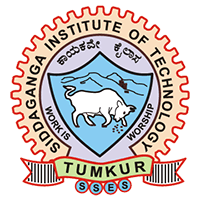
**SIDDAGANGA INSTITUTE OF TECHNOLOGY, Tumakuru- 3**

(An Autonomous institution affiliated to Visvesvaraya Technological University- Belagavi, Approved by AICTE,

Accredited by NAAC with ‘A++’ Grade, Awarded Diamond College Rating by QS I-GAUGE & ISO 9001:2015 certified )



**MINI PROJECT REPORT**

**ON**

**“Anomaly Detection in Heterogeneous Networks using Machine learning and feature-fusion technique.”**

submitted in the partial fulfilment of the requirements for V semester,

Bachelor of Engineering in Computer Science and Engineering

By

|  |  |
| --- | --- |
| **Lalith D** | **1SI22CS091** |
| **Mrinal N** | **1SI22CS106** |
| **Natesh Gaddad** | **1SI22CS110** |

Under the guidance of

**Dr. A S Poornima B.E,M.Tech,Ph.D**

**Professor**

**Department of Computer Science and Engineering**

( Program Accredited by NBA)

**Branch: Computer Science and Engineering**

**Academic Year: 2024-25**

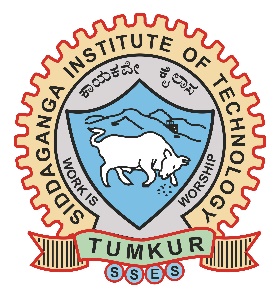
**Siddaganga Institute of Technology, Tumakuru-3**

(An Autonomous institution affiliated to Visvesvaraya Technological University- Belagavi, Approved by AICTE, Accredited by

NAAC with ‘A++’ Grade, Awarded Diamond College Rating by QS I-GAUGE & ISO 9001:2015 certified )

**Department of Computer Science and Engineering**

(Program Accredited by NBA)



# **CERTIFICATE**

This is to certify that the mini project entitled “Anomaly Detection in Heterogeneous Networks using Machine learning and feature-fusion technique” is a bonafide work carried out by **Lalith D (1SI22CS091), Mrinal N (1SI22CS106), Natesh Gaddad (1SI22CS110)** of V semester **Computer science and engineering**, **SIDDAGANGA INSTITUTE OF TECHNOLOGY** during the academic year 2024-2025.

**Signature of the Guide**

**Dr. A S Poornima B.E,M.Tech,Ph.D**

**Professor**

**Signature of the HOD**

Dr. N R Sunitha

**Prof. and Head, Dept. of CSE**

**Name of the Examiners: Signature with Date**



**ACKNOWLEDGEMENT**

We offer our humble pranams at the lotus feet of **His** Holiness, **Dr. Sree Sree Sivakumara Swamigalu**, Founder President and **His** Holiness, **Sree Sree Siddalinga Swamigalu**, President, Sree Siddaganga Education Society, Sree Siddaganga Math for bestowing upon their blessings.

We deem it as a privilege to thank **Dr. M N Channabasappa**, Director, SIT, Tumakuru, **Dr. Shivakumaraiah**, CEO, SIT, Tumakuru, and **Dr. S V Dinesh**, Principal, SIT, Tumakuru for fostering an excellent academic environment in this institution, which made this endeavor fruitful.

We would like to express our sincere gratitude to **Dr. N R Sunitha**, Professor and Head Department of CSE, SIT, Tumakuru for her encouragement and valuable suggestions.

We thank our guide **Dr. A S Poornima**, Assistant Professor, Department of Computer Science and Engineering, SIT, Tumakuru for the valuable guidance, advice and encouragement.

|  |  |
| --- | --- |
| Lalith D | 1SI22CS091 |
| Mrinal N | 1SI22CS106 |
| Natesh Gaddad | 1SI22CS110 |

**Course Outcomes**

After successful completion of mini project, graduates will be able to

CO1: To identify a problem through literature survey and knowledge of contemporary engineering technology.

CO2: To consolidate the literature search to identify issues/gaps and formulate the engineering problem

CO3: To prepare project schedule for the identified design methodology and engage in budget analysis, and share responsibility for every member in the team

CO4: To provide sustainable engineering solution considering health, safety, legal, cultural issues and also demonstrate concern for environment

CO5: To identify and apply the mathematical concepts, science concepts, engineering and management concepts necessary to implement the identified engineering problem.

CO6: To select the engineering tools/components required to implement the proposed solution for the identified engineering problem.

CO7: To analyze, design, and implement optimal design solution, interpret results of experiments and draw valid conclusion.

CO8: To demonstrate effective written communication through the project report, the one-page poster presentation, and preparation of the video about the project and the four page IEEE/Springer/ paper format of the work.

CO9: To engage in effective oral communication through power point presentation and demonstration of the project work.

CO10:To demonstrate compliance to the prescribed standards/ safety norms and abide by the norms of professional ethics.

CO11: To perform in the team, contribute to the team and mentor/lead the team

**CO-PO Mapping**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | PO1 | PO2 | PO3 | PO4 | PO5 | PO6 | PO7 | PO8 | PO9 | PO10 | PO11 | PSO1 | PSO2 |
| CO-1 |  |  |  |  |  |  |  |  |  |  | 3 |  |  |
| CO-2 |  | 3 |  |  |  |  |  |  |  |  |  |  |  |
| CO-3 |  |  |  |  |  |  |  |  |  | 3 |  |  |  |
| CO-4 |  |  |  |  |  | 3 |  |  |  |  |  |  |  |
| CO-5 | 3 | 3 |  |  |  |  |  |  |  |  |  |  |  |
| CO-6 |  |  |  |  | 3 |  |  |  |  |  | 3 |  |  |
| CO-7 |  |  | 3 | 3 |  |  |  |  |  |  |  |  |  |
| CO-8 |  |  |  |  |  |  |  |  |  | 3 |  |  |  |
| CO-9 |  |  |  |  |  |  |  |  |  | 3 |  |  |  |
| CO-10 |  |  |  |  |  |  |  | 3 |  |  |  |  |  |
| CO-11 |  |  |  |  |  |  |  |  | 3 |  |  |  |  |
| Average | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |  |  |

PSO mapping to be done by respective Dept.

Attainment level: - 1: Slight (low) 2: Moderate (medium) 3: Substantial (high)

POs: PO1: Engineering knowledge, PO2: Problem analysis, PO3:Design of solutions, PO4:Conduct investigations of complex problems, PO5: Engineering tool usage, PO6:Engineer and the world, PO7:Ethics, PO8:Individual and collaborative work, PO9:comunication,PO10:project management and finance,PO11: Life-long learning

**ABSTRACT**

Every day, millions of people and institutions communicate over the Internet. As the number of Internet users has rapidly increased over the past two decades, so too has the number of cyberattacks. Traditional signature-based methods, while effective for detecting known threats, fail to identify zero-day attacks. Anomaly-based detection methods offer an alternative, as they can recognize previously unseen attacks. In this study, the CICIDS 2017 dataset, known for its wide attack diversity and up-to-date information, is used to detect network anomalies through machine learning. Feature selection is performed using the Random Forest Regressor algorithm, and several machine learning models have been applied with high success rates, including Random Forest (94%), ID3 (95%), and K-Nearest Neighbors (97%).

This project aims to develop an advanced system for detecting anomalies in heterogeneous networks by utilizing machine learning and feature-fusion techniques. The system profiles normal network traffic patterns and identifies deviations such as spikes, protocol misuse, or suspicious IP addresses. PCAP files are analyzed to create a baseline, while various machine learning models are applied to detect anomalies. The feature-fusion approach combines multiple feature representations to improve detection efficiency and scalability.

Tools such as Wireshark and Python libraries, including Scapy, Pandas, and NumPy, are employed for data preprocessing and analysis. The fusion of different feature representations contributes to robust anomaly detection, particularly for zero-day attacks, thereby enhancing overall network security. The CICIDS 2017 dataset is used to train the models, enabling the system to detect a wide range of attacks in real-world scenarios.

**Contents**

Abstract

1. Introduction 1
2. Literature Survey 2
3. Problem Statement 17
4. System Requirements Specification 18
   1. Functional Requirements **18**
   2. Software and Hardware Requirements **18**
5. System Analysis 20
   1. Problem Identification **20**
   2. Objective of New System **20**
   3. Feasibility and Study **20**
   4. Existing System **21**
   5. Proposed Solution **21**
6. System Design 22

5.1 Data Collection 23

5.2 Preprocessing 23

5.3 Machine Learning and Anomaly Detection 23

5.4 Feature Fusion 23

1. Tools and technologies Used 27

7.1 Programming Languages 27

7.2 Machine Learning and Data Processing Libraries 27

1. Results 29
   1. Preprocessing 29
   2. Attack Filtering 31
   3. Classification Models 33
   4. Feature Fusion Technique 28
2. Applications 43
3. Conclusion 44

Bibilography 46

Appendices 48

**CHAPTER 1**

**INTRODUCTION**

Network security has become an increasingly critical concern as the frequency and sophistication of global cyberattacks continue to rise. Traditional signature-based detection methods, which are effective in identifying known threats, face significant limitations when it comes to detecting zero-day attacks—threats that are not yet cataloged or recognized. As a result, anomaly-based detection techniques are becoming indispensable in modern cybersecurity frameworks. This report proposes an advanced system that utilizes machine learning and feature-fusion techniques to detect anomalies within heterogeneous networks. The proposed system is trained and validated using the CICIDS 2017 dataset, which encompasses a broad spectrum of real-world cyberattacks, providing a robust foundation for model development and evaluation.

The rapid expansion of Internet usage, now exceeding 4 billion users globally, has been paralleled by a notable surge in cyberattacks. Signature-based detection methods rely on extensive databases of known attack patterns, which require frequent updates to remain effective. Despite this, such methods are inherently vulnerable to zero-day attacks—novel exploits that have not yet been incorporated into the signature database. In contrast, anomaly-based detection focuses on identifying deviations from the normal behavior of a network, enabling the detection of previously unknown and zero-day threats that signature-based systems fail to recognize.

Moreover, a significant portion of modern Internet traffic—more than half—is encrypted using SSL/TLS protocols, which presents a challenge for signature-based systems that rely on inspecting the content of data packets. Anomaly-based detection, however, does not depend on the specific content of the traffic but instead analyzes broader patterns, such as connection times, packet sizes, and other network metrics. This makes anomaly-based detection particularly well-suited for analyzing encrypted traffic, as it can detect abnormal patterns without needing to decrypt the data.

The ultimate goal of this study is to develop a scalable and adaptable anomaly detection system that can significantly enhance the security of heterogeneous networks. By integrating machine learning algorithms with feature-fusion techniques, the system aims to provide a comprehensive and efficient approach to detecting network anomalies. The system will be capable of identifying both known and novel threats, including zero-day attacks, by learning the normal patterns of network traffic and detecting deviations from these patterns.

**CHAPTER 2**

**LITERATURE SURVEY**

2.1 Datasets

In the detection of network anomalies using machine learning methods, there is a need for a substantial amount of both harmful and harmless network traffic for the training and testing phases. However, due to privacy concerns, real network traffic cannot be publicly shared. To address this issue, various datasets have been created and continue to be developed. This section provides information on some popular datasets, followed by a comparison and evaluation to determine the most suitable one for use in the implementation phase.

2.1.1 CICIDS 2017

Another notable dataset is the CICIDS 2017 (Intrusion Detection Evaluation Dataset) [16], created by the Canadian Institute for Cybersecurity at the University of New Brunswick. This dataset consists of a 5-day data stream (from July 3rd to July 7th, 2017) generated by computers using modern operating systems such as Windows Vista, 7, 8.1, 10, Mac, Ubuntu 12/16, and Kali. Detailed information about the dataset can be found in Table 1:

**Table 1**. Details of CICIDS2017 Dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Flow Recording Day** (Working Hours) | **pcap File size** | **Duration** | **CSV File Size** | **Attack Name** | **Flow Count** |
| Monday | 10 GB | All Day | 257 MB | No Attack | 529918 |
| Tuesday | 10 GB | All Day | 166 MB | FTP-Patator, SSH-Patator | 445909 |
| Wednesday | 12 GB | All Day | 272 MB | DoS Hulk, DoS  GoldenEye, DoS slowloris,  DoS Slowhttptest,  Heartbleed | 692703 |
| Thursday | 7.7GB | Morning | 87.7 MB | Web Attacks (Brute Force, XSS, Sql Injection) | 170366 |
| Afternoon | 103 MB | Infiltration | 288602 |
| Friday | 8.2GB | Morning | 71.8 MB | Bot | 192033 |
| Afternoon | 92.7 MB | DDoS | 225745 |
| Afternoon | 97.1 MB | PortScan | 286467 |

The CICIDS 2017 dataset offers several advantages over other datasets mentioned earlier:

* **Real-World Data**: The dataset consists of real-world data obtained from a testbed of actual computers, ensuring the data reflects genuine network traffic.
* **Diverse Operating Systems**: The data is collected from computers running up-to-date operating systems, including Windows, Mac, and Linux. This diversity is present in both attacker and victim systems, enhancing the dataset's realism.
* **Labeled Data**: The dataset is labeled, allowing machine learning methods to be applied effectively. Feature extraction, a critical step, has been performed, resulting in 85 features (listed in Appendix A).
* **Data Availability**: Both raw data (pcap files, containing captured network packets) and processed data (CSV files, which are comma-separated data files) are available for analysis, offering flexibility for researchers.
* **Comprehensive Attack Set**: The dataset includes a wide range of attacks, informed by the 2016 McAfee Security Report, ensuring it reflects up-to-date security threats.
* **Protocol Variety**: CICIDS 2017 supports an extensive range of protocols, including HTTPS (in addition to FTP, HTTP, SSH, and email protocols), making it more diverse than other datasets.

However, there are some drawbacks to consider:

* **Large File Sizes**: The raw data files are quite large (47.9 GB), and the processed data files are also substantial (1,147.3 MB), which may require significant storage capacity.
* **No Pre-Split Data for Training and Testing**: Unlike the KDD99 and NSL-KDD datasets, CICIDS 2017 does not provide separate training and testing files. Users must create these sections themselves, a process explained in the "Creation of Training and Test Data" section.
* **Potential Data Issues**: Unlike earlier datasets like DARPA98 and KDD99, which underwent an evolutionary process of improvement, CICIDS 2017 is relatively new and hasn't been widely studied. As a result, it may contain some minor errors, which are addressed in the "Data Cleansing" section.

After a thorough comparison, the decision was made to select the processed data (CSV files) from CICIDS 2017 for the implementation phase. The primary reasons for this choice include the dataset's up-to-date nature, its broader range of protocols and attacks, and the fact that its limited prior use in research presents an opportunity for a significant contribution to the literature.

## Anomaly and Attack types

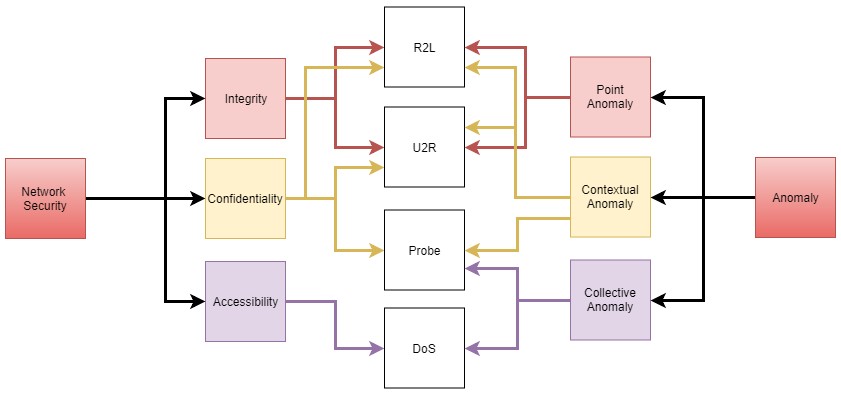
### **2.2.1** Network Attacks Types

Network security aims to protect networks from attacks that threaten the three fundamental principles of confidentiality, integrity, and availability [11, 17].

* **Confidentiality**: Information should be accessible only to authorized users, and unauthorized access must be prevented.
* **Integrity**: Only legitimate users should have the ability to add, modify, or delete information. Unauthorized individuals should not be able to alter data.
* **Availability**: The system should remain accessible to legitimate users at all times.

Network attacks are attempts to breach these essential principles, and they can be categorized into four main types:

1. **Denial of Service (DoS)**: In a DoS attack, the attacker abuses system resources to prevent legitimate users from accessing services. A common example is sending an overwhelming number of requests to a web server, causing it to crash or become unresponsive [11]. DoS attacks can be further divided into:
   * **Bandwidth Depletion**: The attacker floods the victim's network with excessive data flow to exhaust its bandwidth.
   * **Resource Depletion**: The attacker consumes system resources (such as memory or CPU) by sending numerous packets, ultimately rendering the system unresponsive [18].
2. **Probe (Information Gathering)**: These attacks are aimed at collecting critical information about the target system, such as its network structure, operating system, and the types and properties of connected devices. While these attacks do not directly harm the system, they are significant because they gather information that can be used to facilitate more damaging attacks in the future [11].
3. **U2R (User to Root)**: In a U2R attack, the attacker seeks to gain control of the administrative (root) account to access and steal sensitive resources. The attacker may exploit system vulnerabilities or perform brute-force attacks to obtain administrator privileges [11].
4. **R2U / R2L (Remote to User / Remote to Local)**: These attacks involve the attacker infiltrating the victim's network to gain sufficient privileges to send packets from the victim's machine. Similar to U2R, the attacker may exploit system vulnerabilities or use brute-force attacks to escalate their privileges [11].



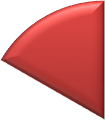
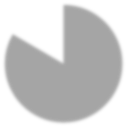
**Figure 2.** Network anomalies v/s network attacks

Classifying the network attacks according to the anomalies they create, can be useful in detecting the attack. Each attack causes differentiation (an anomaly) in the network.

 Attacks :

In this section, the types of attacks that the data set contains are examined in detail.

All data in the dataset are tagged with 15 labels. One (Benign) of these tags represents normal network movements while the other 14 represent attacks. The benign record, formed using Mail services, SSH, FTP, HTTP, and HTTPS protocols represent a non-harmful / normal data stream on the network, created by simulating real user data [6]. The names and numbers of these labels can be seen in Figure 3.



2359289

**83**

**%**

471454

**17**

**%**



Benign



Attack

**Figure 3**. The distribution of data flow and attack types in the dataset.

When examining the frequency of these attacks, it becomes clear that some attacks are significantly more prevalent than others. For example, the DoS HULK attack accounts for nearly half of all recorded attacks, while the PortScan attack represents about one-third of the total. This imbalance in attack distribution can be attributed to the inherent characteristics of these attacks. Both DoS and PortScan attacks generate large volumes of data and packet flows during execution. As a result, it is natural to observe heightened traffic during these attacks, which often leads to an increased presence of both normal traffic and other types of attacks within the dataset.

The attacks in the dataset can be explained as follows:

### DoS HULK

The HULK (HTTP Unbearable Load King) attack, found in the CICIDS 2017 dataset, is a type of DoS/DDoS attack that targets server load, potentially disabling it in under a minute. It works by generating TCP-SYN floods and HTTP-GET flood requests, while hiding the real user agent and using varied attack templates [19]. The HULK attack uses SYN and HTTP-GET flood methods, which can be understood by examining related attacks:

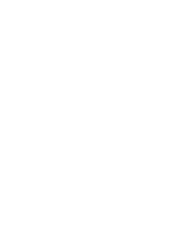
* **SYN Scan**: Sends SYN packets to multiple ports, awaiting a SYN/ACK response, leaving the connection incomplete [25].
* **TCP Connect Scan**: Attempts to establish numerous TCP connections to detect open/closed ports, terminating them after detection [25].
* **ACK Scan**: Sends ACK packets to different ports, expecting a SYN/ACK response and leaving the handshake incomplete [25].
* **FIN Scan**: Sends FIN packets to multiple ports, learning open/closed ports based on RST responses from closed ports [25].
* **NULL Scan**: Sends packets with no flags, relying on RST responses from closed ports to identify them [25].
* **XMAS Scan**: Sends packets with FIN, PSH, and URG flags, similarly detecting open/closed ports via RST responses [25].
* **UDP Scan**: Tries to establish multiple UDP connections across ports [25].

**Fragmentation Attack**: In this attack, packets are split into small chunks to bypass firewalls, with multiple small packets indicating this type of attack [25].

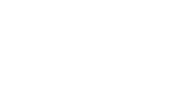
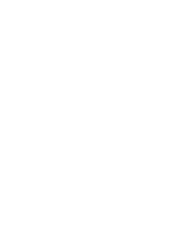
**DDoS**: The DDoS attack in the CICIDS 2017 dataset, performed using the LOIC program, involves sending HTTP, TCP, and UDP requests and is recorded in the Friday afternoon section. Unlike a DoS attack, a DDoS attack originates from multiple computers, often using "zombies" infected by malware, forming a "botnet" to overwhelm a target [26][17].



Attacker



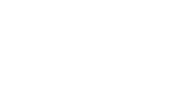
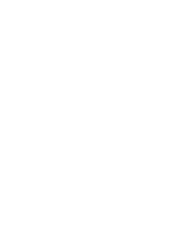
Victim



Agent



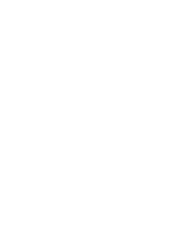
Zombies



Agent



Zombies



**Figure**

**6**

**.**

Representation of a DDoS Attack

**DoS Goldeneye**  
The **DoS Goldeneye** attack [27] appears in the Wednesday records alongside other DoS attacks in the CICIDS 2017 dataset [4]. This Python-based attack aims to consume system resources, preventing legitimate users from accessing services. Goldeneye uses multithreading to efficiently launch an HTTP flood attack, utilizing CPU and memory. It does not encrypt packets or spoof IP addresses, and it works on Linux, Mac, and Windows systems. By using the "Keep-Alive" method and disabling HTTP Cache-Control, Goldeneye maximizes data transmission and quickly consumes resources [20].

**FTP-Patator**  
The **FTP-Patator** attack, recorded on Tuesday morning in the CICIDS 2017 dataset [4], uses the multithreaded **Patator** tool [28]. This brute-force attack targets FTP login credentials by repeatedly attempting different username and password combinations. It is characterized by a high number of failed login attempts in a short period, but the packets involved are small in size, so the attack consumes low bandwidth [29].

**SSH-Patator**  
The **SSH-Patator** attack, recorded on Tuesday afternoon in the CICIDS 2017 dataset [4], also uses the **Patator** tool [28]. The attack is performed in three phases: scanning, brute-forcing, and die-off. First, the attacker scans for SSH hosts, then attempts to brute-force login credentials with numerous combinations. If successful, the attacker gains access to the system. In the scanning phase, incomplete TCP packets with SYN flags are seen, while the brute-force phase shows many small, completed TCP packets [31].

**DoS Slowloris**  
The **DoS Slowloris** attack, recorded on Wednesday in the CICIDS 2017 dataset [4], targets system resources to prevent legitimate access. Written in Perl, it creates a TCP-SYN flood similar to the DoS HULK attack. Slowloris maintains many long-lived connections with a target server, consuming resources by keeping connections open as long as possible. The attack generates many small-sized, incomplete TCP packets, leading to low bandwidth consumption [20].

**DoS SlowHTTPTest**  
The **DoS SlowHTTPTest** attack, also recorded on Wednesday, exploits TCP window size features to overload the target server's resources. By sending a normal request and then setting the window size close to zero, the attacker slows down the data transfer process, causing the server to allocate excessive resources. The attack is low-bandwidth and difficult to detect, as it mimics legitimate HTTP requests. Connections with unusually small window sizes can indicate this attack [33][34].

## Machine Learning

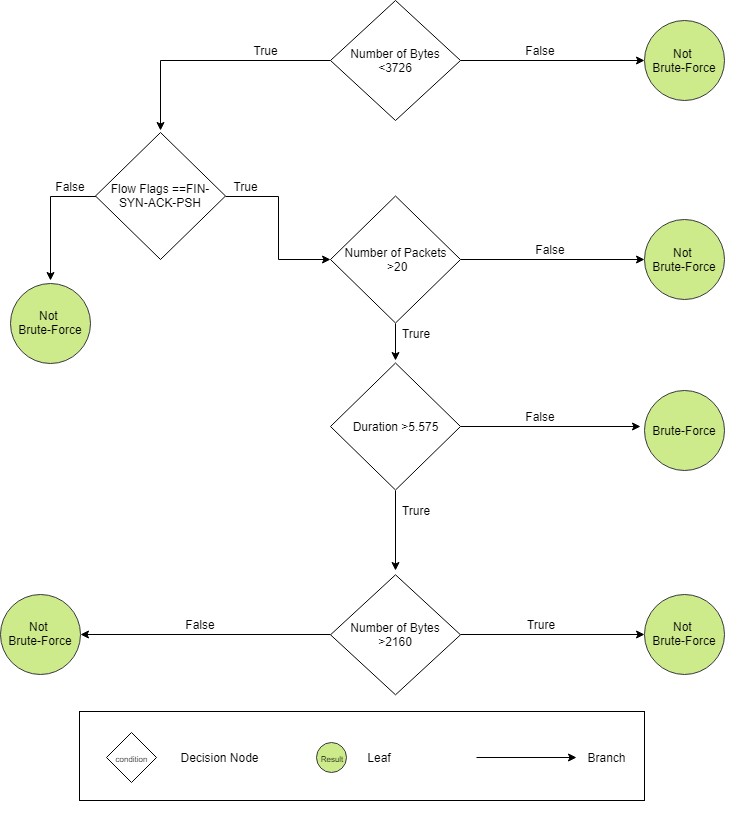
Machine learning enables computers to learn from data with minimal human intervention [40]. It is particularly useful for solving complex problems where traditional methods fail. Machine learning can handle large, intricate datasets, solve problems that are difficult for classical methods, and adapt to dynamic environments.

There are four main types of machine learning:

* **Supervised Learning**: In this method, data is labeled and classified. The algorithm compares its predictions with the known labels to assess its performance. While effective, it is costly due to manual labeling. Common algorithms include Decision Trees, K-Nearest Neighbors, and Random Forests.
* **Unsupervised Learning**: This method does not require labeled data. The algorithm groups data based on similarities, making it useful for anomaly detection and clustering. It is cost-effective because it doesn't require labeled data.
* **Semi-supervised Learning**: This hybrid method combines supervised and unsupervised learning, using a small portion of labeled data with mostly unlabeled data. It balances the high accuracy of supervised learning with the low cost of unsupervised learning.
* **Reinforcement Learning**: Unlike the others, this approach rewards correct actions and penalizes incorrect ones, allowing the algorithm to learn and develop its own rules [40].

In this project, **supervised learning** methods will be used for their high performance and use of labeled datasets. The algorithms chosen for the application phase include Naive Bayes, QDA, Random Forest, ID3, AdaBoost, MLP, and K Nearest Neighbors. These algorithms represent a variety of popular techniques with distinct characteristics.

**Decision Trees** are a widely used classifier in machine learning, known for their simplicity and interpretability. A decision tree consists of nodes (including the root and sub-nodes), branches, and leaves, where each node contains a decision rule. The algorithm follows branches based on these decisions until it reaches a leaf, which represents the outcome.



**Figure 7.** Detection of brute-force attack using decision trees [29]*.*

### Decision trees use a divide-and-conquer approach to transform large, complex datasets into smaller, more meaningful groups, making them ideal for classifying complex data [41]. However, they can suffer from overfitting, which leads to poor generalization. A small misclassification early on can result in misleading branches and inaccurate outcomes. To mitigate this, preprocessing of the data is often required.

### In this project, the **ID3 (Iterative Dichotomiser 3)** decision tree algorithm is used. ID3 is effective when the training set has many features, and it offers a balance between computation and performance. Notably, ID3 can connect more than two branches to decision nodes, making it versatile for various decision-making scenarios [41].

### Random Forest

### **Random Forest** Random forest is a machine learning technique that creates a "forest" of decision trees, each built from a random sample of the training data. The algorithm then combines the results from these trees through a voting process, with the most frequent outcome chosen as the final result [42].

### Advantages of random forest include:

### Effective for large, complex datasets

### Rarely suffers from overfitting

### Can handle missing data by replacing lost values

### Assesses variable importance, making it useful for feature selection [43].

### However, its complexity and difficulty in understanding the model's functioning are notable drawbacks.

### K Nearest Neighbour

KNN is a simple, fast, and popular machine learning algorithm that identifies the class of an unknown sample by examining its nearest neighbors in the dataset. The number of neighbors, K, is specified, and the most common class among the nearest K samples is assigned to the unknown data point [41]. While KNN performs well with multidimensional data and is fast during training, it is slower during the estimation phase [43].



K=

5



K=

2

**Figure 8.** Operation of KNN algorithm for K = 2 and K = 5 values.

### MLP



Input Layer



Hidden Layer

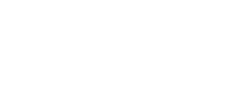


Output Layer



A

neuron



A neuron

connection

**Figure 10.** Demonstration of a three-layer MLP

MLP is a type of artificial neural network inspired by the human brain's structure. It processes data through an input layer, hidden layers, and an output layer. MLP can handle complex problems, missing data, and generalizes well after learning. However, it is challenging to build the network structure, can suffer from overfitting, and is hard to interpret [43, 45].

Various studies have explored machine learning for anomaly detection in computer networks. Notable works include:

* Chebrolu et al. (2005) combined Bayesian networks with Classification and Regression Trees (CART) for better attack detection, achieving high accuracy in most attack types.
* Suresh and Anitha (2011) focused on DDoS attacks using multiple algorithms like Naive Bayesian and SVM, reporting accuracy rates above 96% for most methods.
* Sharafaldin et al. (2017) applied seven machine learning methods (including Naive Bayes, KNN, and Random Forest) on the CICIDS2017 dataset to detect 15 types of attacks, with Random Forest achieving 0.97% accuracy.

This work builds on the Sharafaldin et al. study due to its recent nature, up-to-date dataset, and wide array of machine learning algorithms.

**CHAPTER 3**

**PROBLEM STATEMENT AND OBJECTIVES**

**3.1 PROBLEM STATEMENT**

Anomaly detection in heterogeneous networks using machine learning and feature-fusion techniques focuses on identifying unusual behaviors that could signify potential security threats. Machine learning models are trained to differentiate between normal and attack traffic patterns, enabling the detection of a broad spectrum of cyberattacks. Feature-fusion integrates multiple data sources, improving detection accuracy, even in the presence of complex or encrypted network traffic.

3.2 OBJECTIVES

Train machine learning models on the CICIDS-2017 dataset to maximize accuracy in detecting anomalies within heterogeneous networks.

Implement feature-fusion techniques to select relevant features from the 36 dataset attributes, enhancing the performance of the anomaly detection models.

Evaluate and compare the effectiveness of different machine learning algorithms on the same dataset to identify the best model for anomaly detection.

Measure and compare various performance metrics—such as accuracy, precision, and recall—of the different machine learning algorithms in detecting network anomalies.

**CHAPTER 4**

**SYSTEM REQUIREMENT SPECIFICATION**

**Requirement Overview:**

The study uses the CICIDS-2017 dataset, which includes network traffic data with both normal and attack traffic. This dataset, containing 36 attributes, is key for training and validating machine learning models for anomaly detection in heterogeneous networks. Machine learning algorithms like Random Forest, K-Nearest Neighbours (KNN), and ID3 will be used to analyze the dataset and compare their effectiveness in detecting anomalies.

**4.1 Functional Requirements:**

* **Performance**: Measured using metrics like confusion matrix, F1 score, accuracy, precision, and recall.
* **Usability**: The user interface should be intuitive and user-friendly.
* **Reliability**: The system must maintain high classification accuracy with minimal errors.
* **Scalability**: Should handle increasing data volume and computational demand over time.
* **Maintenance**: Easy to maintain, with provisions for updates, retraining, and bug fixes.

**4.2 Software and Hardware Requirements:**

*Software:*

* **Operating System**: Compatible with Linux, Ubuntu, Windows, and macOS (used in this project).
* **Programming Language**: Python, for its extensive libraries and frameworks.
* **Development Tools**: Jupyter Notebook, Visual Studio Code.
* **Libraries**: TensorFlow, Keras, Numpy, Pandas, Scikit-learn.

*Hardware:*

* **Computational Power**: Sufficient CPU/GPU resources, with GPUs recommended for accelerated training.
* **Memory (RAM)**: Adequate memory for large datasets.
* **Storage**: Enough capacity for datasets, models, and project files.
* **Networking**: Stable internet connection for accessing resources and collaborating.

**CHAPTER 5**

**SYSTEM ANALYSIS**

**Current System Evaluation**

Existing anomaly detection systems in heterogeneous networks largely rely on traditional methods like signature-based detection and rule-based systems. While effective at detecting known threats, these methods struggle with complex attacks, encrypted traffic, and the diverse devices, protocols, and traffic types found in modern networks.

Problem Identification: The key challenges of current systems include:

* Complex Attack Detection: Limited ability to identify evolving or sophisticated attacks.
* Encrypted Traffic: Difficulty analyzing encrypted traffic (e.g., SSL/TLS), which is increasingly prevalent.
* Complex Network Environments: Diverse devices, protocols, and traffic patterns lead to false positives and missed attacks.

Objective of the New System: The proposed system aims to design an anomaly detection framework using machine learning algorithms and feature-fusion techniques. The goal is to detect both known and zero-day attacks by analyzing traffic patterns and enhancing detection accuracy with feature-fusion.

Feasibility Study: The system is feasible with the CICIDS-2017 dataset, which offers a rich set of labeled network traffic data and 36 features for model training and evaluation. Modern computing platforms can support the required computational resources, making this approach scalable for real-time anomaly detection.

Existing Systems: Traditional anomaly detection systems often rely on signature-based methods, which have several drawbacks:

* Unknown Attacks: Unable to detect new or unseen threats.
* Encrypted Traffic: Struggles with encrypted network traffic.
* False Positives: Frequent in heterogeneous networks due to reliance on predefined rules.
* Scalability: Issues arise when scaling to large, dynamic networks.

Proposed Solution: The new solution combines machine learning algorithms and feature-fusion techniques for better anomaly detection:

* Machine Learning Algorithms: Random Forest, K-Nearest Neighbors (KNN), and ID3 are used to classify network traffic as benign or malicious, ensuring high accuracy for both known and unknown attacks.
* Feature-Fusion: Combining relevant features from the CICIDS-2017 dataset (e.g., packet sizes, connection times) improves accuracy by providing a comprehensive view of traffic patterns.
* Wide Attack Detection: The system can detect a broad range of attacks, including DoS, DDoS, PortScan, and Web Attacks.
* Encrypted Traffic Handling: Unlike signature-based systems, this solution focuses on traffic patterns, enabling effective analysis of encrypted traffic.

This approach offers a scalable, adaptable solution for detecting anomalies in diverse and dynamic network environments.

**CHAPTER 6**

**SYSTEM DESIGN**

The system is designed to process network traffic data, profile normal behaviors, and detect anomalies that may indicate malicious activities. It achieves this through a combination of live traffic analysis and pre-existing datasets, utilizing machine learning algorithms for accurate classification. The system handles input data in the form of PCAP files and the CIC IDS 2017 dataset, a benchmark dataset for network intrusion detection. The architecture incorporates a streamlined workflow consisting of data collection, preprocessing, feature fusion, and machine learning analysis to ensure efficiency and scalability.

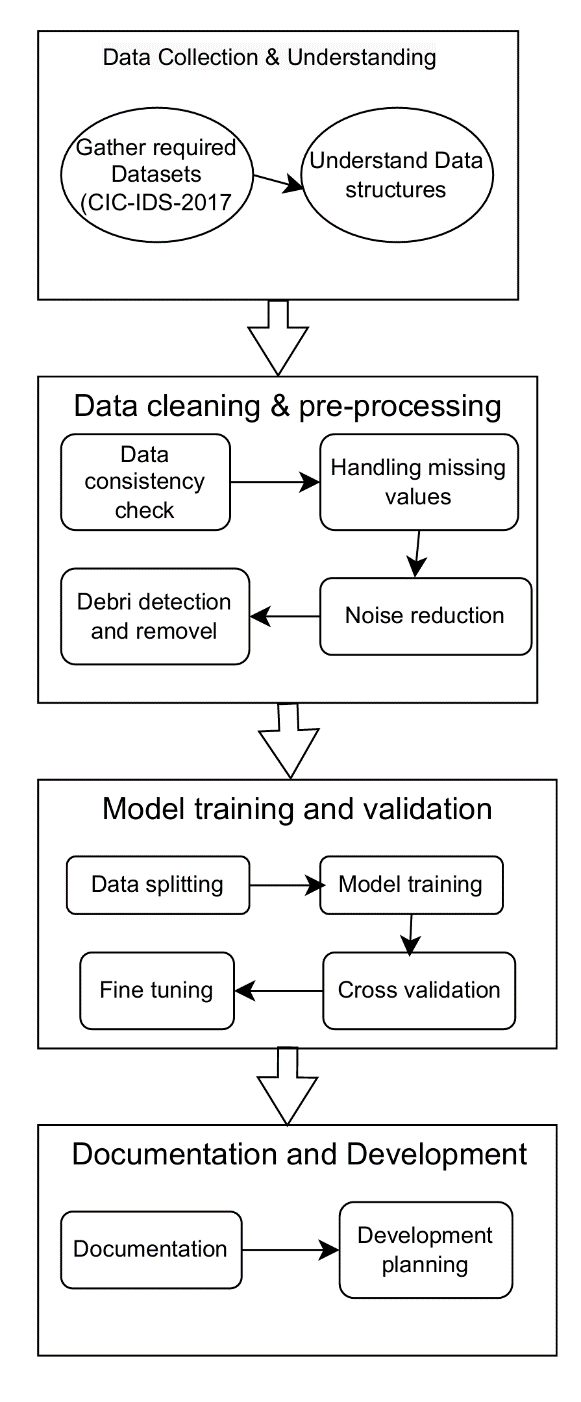


Figure 6.1 System Architecture

Data Collection  
The process begins with collecting network traffic data using tools like Wireshark to capture live traffic in PCAP format. Wireshark records detailed packet information, including source and destination IPs, ports, protocols, timestamps, and packet sizes. In addition to live traffic, the CIC IDS 2017 dataset is used, which contains both normal and malicious traffic patterns, making it ideal for training machine learning models. The dataset includes over 80 features, such as flow duration, packet size, and inter-arrival times.

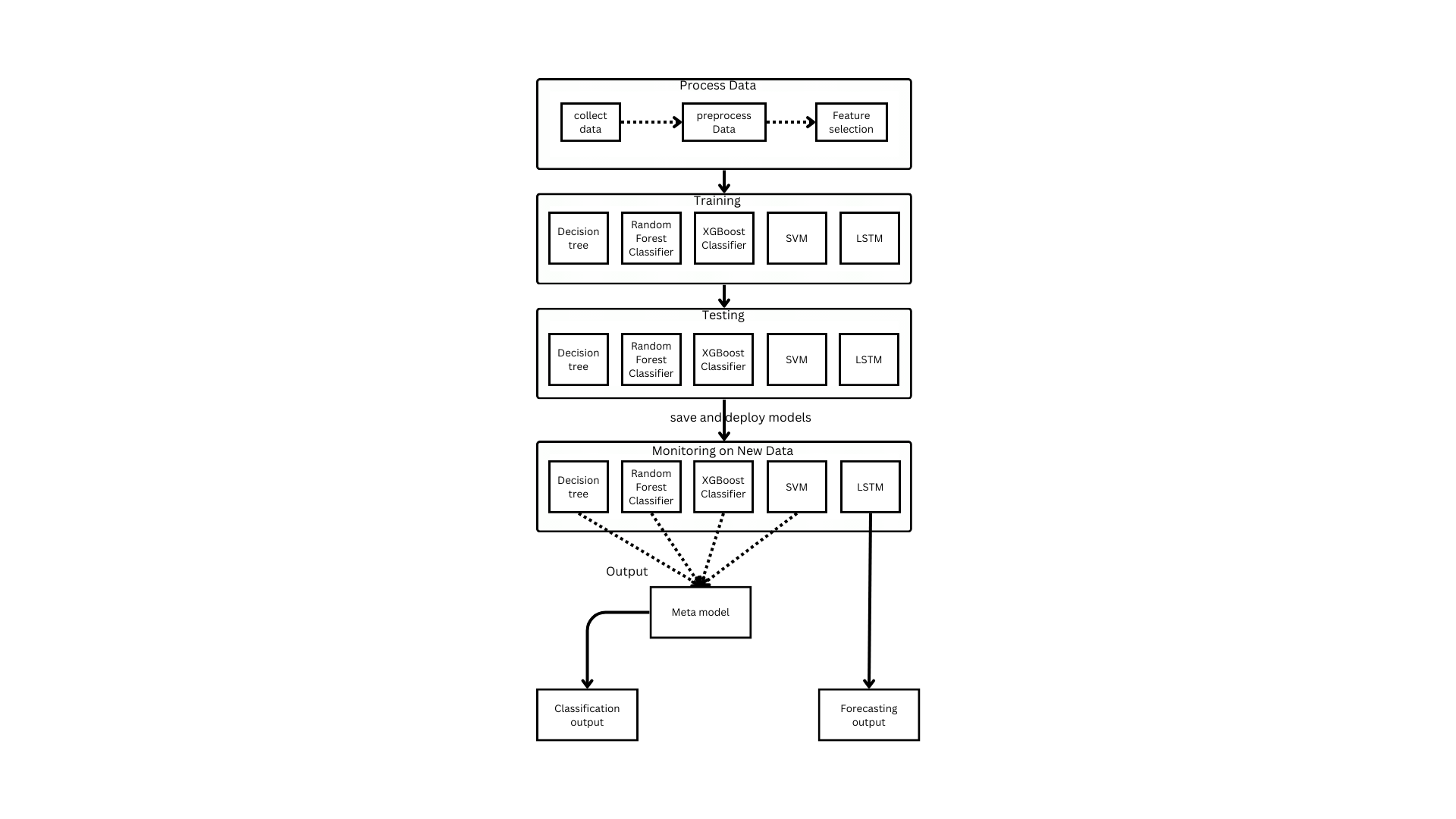
Preprocessing  
Once collected, the data undergoes preprocessing to ensure it's clean and ready for analysis. Raw PCAP files are parsed with libraries like Scapy or PyShark, while the CIC IDS 2017 dataset is imported using Pandas. During preprocessing, missing values are handled, mixed data types are resolved, and continuous features are normalized. This ensures the data is free from errors, allowing for accurate analysis. Libraries like NumPy and Pandas are key in these operations.

Machine Learning and Anomaly Detection  
Before feature fusion, machine learning algorithms are used for profiling and detecting anomalies. Algorithms like Naive Bayes, Random Forest, k-Nearest Neighbors (KNN), and AdaBoost classify traffic patterns from raw and partially processed data. The CIC IDS 2017 dataset, with labeled instances, is used for training and testing, while live traffic is used for validation. These models learn to distinguish between normal and anomalous behaviors, with performance metrics such as precision, recall, F1-score, and confusion matrices used for evaluation.

Feature Fusion  
After machine learning, feature fusion is performed to enhance data representation. Key attributes like packet size, flow duration, and flag counts are combined, and additional features are engineered, such as ratios between forward and backward packet lengths or average byte rates. Dimensionality reduction techniques like Principal Component Analysis (PCA) may also be applied to refine the feature set. The resulting enriched feature matrix enables more detailed analysis and improved anomaly detection.

**HIGH LEVEL DESIGN**

Fig 7.1 High level design of proposed work



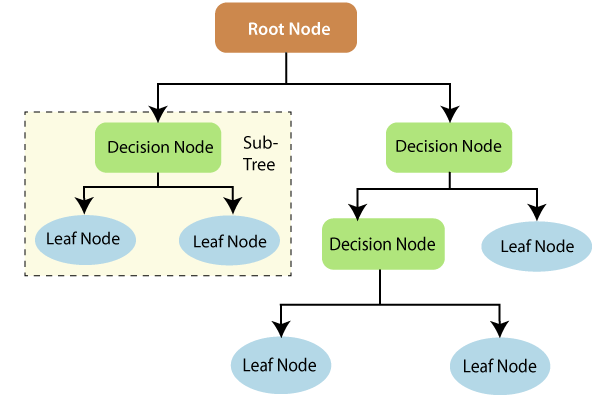
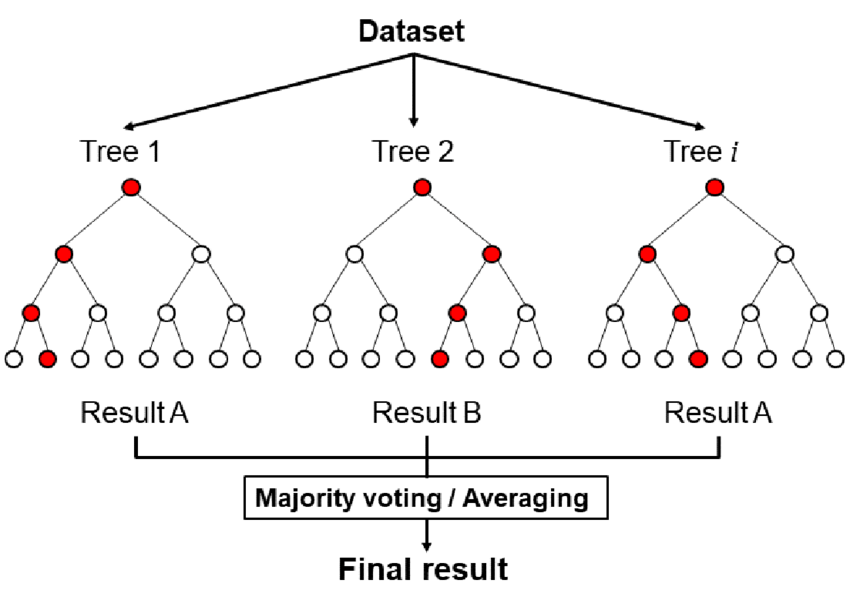


Fig 7.3 Architecture of Random Forest Classifier Model

Fig 7.2 Architecture of Decision Tree Model

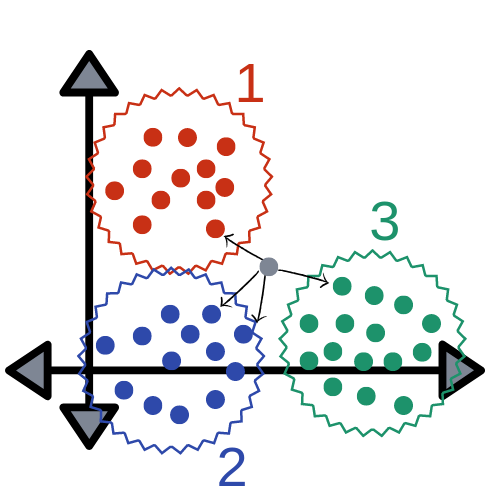
****

Fig 7.4 Architecture of KNN Model

**High level design and working of the proposed model**

Objective: Prepare raw data for analysis and model training.  
Steps:

* Collect Data: Gather raw data from sources like network logs, user activity, or mobile traffic data.
* Preprocess Data: Clean the data by handling missing values, removing duplicates, normalizing values, and encoding categorical variables to ensure the data is ready for model training.
* Feature Selection: Identify the most relevant features for the model using techniques like correlation analysis, mutual information, or algorithms that rank features by importance.

Training  
Objective: Train machine learning models using the processed data.  
Steps:

* Random Forest Classifier: Use an ensemble method to build multiple decision trees and combine their outputs to improve performance and reduce overfitting.
* ID3 (Iterative Dichotomiser 3): Use a decision tree algorithm that selects the feature with the highest information gain to split the data at each node.
* K-Nearest Neighbors (KNN): Use a simple, non-parametric algorithm that classifies data based on the most common class among the k-nearest data points.

Testing  
Objective: Evaluate model performance on a separate test dataset to ensure generalization to unseen data.  
Steps:

* Test each trained model (Random Forest, Decision Tree, XGBoost, SVM, LSTM) using a hold-out test set.
* Evaluate performance using metrics like accuracy, precision, recall, F1-score (for classifiers), and MAE, RMSE (for regression).

Save and Deploy Models  
Objective: Save and deploy models for real-time use.  
Steps:

* Save Models: Serialize the trained models for storage, typically using formats like Pickle (Python) or ONNX (for interoperability).
* Deploy Models: Set up the models in a production environment for real-time predictions, accessed via APIs or other interfaces.

Monitoring on New Data  
Objective: Monitor the deployed models on new data and track performance.  
Steps:

* Monitor Models: Continuously track model performance on incoming data to ensure predictions remain accurate, detect drift, and retrain when necessary.
* Meta Model: Combine individual model outputs using a meta-model or ensemble method to improve overall prediction performance.

Output:

* Classification Output: Predictions related to classification tasks, such as identifying types of traffic or detecting anomalies.

**CHAPTER 7**

**Tools and Technologies Used**

**7.1 Programming Languages Used in the Report:**

**Python**:

Primary language for machine learning tasks due to its extensive libraries (TensorFlow, Keras, Scikit-learn) and data manipulation tools (NumPy, Pandas).

Used for training, testing, and evaluating models like Random Forest, KNN, and ID3.

**R (optional)**:

Sometimes used for statistical analysis and visualization if deeper insights into network data patterns are required.

**Bash**:

To handle server-side deployment scripts or manage computational resources efficiently.

**7.2 Machine Learning and Data Processing Libraries Used in the Report:**

**Machine Learning Libraries**:

**Scikit-learn**: For implementing algorithms like Random Forest, KNN, and ID3, and for evaluation metrics (accuracy, precision, recall).

**TensorFlow**: For advanced modeling and experimentation, especially in feature-fusion techniques.

**Keras**: High-level API for neural networks, integrated with TensorFlow for streamlined model building.

**Data Processing Libraries**:

**NumPy**: For numerical operations, array manipulation, and handling datasets efficiently.

**Pandas**: For data preprocessing, cleaning, and manipulation using DataFrames.

**Visualization Libraries** (optional):

**Matplotlib** and **Seaborn**: For creating detailed visualizations of network behaviors and model performance.

**CHAPTER 8**

**RESULTS**

**8.1 Preprocessing**

Once data is collected, preprocessing ensures it is clean, consistent, and ready for analysis. Raw PCAP files are parsed using tools like Scapy or PyShark, while the CIC IDS 2017 dataset is imported via Pandas. This stage handles missing values, resolves mixed data types, and normalizes continuous features, ensuring data accuracy and consistency for efficient analysis. Python libraries like NumPy and Pandas are key in implementing these steps.

Steps in Data Preprocessing:

* Parsing Raw PCAP Files:  
  Raw PCAP files from tools like Wireshark are parsed with Scapy or PyShark. These libraries extract network traffic features, such as packet sizes, flow duration, source/destination IPs, and protocols, converting them into a structured format suitable for analysis.
* Importing the CIC IDS 2017 Dataset:  
  The CIC IDS 2017 dataset is imported into the analysis environment using Pandas. This dataset includes normal and malicious traffic data with over 80 features, providing a rich resource for training machine learning models.
* Handling Missing Values:  
  Missing or incomplete data is addressed by techniques such as imputation or removing rows with missing values. This ensures the dataset is complete and reliable for model training.
* Resolving Mixed Data Types:  
  The dataset may contain mixed data types (e.g., numeric and categorical). Categorical variables are encoded into numeric values using one-hot or label encoding, ensuring consistency for analysis.
* Normalization of Continuous Features:  
  Continuous features like packet size and flow duration are normalized using techniques such as min-max scaling or z-score normalization. This prevents features with larger ranges from biasing the model.
* Processing Each CSV File:  
  For each CSV file:
  + A new CSV is created and the header is written.
  + Invalid or missing values (e.g., "NaN", "Infinity") are replaced with "0".
  + Special characters (e.g., "–") are replaced with standard ones (e.g., "-").
  + Cleaned data is written to a new file.
* Data Cleaning and Transformation:  
  After cleaning, the dataset is read using Pandas:
  + Missing values are filled with "0" using fillna(0).
  + Non-numeric columns (e.g., "Flow Bytes/s") are converted to numeric, replacing values like "Infinity" with -1 or 0.
  + Categorical columns are encoded with LabelEncoder to convert them into numeric values.
  + Unnecessary columns are removed.
* Merging Files:  
  All cleaned and transformed data from multiple CSV files are merged into a single file (all\_data.csv). The header is written once, and subsequent files are appended with their respective data.

**8.2 Attack Filtering**

Attack filtering separates malicious traffic from benign data in network datasets, helping train machine learning models for attack detection. It organizes traffic into specific attack types, enhancing model accuracy and network security.

**Attacks Analysis**

Identifying and classifying various types of malicious activities targeting networks and systems.

| **Label** | **Count** |
| --- | --- |
| BENIGN | 2,359,289 |
| DoS Hulk | 231,073 |
| PortScan | 158,930 |
| DDoS | 41,835 |
| DoS GoldenEye | 10,293 |
| FTP-Patator | 7,938 |
| SSH-Patator | 5,897 |
| DoS slowloris | 5,796 |
| DoS Slowhttptest | 5,499 |
| Web Attack - Brute Force | 1,507 |
| Web Attack - XSS | 652 |
| Infiltration | 36 |
| Web Attack - SQL Injection | 21 |
| Heartbleed | 11 |

**Description of the features in the dataset**

· **BENIGN**: Normal, non-malicious traffic.

· **DoS Hulk**: A type of Denial of Service (DoS) attack that floods the network with HTTP requests.

· **PortScan**: An attempt to detect open ports or services on the network.

· **DDoS**: Distributed Denial of Service attack, where multiple systems are used to flood a target.

· **DoS GoldenEye**: A DoS attack similar to Hulk but with different packet patterns.

· **FTP-Patator**: Brute-force attacks against the FTP protocol.

· **SSH-Patator**: Brute-force attacks against the SSH protocol.

· **DoS slowloris**: A DoS attack that keeps many connections open to exhaust resources.

· **DoS Slowhttptest**: Another resource-exhausting DoS attack.

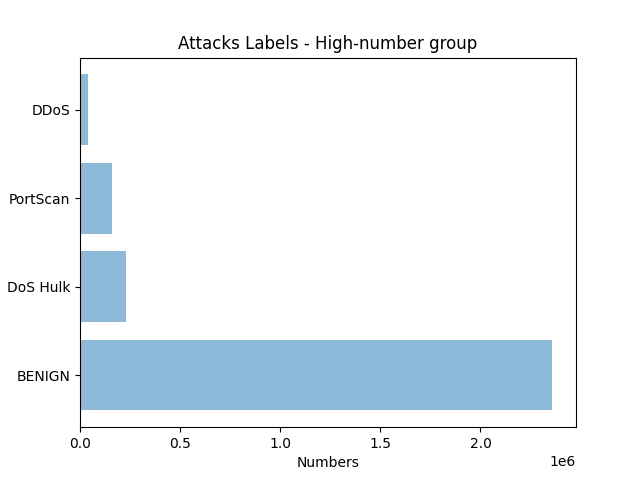
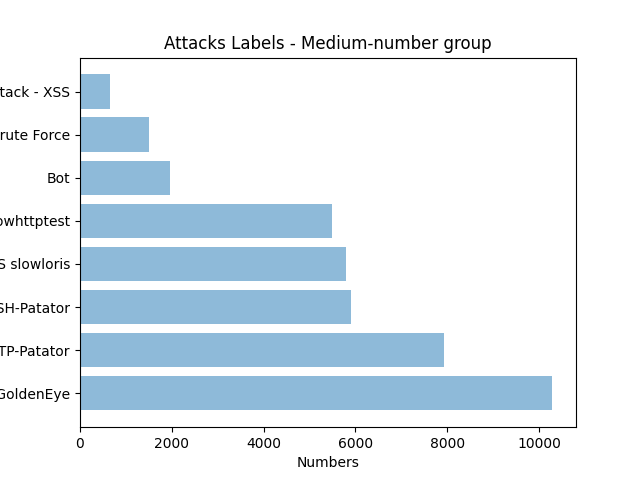
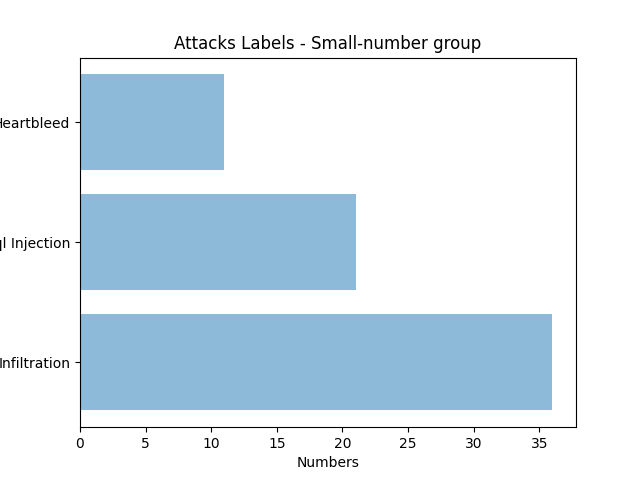
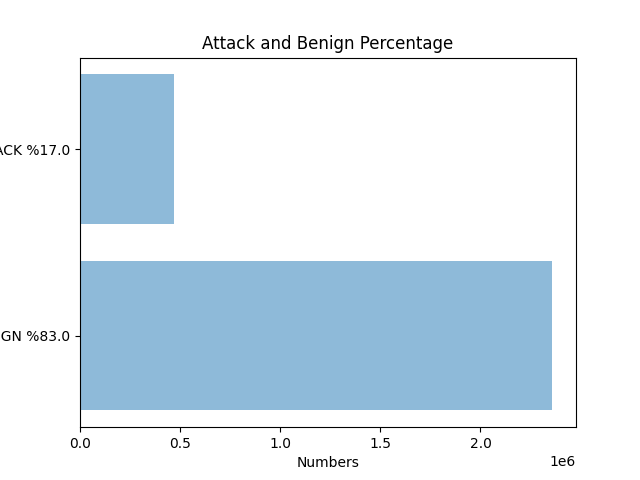
· **Bot**: Malicious bot activity, often used for spreading malware or performing attacks.

· **Web Attack - Brute Force**: Attempts to guess credentials via multiple login attempts on web applications.

· **Web Attack - XSS**: Cross-Site Scripting attack, used to inject malicious scripts into a website.

· **Infiltration**: Attempts to gain unauthorized access or move within a network.

· **Web Attack - Sql Injection**: SQL Injection attack, targeting databases by injecting malicious SQL queries. And lastly **Heartbleed**: An attack exploiting the Heartbleed vulnerability in OpenSSL to extract sensitive

data.

**8.3 Classification models used:**

Following are the models used and their pseudocode

* **ID3 Algorithm**

Pseudocode:

Input: Dataset S, Attributes A, Target T (class labels)

Output: Decision Tree

1. If all instances in S belong to the same class:

Return a leaf node with that class.

2. If A is empty:

Return a leaf node with the majority class in S.

3. Select the attribute A\_best with the highest Information Gain.

4. Create a root node labeled A\_best.

5. For each value v of A\_best:

a. Let S\_v = subset of S where A\_best = v.

b. If S\_v is empty:

Attach a leaf node with the majority class in S.

Else:

Recursively apply ID3 on S\_v and A - {A\_best}.

6. Return the constructed tree.

ID3 for different data sets:

1. Dataset with 1000 tuples:

A graph with numbers and a bar chart

Description automatically generated with medium confidence A screenshot of a computer screen

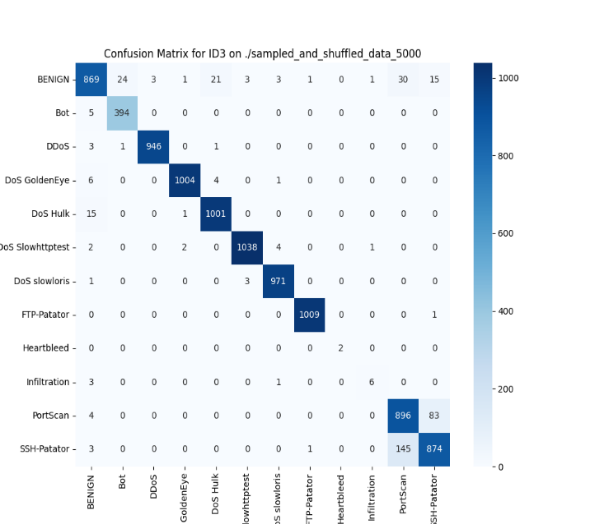
Description automatically generated

classification report for ID3 with 1000 tuples

Confusion matrix of ID3

With 1000 tuples

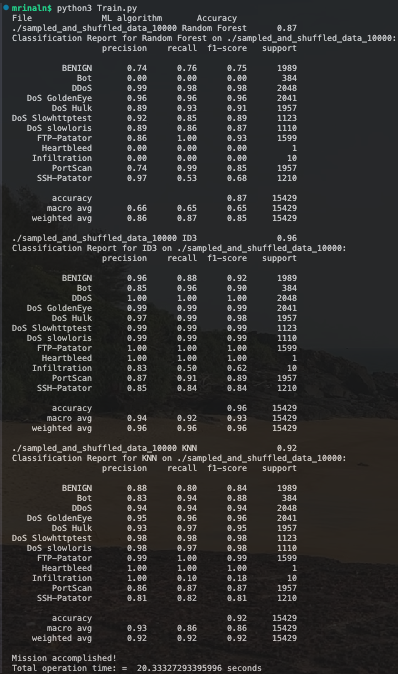
1. Dataset with 5000 tuples:

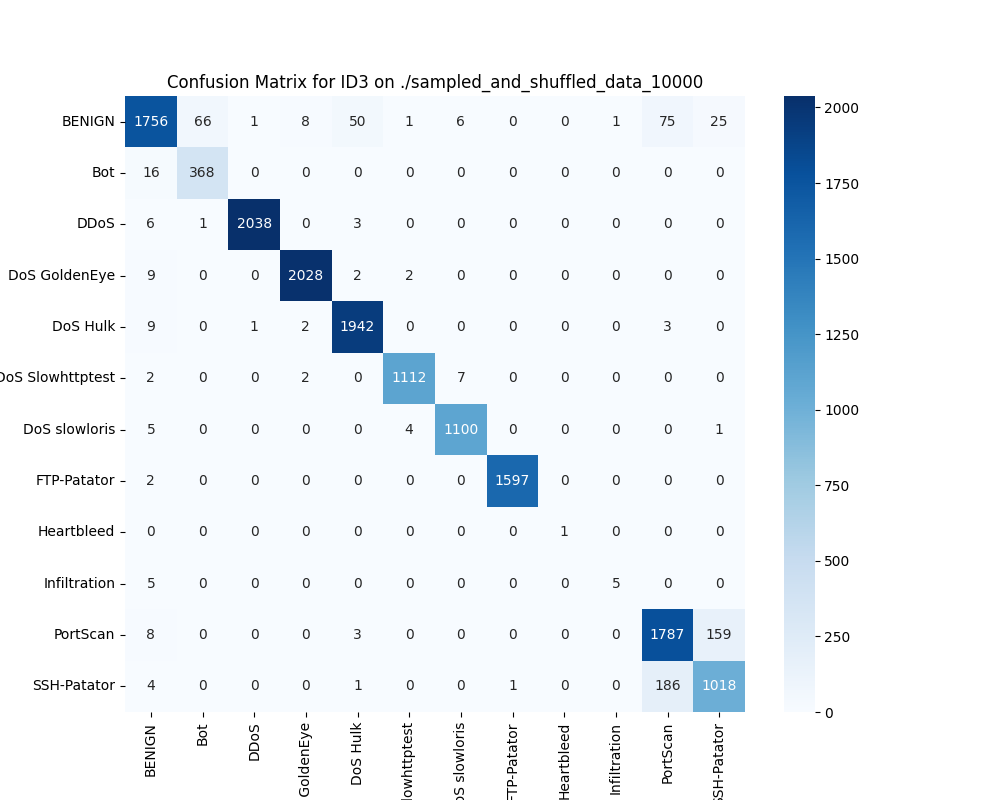


A screenshot of a computer screen

Description automatically generated

1. Dataset with 10000 tuples:





* **Random Forest classifier**

Pseudocode:

Input: Training dataset D, Number of trees T, Features per split F

Output: Random Forest Model

1. For each tree (i = 1 to T):

a. Bootstrap sample from D.

b. Build a decision tree using F features per split.

2. For classification:

a. Get majority vote from all T trees.

3. For regression:

a. Average predictions from all T trees

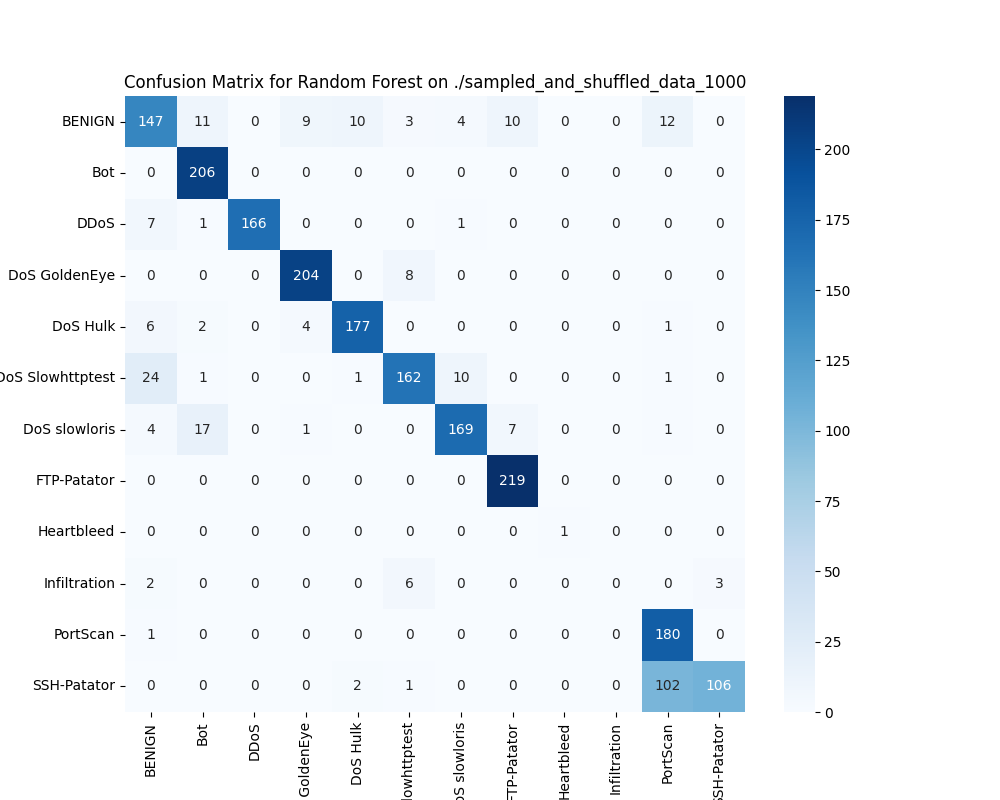
4. Return the Random Forest model.

Random forest for different data sets:

1. Data set with 1000 tuples:

A screenshot of a computer screen

Description automatically generated



1. Dataset with 5000 tuples:

A graph with numbers and a bar chart

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. Dataset with 10000 tuples:

A screenshot of a computer screen

Description automatically generated

A graph with numbers and a bar chart

Description automatically generated with medium confidence

* **K-Nearest Neighbours algorithm:**

Input: Training dataset D, Query Q, Number of neighbors K

Output: Predicted class for Q

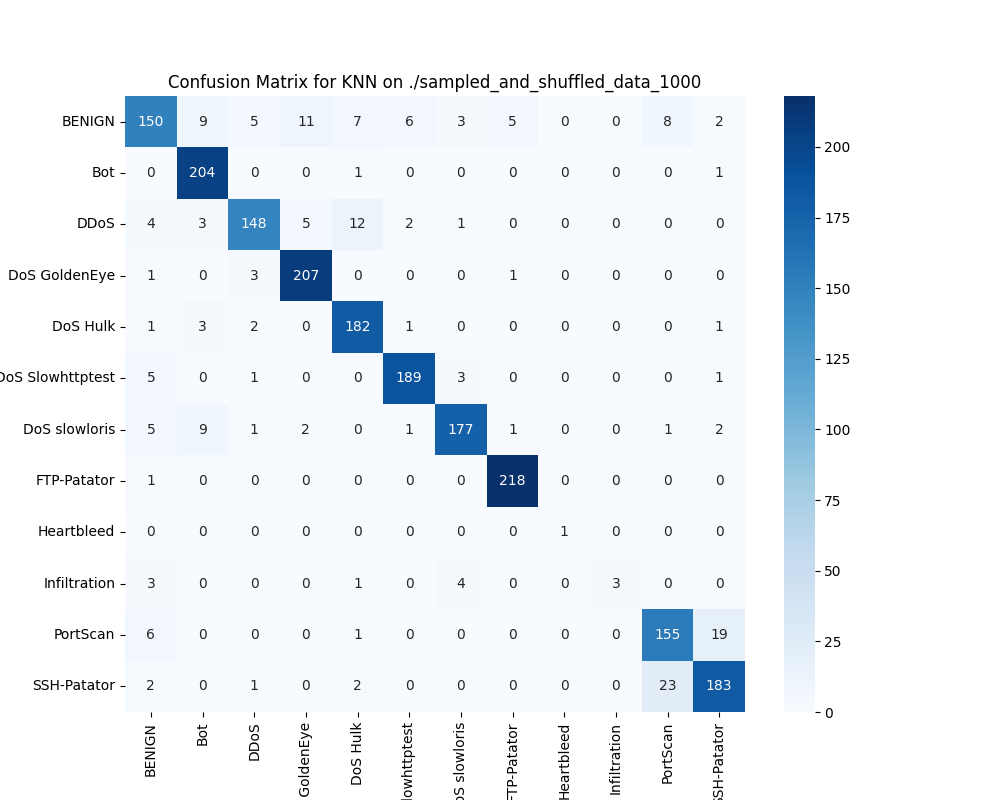
1. Calculate the distance between Q and each instance in D.

2. Sort instances by distance.

3. Select the K nearest neighbors.

4. Assign the majority class label among the K neighbors to Q.

1. Dataset with 1000 tuples:



A screenshot of a computer screen

Description automatically generated

1. Dataset with 5000 tuples:

A screenshot of a computer screen

Description automatically generated

A graph with numbers and a bar chart

Description automatically generated

1. Dataset with 10000 tuples:

A screenshot of a computer screen

Description automatically generated

A graph with numbers and a bar graph

Description automatically generated with medium confidence

**8.4 Feature Fusion Technique:**

Feature fusion is a technique used in machine learning and data processing where features from multiple sources or modalities are combined into a single unified feature set. The goal of feature fusion is to enrich the data representation, improving the performance of machine learning models by integrating useful information from different feature sets.

**Advantages of Feature Fusion**

1. Improved Performance: By combining multiple feature sets, models can gain a more comprehensive understanding of the data, leading to better predictions and generalization.

2. Rich Data Representation: Feature fusion allows the model to use more diverse data, which helps in capturing a wider range of patterns and relationships within the data.

3. Reduced Overfitting: Combining different features can help reduce overfitting as it makes the model less reliant on a single set of features, leading to better generalization.

4. Cross-Modal Information: Feature fusion is especially useful when dealing with multiple data sources, such as combining visual and textual data in multi-modal applications (e.g., image captioning or multi-sensor data fusion).

**Importance of Feature fusion technique:**

Feature fusion is used in many machine learning applications to combine different types of data, improving the overall quality of the features provided to the model. It is especially useful when different feature sets provide complementary information, leading to improved prediction accuracy.

Results before and after using the feature fusion:

**With 100 samples**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.82 | 0.86 | 0.79 | 0.79 |
| 2 | ID3 | 0.88 | 0.86 | 0.85 | 0.85 |
| 3 | KNN | 0.70 | 0.75 | 0.70 | 0.68 |

Figure 1:sampled\_and\_shuffled\_data.csv

Feature fusion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.83 | 0.84 | 0.73 | 0.81 |
| 2 | ID3 | 0.86 | 0.86 | 0.85 | 0.84 |
| 3 | KNN | 0.85 | 0.87 | 0.75 | 0.84 |

**With 500 samples**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.83 | 0.89 | 0.84 | 0.84 |
| 2 | ID3 | 0.91 | 0.92 | 0.91 | 0.91 |
| 3 | KNN | 0.79 | 0.71 | 0.74 | 0.72 |

Figure 2:sampled\_and\_shuffled\_data\_500.csv

**With Feature fusion**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.85 | 0.86 | 0.85 | 0.84 |
| 2 | ID3 | 0.91 | 0.91 | 0.91 | 0.91 |
| 3 | KNN | 0.92 | 0.92 | 0.92 | 0.92 |

Figure 3:sampled\_and\_shuffled\_data\_500.csv

With 100 samples

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.84 | 0.82 | 0.74 | 0.75 |
| 2 | ID3 | 0.92 | 0.92 | 0.90 | 0.91 |
| 3 | KNN | 0.82 | 0.83 | 0.77 | 0.78 |

Figure 4:sampled\_and\_shuffled\_data\_1000.csv

Feature fusion

With 500 samples

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.87 | 0.88 | 0.87 | 0.87 |
| 2 | ID3 | 0.94 | 0.94 | 0.94 | 0.94 |
| 3 | KNN | 0.94 | 0.94 | 0.94 | 0.94 |

**With 5000 samples:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.85 | 0.74 | 0.68 | 0.68 |
| 2 | ID3 | 0.94 | 0.95 | 0.91 | 0.92 |
| 3 | KNN | 0.86 | 0.81 | 0.80 | 0.80 |

Figure 5sampled\_and\_shuffled\_data\_5000.csv

Feature fusion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.87 | 0.87 | 0.87 | 0.86 |
| 2 | ID3 | 0.97 | 0.97 | 0.97 | 0.97 |
| 3 | KNN | 0.97 | 0.97 | 0.97 | 0.97 |

**With 10000 samples:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.90 | 0.77 | 0.73 | 0.73 |
| 2 | ID3 | 0.95 | 0.96 | 0.91 | 0.93 |
| 3 | KNN | 0.87 | 0.81 | 0.80 | 0.80 |

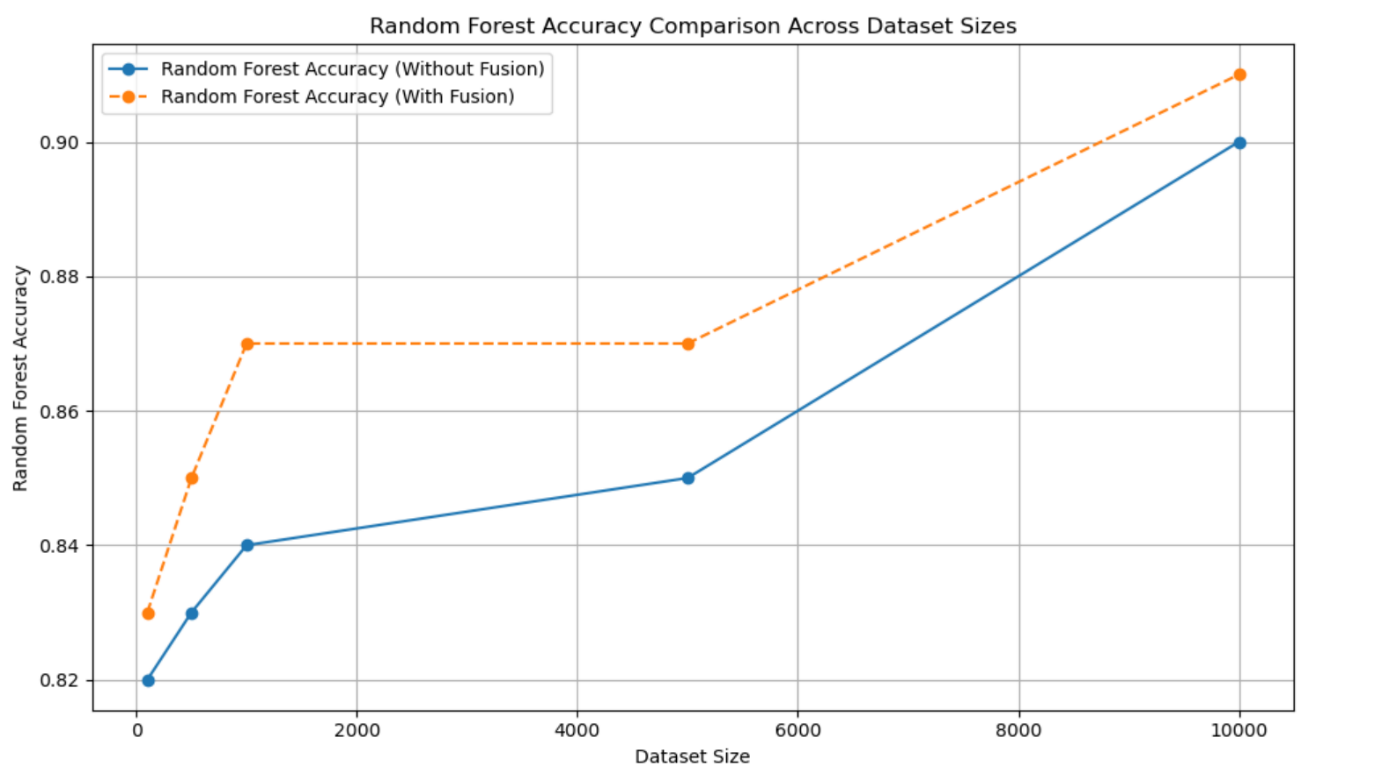
Figure 6:sampled\_and\_shuffled\_data\_10000.csv

Feature fusion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl no | ML Algorithm | Accuracy | Precision | Recall | F1-score |
| 1 | Random Forest | 0.91 | 0.90 | 0.91 | 0.90 |
| 2 | ID3 | 0.98 | 0.98 | 0.98 | 0.98 |
| 3 | KNN | 0.98 | 0.98 | 0.98 | 0.98 |

**Graphical representation of the variation in the parameters after using feature fusion:**

1.Random forest Algorithm:



2.KNN Algorithm:

A graph with a line and a line

Description automatically generated

3.ID3 algorithm:

A graph of a number of data

Description automatically generated with medium confidence

**CHAPTER 9**

**APPLICATIONS**

1. **Network Security Monitoring:**

Detects anomalies in real-time, helping organizations identify and mitigate potential cyber threats like DoS, DDoS, and PortScan attacks.

1. **Intrusion Detection Systems (IDS):**

Enhances existing IDS by detecting both known and zero-day attacks using machine learning and feature-fusion techniques.

1. **Cloud Security:**

Protects cloud-based infrastructures by analyzing encrypted traffic patterns and identifying malicious activities.

1. **IoT Device Protection:**

Secures heterogeneous IoT networks by handling diverse protocols and traffic types effectively.

1. **Critical Infrastructure Protection:**

Safeguards critical systems (e.g., power grids, transportation networks) by identifying unusual network behavior that may indicate cyber threats.

1. **Enterprise IT Security:**

Assists in monitoring large-scale enterprise networks with scalable and accurate anomaly detection.

1. **Fraud Detection:**

Identifies unauthorized access or anomalous behavior in financial and transactional data networks.

**CHAPTER 10**

**CONCLUSION**

In conclusion, the implementation of multiple machine learning models, including Decision Trees and K-Nearest Neighbors, to classify network attack types demonstrates the effectiveness of machine learning in cybersecurity. By preprocessing the dataset, selecting relevant features, and training the models on various attack categories, we were able to evaluate model performance through accuracy, classification reports, and confusion matrices. These results provide valuable insights into the strengths and weaknesses of each model in terms of predicting different attack types. The use of these models for network traffic classification is a powerful tool in enhancing cybersecurity, offering both high accuracy and interpretability, which are critical for real-time attack detection and mitigation.

Additionally, the analysis highlights the importance of feature selection and preprocessing in optimizing model performance. By carefully choosing relevant features such as packet lengths and inter-arrival times, and handling missing data appropriately, the models were able to achieve better results. The use of confusion matrices and classification reports further allowed for a detailed understanding of each model's performance across different attack categories. This approach not only provides a solid foundation for identifying malicious activities but also paves the way for future improvements, such as integrating more advanced models or tuning hyperparameters, to enhance the accuracy and robustness of network intrusion detection systems.

**FUTURE WORK**

This work is open to future improvements. In this section, a few of these improvement possibilities will be presented.

In this study, a data set consisting of CSV files containing features obtained from the network flow was used as the training and test data. Unfortunately, this method is not practically viable in real systems. However, this problem can be solved by adding a module that catches real network data and makes it workable with the machine learning algorithm.

Another point was that during this study, various machine learning methods were applied independently from each other and experimental results were obtained. However, this method has a weak practical applicability in real life. In order to deal with this problem, a multi-layered / hierarchical machine learning structure can be designed. Moreover, thanks to such a structure, it is possible to save time, CPU power and memory. For example, in a two-tiered structure, the first layer can be created from fast and computationally cheap algorithms such as Naive Bayes or QDA, so network traffic can be observed continuously and at minimal cost. The first step, when detecting any anomaly, transmits it to an upper layer composed of algorithms with higher performance level. This layer measures by forming the decision mechanism. The first step, when detecting an anomaly, transmits it to an upper layer composed of algorithms with higher performance such as ID3, AdaBoost, and KNN. The final layer that makes up the determination mechanism takes the precautionary decision to protect the network against attack.

**BIBLIOGRAPHY**

1. K. Kostas, "Anomaly Detection in Networks Using Machine Learning," Research Proposal, 23 Mar 2018, 2018.
2. “Internet Growth Statistics,” *Miniwatts Marketing Group, 2 Mar 2018.* [Online]. Available: [https://www.internetworldstats.com/emarketing.htm.](https://www.internetworldstats.com/emarketing.htm) [Accessed 26 Aug 2018].
3. K. Leung and C. Leckie, "Unsupervised anomaly detection in network intrusion detection using clusters," in *Proceedings of the Twenty-eighth Australasian conference on Computer Science-Volume 38*, 2005, pp. 333-342: Australian Computer Society, Inc.
4. I. Sharafaldin, A. Gharib, A. H. Lashkari, and A. A. Ghorbani, "Towards a reliable intrusion detection benchmark dataset," *Software Networking,* vol. 1, no. 1, pp. 177200, 2017.
5. "1998 DARPA Intrusion Detection Evaluation Data Set," *Lincoln Laboratory, Massachusetts Institute of Technology,* [Online]. Available: [https://www.ll.mit.edu/rd/datasets/1998-darpa-intrusion-detection-evaluation-data-set.](https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-data-set) [Accessed 05 Aug 2018].
6. C. Thomas, V. Sharma, and N. Balakrishnan, "Usefulness of DARPA dataset for intrusion detection system evaluation," in *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2008*, 2008, vol. 6973, p. 69730G: International Society for Optics and Photonics.
7. A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in *Information Science and Security (ICISS), 2016 International Conference on*, 2016, pp. 1-6: IEEE.
8. "KDD Cup 1999 Data," *University of California, Irvine,* [Online]. Available: [http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.](http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html) [Accessed 05 Aug 2018].
9. A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," *PeerJ PrePrints,* vol. 4, p. e1954v1, 2016.
10. "CAIDA OC48 Peering Point Traces dataset," *Center for Applied Internet Data Analysis,* [Online]. Available:

[https://www.caida.org/data/passive/passive\_oc48\_dataset.xml.](https://www.caida.org/data/passive/passive_oc48_dataset.xml) [Accessed06 Aug 2018].

1. M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications,* vol. 60, pp. 19-31, 2016.
2. "NSL-KDD dataset," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available: [http://www.unb.ca/cic/datasets/nsl.html.](http://www.unb.ca/cic/datasets/nsl.html) [Accessed 06 Aug 2018].
3. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on*, 2009, pp. 1-6: IEEE.
4. "Intrusion detection evaluation dataset (ISCXIDS2012)," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available:

[http://www.unb.ca/cic/datasets/ids.html.](http://www.unb.ca/cic/datasets/ids.html) [Accessed 07 Aug 2018].

1. A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *computers & security,* vol. 31, no. 3, pp. 357-374, 2012.
2. "Intrusion Detection Evaluation Dataset (CICIDS2017)," *Canadian Institute for Cybersecurity, University of New Brunswick,* [Online]. Available:

[http://www.unb.ca/cic/datasets/ids-2017.html.](http://www.unb.ca/cic/datasets/ids-2017.html) [Accessed 08 Aug 2018].

1. W. Stallings, L. Brown, M. D. Bauer, and A. K. Bhattacharjee, *Computer security: principles and practice*. Pearson Education, 2012.
2. M. Chhabra, B. Gupta, and A. Almomani, "A novel solution to handle DDOS attack in MANET," *Journal of Information Security,* vol. 4, no. 03, p. 165, 2013.
3. "Hulk DoS tool," *GitHub Inc,* [Online]. Available: [https://github.com/grafov/hulk.](https://github.com/grafov/hulk) [Accessed 08 Aug 2018].
4. S. Behal and K. Kumar, "Characterization and Comparison of DDoS Attack Tools and Traffic Generators: A Review," *IJ Network Security,* vol. 19, no. 3, pp. 383-393, 2017.
5. S. Haris, R. Ahmad, and M. Ghani, "Detecting TCP SYN flood attack based on anomaly detection," in *Network Applications Protocols and Services (NETAPPS), 2010 Second International Conference on*, 2010, pp. 240-244: IEEE.
6. J. Choi, C. Choi, B. Ko, D. Choi, and P. Kim, "Detecting Web based DDoS Attack using MapReduce operations in Cloud Computing Environment," *J. Internet Serv. Inf. Secur.,* vol. 3, no. 3/4, pp. 28-37, 2013.
7. "Nmap: *the Network Mapper - Free Security Scanner," Nmap.org. (2018),*  [online] Available: <https://nmap.org/>[Accessed 10 Aug. 2018].
8. R. Christopher, "Port scanning techniques and the defense against them," *SANS Institute,* 2001.
9. J. Gadge and A. A. Patil, "Port scan detection," in *Networks, 2008. ICON 2008. 16th IEEE International Conference on*, 2008, pp. 1-6: IEEE.
10. “LOIC,” *SourceForge,* [Online]. Available: [https://sourceforge.net/projects/loic/.](https://sourceforge.net/projects/loic/) [Accessed: 11-Aug-2018].
11. “GoldenEye,” *GitHub, 20-Jun-2018.* [Online]. Available: Available:

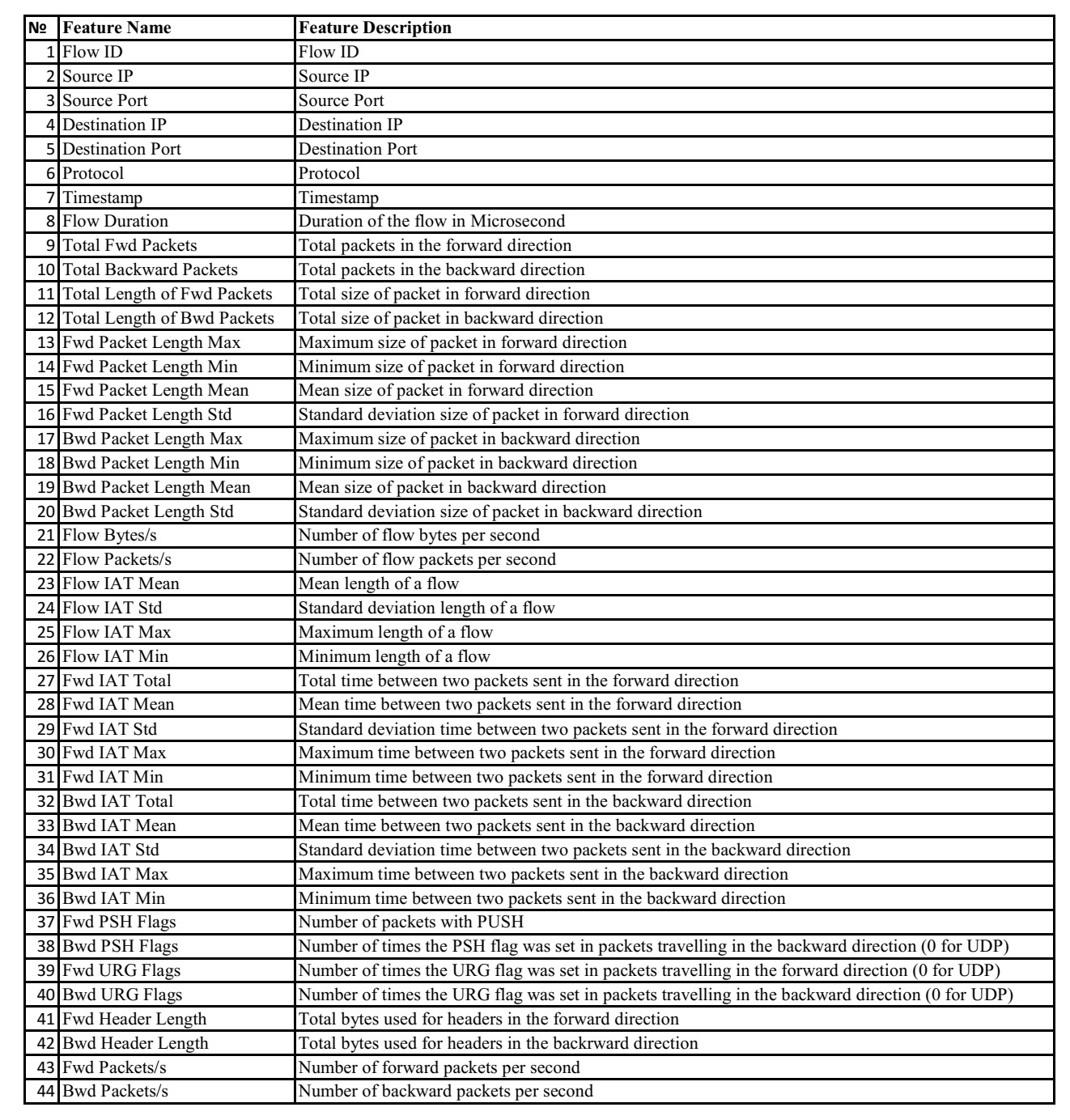
[https://github.com/jseidl/GoldenEye.](https://github.com/jseidl/GoldenEye) [Accessed: 11-Aug-2018].

1. “Patator,” *GitHub, 04-Aug-2018.* [Online]. Available:

[https://github.com/lanjelot/patator.](https://github.com/lanjelot/patator) [Accessed: 11-Aug-2018].

1. M. M. Najafabadi, T. M. Khoshgoftaar, C. Kemp, N. Seliya, and R. Zuech, "Machine learning for detecting brute force attacks at the network level," in *Bioinformatics and Bioengineering (BIBE), 2014 IEEE International Conference on*, 2014, pp. 379-385: IEEE.

**APPENDICES**



**Self- Assessment of the Project :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Level | | | | | |
| Poor 1 | | Good 2 | | Excellent 3 | |
|  | PO PSO | | Contribution from the project | | Level |
| 1 | **Engineering Knowledge:**  Knowledge of mathematics, engineering fundamentals engineering specialization to form of complex engineering problems | | Applying FuzzyC and K-Means algorithms to perform image processing tasks | | 3 |
| 2 | **Problem Analysis:** Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions with consideration for sustainable development. | | Model image segmentation problem and automatically segment the brain tumor from the MRI images. | | 3 |
| 3 | **Design/development of solutions:** Design creative solutions for complex engineering problems and design/develop systems/components/processes to meet identified needs with consideration for the public health and safety, whole-life cost, net zero carbon, culture, society and environment as required | | To learn MATLAB graphical user interface (GUI) and develop a code according to problem statement.  To segment the brain tumor using combinations of clustering algorithms with some morphological operations on the image. | | 3 |
| 4 | **Conduct investigations of complex problems:**  Conduct investigations of complex engineering problems using research-based knowledge including design of experiments, modelling, analysis & interpretation of data to provide valid conclusions. | | Project work is carried out based on the thesis. Understanding the concepts of image processing and tumors and pixels intensity. Comparison of two clustering algorithms K-means and Fuzzy to find the area of tumor and to find the patient’s condition. | | 3 |
| 5 | **Modern tool usage:** Create, select and apply appropriate techniques, resources and modern engineering & IT tools, including prediction and modelling recognizing their limitations to solve complex engineering problems. | | Project was carried out using MATLAB and GUI using graphical user interface development environment. | | 3 |
| 6 | **The Engineer and the world:**  Analyze and evaluate societal and environmental aspects while solving complex engineering problems for its impact on sustainability with reference to economy, health, safety, legal framework, culture and environment. | | It is cost effective and automated process. To complete the area of tumor from the MRI images and check the patient’s condition based on the area, which makes the whole process automatic. | | 3 |
| 7 | **Ethics:** Apply ethical principles and commit to professional ethics, human values, diversity and inclusion; adhere to national & international laws. | | Project work and report followed honor code which is verified by Plagiarism check (25%) and report conforming to Industry standard | | 3 |
| 8 | **Individual and Team Work:** Function effectively as an individual, and as a member or leader in diverse/multi-disciplinary teams. | | Equal and active participation is done among the team members. | | 3 |
| 9 | **Communication:**Communicate effectively and inclusively within the engineering community and society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations considering cultural, language, and learning differences | | Effective documentation is done using Latex (Overleaf) and presented using structure easy to understand  Effective presentation is also prepared highlighting the novelty, design solution, result analysis and inference giving directions for further improvement | | 3 |
| 10 | **Project Management and Finance:** Apply knowledge and understanding of engineering management principles and economic decision-making and apply these to one’s own work, as a member and leader in a team, and to manage projects and in multidisciplinary environments.. | | Scheduling and plan of action was prepared at the beginning  Plan of action and implementation was recorded in diary maintained  Execution was done using Cost efficient, scalable and customization. approach | | 2 |
| 11 | **Life-long Learning:** Recognize the need for, and have the preparation and ability for i) independent and life-long learning ii) adaptability to new and emerging technologies and iii) critical thinking in the broadest context of technological change | | As project is about the ongoing and upcoming technologies, further the medical image processing is applied to detect the tumors, diseases and recent covid affected lungs severity could also be found using clustering algorithms.  Self learning ability improved | | 3 |
| 13 | **PSO1:** Apply the concepts of electronic circuits and systems to analyses and design systems related to Microelectronics, Communication, Signal processing and Embedded systems for solving real world problems | | Applied and analyzed the concept of digital image processing concepts. To segment the image the clustering techniques are used. | | 3 |
| 14 | **PSO2:** To identify problems in the area of communication and embedded systems and provide efficient solutions using modern tools/algorithms working in a team | | Code is generated using MATLAB software. | | 3 |

SUSTAINABLE DEVELOPMENT GOALS addressed in the project:

Levels: Poor:1, Good :2, Excellent:3

|  |  |
| --- | --- |
| SDG | Level |
| No Poverty |  |
| Zero Hunger |  |
| Good Health and Well-being |  |
| Quality education |  |
| Gender Quality |  |
| Clean water and Sanitation |  |
| Affordable and Clean Energy |  |
| Decent work and Economic Growth |  |
| Industry, Innovation and Infrastructure |  |
| Reduced Inequalities |  |
| Sustainable cities and Communities |  |
| Responsible Consumption and production |  |
| Climate action |  |
| Life below water |  |
| Life on Land |  |
| Peace, Justice and Strong Institutions |  |
| Partnership’s for the Goals |  |