Intrusion Detection System Using Machine Learning

Enrol.No.(s)-20103303,20103302,20103298,20103299

Name of Student(s) -Prakhar Jain, Chirag Sharma, Sanskar Goel,Vasu Verma

Name of Supervisor -Dr. Tarun Agrawal

MAJOR PROJECT (Dec-2023)

Submitted in partial fulfillment of the Degree of Bachelor of Technology In Computer Science Engineering

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING & INFORMATION TECHNOLOGY

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY,NOIDA

TABLE OF CONTENTS

Chapter-6 Conclusion and Future Work

ACKNOWLEDGEMENT

We would like to place on record our deep sense of gratitude to **Prof.Dr. Tarun Agrawal**, Assistant Professor (senior grade), Jaypee Institute of Information Technology, India for his generous guidance, help and useful suggestions. We express our sincere gratitude to her for her stimulating guidance, continuous encouragement and supervision throughout the course of present work.

We also wish to extend my thanks to all our teachers, classmates and parents for their insightful comments and constructive suggestions to improve the quality of this project work and helping in finalizing this project within the limited time frame.

Signature(s) of Students

Prakhar Jain(20103303)
Chirag Sharma(20103302)
Sanskar Goel(20103298)
Vasu Verma(20103299)

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and beliefs, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma from a university or other institute of higher learning, except where due acknowledgment has been made in the text.

Place: Jaypee Institute of Information Technology, Noida
Date: 29 Nov, 2023

Name- Prakhar Jain
Enrollment no.-20103303
Batch-B10

Name- Chirag Sharma
Enrollment no.-20103302
Batch-B10

Name- Sanskar Goel
Enrollment no.- 20103298
Batch- B10

Name- Vasu Verma
Enrollment no:-20103299
Batch:-B9

CERTIFICATE

This is to certify that the work titled **"Intrusion Detection System Using Machine Learning**" submitted by "**Prakhar Jain, Chirag Sharma,Vasu Verma and Sanskar Goel**" of B. Tech of Jaypee Institute of Information Technology,Sec-62 Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of any other degree or diploma.

Signature:

Name: Dr. Tarun Agrawal
Designation: Assistant Professor (Sr.Grade)
Date: 29 Nov, 2023

SUMMARY

The project focused on the development and implementation of an advanced Intrusion Detection System (IDS) tailored to meet the evolving challenges in contemporary cybersecurity landscapes. Recognizing the limitations of traditional rule-based systems, this IDS was designed to leverage cutting-edge Machine Learning (ML) algorithms, including deep learning, decision trees, Support Vector Machines (SVM), and ensemble methods, to fortify network defenses against sophisticated cyber threats.

The primary goal was to create a proactive defense mechanism capable of analyzing network traffic patterns, system logs, and behavioral anomalies to swiftly detect and categorize intrusion attempts. Through extensive data collection, preprocessing, and model training, the IDS demonstrated a heightened ability to differentiate between normal network behavior and malicious activities, significantly reducing false positives and enhancing detection accuracy.

Key highlights of the project encompassed the implementation of scalable and real-time processing capabilities to handle dynamic network environments. This empowered administrators and security personnel with timely alerts and actionable insights, enabling proactive responses to potential breaches and bolstering the overall security posture of the network infrastructure.

The successful deployment of this IDS showcased promising results in fortifying network resilience, safeguarding critical assets, and elevating the cybersecurity framework. The project's outcomes not only contributed to enhancing the security paradigm but also underscored the significance of leveraging advanced ML techniques in combating evolving cyber threats.

The culmination of this project signifies a step forward in redefining network security paradigms, offering a robust defense mechanism against an ever-evolving threat landscape, and ensuring a safer digital environment for organizations and users alike.

Signature of Students **Signature of Supervisor**

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 General Introduction:

The existing network security framework faces persistent challenges due to the increasing sophistication of cyber threats. The current intrusion detection systems often struggle with accurately identifying and responding to evolving threats in real-time, leading to vulnerabilities and potential breaches. The absence of an agile, adaptive, and proactive system hampers the network's resilience against intrusions, jeopardizing the integrity of data and system operations.

The project aims to develop an innovative Intrusion Detection System (IDS) that mitigates these shortcomings. The current landscape demands a solution that surpasses the limitations of conventional rule-based detection systems. This system intends to harness the power of Machine Learning (ML) and anomaly detection techniques to analyze network traffic patterns, system logs, and behavioral anomalies, thereby identifying and categorizing diverse intrusion attempts.

The envisioned IDS will offer a robust defense mechanism against various cyber threats such as DoS attacks, malware infections, and unauthorized access attempts. The goal is to create a system capable of promptly distinguishing between normal network behavior and suspicious activities, enabling proactive responses to potential breaches.

The proposed solution aims to redefine network security by:

1. Implementing advanced ML algorithms, including deep learning, decision trees, SVM, or ensemble methods, to enhance detection accuracy and reduce false positives.
2. Collecting and preprocessing extensive and diverse datasets to train and validate the IDS models effectively.
3. Ensuring scalability and real-time processing capabilities to handle dynamic network environments and provide timely alerts to system administrators and security personnel.
4. Empowering network defenders with actionable insights and response mechanisms to mitigate risks promptly, bolstering the overall security posture.

The successful implementation of this IDS promises to fortify network resilience, safeguard critical assets, and elevate the cybersecurity framework, thereby ensuring a safer digital environment for organizations and users alike.

1.2 Problem Statement:

"In today's fast-paced digital landscape where cyber threats are constantly evolving, the need for a robust defense mechanism against intrusions has never been more crucial. This project revolves around the development of an advanced Intrusion Detection System (IDS) that harnesses the power of cutting-edge Machine Learning techniques. The aim is to create an intelligent system capable of proactively identifying and categorizing potential threats by analyzing complex network data, including traffic patterns, system logs, and behavioral anomalies.

The envisioned IDS serves as a shield, empowering networks to detect and respond swiftly to emerging threats, significantly minimizing the risk of security breaches. By leveraging Machine Learning algorithms, such as deep learning, decision trees, Support Vector Machines (SVM), or ensemble methods, the system will evolve to discern between normal network operations and suspicious activities, thereby reducing false alarms and ensuring accurate threat detection.

This project represents a pivotal step towards fortifying cybersecurity measures, offering a shield that adapts and learns in real-time to safeguard critical data and infrastructure. The system's capability to adapt and learn from new threats is pivotal in creating a secure digital environment, providing network administrators with the tools to stay one step ahead of the constantly evolving threat landscape. Ultimately, the goal is to establish a resilient defense mechanism that assures the safety and integrity of our digital spaces in the face of ever-growing cyber risks."

1.3 Significance Of The Problem:

The significance of developing an advanced Intrusion Detection System (IDS) using Machine Learning techniques lies in its pivotal role in safeguarding digital environments against an increasingly complex and evolving landscape of cyber threats. Here are some key points highlighting its significance:

1. Enhanced Cybersecurity: The IDS serves as a frontline defense, significantly bolstering cybersecurity measures by proactively identifying and mitigating potential threats. It helps prevent unauthorized access, data breaches, and system vulnerabilities, safeguarding sensitive information and critical infrastructure.

2. Adaptive Defense Mechanism: Leveraging Machine Learning allows the IDS to continuously learn and adapt to new threat patterns. This adaptability is crucial in countering the ever-evolving tactics employed by cyber attackers, ensuring a more resilient defense.

3. Reduced False Alarms: By employing advanced ML algorithms, the IDS aims to minimize false positives, enabling accurate detection of genuine threats. This reduction in false alarms saves time and resources for security teams, allowing them to focus on addressing actual security issues.

4. Real-time Threat Response: The IDS's capability to provide real-time alerts and insights empowers administrators and security personnel to respond promptly to potential threats. This agility in response helps in containing and neutralizing attacks before they can cause significant damage.

5. Improving Overall Security Posture: Implementing an effective IDS contributes to an organization's overall security posture. It instills confidence in stakeholders, clients, and users by demonstrating a proactive approach to cybersecurity, thereby preserving trust and reputation.

6. Regulatory Compliance: In various industries, compliance with cybersecurity regulations and standards is mandatory. A robust IDS helps meet these requirements by ensuring adequate protection against potential threats and vulnerabilities, thereby avoiding regulatory penalties and fines.

7. Enabling Safe Innovation: A reliable IDS promotes a secure environment for innovation and technological advancements within organizations. It encourages the adoption of new technologies and practices without compromising security, fostering growth and development.

In essence, the development of an advanced IDS using Machine Learning techniques not only addresses current cybersecurity challenges but also sets the stage for a proactive and adaptive defense strategy against future threats, thereby safeguarding the integrity, confidentiality, and availability of digital assets and infrastructure.

1.4 Empirical Study:

Comparative Analysis of Machine Learning Algorithms for Intrusion Detection

Objective: The objective of this empirical study was to evaluate and compare the performance of various Machine Learning algorithms in the context of an IDS for network security.

Methodology:

1. Dataset Selection: A comprehensive and diverse dataset, such as the CICIDS2017 dataset, was utilized for training and testing the Machine Learning models. This dataset contains both normal and attack instances, providing a realistic simulation of network traffic.
2. Algorithm Selection: Several prominent Machine Learning algorithms were selected for comparison, including Decision Tree, Random Forest, Logistic Regression,Adaptive Boosting and k-Nearest Neighbors (KNN).
3. Feature Engineering: Preprocessing techniques were applied to extract relevant features from the dataset, optimizing the input for the selected algorithms.
4. Model Training and Evaluation: Each algorithm was trained on a portion of the dataset and validated using cross-validation techniques to ensure robustness. Evaluation metrics such as accuracy, precision, recall, and F1-score were employed to measure the performance of each model.

Results:

1. Performance Metrics: The performance of each algorithm was assessed based on multiple evaluation metrics. The results indicated varying levels of accuracy, precision, recall, and F1-score across different algorithms.

2. Comparison and Analysis: Comparative analysis revealed that certain algorithms exhibited superior performance in detecting specific types of intrusions. For instance, Random Forest gives higher accuracy.

3. Trade-offs: Alongside performance metrics, considerations such as computational complexity, training time, and resource requirements were factored in. This provided a holistic view of the trade-offs between accuracy and computational efficiency for each algorithm.

Key findings:

1. Algorithm Performance Variation: The study revealed significant variations in the performance of Machine Learning algorithms when applied to intrusion detection. Some algorithms exhibited higher accuracy, precision, and recall rates compared to others, showcasing their effectiveness in identifying different types of intrusions.

2. Accuracy vs. Computational Complexity: Certain algorithms demonstrated exceptional accuracy but required more computational resources, leading to longer training times or increased computational complexity. Conversely, other algorithms offered a balance between accuracy and computational efficiency, making them more practical for real-time intrusion detection.

3. Algorithm Suitability for Specific Threats: Specific Machine Learning algorithms showcased better capabilities in detecting certain types of intrusions. For example, Deep Neural Networks excelled in recognizing complex patterns associated with sophisticated attacks, while ensemble methods like Random Forest proved adept at handling diverse attack scenarios with minimal overfitting.

4. Robustness and Generalization: The study indicated the robustness and generalization capabilities of selected algorithms across different network environments. Some models displayed consistent performance even when tested on previously unseen data, indicating their potential for deployment in real-world scenarios.

5. Impact of Feature Engineering: The preprocessing and feature engineering techniques employed significantly influenced the performance of Machine Learning models. Feature selection and extraction played a crucial role in improving the algorithms' ability to discern between normal network behavior and potential intrusions.

6. Trade-offs between Performance Metrics: While striving for higher accuracy, there were trade-offs between precision, recall, and false positives. Algorithms with higher accuracy might compromise on other metrics, emphasizing the importance of considering a balanced evaluation approach based on the specific needs of the network.

7. Real-Time Application Feasibility: Certain algorithms demonstrated better suitability for real-time application due to their ability to process data swiftly and provide timely responses to potential threats, aligning with the practical requirements of an IDS in dynamic network environments.

1.5 Brief Description Of The Solution Approach:

Our approach to developing an advanced Intrusion Detection System (IDS) using Machine Learning is founded on a systematic and comprehensive methodology that prioritizes accuracy, adaptability, and real-time responsiveness in combating cyber threats.

1. Data Collection and Preprocessing:

We initiate the process by curating diverse and extensive datasets encompassing various network activities and potential intrusion scenarios. The datasets are meticulously preprocessed to extract relevant features, normalize data, and eliminate noise, ensuring optimal input for the Machine Learning models.

2. Algorithm Selection and Optimization:

Our team meticulously evaluates a spectrum of Machine Learning algorithms, including but not limited to Decision Tree, Random Forest, Logistic Regression,Adaptive Boosting and k-Nearest Neighbors (KNN). . The selection is based on their capacity to handle diverse network traffic patterns and effectively discern anomalies. Hyperparameter tuning and optimization techniques are employed to enhance algorithm performance and generalization.

3. Model Training and Validation:

Rigorous model training and validation take center stage, utilizing a portion of the dataset for training and another for validation. Cross-validation techniques ensure robustness and prevent overfitting, allowing the models to generalize well to unseen data. Performance metrics such as accuracy, precision, recall, and F1-score are extensively analyzed to gauge the efficacy of each model.

4. Continuous Learning and Adaptation:

The IDS is engineered for continuous learning, dynamically adapting to evolving threat landscapes. By integrating feedback mechanisms and retraining the models periodically, the system stays abreast of emerging threats, ensuring its efficacy in safeguarding the network infrastructure.

5. Performance Evaluation and Optimization Iterations:

The performance of the IDS undergoes rigorous evaluation at each stage. Iterative optimization processes based on empirical results and ongoing research ensure that the system achieves and maintains a high level of accuracy, minimizes false positives, and optimizes resource utilization.

Our solution approach encompasses a holistic framework, integrating cutting-edge technologies and methodologies to create an IDS that not only identifies and mitigates potential threats but also evolves continuously to counteract emerging cyber risks. Through meticulous analysis, robust model development, and real-time responsiveness, our IDS stands as a stalwart guardian of network security, ensuring a resilient defense against cyber intrusions.

CHAPTER 2

LITERATURE SURVEY

2.1 Summary Of The Paper Studied:

1. Title :- Intrusion Detection System with Machine Learning Algorithms and Comparison Analysis

Summary :- This research paper delves into the critical realm of cybersecurity, given the surge in technology use and data processing. It introduces an Intrusion Detection System (IDS) that employs Machine Learning Algorithms to bolster our security measures. The paper aims to alleviate the issue of false alarms while enhancing the system's ability to spot security threats. It achieves this by evaluating and comparing the performance of different machine learning approaches. In essence, it offers a contemporary solution to effectively identify and categorize potential security breaches.

2. Title :- Intrusion Detection System with Correlation Engine and Vulnerability Assessment

Summary :- The research paper introduces an innovative Intrusion Detection System (IDS) designed to enhance network security. This IDS not only correlates alerts from both Host-based and Network-based IDS, following a concept similar to modern SIEM systems but also employs advanced techniques such as machine learning and rule-based methods to offer more precise alerts while minimizing false positives. The paper also sheds light on the shortcomings of current IDS systems, like their inability to detect newly developed attacks and their tendency to generate many false alerts. It underscores the pressing need for an affordable yet advanced IDS to tackle these issues and strengthen network security.

3. Title :- Design an Intrusion Detection System based on Feature Selection Using ML Algorithms

Summary :- It stresses the importance of safeguarding against cyber threats and explores how machine learning can play a pivotal role in enhancing Intrusion Detection Systems (IDS).The study employs a well-known cybersecurity dataset, NSL-KDD, to build an IDS. It focuses on the selection of key features to improve the system's accuracy in detecting intrusions. Three popular machine learning algorithms—Decision Tree, Random Forest, and SVM—are tested and compared to determine which one performs best in this context.

4. Title :- Intrusion Detection Systems, Issues, Challenges, and Needs

Summary:- This research paper delves into the world of Intrusion Detection Systems (IDSs) and their challenges. It highlights the limitations of common classification algorithms when used in IDS and discusses the importance of optimization algorithms to improve their performance.
The paper outlines the ongoing security issues in computer systems and networks, emphasizing that no system can be entirely secure. It introduces IDSs as a crucial tool for monitoring and detecting various network attacks.

5. Title:-Research Trends in Network-Based Intrusion Detection Systems: A Review

Summary:- In this research paper they conduct a comprehensive analysis of NIDS research trends, looking at the popularity of different approaches, commonly utilized datasets, and the metrics used to evaluate system performance. Their study, which spans from 2005 to 2020, relies on citation analysis of articles within the NIDS domain, offering valuable insights into the current state of NIDS research.

2.2 Integrated Summary Of The Literature Studied:

In the realm of cybersecurity, these research papers delve deeply into the critical task of fortifying network defenses against the ever-evolving landscape of cyber threats. They collectively explore different methodologies, technologies, and challenges aimed at enhancing the efficacy of Intrusion Detection Systems (IDS) to ensure robust network security.

The initial studies shine a spotlight on the integration of Machine Learning (ML) algorithms within IDS frameworks. Their primary focus is on leveraging ML techniques to bolster the effectiveness of IDS, reducing false alarms, and enhancing the system's ability to identify and categorize security threats accurately. A core aspect highlighted in these papers is the importance of conducting comparative analyses among different ML approaches. By doing so, researchers aim to identify the most effective algorithms that strike a balance between accuracy and efficiency while safeguarding networks against malicious activities.

Beyond the integration of ML, another research paper introduces an innovative approach by amalgamating correlation engines and vulnerability assessments within IDS architectures. This holistic approach not only correlates alerts from various sources but also employs advanced techniques such as ML and rule-based methods. The goal is to refine alerts, minimize false positives, and offer a more precise and reliable intrusion detection mechanism. This paper critically evaluates existing IDS limitations and suggests innovative solutions, emphasizing the

need for an affordable yet sophisticated IDS to bolster network security.

Further exploring the intricacies of ML integration within IDS, another study underscores the significance of feature selection. Utilizing well-known datasets like NSL-KDD, the research focuses on building an IDS while emphasizing the pivotal role of feature selection in improving detection accuracy. The study's comparative analysis of popular ML algorithms—Decision Tree, Random Forest, and Support Vector Machines (SVM)—provides nuanced insights into algorithmic performance concerning feature selection, shedding light on the key factors influencing detection capabilities.

Shifting the focus, an in-depth exploration of IDS challenges emerges within the literature. This paper illuminates the inherent limitations of current IDS systems, particularly concerning classification algorithms' constraints and the importance of optimization algorithms to improve performance. Despite acknowledging the inherent security vulnerabilities within computer systems and networks, these studies highlight the indispensable role of IDS as a crucial tool for monitoring and detecting various network attacks.

Lastly, a comprehensive review study delves into the evolving landscape of Network-Based Intrusion Detection Systems (NIDS) research trends. Conducting a thorough analysis spanning over a significant timeframe, this research delves into the popularity of different approaches, commonly utilized datasets, and evaluation metrics used to assess system performance. By employing citation analysis within the NIDS domain, this study offers a panoramic view of prevalent methodologies and evaluation criteria over time, providing valuable insights into the ever-evolving field of intrusion detection research.

Collectively, these diverse research papers contribute richly to our understanding of IDS, from its integration with advanced technologies like ML and correlation engines to the prevailing challenges, trends, and evolving research landscapes within the realm of intrusion detection. These studies underscore the critical role IDS plays in fortifying network security and offer invaluable insights for developing more resilient, adaptive, and effective intrusion detection mechanisms in the face of burgeoning cyber threats.

CHAPTER 3
REQUIREMENT ANALYSIS AND SOLUTION APPROACH

3.1 Overall description of the project:

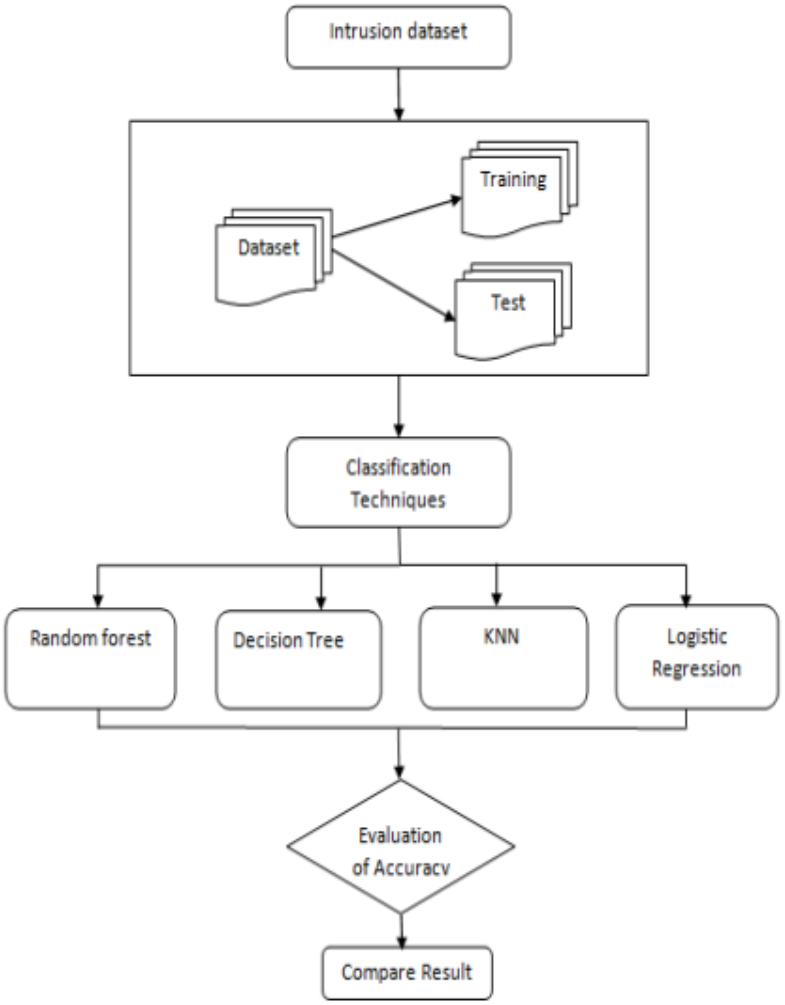

Fig 1: System Architecture

Proposed smart intrusion detection system (IDS) is viewed as an effective solution for network security and protection against external threats. However, the existing IDS often has a lower detection rate under new attacks and has a high overhead when working with audit data, and thus machine learning methods have been widely applied in intrusion detection.

In order to overcome the problem of class imbalance, feature selection based on Correlation-based feature selection (CFS) is used to determine a subset of the original features to eliminate irrelevant features. The detection framework of the proposed ML- Based consists of three stages including: feature selection, build and training the ensemble classifier and attack recognition.

In our proposed method, Decision Tree, Logistic regression, Random Forest and KNN are developed as learning methods in solving the classification problem of pattern recognition and intrusion identification.

3.2 Requirement Analysis:

The Requirement Analysis includes Software and Hardware requirement, which are provided below:

- **Software Requirements:**

Operating System :	Windows 7 or higher
Programming :	Python 3.6 and related libraries
Software Environment :	JupyterLab

➢ **Python:**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation.

➢ **Scikit-learn:**

Scikit-learn is a simple and efficient tool for data mining and data analysis. Scikit Learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

Some popular groups of models provided by scikit-learn include:

Clustering: for grouping unlabeled data such as KMeans.

Cross Validation: for estimating the performance of supervised models on unseen data. Datasets: for test datasets and for generating datasets with specific properties for investigating model behavior.

Dimensionality Reduction: for reducing the number of attributes in data for summarization,

visualization and feature selection such as Principal component analysis.

Ensemble methods: for combining the predictions of multiple supervised models.

Feature extraction: for defining attributes in image and text data.

Feature selection: for identifying meaningful attributes from which to create supervised models.

Parameter Tuning: for getting the most out of supervised models.

Manifold Learning: For summarizing and depicting complex multi-dimensional data.

Supervised Models: a vast array not limited to generalized linear models, discriminant analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

- **Hardware Requirements**

Processor :	Any Processor above 500 MHz.
Ram :	4 GB
Hard Disk :	4 GB
Input device :	Standard Keyboard and Mouse.
Output device :	VGA and High Resolution Monitor.

- **Functional Requirements:**

1. Data Collection and Preprocessing:

   The system should be able to collect network data in real-time. It must support the preprocessing of raw data to prepare it for the machine learning models.

2. Feature Extraction:

   The system should extract relevant features from network data, considering both packet-level and session-level features.

3. Model Integration:

   The IDS must integrate machine learning models, including Decision Tree, k-Nearest Neighbors, Random Forest, and Logistic Regression.

4. Training:

   The system should be capable of training the machine learning models using labeled datasets. It must allow periodic retraining to adapt to evolving network patterns.

5. Anomaly Detection:

   The IDS should identify anomalous patterns in network traffic using the trained models. It should have the capability to distinguish between normal and malicious activities.

6. Integration with Existing Security Infrastructure:

   The IDS should seamlessly integrate with existing security infrastructure, including firewalls, antivirus software, and log management systems.

7. Scalability:

   The system should be scalable to handle increasing network traffic and data volumes.

- **Non-Functional Requirements:**
    1. Performance:

       The detection accuracy of the models should meet or exceed industry standards.
    2. Reliability:

       The IDS should have high reliability, ensuring that it operates effectively in diverse network environments. It must be resistant to false positives and negatives.
    3. Compatibility:

       The system should be compatible with various network architectures and protocols. It must support interoperability with other security solutions.
    4. Maintainability:

       The system should be easy to maintain, allowing for updates, patches, and model retraining without significant downtime.
    5. Scalability:

       The IDS should scale horizontally to accommodate the growth of network traffic and data.
    6. Usability:

       The user interface should be intuitive and easy to use for security analysts with varying levels of expertise.
    7. Resource Utilization:

       The system should utilize computing resources efficiently to prevent bottlenecks and ensure optimal performance.

3.3 Solution Approach:

1. Data Cleaning:

Most Machine Learning algorithms cannot work with missing features, so let's create a few functions to take care of them. You noticed earlier that the total bedrooms attribute has some missing values, so let's fix this.

You have three options:

- Get rid of the corresponding districts.
- Get rid of the whole attribute.
- Set the values to some value (zero, the mean, the median, etc.).

2. Feature Selection:

The aim of feature selection is to find a subset of the attributes from the original set which are representative enough for the data, and the attributions in the subset are highly relevant to the

prediction.

In this project, a CFS approach is used to optimize the efficiency of the feature selection process and enhance the accuracy of the classification:

- **Correlation-based feature selection (CFS):** CFS is one of classical filter algorithms that choose features according to the result of the heuristic (correlation-based) assessment function. The preference of this function is to select subsets whose features are extraordinarily related with the class but uncorrelated with each other.

3. Machine Learning:

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

- **Supervised Learning:** In supervised learning, the training data you feed to the algorithm includes the desired solutions, called labels.

  Here are some of the most important supervised learning algorithms:
  • k-Nearest Neighbors
  • Logistic Regression
  • Decision Trees
  • Random Forests
- **Confusion Matrix in Machine Learning:**

  In the field of machine learning and specifically the problem of statistical classification, a confusion matrix, also known as an error matrix.

  A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. It allows the visualization of the performance of an algorithm. It allows easy identification of confusion between classes

  e.g. one class is commonly mislabeled as the other. Most performance measures are computed from the confusion matrix.

  Confusion Matrix: A confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is the key to the confusion matrix. The confusion matrix shows the ways in which your classification model is confused when it makes predictions. It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.

Classification Rate/Accuracy:

Classification Rate or Accuracy is given by the relation:

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

However, there are problems with accuracy. It assumes equal costs for both kinds of errors. A 99% accuracy can be excellent, good, mediocre, poor or terrible depending upon the problem.

Recall:

Recall can be defined as the ratio of the total number of correctly classified positive examples divided to the total number of positive examples. High Recall indicates the class is correctly recognized (a small number of FN).

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

Precision: To get the value of precision we divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (a small number of FP).

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

High recall, low precision: This means that most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.

Low recall, high precision: This shows that we miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP).

CHAPTER 4

MODELING AND IMPLEMENTATION DETAILS

4.1 Design Diagrams:

4.1.1 Use Case Diagram:

Use Case Diagram captures the system's functionality and requirements by using actors and use cases. Use Cases model the services, tasks, functions that a system needs to perform. Use cases represent high-level functionalities and how a user will handle the system. Use-cases are the core concepts of Unified Modelling language modeling.

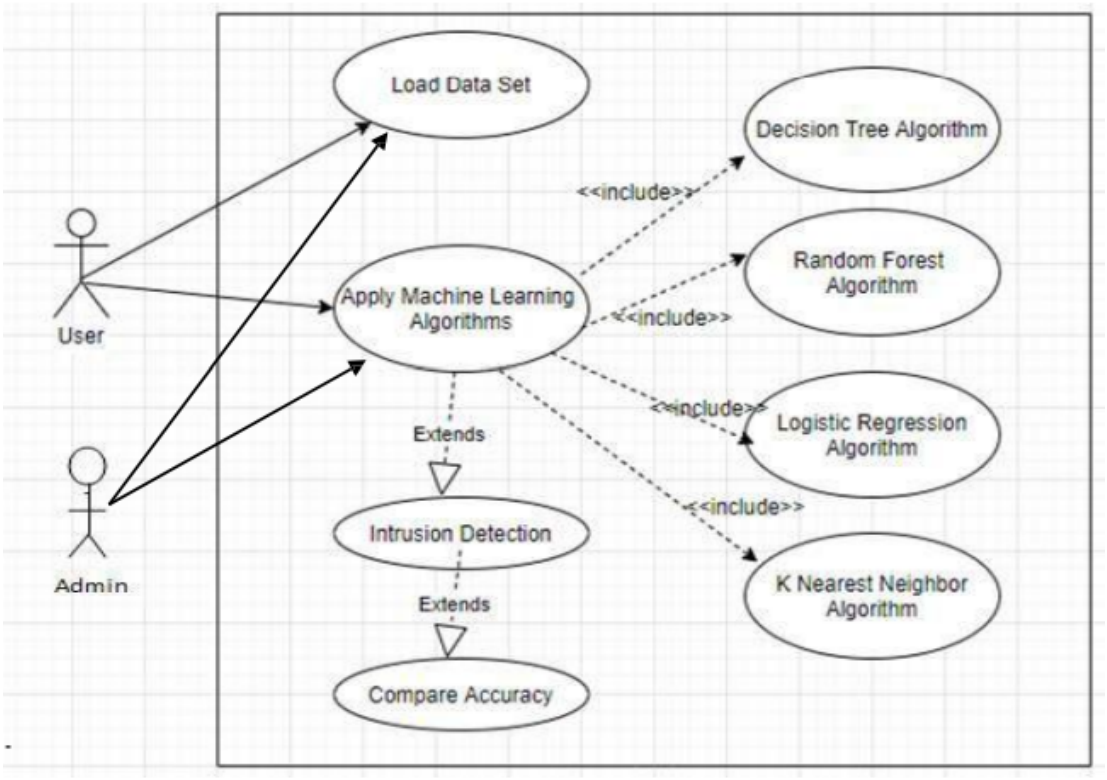

Fig 2: Use Case Diagram for the Proposed System

4.1.2 Sequence Diagram:

Sequence Diagrams display the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).

Object: Object can be viewed as an entity at a particular point in time with a specific value and as a holder of identity that has different values over time.

Actor: An Actor represents a coherent set of roles that users of a system play when interacting with the use cases of the system.

Message: A Message is sending of a signal from one sender object to other receiver objects

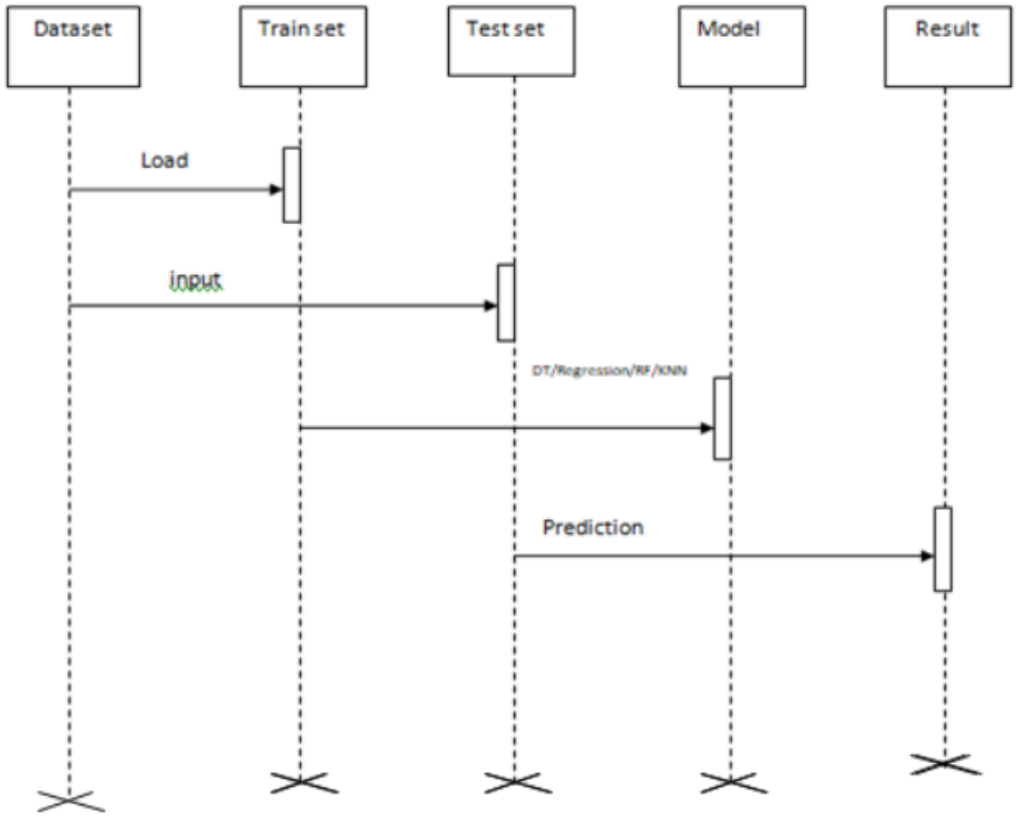

Fig 3: Sequence Diagram of the Proposed System

4.1.3 Activity Diagram:

An activity diagram shows the flow from activity to activity. An activity is a nonatomic execution within a state machine. An activity results in some action, results in a change of state or return of a value. Activity Diagram commonly contains:

- Activity states and action states.
- Transitions.
- Objects may contain nodes and constraints .

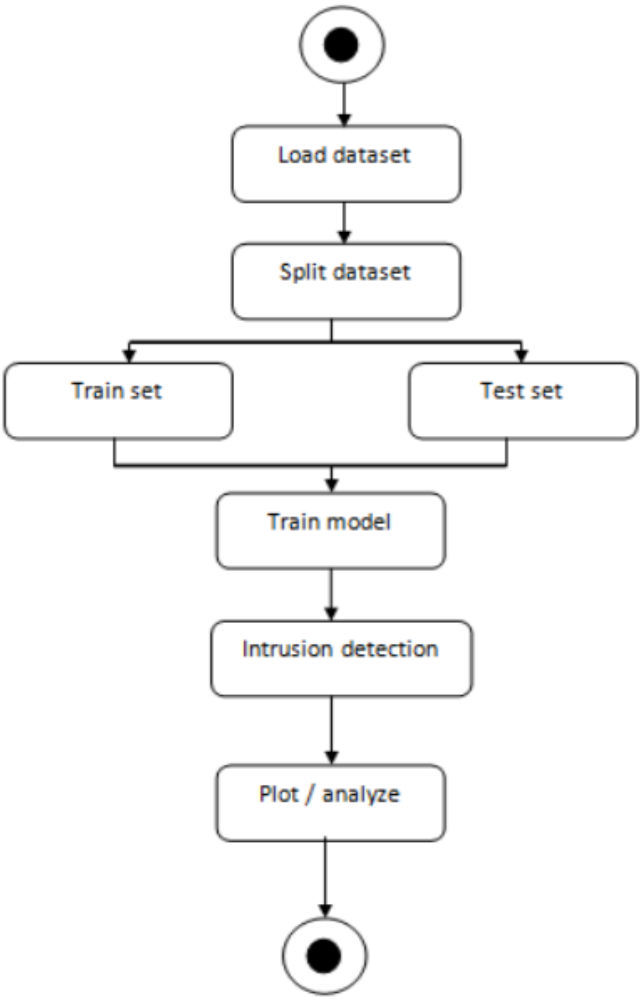

Fig 4:Activity Diagram of the Proposed System

4.1.4 Component Diagram:

Component Diagram describes the organization and wiring of the Physical Components in a system. It can be presented to key project stakeholders and implementation staff. Component Diagram can be used:

- To model the components of a system.
- To model the database schema.
- To model the executability of an application.
- To model the system source code.

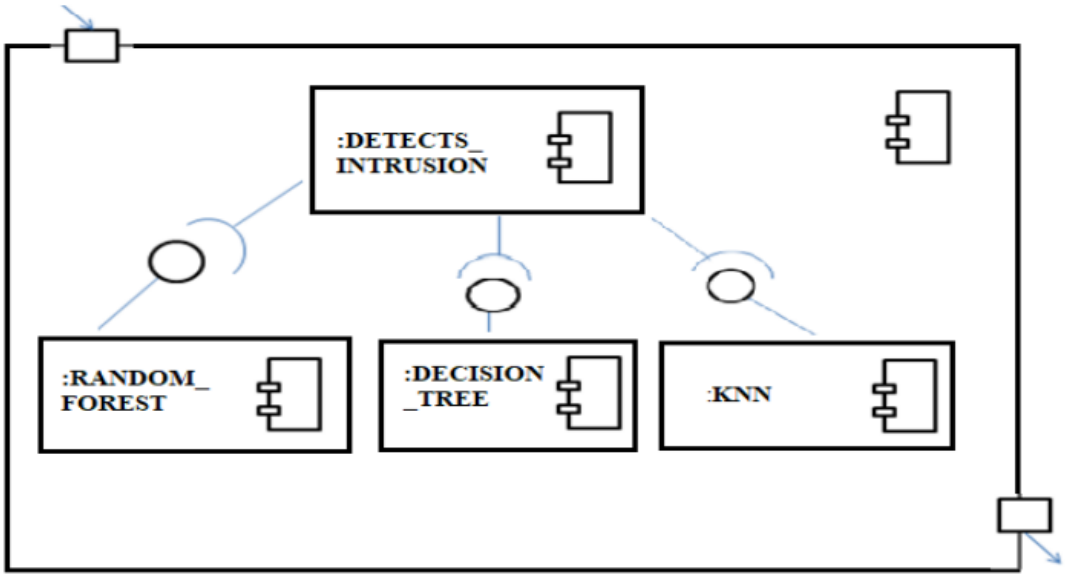

Fig 5:Component Diagram of the Proposed System

4.1.5 Deployment Diagram:

Deployment Diagram is used to visualize the Topology of the physical component of a system, where the software components are deployed. A Deployment Diagram consists of Nodes and their relationships. Deployment Diagram is used to show how they are deployed in hardware.

These are useful for System Engineers. An efficient Deployment Diagram is necessary as it controls the performance, maintainability and portability.

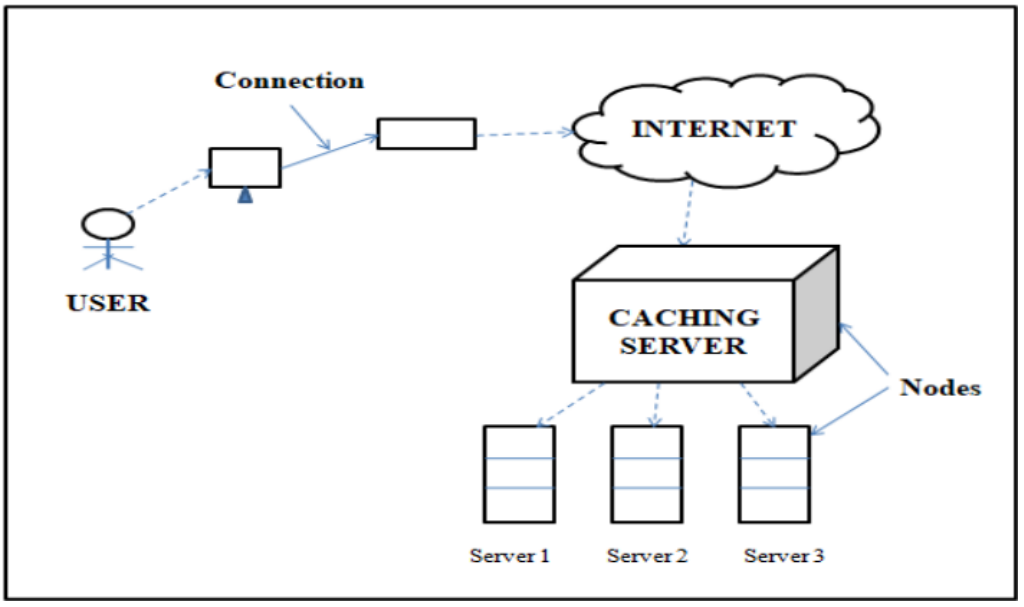

Fig 6: Deployment Diagram for the Proposed System

4.2 Implementation Details:

Intrusion Detection is considered with four different machine learning algorithms such as Decision Tree, Logistic Regression, KNN and Random Forest. The predicted value is compared for the predicted accuracy and error values.

A. Decision Tree Algorithm:

Decision tree is a type of supervised learning algorithm that is mostly used in classification problems. It works for both categorical and continuous input and output variables. In this technique, we split samples into two or more homogeneous sets (or sub-populations) based on the most significant splitter / differentiator in input variables. In the decision tree the internal node represents a test on the attribute, the branch depicts the outcome and the leaf represents the decision made after computing the attribute.

The general motive of using Decision Tree is to create a training model which can be used to predict class or a value of target variables by learning decision rules inferred from prior data (training data).

The understanding level of the Decision Tree algorithm is so easy compared with other classification algorithms. The Decision Tree algorithm tries to solve the problem, by using tree representation. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label.

Decision Tree works in the following manner:

Place the best attribute of the dataset at the root of the tree.

Split the training set into subsets.

Subsets should be made in such a way that each subset contains data with the same value for an attribute.

Repeat step 1 and step 2 on each subset until you find leaf nodes in all the branches of the tree. In decision trees, for predicting a class label for a record we start from the root of the tree. Then compare the values of the root attribute with the record's attribute. On the basis of comparison, follow the branch corresponding to that value and jump to the next node.

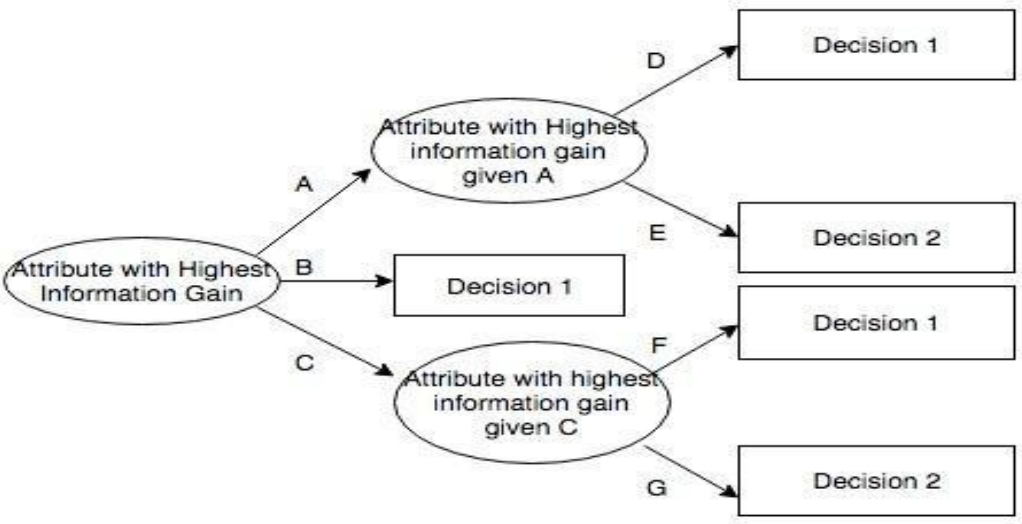

Fig 7: Flowchart Decision Tree algorithm

B. Logistic Regression Algorithm:

Logistic Regression is basically a predictive model analysis technique where the output (target) variables are discrete values for a given set of features or input (X). It is a very powerful yet simple supervised classification algorithm in machine learning. Around 60% of the world's classification problems can be solved by using the logistic regression algorithm.

Logistic Regression is one of the most common algorithms used for binary classification. It predicts the probability of occurrence of a binary outcome using a logit function. It is a special case of linear regression as it predicts the probabilities of outcome using log function.

In simple terms, Linear Regression predicts scores on one variable from the scores on a second variable. The variable that is predicted is called Criterion Variable. The variable base for predictions is called the predictor variable. When there is only one predictor variable, the prediction method is called simple regression. We use the activation function (sigmoid) to convert the outcome into categorical value.

There are many examples where we can use Logistic Regression for example, it can be used for Fraud Detection, Spam Detection, Cancer Detection, etc.

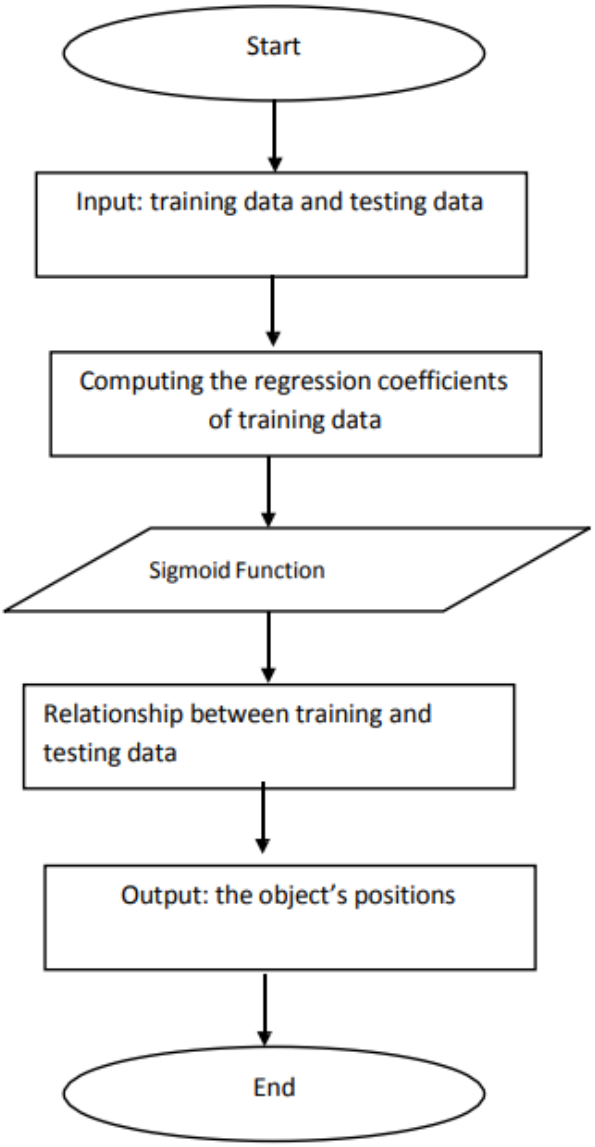

Fig 8: Flow chart of Logistic Regression algorithm

C. Random Forest Algorithm:

Given there are n cases in the training dataset. From these n cases, sub-samples are chosen at random with replacement. These random sub-samples chosen from the training dataset are used to build individual trees.

Assuming there are k variables for input, a number m is chosen such that m < k. m variables are selected randomly out of k variables at each node. The split which is the best of these m variables is chosen to split the node.

The value of m is kept unchanged while the forest is grown. Each tree is grown as large as possible without pruning. The class of the new object is predicted based upon the majority of votes received from the combination of all the decision trees.

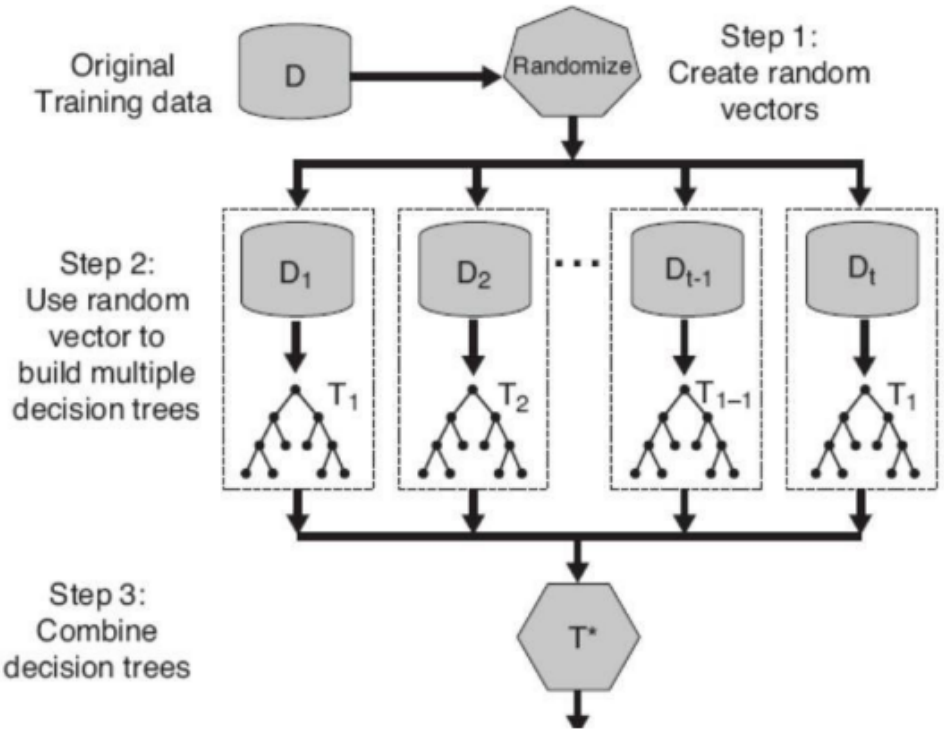

Fig 9: Flow chart of Random Forest Algorithm

D. K Nearest Neighbor (KNN) Algorithm:

The model for KNN is the entire training dataset. When a prediction is required for an unseen data instance, the KNN algorithm will search through the training dataset for the k-most similar instances. The prediction attribute of the most similar instances is summarized and returned as the prediction for the unseen instance.

The similarity measure is dependent on the type of data. For real-valued data, the Euclidean distance can be used. Other types of data such as categorical or binary data, Hamming distance can be used. Instance-based algorithms are those algorithms that model the problem using data instances (or rows) in order to make predictive decisions.

The KNN algorithm is an extreme form of instance-based methods because all training observations are retained as part of the model. It is a competitive learning algorithm, because it internally uses competition between model elements (data instances) in order to make a predictive decision. The objective similarity measure between data instances causes each data instance to compete to "win" or be most similar to a given unseen data instance and contribute to a prediction.

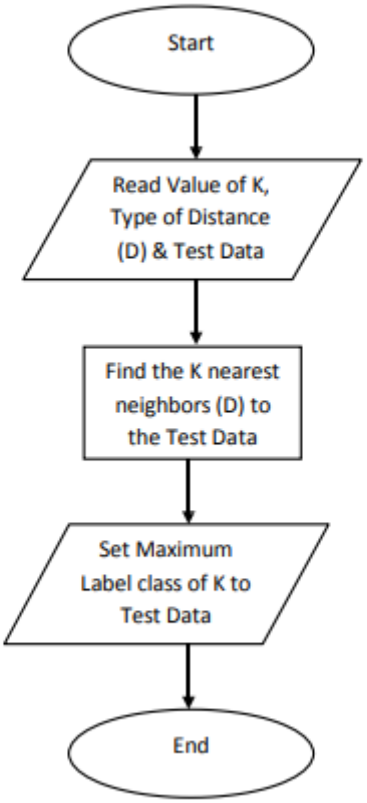

Fig 10: Flow chart of KNN Algorithm

CHAPTER 5

TESTING

5.1 Testing Plan:

The aim of testing is often to demonstrate that a program works by showing that it has no errors. The basic purpose of the testing phase is to detect the errors that may be present in the program. Hence one should not start testing with the intent of showing that a program works but the intent should be to show that a program doesn't work. Testing is the purpose of executing a program with the intent of finding others.

Testing Objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time, Stating formally, we can say

- Testing is a process of executing a program for finding an error.
- A successful test is one that uncovers as yet undiscovered errors.
- A good test case is one that has a high probability of finding errors.
- The tests are inadequate to detect possibly present errors.
- The software more or less confirms to the quality

Types of Testing:

- Unit Testing.
- Integration Testing.
- System Testing.
- Acceptance Testing.

1. Unit Testing: Unit testing focuses verification effort on the smallest unit of software i.e. module. Using the detailed design and the process specification testing is done to uncover errors within the boundary of the module. All modules must be successful in the unit test before the start of the integration testing begins. In this project each service can be thought of as a module.
2. Integration Testing: After the unit testing we have to perform integration testing. The goal here is to see if modules can be integrated properly, the emphasis being on testing interfaces between modules. This testing activity can be considered as testing the design and hence the emphasis on testing module interactions. In this project integrating the entire module forms the main system. When integrating all the modules I have checked whether the integration affects the working of any of the services by giving different combinations of inputs with which the two services run perfectly before integration.

- Analogy:

  During the process of manufacturing a ballpoint pen, the cap, the body, the tail and clip, the ink cartridge and the ballpoint are produced separately and unit tested separately. When two or more units are ready, they are assembled and Integration Testing is performed. For example, whether the cap fits into the body or not.

- Method:

  Any of Black Box Testing, White Box Testing and Gray Box Testing methods can be used. Normally, the method depends on your definition of 'unit'.

  Tasks

  - Integration Test Plan
    - Prepare
    - Review
    - Rework
    - Baseline
  - Integration Test Cases/Scripts
    - Prepare
    - Review
    - Rework
    - Baseline
    - Integration Test

  Integration Testing is the second level of testing performed after Unit Testing and before System Testing. Developers themselves or independent testers perform Integration Testing.

3. System Testing: Here the entire software system is tested. The reference document for this process is the requirements of the document. And the goal is to see if software needs its requirements.
4. Acceptance Testing:

   ● Acceptance test is performed with realistic data of the client to demonstrate that the software is working satisfactory. Testing here is focused on external behavior of the system: the internal logic of the program is not emphasized.

   ● Test cases should be selected so that the largest number of attributes of an aquiline class is exercised at once. The testing phase is an important path of software development. It is whether the objectives are met and the user requirements are satisfied.

● In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.

● System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system and also the expectations of the customer.

5.2 Output :

[13]:

	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	Fwd Packet Length Std	...	min_seg_size_forward	Active Mean	Active Std	Active Max	Active Min
0	49188	4	2	0	12	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
1	49188	1	2	0	12	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
2	49188	1	2	0	12	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
3	49188	1	2	0	12	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
4	49486	3	2	0	12	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1087559	80	163071	4	0	24	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
1087560	80	2036	3	6	26	11607	20	0	8.666667	10.263203	...	20	0.0	0.0	0	0
1087561	80	159465	4	0	24	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0
1087562	80	1453	3	6	26	11607	20	0	8.666667	10.263203	...	20	0.0	0.0	0	0
1087563	80	155039	4	0	24	0	6	6	6.000000	0.000000	...	20	0.0	0.0	0	0

1087564 rows × 66 columns

Fig 11: Dataset

```
BENIGN                          529918
DoS Hulk                        231073
PortScan                        158930
DDoS                            128027
DoS GoldenEye                    10293
FTP-Patator                       7938
SSH-Patator                       5897
DoS slowloris                     5796
DoS Slowhttptest                  5499
Bot                               1966
Web Attack ? Brute Force          1507
Web Attack ? XSS                   652
Infiltration                        36
Web Attack ? Sql Injection          21
Heartbleed                          11
Name: Label, dtype: int64
```

Fig 12: Types Of Attack

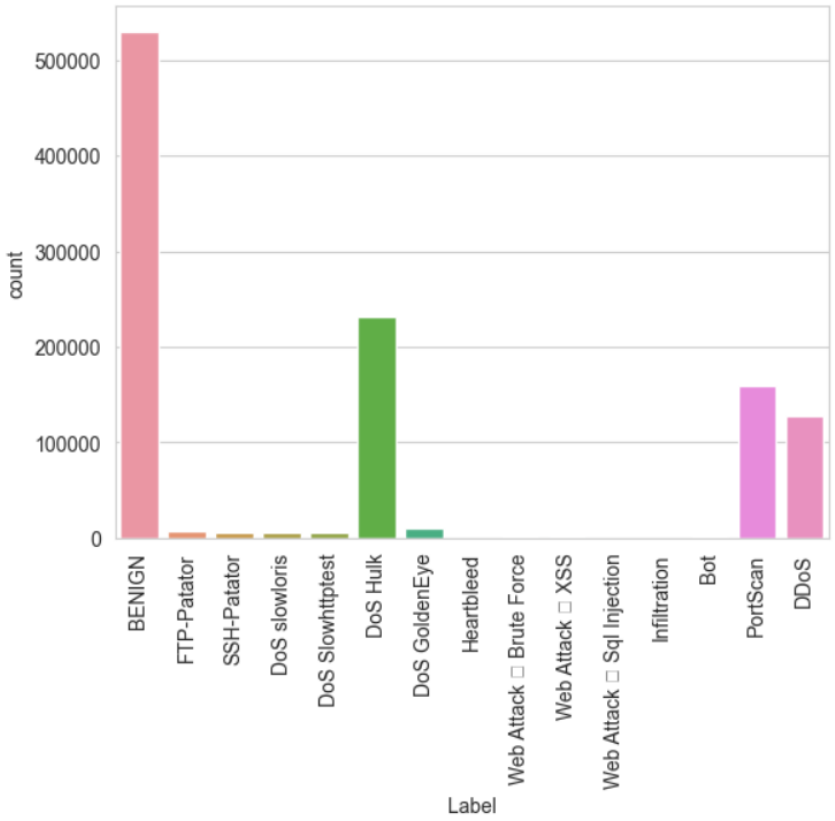

Fig 13: Countplot of Attacks in Dataset

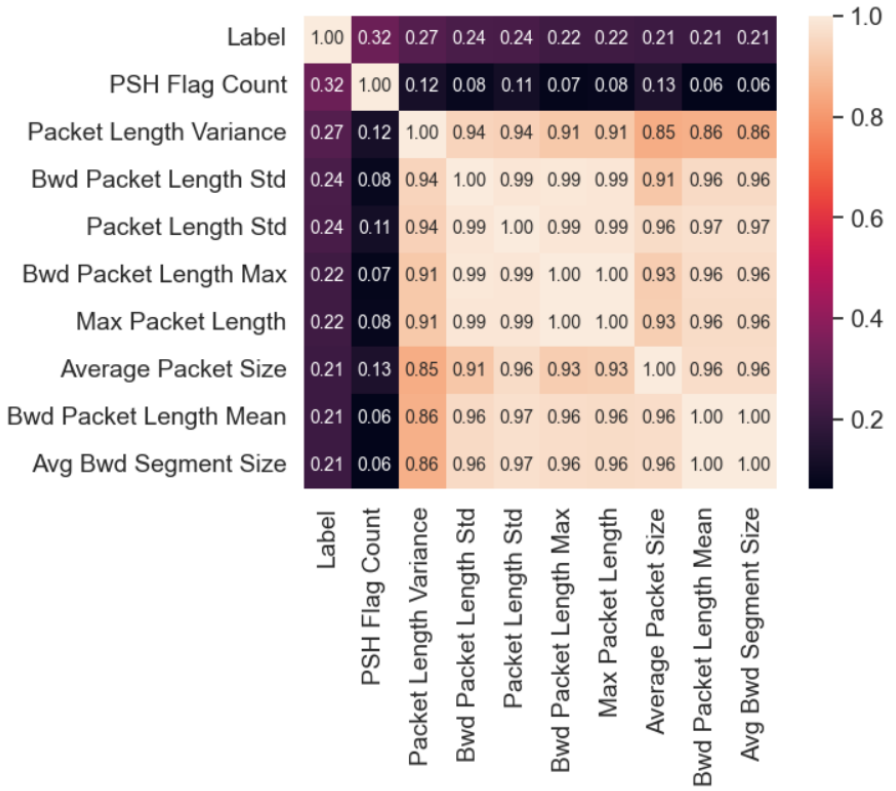

Fig 14: Correlation Matrix Of Top 10 Features including Label

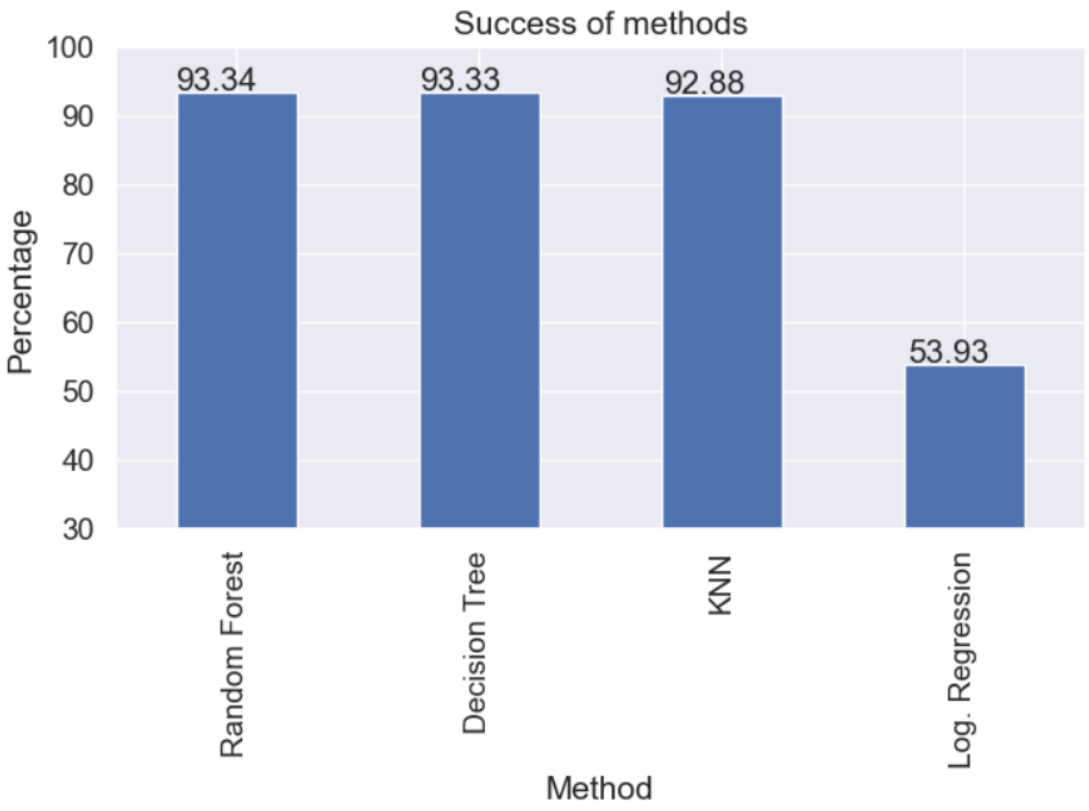

Fig 15: Accuracy Of Diff Algorithms used for training the dataset

[41]:

	Destination Port	Label
0	432524	1
1	70453	3
2	852636	2
3	260402	0
4	276178	0
5	304190	0
6	712069	0
7	201743	0
8	488404	0
9	435626	0
10	849805	2

Fig 16: Predictions on Test Data

CHAPTER 6:

CONCLUSION and FUTURE WORK

6.1 Conclusion:

In conclusion, the development and implementation of an Intrusion Detection System (IDS) are pivotal in today's technology-driven landscape where the threat of cyberattacks looms large. The primary objective of an IDS is to effectively detect and mitigate potential threats and malicious activities within a network. Achieving this requires a sophisticated system capable of discerning between genuine threats and false positives.

Machine learning algorithms play a significant role in enhancing the accuracy and reliability of IDS outputs. These algorithms enable the system to learn and adapt to evolving threats, thereby improving its capability to detect and respond to various types of attacks. By leveraging machine learning, IDS can continually refine its detection mechanisms, thereby increasing its efficacy in safeguarding the network.

The exponential growth in technological advancements has resulted in vast amounts of data being generated, necessitating secure processing and storage measures. Security remains paramount in maintaining user privacy and ensuring the integrity of sensitive data. A robust IDS not only identifies and mitigates threats but also contributes significantly to fortifying the security infrastructure, thereby enhancing user trust and confidence in the system.

The strength of an IDS lies in its ability to ensure the confidentiality, integrity, and availability of data. By providing a secure environment, an IDS contributes significantly to the overall reliability of the system. Therefore, an IDS that effectively protects user data and maintains a high level of security can be deemed as a valuable asset, contributing to the trustworthiness and reliability of the entire network infrastructure. Ultimately, a well-designed and efficiently implemented IDS is instrumental in safeguarding against threats, thereby reinforcing the system's security posture and ensuring user confidence in the digital ecosystem.

6.2 Future Work:

In advancing the proposed intrusion detection system, future work should focus on augmenting its reliability and efficiency. One avenue for improvement involves integrating additional machine learning algorithms, such as Support Vector Machines, Naive Bayes, and Neural Networks, to diversify the detection capabilities. Employing ensemble learning techniques, like stacking or boosting, could further enhance the overall performance by leveraging the strengths of multiple algorithms. Feature engineering and selection techniques should be explored to identify and incorporate more relevant features, empowering the models to better discern normal and malicious network behaviors. The implementation of dynamic model adaptation mechanisms would enable the system to flexibly adjust its strategies in response to evolving attack patterns and network dynamics. Additionally, extending the system's classification abilities to encompass various types of cyber threats beyond traditional intrusions, including advanced persistent threats and zero-day attacks, would contribute to a more comprehensive security framework. Further research into behavioral analysis techniques, dynamic dataset expansion, and the integration of user and entity behavior analytics could collectively refine the system's anomaly detection capabilities. Collaboration with other security systems, improved model interpretability, and a focus on privacy-preserving techniques would further contribute to the system's efficacy. Lastly, integrating human-in-the-loop approaches would allow security analysts to provide valuable feedback, fostering continuous learning and adaptation to emerging threats.

6.3 References:

[1] V. U. Bala, N. D. S. Keerthana, and K. Bharathi, "Intrusion Detection System with Machine Learning Algorithms and Comparison Analysis," *International Research Journal of Engineering and Technology*, Apr. 2020.
Available at: https://www.irjet.net/archives/V7/i4/IRJET-V7I484.pdf

[2] D. W. Y. O. Waidyarathna, W. V. A. C. Nayantha, W. M. T. C. Wijesinghe, and K. Y. Abeywardena, "Intrusion Detection System with Correlation Engine and Vulnerability Assessment," *International Journal of Advanced Computer Science and Applications*, 2018.
Available at:https://thesai.org/Downloads/Volume9No9/Paper_47-Intrusion_Detection_System.pdf

[3] S. Venkatesan, "Design of an Intrusion Detection System based on Feature Selection Using Machine Learning Algorithms," *Mathematical Statistician and Engineering Applications*, 2023.
Available at:https://philstat.org/index.php/MSEA/article/view/2000/1483

[4] M. Aljanabi, M. A. Ismail, and A. H. Ali, "Intrusion Detection Systems: Issues, Challenges, and Needs," *International Journal of Computational Intelligence Systems*, 2021.
Available at:https://www.atlantis-press.com/journals/ijcis/125951139

[5] S. Kumar, S. Gupta, and S. Arora, "Research Trends in Network-Based Intrusion Detection Systems: A Review," *IEEE Xplore*, 2021.
Available at:https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=96234