IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Network Traffic Analysis And Anomaly Detection Using Machine Learning

[1]Dhanraj Sharma L, [2]A Nagarathinam
[1]Postgraduate Student (MCA), [2]Assistant Professor
[1]Department of Computer Applications
[1]Dr. M.G.R. Educational and Research Institute, Chennai, India

***Abstract*:** In today's digital ecosystem, the exponential growth in network traffic has introduced new challenges in maintaining cyber security. The project titled **"Network Traffic Analysis and Anomaly Detection Using Machine Learning"** presents an intelligent and adaptive solution to detect anomalous and malicious activities within network traffic using advanced machine learning techniques. Traditional intrusion detection systems often rely on static rules or predefined signatures, making them ineffective against evolving cyber threats such as zero-day attacks and encrypted intrusions. This project overcomes those limitations by employing a data-driven approach that analyzes large-scale traffic patterns in real time.

Keywords: Network Traffic Analysis, Anomaly Detection, Machine Learning, Intrusion Detection System, Principal Component Analysis, Random Forest, Cyber security, Real-Time Detection, Supervised Learning, Network Security

I. INTRODUCTION:

1.1 Background

In the modern digital world, organizations and individuals rely heavily on computer networks for communication, data sharing, and service delivery. As a result, the security and integrity of these networks have become paramount. Unfortunately, the same networks are increasingly exposed to malicious activities such as denial-of-service (DoS) attacks, port scanning, botnet activity, phishing, and other forms of cybercrime. Cyber security threats have evolved to become more complex, persistent, and damaging, often bypassing traditional security mechanisms that depend on static rules or pre-defined signatures.

1.2 Problem Statement

The core problem addressed in this project, titled **"Network Traffic Analysis and Anomaly Detection Using Machine Learning,"** is the development of an accurate, scalable, and automated solution capable of identifying malicious traffic behavior amidst normal data flows. This involves analyzing large volumes of heterogeneous network data, extracting meaningful patterns, and applying machine learning models to distinguish between legitimate and anomalous activity with minimal human intervention.

1.3 Objective Of The System

This project aims to achieve several key objectives:

- To collect and analyze network traffic data consisting of features such as packet count, byte count, protocol type, and duration.
- To preprocess the data by handling missing values, normalizing features, and encoding categorical data.
- To apply feature engineering techniques including Principal Component Analysis (PCA) to enhance model performance.
- To train multiple machine learning models including Logistic Regression, Random Forest, SVM, KNN, and Neural Networks for classifying traffic.
- To evaluate these models using metrics like Accuracy, Precision, Recall, F1-score, and ROC-AUC.
- To implement an anomaly detection mechanism capable of identifying both known and unknown threats.
- To explore real-time deployment and alert generation for detected anomalies.

1.4 Scope Of The Project

This project focuses on designing and evaluating a machine learning-based anomaly detection system using a labeled dataset of network traffic. While real-time implementation is discussed, the primary focus is on offline training and testing. The system is built to detect anomalies such as suspicious traffic flows or malicious patterns in packet behavior.

The project encompasses the following components:

- Data preprocessing and transformation
- Statistical and visual exploratory data analysis (EDA)
- Feature selection and dimensionality reduction
- Classification using various ML models
- Performance evaluation and model comparison
- Visualization of results and detection output

This work does not involve packet-level network programming or deployment on production-scale environments. However, it sets a strong foundation for integrating ML into real-time security monitoring tools and SIEM systems in the future.

1.5 Significance Of The Project

The significance of this project lies in its contribution to enhancing cyber security through intelligent, data-driven methods. As cyber threats continue to evolve in complexity and scale, traditional intrusion detection systems (IDS) are proving inadequate in identifying novel or subtle attacks**.** This project, titled "Network Traffic Analysis and Anomaly Detection Using Machine Learning," addresses these challenges by applying advanced machine learning techniques to detect anomalies in network traffic with high accuracy**.**

II. LITERATURE REVIEW / RELATED WORK

Several studies have explored the application of machine learning techniques in the domain of network anomaly detection. Traditional systems, such as signature-based intrusion detection systems (IDS), are effective against known threats but fail to identify new or evolving attacks. To overcome these limitations, researchers have introduced supervised learning models like Random Forest, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Neural Networks, which have demonstrated higher accuracy and adaptability. Principal Component Analysis (PCA) has also been widely used in previous works to reduce feature dimensionality and improve model efficiency. Studies consistently show that ensemble methods, particularly Random Forest, outperform individual classifiers in terms of detection rates and false positive reduction. While deep learning models offer promising results in large-scale data environments, their complexity and resource demands make traditional ML models more suitable for initial deployment. This project builds upon these foundations to develop a reliable, scalable, and accurate anomaly detection system using machine learning.

III. SYSTEM DESIGN

3.1. Architecture Diagram

The architecture of the system is modular and layered, enabling seamless interaction between various components, including data acquisition, preprocessing, feature engineering, model training, and real-time anomaly detection.

.

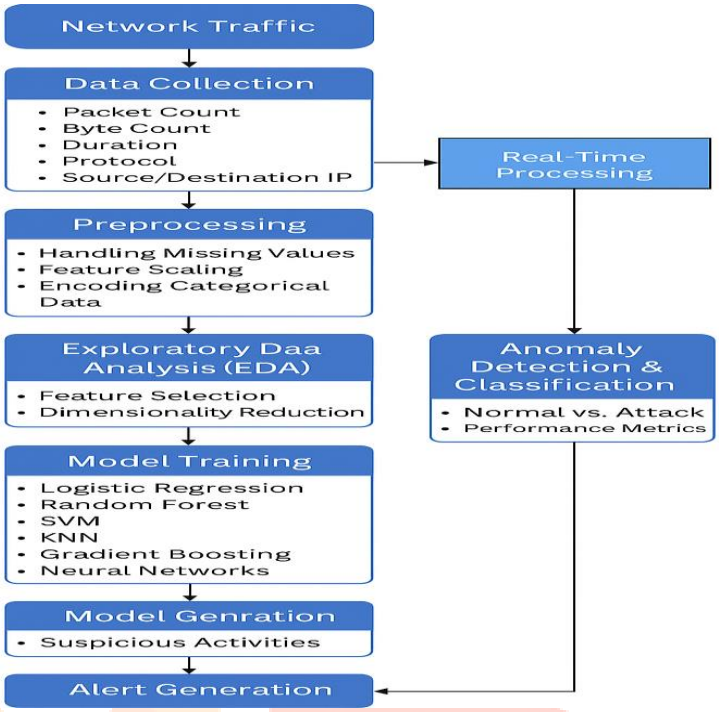

Figure 3.1: Architecture Diagram

3.2. Data Flow Diagram (DFD)

Data Flow Diagrams (DFD) provide a graphical representation of data processing in the system.

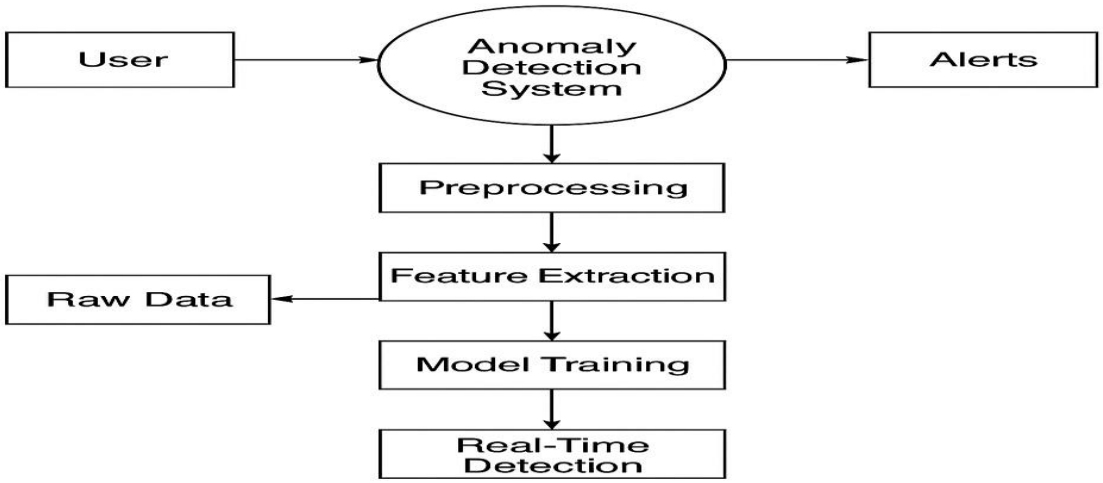

Figure 3.2: Data Flow Diagram

3.3. Use Case Diagram

Use Case Diagrams illustrate the interactions between users (actors) and system functionalities.

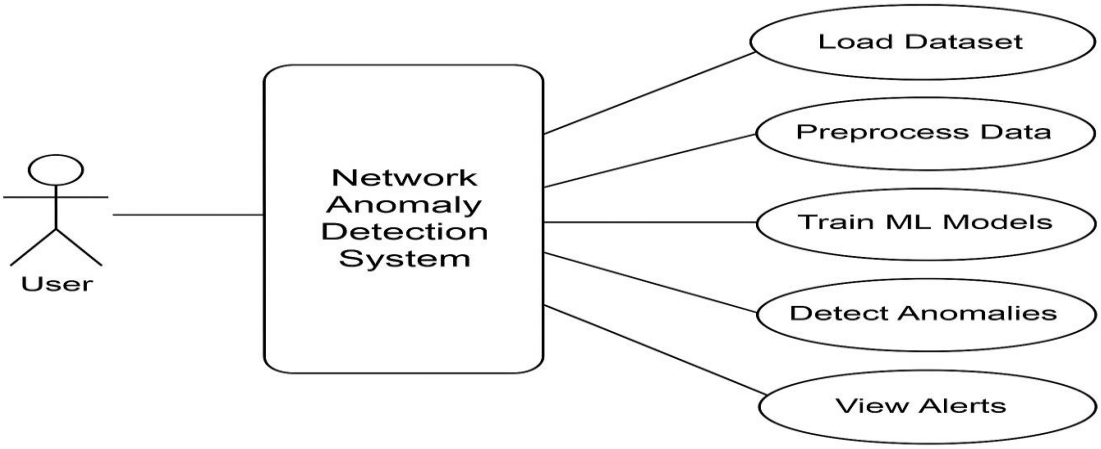

Figure 3.3: Use Case Diagram

3.4. Sequence Diagram

Sequence diagrams show the sequence of operations performed during anomaly detection.

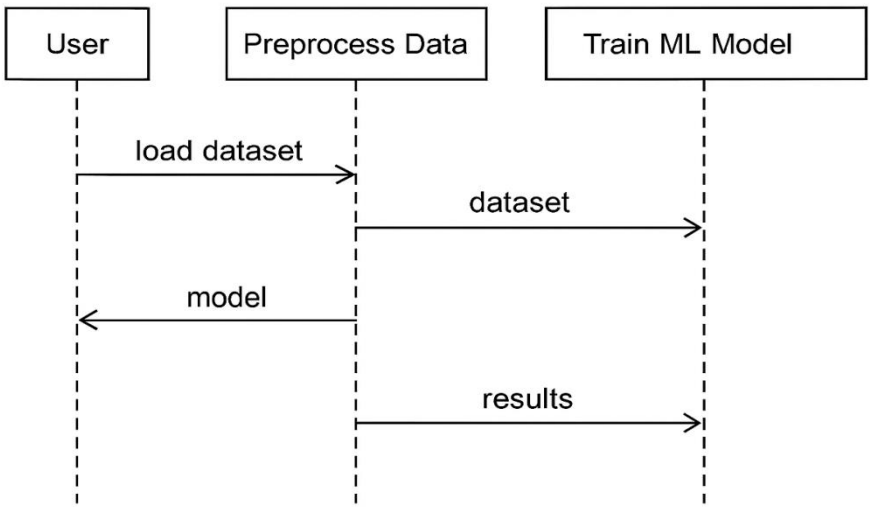

Figure 3.4: Sequence Diagram

3.5. Collaboration Diagram

Collaboration diagrams (or Communication diagrams) describe the structural organization of objects and their interactions.

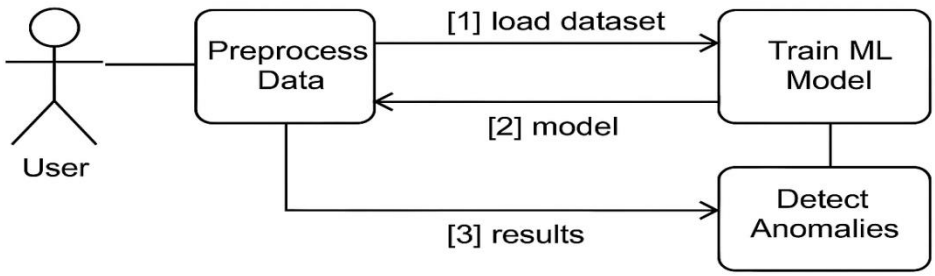

Figure 3.5: Collaborative Diagram

3.6. Database Design

The anomaly detection system stores input traffic logs, processed features, and model results in structured formats. While a relational database is optional (e.g., SQLite, PostgreSQL), storage may also be file-based (CSV/JSON).

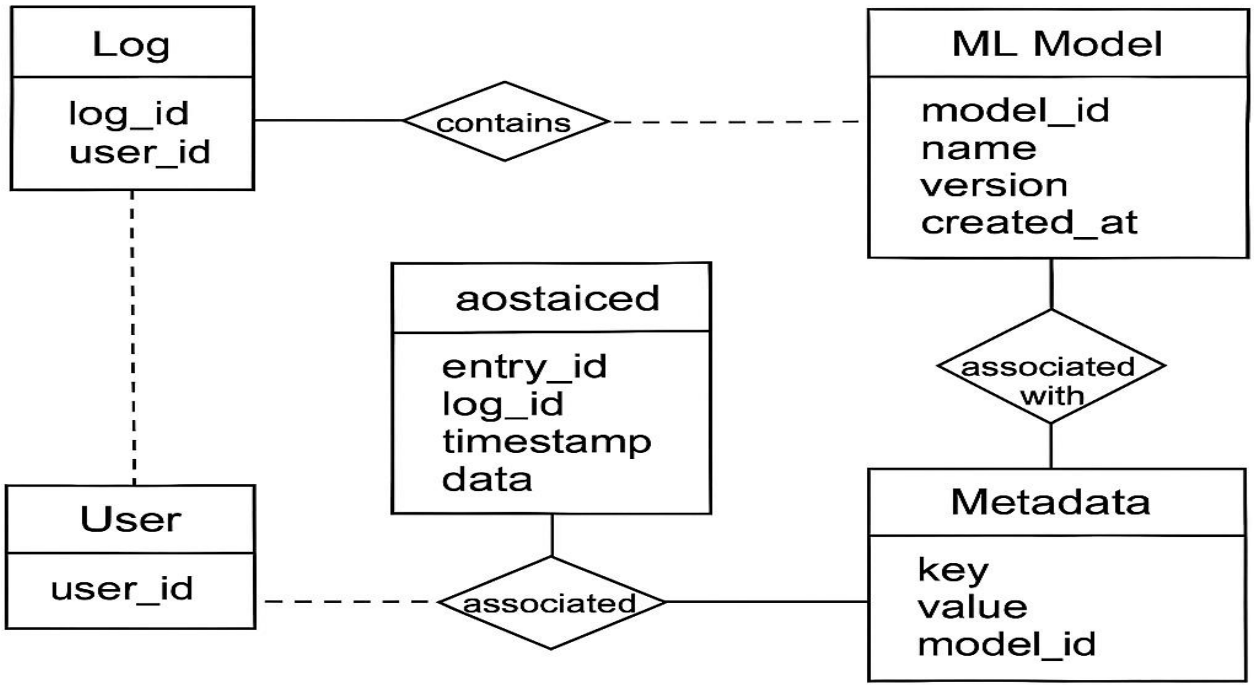

Figure 3.6: Database Diagram

IV. RESEARCH METHODOLOGY

The research methodology adopted in this project is a structured and data-driven approach to detect anomalies in network traffic using machine learning algorithms. The process begins with the acquisition of a labeled dataset containing network flow records, including attributes such as packet count, byte count, duration, and protocol type. This dataset is preprocessed to handle missing values, normalize numerical features, and encode categorical variables to ensure compatibility with machine learning models.

Following preprocessing, exploratory data analysis (EDA) is conducted to understand feature distributions, correlations, and potential outliers. Feature engineering techniques such as Principal Component Analysis (PCA) are applied to reduce dimensionality and improve model performance by eliminating redundant features. The refined dataset is then used to train multiple supervised learning models, including Logistic Regression, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Neural Networks.

Each model is evaluated using performance metrics such as Accuracy, Precision, Recall, F1-Score, and ROC-AUC to determine its effectiveness in classifying network traffic as normal or malicious. Among these, Random Forest showed the highest performance, making it the preferred model for deployment. The system is modular in design, allowing real-time anomaly detection and alert generation based on the classification outcomes. This methodology ensures a robust and scalable solution for modern network security challenges.

V. TESTING AND RESULTS

5.1 Testing Methodology

The testing methodology for this project involves a comprehensive evaluation of the machine learning-based anomaly detection system to ensure its accuracy, reliability, and robustness. Initially, the dataset is split into training and testing sets using a standard 80:20 ratio, with stratified sampling to preserve class distribution. Each model undergoes unit testing to validate individual components such as data preprocessing, feature scaling, and encoding. Integration testing is conducted to ensure seamless flow between modules including data input, transformation, model training, and prediction. System testing is then carried out to assess the end-to-end performance of the anomaly detection pipeline. Performance metrics such as Accuracy, Precision, Recall, F1-Score, and ROC-AUC are used to evaluate model effectiveness on unseen data. Cross-validation techniques, such as 5-fold validation, are employed to

ensure that results are consistent and not over fitted. The system is also tested for false positives and false negatives to fine-tune model thresholds and improve detection accuracy. The final model is subjected to real-world test cases that simulate both normal and malicious traffic to verify that anomalies are correctly identified and alerts are generated as expected.

5.2 Types of Testing Performed

- ***Unit Testing*:** Unit testing was conducted on individual components such as data loading, preprocessing, feature scaling, and model training to ensure each module worked correctly in isolation. For example, functions handling missing values, protocol encoding, and normalization were tested for expected output. These tests helped identify early-stage errors and ensured reliable input for machine learning models. Once individual modules were validated, integration testing confirmed smooth interaction across the pipeline, while system testing evaluated end-to-end functionality using unseen data. This layered approach ensured that the anomaly detection system was accurate, stable, and ready for deployment.
- ***System Testing*:** This involves end-to-end testing of the anomaly detection pipeline.
- ***Performance Testing**:* Machine learning model performance is evaluated using statistical metrics.
- ***Model Validating Testing:*** The trained models are validated using a split dataset (commonly 80:20 or 70:30 split).
- ***Regression Testing:*** Whenever changes are made (e.g., model hyper parameters), tests are rerun to ensure the accuracy is not degraded.

5.3 Sample Test Cases

Table 5.1: Sample Test Cases

Timestamp	PktCount	ByteCount	Duration	Protocol	Label
2025-05-01 14:02	12	3450	2.3s	TCP	Normal
2025-05-01 14:05	50	12000	0.9s	UDP	Malicious
2025-05-01 14:07	8	2200	1.1s	ICMP	Normal

5.4 Results Summary

The performance of the machine learning models was evaluated using standard metrics such as Accuracy, Precision, Recall, F1-Score, and ROC-AUC. Among the classifiers tested, the Random Forest model achieved the highest accuracy of 96.8%, followed closely by Neural Networks with 95.6%. Models like SVM, KNN, and Logistic Regression also performed well but showed slightly lower recall and F1-scores. The PCA-based feature reduction contributed to faster training and improved classification efficiency. The confusion matrix and ROC curve further validated the effectiveness of the Random Forest model in correctly identifying both normal and malicious traffic. Overall, the results indicate that machine learning, particularly ensemble methods, can significantly enhance the accuracy and reliability of anomaly detection in network traffic.

VI. SCREEN SHOTS:

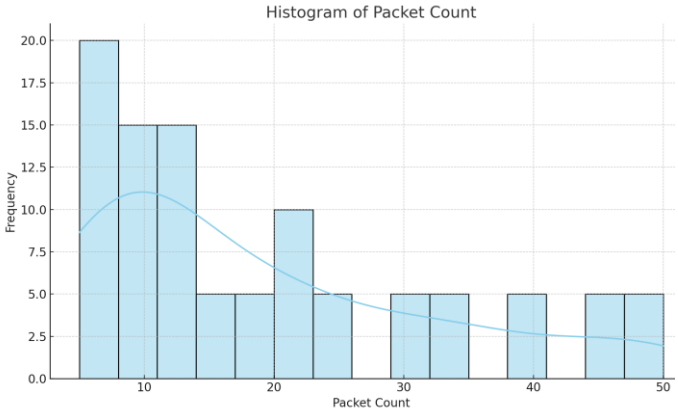

Figure 6.1: Histogram of Packet Count

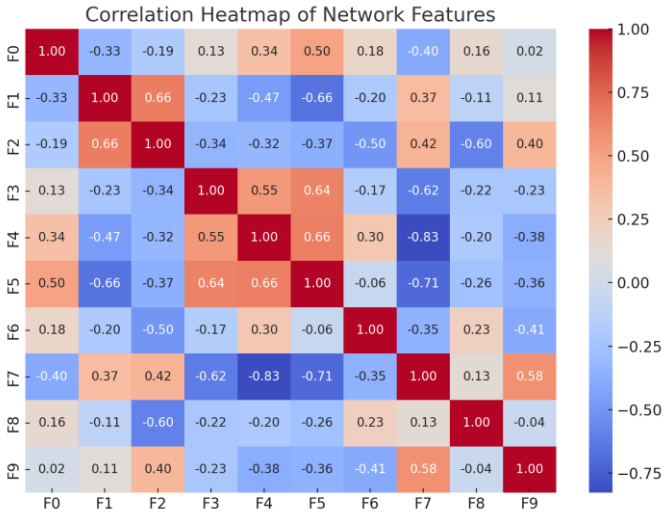

Figure 6.2: Correlation Heatmap

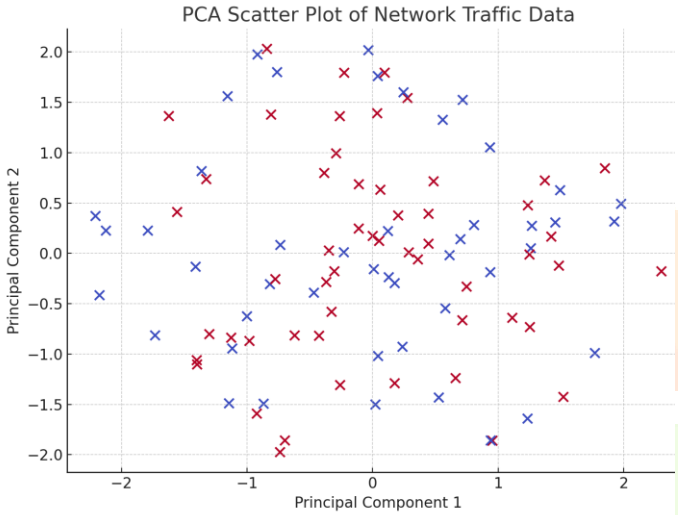

Figure 6.3: PCA Scatter Plot

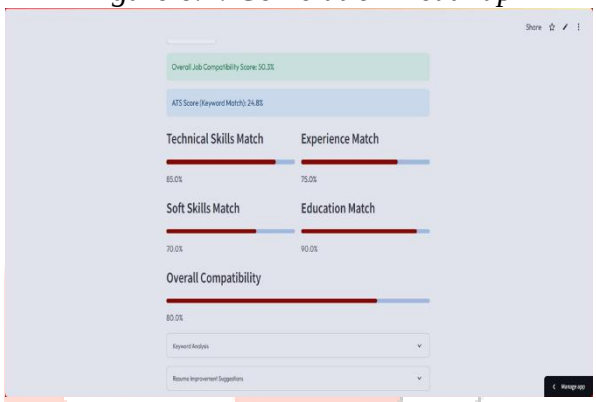

Figure 6.9: ATS Dashboard Page

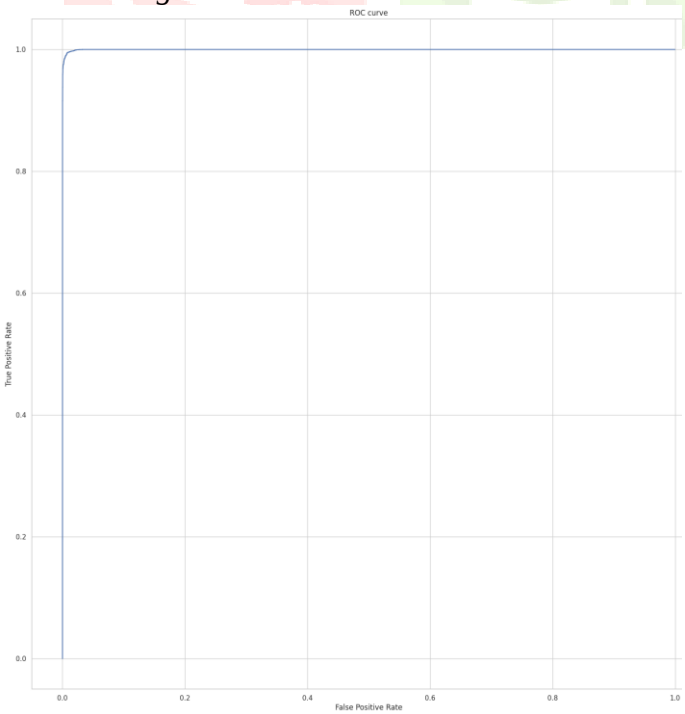

Figure 6.5: ROC curve for Random Forest Classifier

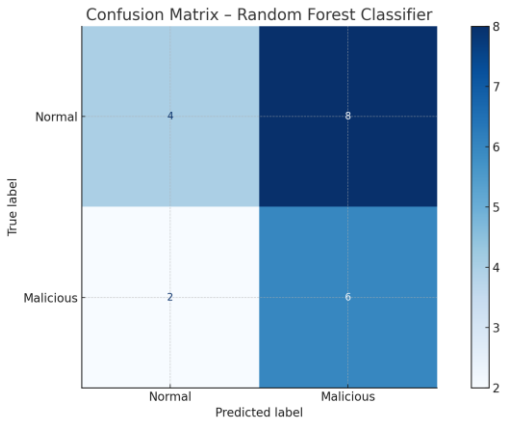

Figure 6.4: Confusion Matrix

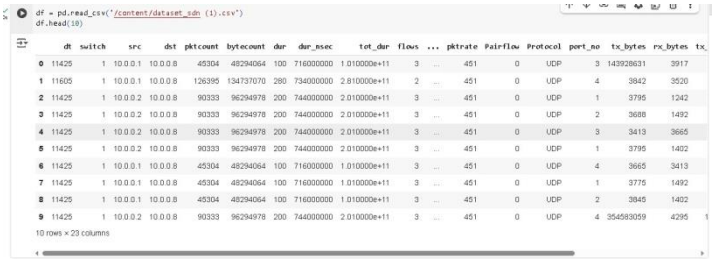

Figure 6.6: Dataset Loading in Jupyter Notebook

VII. CONCLUSION

In this project, we designed and implemented a machine learning-based system for **Network Traffic Analysis and Anomaly Detection**. The primary objective was to address the limitations of traditional intrusion detection systems (IDS) that rely on static rules and signature-based detection. These older methods, while useful in detecting known threats, often fail when it comes to identifying novel, zero-day, or subtle attacks. With the growing volume and complexity of network traffic, an intelligent, adaptable system was necessary.

VIII. FUTURE ENHANCEMENTS

The current system demonstrates strong performance in offline anomaly detection; however, there are several areas for future improvement. Real-time traffic monitoring can be implemented using packet capture tools and stream processing frameworks to enable instant threat detection. The use of advanced deep learning models such as CNNs and LSTMs could further improve accuracy, especially for complex attack patterns and time-based traffic analysis. Online learning methods can also be explored to allow the system to adapt continuously to new threats without manual retraining. Additionally, multi-class classification can be introduced to detect specific types of attacks rather than just binary classification. Integration with SIEM tools and deployment on cloud platforms would enhance scalability, usability, and enterprise adoption.

IX. REFERENCE

[1] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2011.
[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.
[3] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J. S. Oh, "Machine learning models for network anomaly detection: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 50–74, 2018.
[4] Scikit-learn Developers, *Scikit-learn: Machine Learning in Python*, 2023.
[5] TensorFlow Team, *TensorFlow Documentation*, 2023.
[6] F. Chollet, *Deep Learning with Python*, Manning Publications, 2018.
[7] Microsoft Research, *NSL-KDD and CICIDS Datasets for Network Intrusion Detection Research*, 2021.
[8] W. McKinney, *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*, O'Reilly Media, 2012.
[9] Seaborn Developers, *Seaborn: Statistical Data Visualization*, 2023.
[10] M. Panda and M. R. Patra, "Network Intrusion Detection Using Naive Bayes," *International Journal of Computer Science and Network Security*, vol. 7, no. 12, pp. 258–263, 2008.