time. Once the form is filled out and the "Predict" button is clicked, the model processes the inputs and returns a classification — either **normal** or **potential threat**.

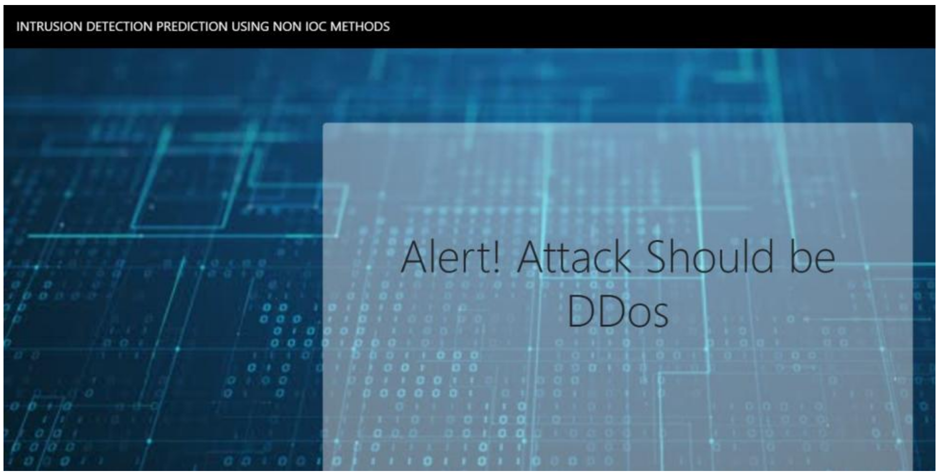


Feature Name	Value	Description
Attack Neptune	0	Indicates if the attack is Neptune (1 = Yes, 0 = No)
Attack Normal	1	Indicates normal traffic (1 = Yes, 0 = No)
Attack Satan	1	Indicates if the attack is Satan (1 = Yes, 0 = No)
Count	497	Number of connections to the same host in the past 2 seconds
Dst Host Diff Srv Rate	1	Rate of connections to different services on the destination host
Dst Host Same Src Port Rate	0	Rate of same source port usage to the destination host
Dst Host Same Srv Rate	0	Rate of connections to the same service on the destination host
Dst Host Srv Count	0	Count of connections to the same service
Flag S0	1	Indicates connection state: S0 = connection attempt seen, no reply
Flag SF	1	Indicates a normal connection (SYN-ACK flags)
Last Flag	0	Encoded representation of last observed flag
Logged In	20	Indicates login status (1 = logged in, 0 = not logged in)
Same Srv Rate	0	Rate of same service used over all connections
Serror Rate	0	Rate of SYN errors in connections
Service HTTP	0	Encoded value for HTTP service

Table 3: Input Feature Parameters for Network Anomaly Detection System

Category	Feature Name	Value	Description
Attack Information	Attack Neptune	0	Indicates if the attack is Neptune (1 = Yes, 0 = No)
	Attack Normal	1	Indicates normal traffic (1 = Yes, 0 = No)
	Attack Satan	1	Indicates if the attack is Satan (1 = Yes, 0 = No)
Connection Rates	Dst Host Diff Srv Rate	1	Rate of connections to different services on the destination host
	Dst Host Same Src Port Rate	0	Rate of same source port usage to the destination host
	Dst Host Same Srv Rate	0	Rate of connections to the same service on the destination host
	Same Srv Rate	0	Rate of same service used over all connections
	Serror Rate	0	Rate of SYN errors in connections
Connection Counts	Count	497	Number of connections to the same host in the past 2 seconds
	Dst Host Srv Count	0	Count of connections to the same service
Flags & Status	Flag S0	1	Indicates connection state: S0 = connection attempt seen, no reply
	Flag SF	1	Indicates a normal connection (SYN-ACK flags)
	Last Flag	0	Encoded representation of last observed flag
	Logged In	20	Indicates login status (1 = logged in, 0 = not logged in)
Service	Service HTTP	0	Encoded value for HTTP service

Table 4: Categorized Input Features for Network Anomaly Detection System

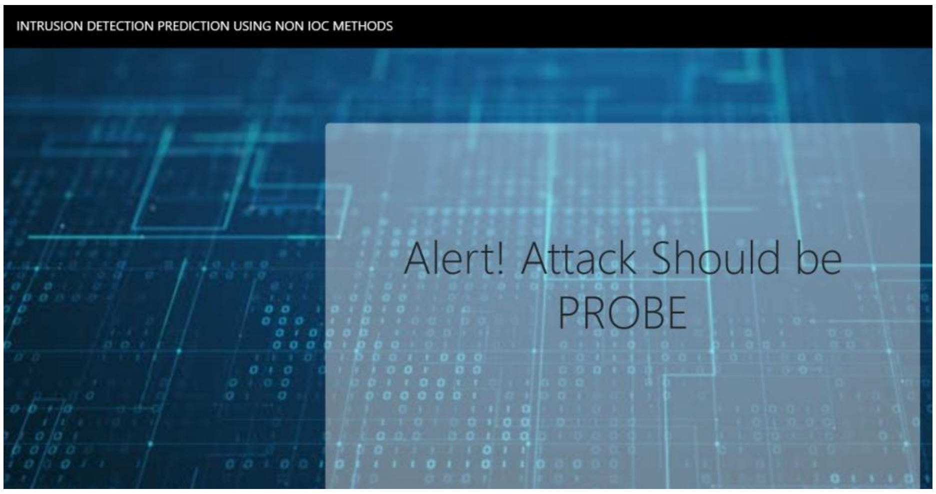


Alert! Attack Should be DDos

Fig 21: Intrusion Detection System for DDoS Attack

This figure represents the alert interface of an Intrusion Detection System (IDS) that identifies network threats using Non-Indicator of Compromise (IoC) methods.

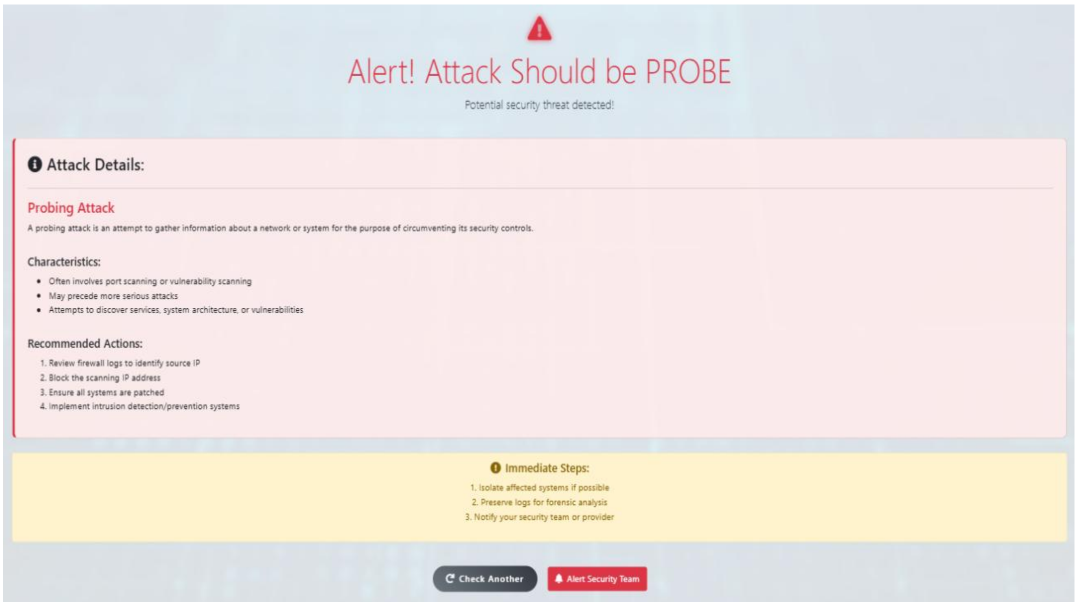
The system has detected an anomaly in network traffic and classified the attack as a DDoS (Distributed Denial of Service) attack. The alert message informs the user about the potential threat, enabling further investigation and mitigation actions.



Alert! Attack Should be PROBE

Fig 22: Intrusion Detection System for Probe Attack

This figure displays the alert interface of an Intrusion Detection System (IDS) that uses Non-Indicator of Compromise (IoC) methods to detect network anomalies. The system has identified suspicious network activity and classified it as a Probe attack, which typically involves scanning a network to gather information about its structure and vulnerabilities.



Alert! Attack Should be PROBE

Potential security threat detected!

Attack Details:

Probing Attack

A probing attack is an attempt to gather information about a network or system for the purpose of circumventing its security controls.

Characteristics:

- Often involves port scanning or vulnerability scanning
- May precede more serious attacks
- Attempts to discover services, system architecture, or vulnerabilities

Recommended Actions:

1. Review firewall logs to identify source IP
2. Block the scanning IP address
3. Ensure all systems are patched
4. Implement intrusion detection/prevention systems

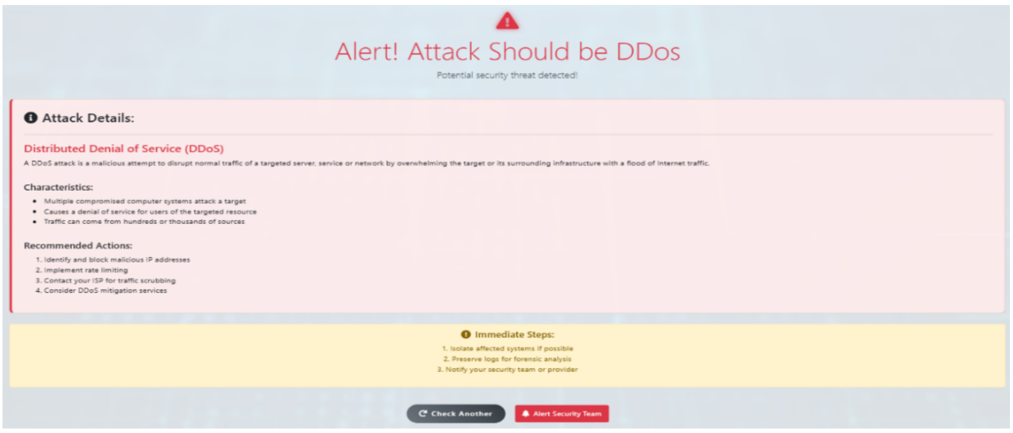
Immediate Steps:

1. Isolate affected systems if possible
2. Preserve logs for forensic analysis
3. Notify your security team or provider

Fig 23: Probing Attack Detection Alert Interface

Explanation:

This interface displays an alert indicating a detected **Probing Attack**, which involves scanning for vulnerabilities in a network. It outlines the attack characteristics, recommended mitigation steps, and immediate actions for security response.



Alert! Attack Should be DDos

Potential security threat detected!

Attack Details:

Distributed Denial of Service (DDoS)

A DDoS attack is a malicious attempt to disrupt normal traffic of a targeted server, service or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic.

Characteristics:

- Multiple compromised computer systems attack a target
- Causes a denial of service for users of the targeted resource
- Traffic can come from hundreds or thousands of sources

Recommended Actions:

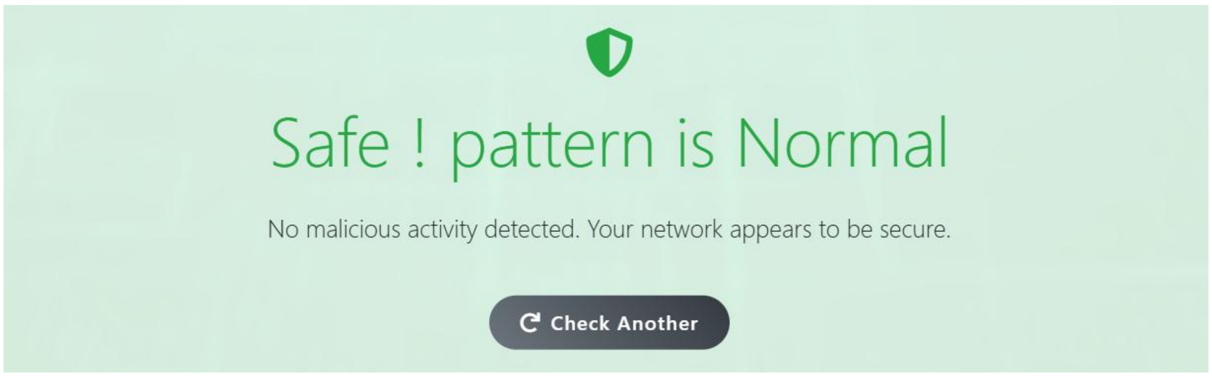
1. Identify and block malicious IP addresses
2. Implement rate limiting
3. Contact your ISP for traffic scrubbing
4. Consider DDoS mitigation services

Immediate Steps:

1. Isolate affected systems if possible
2. Preserve logs for forensic analysis
3. Notify your security team or provider

Fig 24: DDoS Attack Detection Alert Interface

Explanation: This alert screen identifies a **Distributed Denial of Service (DDoS)** attack, where multiple systems flood a target with traffic to disrupt services. It provides key characteristics, recommended countermeasures, and immediate response steps for containment.


Safe ! pattern is Normal

No malicious activity detected. Your network appears to be secure.

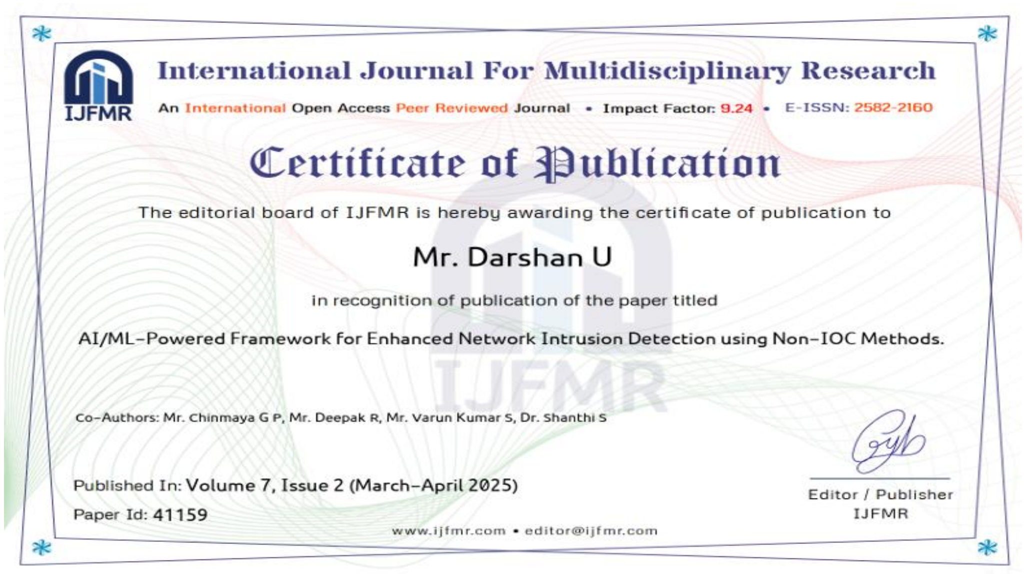
Fig 25: Normal Traffic Pattern Confirmation Interface

Explanation: This screen indicates that the network traffic is **normal** and no malicious activity has been detected. It reassures users that the network is currently **secure and safe** from threats.

APPENDIX-C

ENCLOSURES

1. Journal publication/Conference Paper Presented Certificates of all students.



International Journal For Multidisciplinary Research

An International Open Access Peer Reviewed Journal • Impact Factor: 9.24 • E-ISSN: 2582-2160

Certificate of Publication

The editorial board of IJFMR is hereby awarding the certificate of publication to

Mr. Darshan U

in recognition of publication of the paper titled

AI/ML-Powered Framework for Enhanced Network Intrusion Detection using Non-IOC Methods.

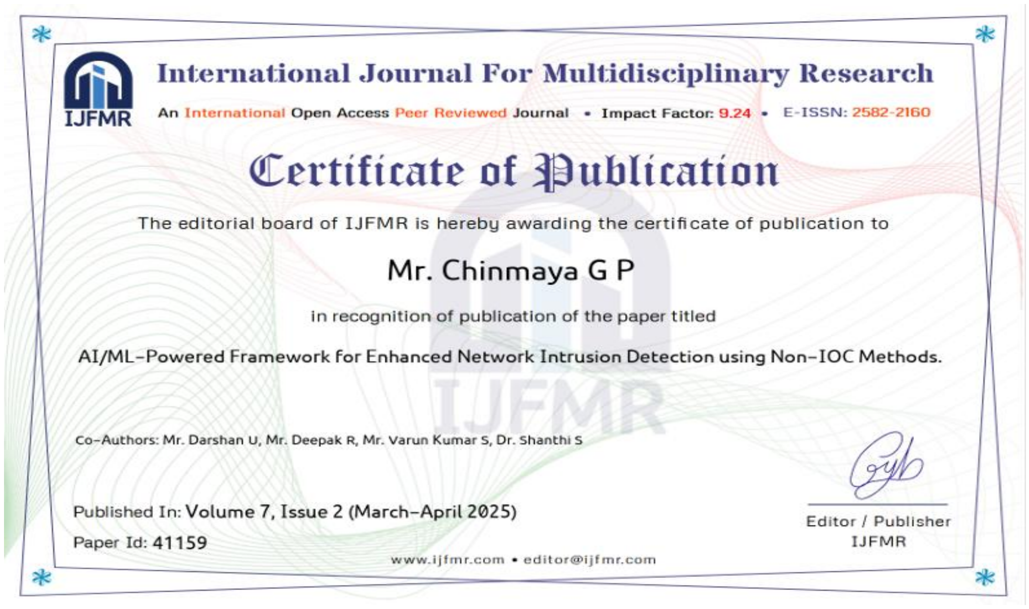
Co-Authors: Mr. Chinmaya G P, Mr. Deepak R, Mr. Varun Kumar S, Dr. Shanthi S

Published In: Volume 7, Issue 2 (March-April 2025)

Paper Id: 41159

Editor / Publisher
IJFMR

www.ijfmr.com • editor@ijfmr.com



International Journal For Multidisciplinary Research

An International Open Access Peer Reviewed Journal • Impact Factor: 9.24 • E-ISSN: 2582-2160

Certificate of Publication

The editorial board of IJFMR is hereby awarding the certificate of publication to

Mr. Chinmaya G P

in recognition of publication of the paper titled

AI/ML-Powered Framework for Enhanced Network Intrusion Detection using Non-IOC Methods.

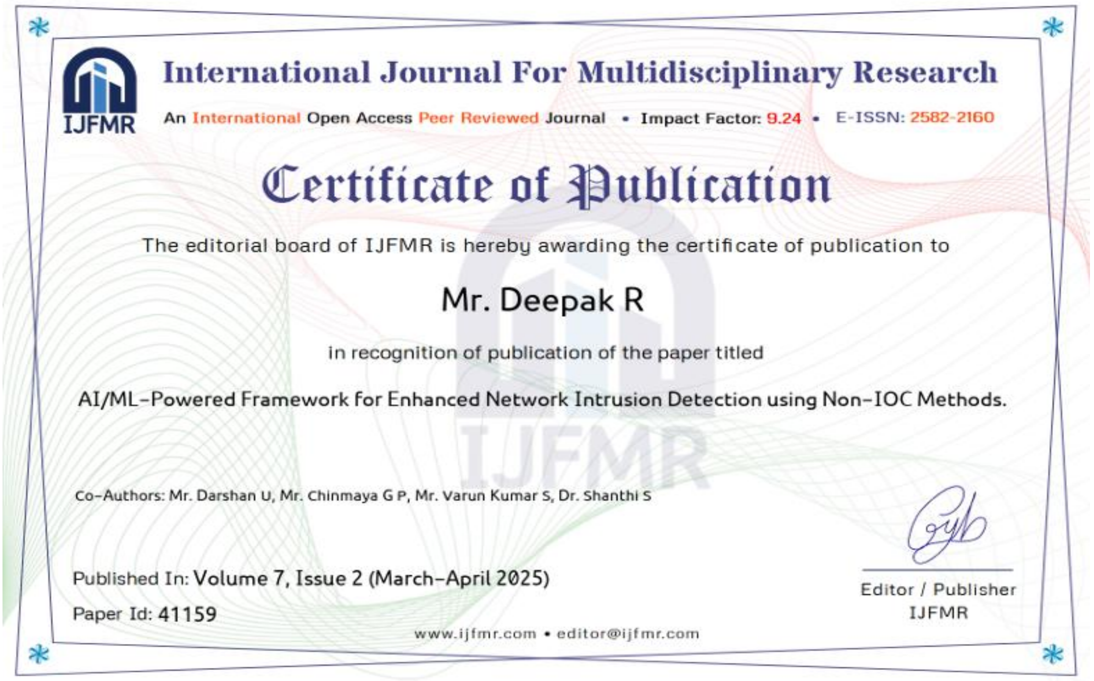
Co-Authors: Mr. Darshan U, Mr. Deepak R, Mr. Varun Kumar S, Dr. Shanthi S

Published In: Volume 7, Issue 2 (March-April 2025)

Paper Id: 41159

Editor / Publisher
IJFMR

www.ijfmr.com • editor@ijfmr.com



IJFMR

International Journal For Multidisciplinary Research

An International Open Access Peer Reviewed Journal • Impact Factor: 9.24 • E-ISSN: 2582-2160

Certificate of Publication

The editorial board of IJFMR is hereby awarding the certificate of publication to

Mr. Deepak R

in recognition of publication of the paper titled

AI/ML-Powered Framework for Enhanced Network Intrusion Detection using Non-IOC Methods.

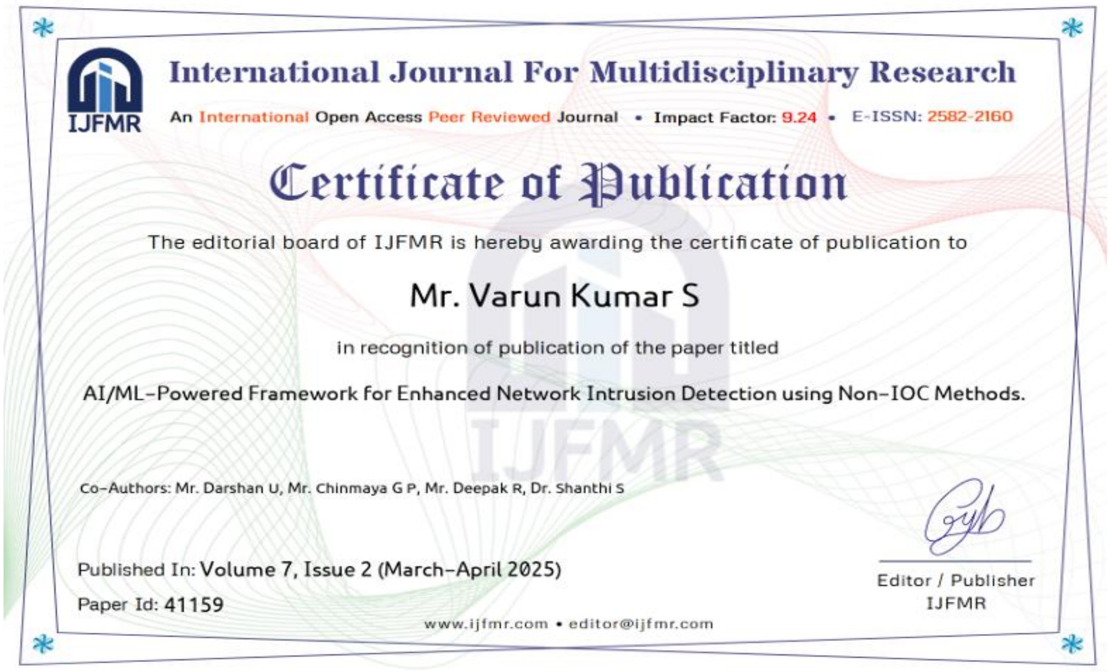
Co-Authors: Mr. Darshan U, Mr. Chinmaya G P, Mr. Varun Kumar S, Dr. Shanthi S

Published In: Volume 7, Issue 2 (March-April 2025)

Paper Id: 41159

Editor / Publisher
IJFMR

www.ijfmr.com • editor@ijfmr.com



IJFMR

International Journal For Multidisciplinary Research

An International Open Access Peer Reviewed Journal • Impact Factor: 9.24 • E-ISSN: 2582-2160

Certificate of Publication

The editorial board of IJFMR is hereby awarding the certificate of publication to

Mr. Varun Kumar S

in recognition of publication of the paper titled

AI/ML-Powered Framework for Enhanced Network Intrusion Detection using Non-IOC Methods.

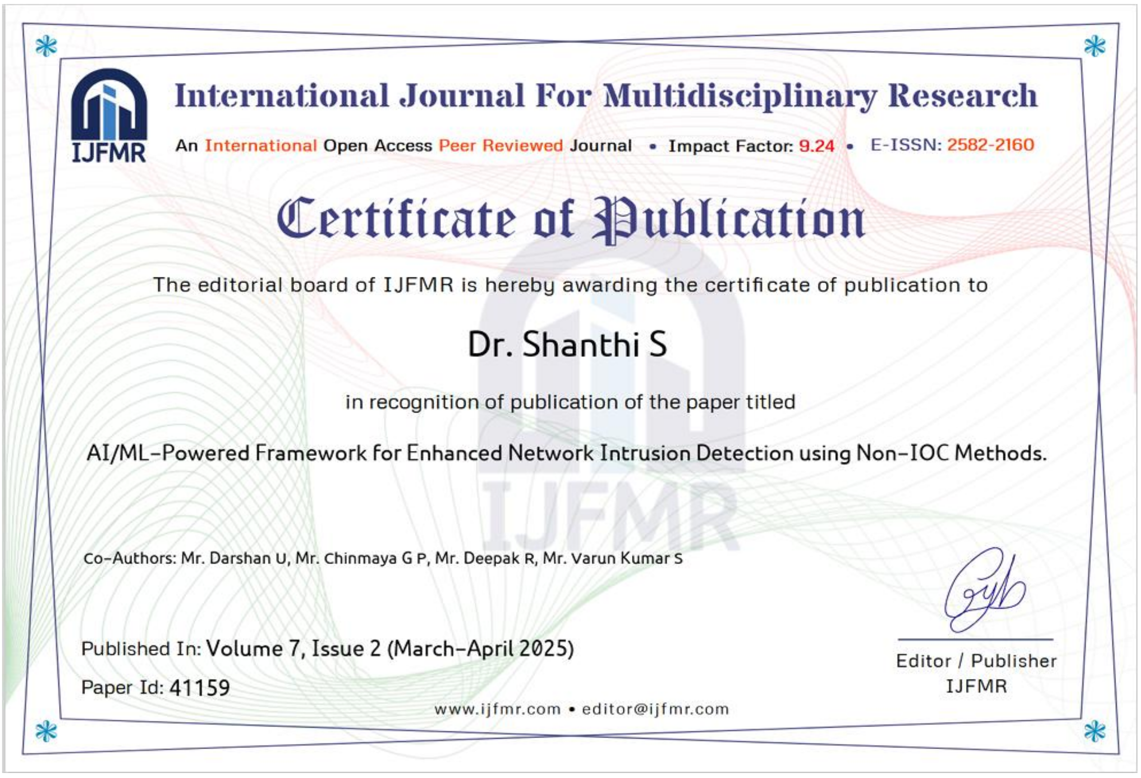
Co-Authors: Mr. Darshan U, Mr. Chinmaya G P, Mr. Deepak R, Dr. Shanthi S

Published In: Volume 7, Issue 2 (March-April 2025)

Paper Id: 41159

Editor / Publisher
IJFMR

www.ijfmr.com • editor@ijfmr.com



International Journal For Multidisciplinary Research

An International Open Access Peer Reviewed Journal • Impact Factor: 9.24 • E-ISSN: 2582-2160

Certificate of Publication

The editorial board of IJFMR is hereby awarding the certificate of publication to

Dr. Shanthi S

in recognition of publication of the paper titled

AI/ML-Powered Framework for Enhanced Network Intrusion Detection using Non-IOC Methods.

Co-Authors: Mr. Darshan U, Mr. Chinmaya G P, Mr. Deepak R, Mr. Varun Kumar S

Published In: Volume 7, Issue 2 (March-April 2025)

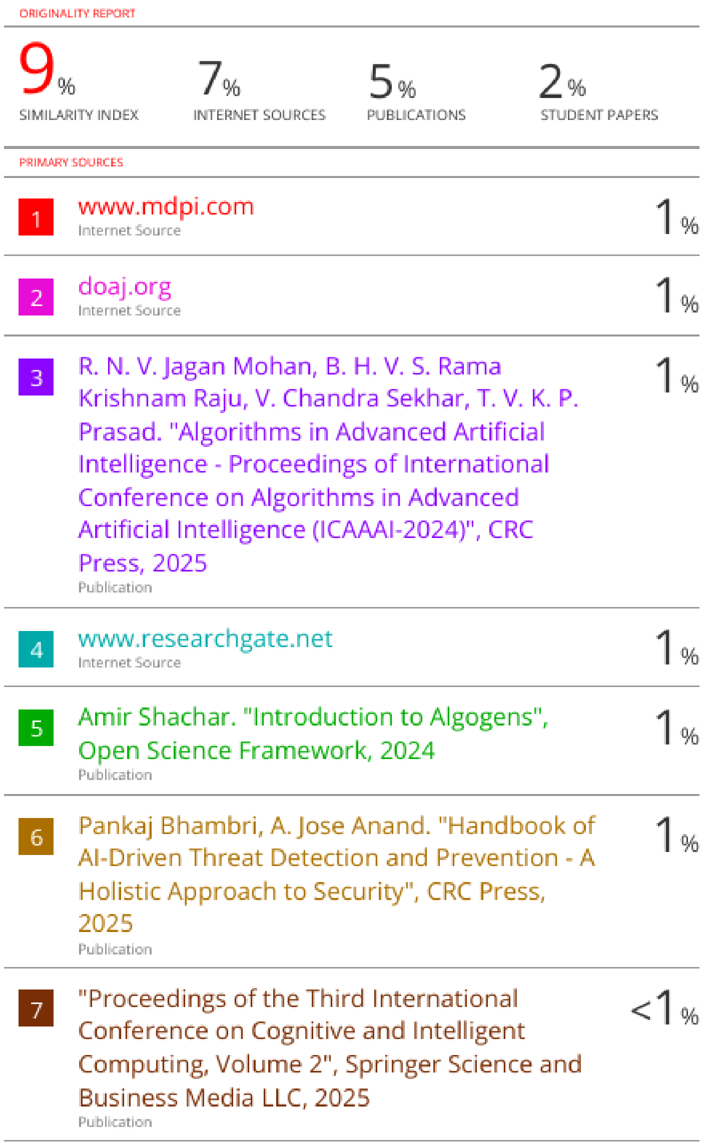
Paper Id: 41159

[illegible]

Editor / Publisher
IJFMR

www.ijfmr.com • editor@ijfmr.com

2. Similarity Index / Plagiarism Check report clearly showing the Percentage (%). No need for a page-wise explanation.



ORIGINALITY REPORT

9%	7%	5%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.mdpi.com Internet Source	1%
2	doaj.org Internet Source	1%
3	R. N. V. Jagan Mohan, B. H. V. S. Rama Krishnam Raju, V. Chandra Sekhar, T. V. K. P. Prasad. "Algorithms in Advanced Artificial Intelligence - Proceedings of International Conference on Algorithms in Advanced Artificial Intelligence (ICAAAI-2024)", CRC Press, 2025 Publication	1%
4	www.researchgate.net Internet Source	1%
5	Amir Shachar. "Introduction to Algogens", Open Science Framework, 2024 Publication	1%
6	Pankaj Bhambri, A. Jose Anand. "Handbook of AI-Driven Threat Detection and Prevention - A Holistic Approach to Security", CRC Press, 2025 Publication	1%
7	"Proceedings of the Third International Conference on Cognitive and Intelligent Computing, Volume 2", Springer Science and Business Media LLC, 2025 Publication	<1%

3. Details of mapping the project with the Sustainable Development Goals (SDGs).

SDG MAPPING

1 NO POVERTY
2 ZERO HUNGER
3 GOOD HEALTH AND WELL-BEING
4 QUALITY EDUCATION
5 GENDER EQUALITY
6 CLEAN WATER AND SANITATION
7 AFFORDABLE AND CLEAN ENERGY
8 DECENT WORK AND ECONOMIC GROWTH
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE
10 REDUCED INEQUALITIES
11 SUSTAINABLE CITIES AND COMMUNITIES
12 RESPONSIBLE CONSUMPTION AND PRODUCTION
13 CLIMATE ACTION
14 LIFE BELOW WATER
15 LIFE ON LAND
16 PEACE, JUSTICE AND STRONG INSTITUTIONS
17 PARTNERSHIPS FOR THE GOALS
SUSTAINABLE DEVELOPMENT GOALS

How My Project Aligns with the Sustainable Development Goals

Cybersecurity has become a vital foundation for the functioning of modern economies, governments, and societies. In recognition of this reality, the United Nations' Sustainable Development Goals (SDGs) emphasize the importance of resilient infrastructure, peace, and innovation. My project, AI/ML-Powered Framework for Enhanced Network Intrusion Detection Using Non-IOC Methods, is a forward-looking initiative that directly supports these global objectives. By focusing on proactive, behavior-based threat detection rather than relying solely on known indicators of compromise, it offers an innovative solution to the escalating cybersecurity challenges of our digital age.

Alignment with SDG 9: Industry, Innovation, and Infrastructure

At the core of my project is the goal of strengthening digital infrastructure through innovative technology, aligning closely with **SDG 9**, which focuses on building resilient infrastructure, promoting inclusive and sustainable industrialization, and fostering innovation.

Target 9.1: Develop Quality, Reliable, Sustainable, and Resilient Infrastructure

Secure digital infrastructure is critical to the modern world. My project enhances the resilience of networks against cyber threats by introducing AI and machine learning techniques that can detect anomalies even when traditional threat signatures are absent. This proactive approach ensures that vital industries, businesses, and services can operate securely and sustainably in an increasingly digitized environment.

Target 9.5: Enhance Scientific Research and Upgrade Technological Capabilities

My project leverages cutting-edge research in AI and cybersecurity to create a novel method of detecting intrusions. By pushing the boundaries of traditional cybersecurity frameworks, it not only upgrades existing technological capabilities but also fosters new avenues for scientific exploration, supporting innovation at multiple levels.

Alignment with SDG 16: Peace, Justice, and Strong Institutions

Trust in institutions relies heavily on the security of the systems that support them. My project contributes directly to **SDG 16**, which seeks to promote peaceful and inclusive societies, provide access to justice for all, and build effective, accountable, and inclusive institutions at all levels.

Target 16.1: Significantly Reduce All Forms of Violence and Related Death Rates Everywhere

While cyberattacks may not always involve physical violence, the damage they inflict on economies, personal privacy, and national security can be profound. By strengthening defenses against sophisticated cyber threats, my project helps reduce the potential for digital violence and disruption, fostering greater safety in the online and offline worlds.

Target 16.6: Develop Effective, Accountable, and Transparent Institutions

Cybersecurity is a foundation for institutional accountability and trust. My project enhances the transparency and effectiveness of organizations by ensuring that their digital operations remain uncompromised, thus upholding public trust in critical services such as healthcare, education, governance, and finance.

Broader Impact on Related SDGs

While my project primarily aligns with SDG 9 and SDG 16, its impact extends into other important areas:

SDG 8: Decent Work and Economic Growth

A secure digital environment is crucial for enabling economic activity. By preventing cyber disruptions, my project supports the uninterrupted growth of businesses and digital entrepreneurship, aligning with SDG 8's goals of promoting sustained, inclusive, and sustainable economic growth.

SDG 4: Quality Education

By contributing research and developing new AI/ML models, my project creates learning opportunities for future professionals in cybersecurity and data science.

In doing so, it supports SDG 4's target of increasing the number of youth and adults with relevant skills for decent jobs and entrepreneurship.

Human-Centric Impact

At its heart, my project addresses a deeply human need: the right to security in an increasingly digital world. Cyberattacks can disrupt hospitals, compromise public services, and endanger individuals' private data. By developing smarter and more proactive defense mechanisms, my project not only protects systems but safeguards people's lives, rights, and livelihoods.
Moreover, the project's non-IOC approach makes it particularly valuable during unpredictable, large-scale events such as new forms of cyberwarfare or unknown malware outbreaks. By detecting and responding to abnormal behavior patterns, it ensures that defenses remain robust even against novel threats, helping societies to remain resilient and adaptable in the face of emerging digital risks.

Conclusion

My project is a clear demonstration of how innovative technology can be used to address some of the most urgent security challenges of the digital era. By proactively enhancing network security, it advances the core principles of **SDG 9** and **SDG 16**, while also contributing meaningfully to **SDG 8** and **SDG 4**. Beyond its technical achievements, the project has the potential to empower institutions, protect individuals, foster trust, and inspire future innovation towards a safer and more sustainable digital future.