ROBUST NETWORK INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

Btech Final Year Project Report

Supervised by: Dr. Abdur Rahaman Sardar

Department of Computer Science and Engineering (CSE)
Ramkrishna Mahato Government Engineering College, Purulia

May, 2023

CERTIFICATE OF ORIGINALITY

I hereby certify that the work which is being presented in the B.Tech. Final Year Project Report entitled **"Robust Network Intrusion Detection System using Machine Learning",** in partial fulfillment of the requirements for the award of the **Bachelor of Technology in Computer Science & Engineering** and submitted to the Department of Computer Science & Engineering of Ramkrishna Mahato Government Engineering College, Purulia, West Bengal is an authentic record of my own work carried out during a period from June, 2022 to June, 2023 under the supervision of **Dr. Abdur Rahaman Sardar, CSE Department**.

The matter presented in this thesis has not been submitted by me for the award of any other degree elsewhere.

Signature of Candidates:

Rahul Kumar Yadav
(35000119029)
CSE 4th Year

Bondala Deeksha
(35000119039)
CSE 4th Year

Saugata Choudhury
(35000119019)
CSE 4th Year

CERTIFICATE OF RECOMMENDATION

This is to certify that the Project entitled **"Robust Network Intrusion Detection System using Machine Learning"** has been submitted by **Rahul Kumar Yadav, Bondala Deeksha and Saugata Choudhury** under my guidance in partial fulfillment of the degree of Bachelor of Technology in Computer Science & Engineering from Ramkrishna Mahato Government Engineering College, Purulia, WB during the academic year 2022-2023..

Signature of Supervisor(s)

Name & Designation
Project Supervisor(s)

Head of the Department

Department of Computer Science & Engineering

Ramkrishna Mahato Government Engineering College, Purulia, West Bengal

Date: 26:05:23

Place: Purulia

ACKNOWLEDGEMENT

We would like to extend our heartfelt gratitude to all individuals and organizations who have contributed to the successful completion of this project and the preparation of this report.

First and foremost, we would like to express our deepest appreciation to our project supervisor **Dr. Abdur Rahaman Sardar** for their unwavering guidance, expertise, and support throughout the entire duration of this research endeavor. Their valuable insights, constructive feedback, and continuous encouragement have been instrumental in shaping the direction of our project and ensuring its successful completion.

We would also like to extend our gratitude to the faculty members and staff of **Ramkrishna Mahato Government Engineering College** for their immense knowledge, guidance, and support. Their dedication to providing a conducive learning environment and their willingness to assist us with our queries and challenges have been invaluable to our academic growth and the successful execution of this project.

In conclusion, we express our sincere gratitude to all individuals and organizations mentioned above, as well as anyone else who has directly or indirectly contributed to this project. Your support, guidance, and encouragement have been instrumental in our success, and we are genuinely grateful for the opportunity to undertake this project and present this report.

Rahul Kumar Yadav

Bondala Deeksha

Saugata Choudhury

ABSTRACT

As the reliance on networked systems continues to grow, the threat landscape of cyberattacks becomes increasingly sophisticated and complex. Network intrusion detection systems (NIDS) play a crucial role in safeguarding the integrity and security of computer networks. This research project focuses on the development of a robust and efficient NIDS utilizing machine learning and deep learning techniques to detect and mitigate various types of network intrusions.

The primary objective of this project is to enhance the detection accuracy and responsiveness of NIDS by leveraging the power of machine learning algorithms. Traditional signature-based intrusion detection systems often struggle to detect novel and previously unseen attacks. In contrast, machine learning models can learn complex patterns and features from network traffic data, enabling the identification of both known and unknown intrusion attempts.

The proposed NIDS architecture consists of several key components. Firstly, a comprehensive dataset comprising normal network traffic and various attack scenarios is collected and preprocessed to create a training set for the machine learning model. Next, a suitable machine learning algorithm, such as support vector machines, decision trees, or random forests, is employed to train the model using the dataset.

The outcomes of this research project contribute to the advancement of network security by providing a robust and efficient NIDS framework based on machine learning algorithms. By leveraging the capabilities of machine learning, organizations can enhance their ability to detect and respond to network intrusions effectively. This, in turn, helps prevent data breaches, service disruptions, and unauthorized access to sensitive information.

INTRODUCTION

The rapid advancements in technology and the widespread reliance on computer networks have brought about numerous benefits and opportunities. However, they have also introduced new challenges, particularly in the realm of network security. The ever-evolving threat landscape poses significant risks to the confidentiality, integrity, and availability of data and systems. Network intrusion, in particular, has emerged as a major concern, with malicious actors constantly seeking ways to infiltrate and compromise networks for their nefarious purposes.

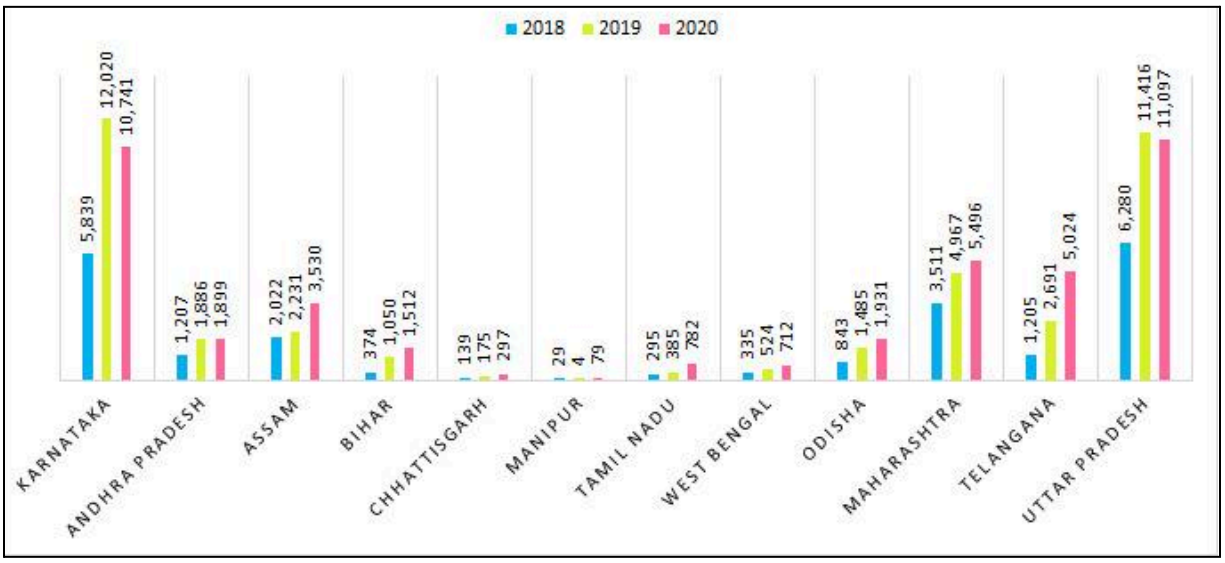


Fig 1.1: Statewise cyber crimes recorded in India [16]

The purpose of this report is to present a comprehensive study on network intrusion detection systems (NIDS) and their role in enhancing cybersecurity. NIDS play a crucial role in safeguarding computer networks by detecting and mitigating various types of network intrusions. By monitoring network traffic and analyzing patterns, anomalies, and known attack signatures, NIDS strive to identify and respond to malicious activities in real-time.

The objectives of this research project are to explore and evaluate different approaches and techniques employed in NIDS, with a specific focus on machine learning algorithms. Traditional signature-based detection methods have limitations in detecting novel and previously unseen

attacks. Machine learning offers a promising avenue to overcome these limitations by enabling the NIDS to learn and adapt from large volumes of network traffic data, enhancing its ability to detect both known and unknown intrusions.

The report aims to provide a comprehensive understanding of the underlying concepts, methodologies, and challenges associated with NIDS. It will delve into the various types of network intrusions, the characteristics and patterns that can be leveraged for detection, and the importance of timely response and mitigation strategies. Additionally, this report will explore the potential of machine learning algorithms, such as support vector machines, decision trees, and deep neural networks, in enhancing the accuracy and effectiveness of NIDS. Furthermore, this research project will involve practical implementation and experimentation to evaluate the performance and effectiveness of the machine learning-based NIDS.

The outcomes of this project are expected to contribute to the advancement of network security by providing valuable insights into the capabilities and limitations of machine learning-based NIDS. The findings and recommendations from this research endeavor can aid organizations in making informed decisions regarding the selection, deployment, and configuration of NIDS to enhance their cybersecurity posture.

In conclusion, this report sets out to explore the significance of network intrusion detection systems in mitigating the risks posed by network intrusions. By leveraging the power of machine learning algorithms, the aim is to develop more robust and efficient NIDS capable of detecting and responding to both known and emerging threats. Through a comprehensive analysis and evaluation, this research project seeks to contribute to the ongoing efforts in strengthening the cybersecurity landscape and protecting computer networks from unauthorized access and malicious activities.

LITERATURE SURVEY

Introduction:

Network intrusion detection systems (NIDS) are essential tools in ensuring the security and integrity of computer networks. They play a critical role in identifying and mitigating various types of network intrusions, including unauthorized access, data breaches, and malicious activities. In recent years, researchers have made significant advancements in the field of NIDS, leveraging various techniques to enhance detection accuracy and improve the ability to respond to evolving threats. This literature survey aims to provide an overview of the key research studies, methodologies, and trends in NIDS, focusing on machine learning approaches.

Evolution of Intrusion Detection Techniques:

Early intrusion detection systems primarily relied on rule-based approaches, employing predefined signatures and patterns to detect known attacks. However, these systems struggled to detect novel and previously unseen attacks, necessitating the development of more sophisticated techniques. Machine learning algorithms have gained popularity due to their ability to learn from data and adapt to new attack patterns. Notable techniques include support vector machines (SVM), decision trees, random forests, and deep neural networks.

Machine Learning-Based Approaches:

Machine learning algorithms have revolutionized the field of NIDS, enabling more accurate and efficient detection of network intrusions. Studies have demonstrated the effectiveness of SVM in classifying network traffic as normal or malicious based on extracted features. Decision tree algorithms, such as C4.5 and Random Forests, have been applied for feature selection and intrusion detection. Deep learning models, such as convolutional neural networks (CNN) and recurrent neural networks (RNN), have shown promise in capturing intricate patterns and anomalies in network traffic data.

Feature Extraction and Selection:

Effective feature extraction and selection techniques play a crucial role in the success of machine learning-based NIDS. Various studies have explored different feature sets, including statistical, content-based, and traffic flow-based features. Statistical features, such as packet sizes, protocol distribution, and inter-arrival times, capture the statistical properties of network traffic. Content-based features, such as payload characteristics and header information, focus on the content of network packets. Traffic flow-based features analyze the flow of network communication between hosts, considering factors like source and destination IP addresses, port numbers, and packet timing.

Performance Evaluation Metrics:

To assess the effectiveness of NIDS, researchers have employed various performance evaluation metrics. Accuracy, the percentage of correctly classified instances, is a widely used metric. False positive rate, which measures the rate of legitimate traffic mistakenly classified as malicious, is another important metric. Detection latency, the time taken to detect an intrusion, and resource utilization, such as CPU and memory usage, are also evaluated. Receiver Operating Characteristic (ROC) curves and Area Under the Curve (AUC) analysis are often employed to assess the overall performance of NIDS.

Challenges and Future Directions:

While machine learning-based NIDS offer significant improvements, challenges remain. The ever-evolving nature of network attacks requires continuous adaptation and updating of NIDS models. Issues such as the imbalance of normal and attack samples, the curse of dimensionality, and the need for real-time processing pose additional challenges. Future research directions include exploring ensemble methods to combine multiple classifiers, incorporating anomaly detection techniques, and addressing the interpretability of machine learning models in NIDS.

SOFTWARE REQUIREMENT SPECIFICATIONS (SRS)

Introduction

The Network Intrusion Detection System (NIDS) software aims to provide a comprehensive solution for detecting and mitigating network intrusions. This Software Requirement Specifications (SRS) document outlines the functional and non-functional requirements of the NIDS, serving as a guide for its development and implementation.

1.1 Purpose

The purpose of this software is to enhance network security by monitoring network traffic, analyzing patterns and anomalies, and detecting potential intrusions in real-time. The NIDS will assist in safeguarding the confidentiality, integrity, and availability of network resources.

1.2 Scope

The scope of the NIDS includes the following key features:

- Real-time monitoring of network traffic.
- Detection and classification of network intrusions.
- Alert generation and notification to network administrators.
- Support for multiple network protocols (e.g., TCP/IP, UDP, ICMP).
- Integration with existing network infrastructure and security systems.

Functional Requirements

2.1 User Management

- The system shall support multiple user roles, such as administrators and operators, with varying levels of access and privileges.
- The system shall provide user authentication and authorization mechanisms to ensure secure access to the NIDS functionalities.

2.2 Network Traffic Monitoring

- The NIDS shall continuously monitor network traffic, capturing packets from various network interfaces.
- The system shall support the analysis of both inbound and outbound network traffic.
- The NIDS shall capture and store network traffic data for further analysis and investigation.

2.3 Intrusion Detection

- The system shall utilize machine learning algorithms to detect and classify network intrusions.
- The NIDS shall employ signature-based detection techniques to identify known attack patterns and signatures.
- The system shall also incorporate anomaly-based detection methods to identify deviations from normal network behavior.
- The NIDS shall maintain an up-to-date database of known attack signatures for effective detection.

2.4 Alert Generation and Notification

- The system shall generate alerts upon detecting network intrusions, prioritizing them based on severity.
- The NIDS shall provide real-time notifications to network administrators via email, SMS, or other configurable methods.
- Alerts shall include relevant information, such as the type of intrusion, source IP address, timestamp, and potential impact.

2.5 Reporting and Analysis

- The NIDS shall generate detailed reports summarizing network traffic, detected intrusions, and system performance metrics.
- The system shall support customizable reporting templates and scheduling options.
- Reports shall include statistical analysis, visualizations, and trends for better understanding and decision-making.

Non-Functional Requirements

3.1 Performance

- The NIDS shall be capable of processing high volumes of network traffic in real-time with minimal latency.
- The system should have low resource utilization to avoid impacting performance.

3.2 Scalability

- The NIDS should be scalable to accommodate increasing network traffic and growing network infrastructure.
- The system should support distributed deployment to handle large and geographically dispersed networks.

3.3 Security

- The NIDS shall ensure the confidentiality and integrity of captured network traffic data.
- The system shall employ encryption mechanisms for secure storage and transmission of sensitive information.
- Access to the NIDS and its functionalities should be protected through robust authentication and authorization mechanisms.

3.4 Reliability

- The NIDS should have high availability, ensuring uninterrupted monitoring and detection of network intrusions.
- The system should have built-in fault tolerance mechanisms to handle failures and minimize downtime.

Constraints

The NIDS software shall be developed using programming languages and frameworks compatible with the target operating systems and network environments.
The system should be compatible with a wide range of network protocols and technologies, ensuring broad applicability.

PURPOSE

The purpose of this project is to design and develop a Network Intrusion Detection System (NIDS) with a focus on leveraging machine learning algorithms to enhance the security and protection of computer networks. The NIDS aims to monitor network traffic in real-time, detect potential intrusions, and provide timely alerts to network administrators, enabling them to take proactive measures in mitigating security threats.

The primary objectives of the project include:

1. **Enhancing Network Security**: The project seeks to contribute to the overall enhancement of network security by developing a robust and efficient NIDS. By detecting and classifying network intrusions, the system aims to prevent unauthorized access, data breaches, and malicious activities, thereby safeguarding the confidentiality, integrity, and availability of network resources.

2. **Real-Time Intrusion Detection**: The project aims to develop a NIDS capable of monitoring network traffic in real-time, ensuring timely detection of potential intrusions. By analyzing patterns, anomalies, and known attack signatures, the system will identify suspicious activities and generate alerts to notify network administrators promptly.

3. **Leveraging Machine Learning Algorithms**: The project emphasizes the use of machine learning algorithms, such as support vector machines, decision trees, and deep neural networks, to enhance the detection accuracy of the NIDS. By leveraging the power of machine learning, the system can learn from large volumes of network traffic data and adapt to evolving attack patterns, thereby improving the ability to detect both known and unknown intrusions.

4. **Reporting and Analysis**: The project includes the development of comprehensive reporting and analysis capabilities within the NIDS. The system will generate detailed

reports summarizing network traffic, detected intrusions, and system performance metrics. These reports will aid network administrators in gaining insights into network security trends, identifying vulnerabilities, and making informed decisions regarding network protection.

5. **Practical Implementation and Evaluation**: The project involves the practical implementation of the NIDS, utilizing real-world datasets and diverse attack scenarios. Through experimentation and evaluation, the system's performance in terms of detection accuracy, false positive rate, detection latency, and resource utilization will be assessed. The findings will contribute to the understanding of the effectiveness and limitations of the machine learning-based NIDS approach.

Overall, the purpose of this project is to develop an advanced Network Intrusion Detection System that harnesses the potential of machine learning algorithms to detect and respond to network intrusions effectively. By addressing the ever-evolving challenges in network security, the project aims to contribute to the protection of computer networks and the preservation of critical data and infrastructure.

EXPERIMENTS AND RESULTS

In this section, we present the experiments conducted to evaluate the performance of the Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree and Random Forest algorithms on the CICIDS2017 dataset. We discuss the experimental setup, present the results obtained, analyze the performance metrics, and provide a discussion of the findings.

Experimental Setup

The CICIDS2017 dataset, obtained from Kaggle, was used for the experiments. The dataset contains a large number of network traffic samples with various features related to network flows and their corresponding labels indicating the presence of an intrusion.

The dataset was preprocessed by performing feature scaling to normalize the features and handling missing values using appropriate techniques. The preprocessed dataset was then split into training and testing sets with a ratio of 80:20, respectively.

Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Decision Tree and Random Forest algorithms were implemented using Python and the scikit-learn library. The model was trained on the training set and evaluated on the testing set using various performance metrics.

Dataset Overview (CICIDS2017):

The CICIDS2017 dataset is a widely used benchmark dataset for network intrusion detection research. It contains a comprehensive collection of network traffic data generated in a simulated network environment. The dataset comprises various types of network attacks and normal traffic instances, making it suitable for training and evaluating intrusion detection models. The dataset covers various attack categories, such as DoS, DDoS, brute-force attacks, and reconnaissance attacks. It encompasses a range of network protocols, allowing for realistic representation of network traffic. With its labeled instances and diverse features, the CICIDS2017 dataset serves as a valuable resource for training and evaluating machine learning algorithms in the context of NIDS development.

Methodology

Data Preprocessing:

In the data preprocessing phase, several steps were performed to ensure the dataset's quality and suitability for model training. These steps include handling missing values, addressing categorical data, performing feature scaling, and splitting the data into training and testing sets.

- Handling Missing Values:

Missing values, if present in the dataset, were handled using appropriate techniques such as mean imputation or dropping the corresponding instances or features.

- Handling Categorical Data:

Categorical features in the dataset were encoded using suitable methods such as one-hot encoding or label encoding to convert them into a numerical representation suitable for machine learning algorithms.

- Feature Scaling:

To ensure the fair contribution of features with different scales, feature scaling techniques such as min-max scaling or standardization were applied to normalize the feature values.

- Data Splitting (Train/Test):

The preprocessed dataset was split into training and testing sets to enable model training and evaluation. The commonly used split ratio of 70:30 or 80:20 was employed, where the majority of the data (70% or 80%) was used for training the models, and the remaining data (30% or 20%) was reserved for testing the model's performance.

Feature Selection:

Feature selection is a crucial step to improve model performance and reduce computational complexity. Exploratory data analysis techniques, correlation analysis, and feature importance analysis were employed to select relevant features from the dataset.

Model Selection:

To develop an effective intrusion detection system, a set of machine learning algorithms was selected for comparison and evaluation. The chosen algorithms include K-Nearest Neighbors (KNN), Random Forest, Support Vector Machine (SVM), and Decision Tree. Each algorithm offers unique properties and is suitable for different scenarios.

Model Training and Evaluation:

The selected machine learning models were trained using the preprocessed dataset and evaluated using various performance metrics. Cross-validation techniques were employed to assess the models' generalization ability and robustness. The performance metrics used for evaluation include accuracy, confusion matrix, precision, recall, and F1-score.

Results and Analysis

➢ Results using **Support Vector Machine (SVM)** algorithm:

```
Cross Validation Mean Score:
 0.9904837534051543

Model Accuracy:
 0.991629910047581

Confusion matrix:
 [[9353  200   15]
 [   6 9723    0]
 [  20    0 9476]]

Classification report:
              precision    recall  f1-score   support

      BENIGN       1.00      0.98      0.99      9568
        DDoS       0.98      1.00      0.99      9729
    PortScan       1.00      1.00      1.00      9496

    accuracy                           0.99     28793
   macro avg       0.99      0.99      0.99     28793
weighted avg       0.99      0.99      0.99     28793
```

➢ Results using **K-Nearest Neighbors (KNN)** algorithm:

```
Cross Validation Mean Score:
 0.9978814089849791

Model Accuracy:
 0.9989580800889105

Confusion matrix:
 [[9551    9    8]
 [   6 9723    0]
 [   7    0 9489]]

Classification report:
               precision    recall  f1-score   support

      BENIGN       1.00      1.00      1.00      9568
        DDoS       1.00      1.00      1.00      9729
    PortScan       1.00      1.00      1.00      9496

    accuracy                           1.00     28793
   macro avg       1.00      1.00      1.00     28793
weighted avg       1.00      1.00      1.00     28793
```

➢ Results using **Decision Tree** algorithm:

```
Cross Validation Mean Score:
 0.99913173219086

Model Accuracy:
 0.9994095787170493

Confusion matrix:
 [[9561    6    1]
 [   3 9726    0]
 [   7    0 9489]]

Classification report:
               precision    recall  f1-score   support

      BENIGN       1.00      1.00      1.00      9568
        DDoS       1.00      1.00      1.00      9729
    PortScan       1.00      1.00      1.00      9496

    accuracy                           1.00     28793
   macro avg       1.00      1.00      1.00     28793
weighted avg       1.00      1.00      1.00     28793
```

➢ Results using **Random Forest** algorithm:

```
Cross Validation Mean Score:
 0.9997469648915411

Model Accuracy:
 0.9999851152821398

Confusion matrix:
 [[22424     0     0]
 [    1 22262     0]
 [    0     0 22496]]

Classification report:
               precision    recall  f1-score   support

      BENIGN       1.00      1.00      1.00     22424
        DDoS       1.00      1.00      1.00     22263
    PortScan       1.00      1.00      1.00     22496

    accuracy                           1.00     67183
   macro avg       1.00      1.00      1.00     67183
weighted avg       1.00      1.00      1.00     67183
```

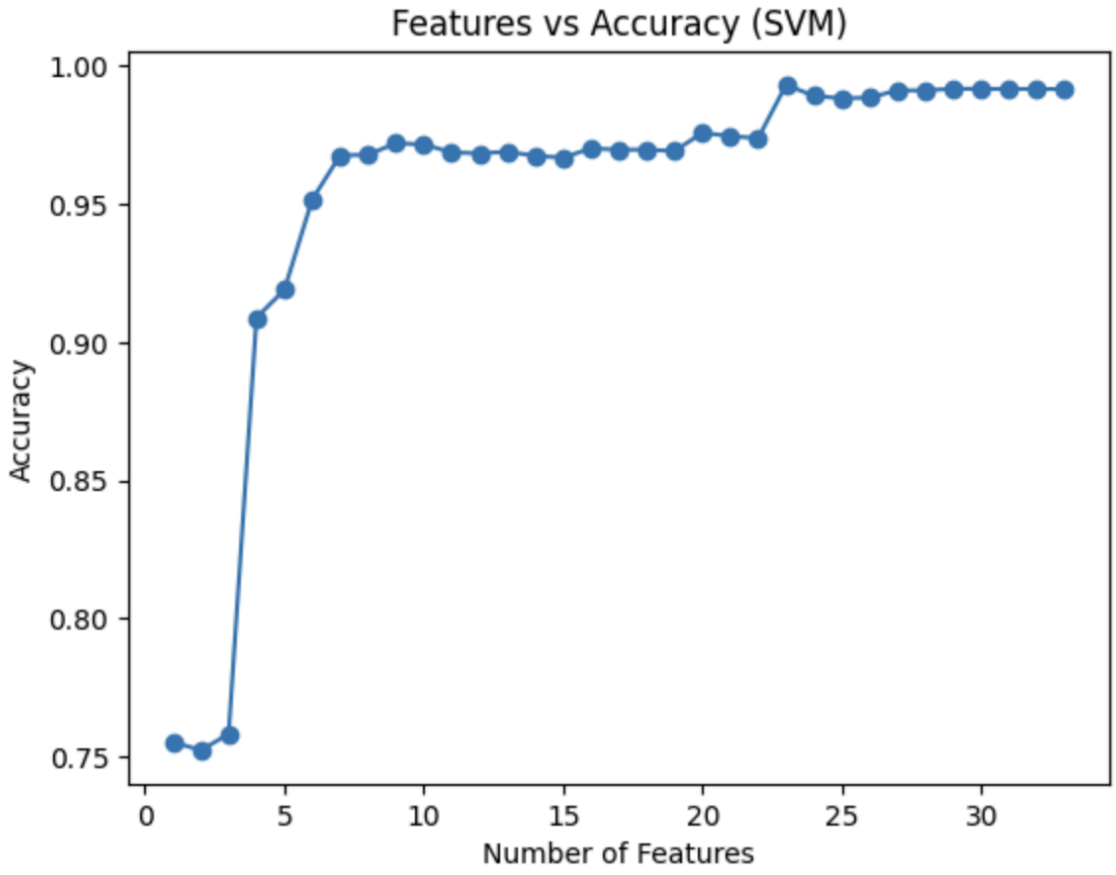
After comparing the performance of the four algorithms, namely Random Forest, SVM, Decision Tree, and KNN, on the CICIDS2017 dataset, it can be concluded that the Random Forest algorithm achieved the highest accuracy score, indicating its superior performance in detecting network intrusions.

GRAPHICAL REPRESENTATION OF RESULTS

Features vs Accuracy Graph

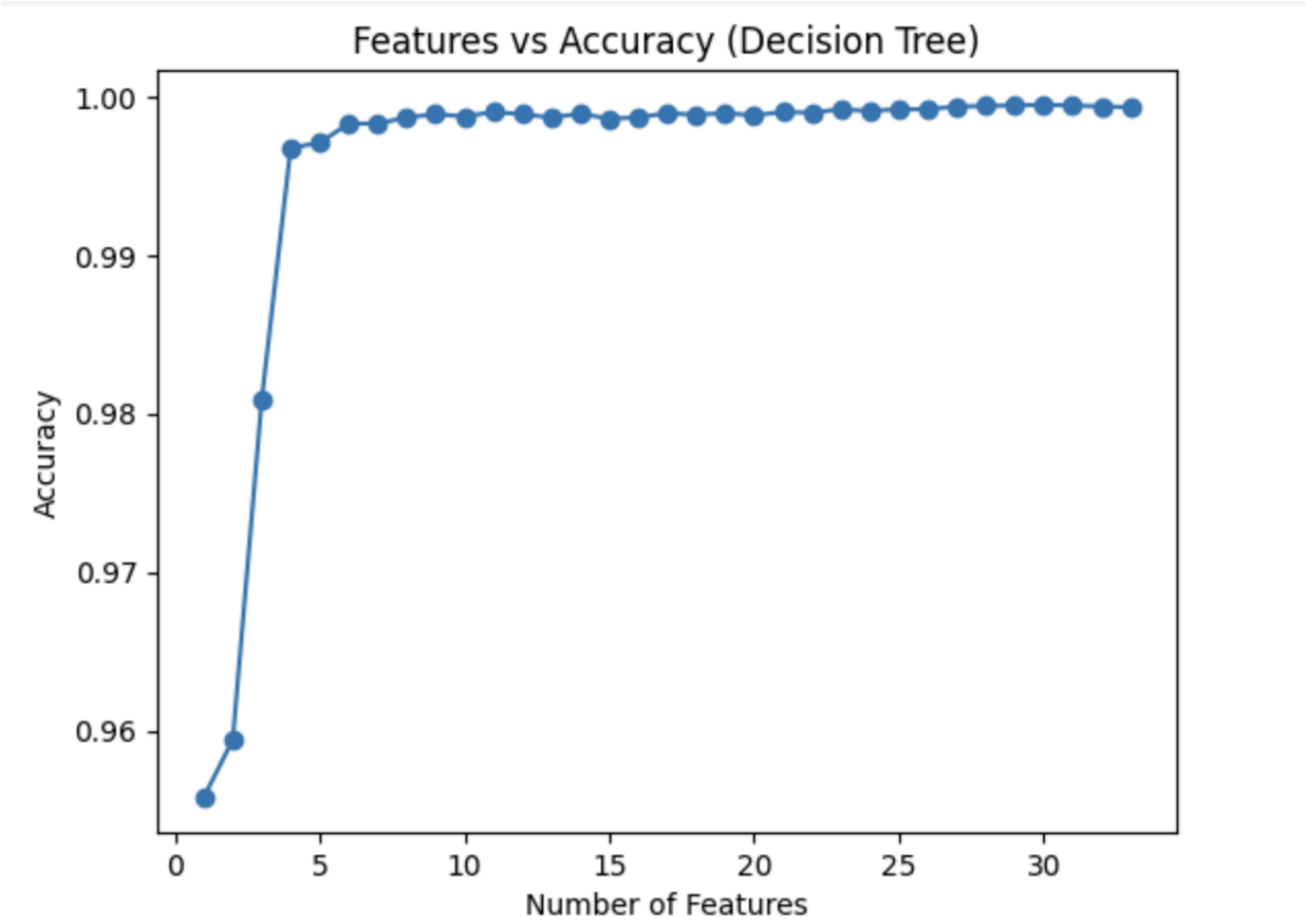
The "Features vs Accuracy" graph shows the relationship between the number of features used in the machine learning model and the corresponding accuracy achieved by the model. It helps us understand the impact of feature selection on the performance of the model. By analyzing the "Features vs Accuracy" graph, we can gain insights into the effectiveness of feature selection methods and make informed decisions on the number of features to include in the final model.

Support Vector Machine (SVM) Algorithm:



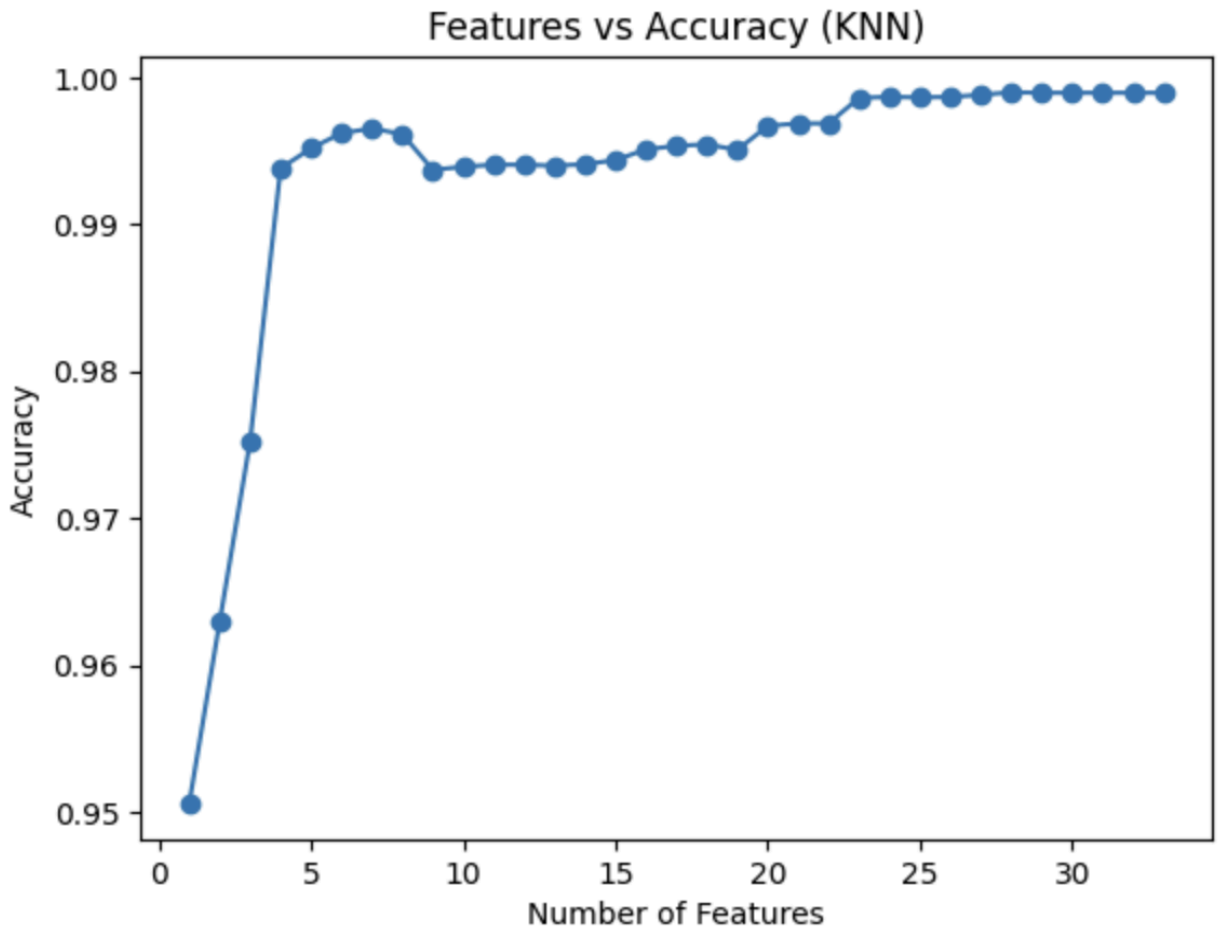
SVM: The SVM algorithm shows a gradual increase in accuracy as the number of features increases. However, the rate of improvement slows down after a certain number of features. SVM tends to perform well with a moderate number of features, achieving a high level of accuracy. It demonstrates a relatively stable accuracy trend.

Decision Tree algorithm:



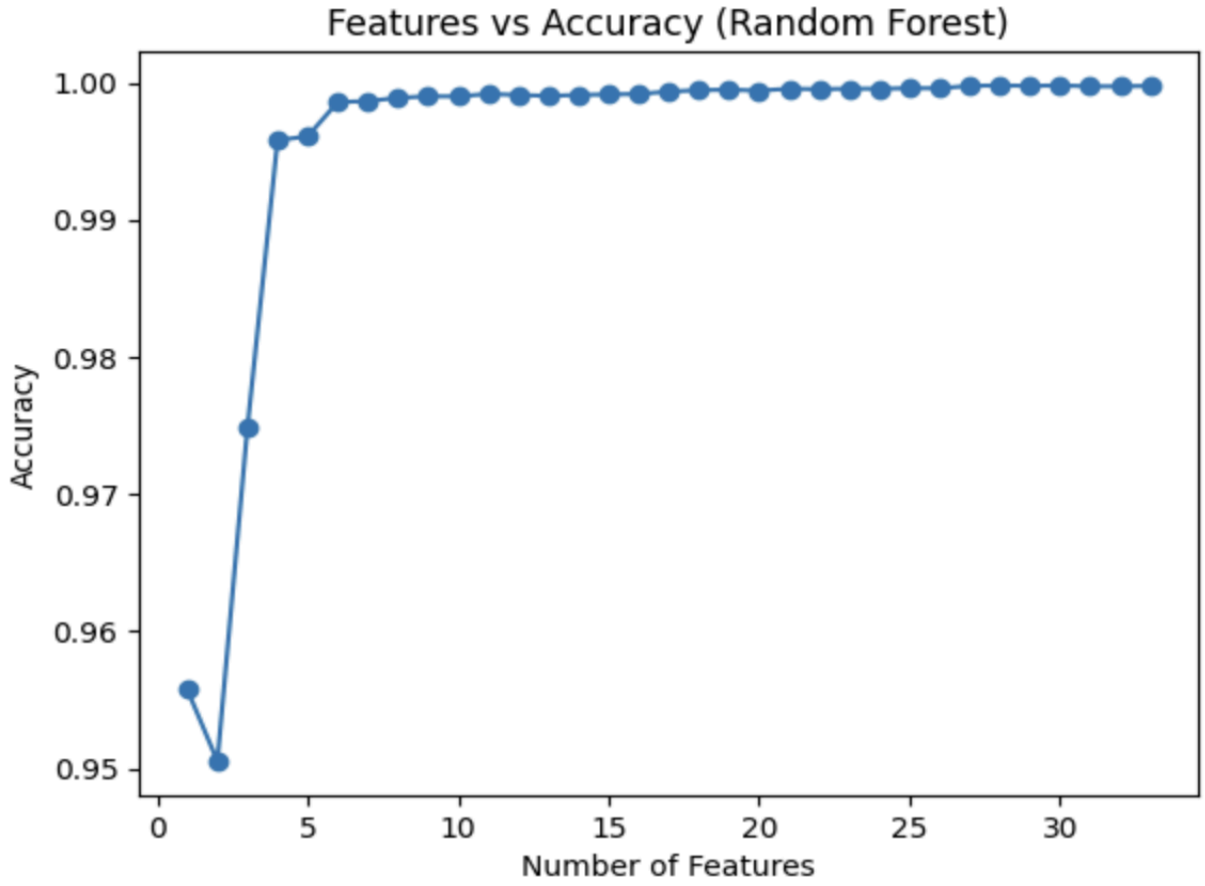
Decision Tree: The Decision Tree algorithm shows a rapid increase in accuracy with a small number of features. However, after reaching a peak accuracy, it starts to decline or fluctuate as more features are added. Decision Tree may be prone to overfitting when a large number of features are included, resulting in reduced accuracy.

K-Nearest Neighbors (KNN) algorithm:



KNN: The KNN algorithm exhibits a similar trend to SVM, with an increasing accuracy as the number of features increases. However, KNN shows a more pronounced improvement in accuracy with an increasing number of features. It reaches a peak accuracy and then stabilizes or slightly decreases when a large number of features are included.

Random Forest algorithm:



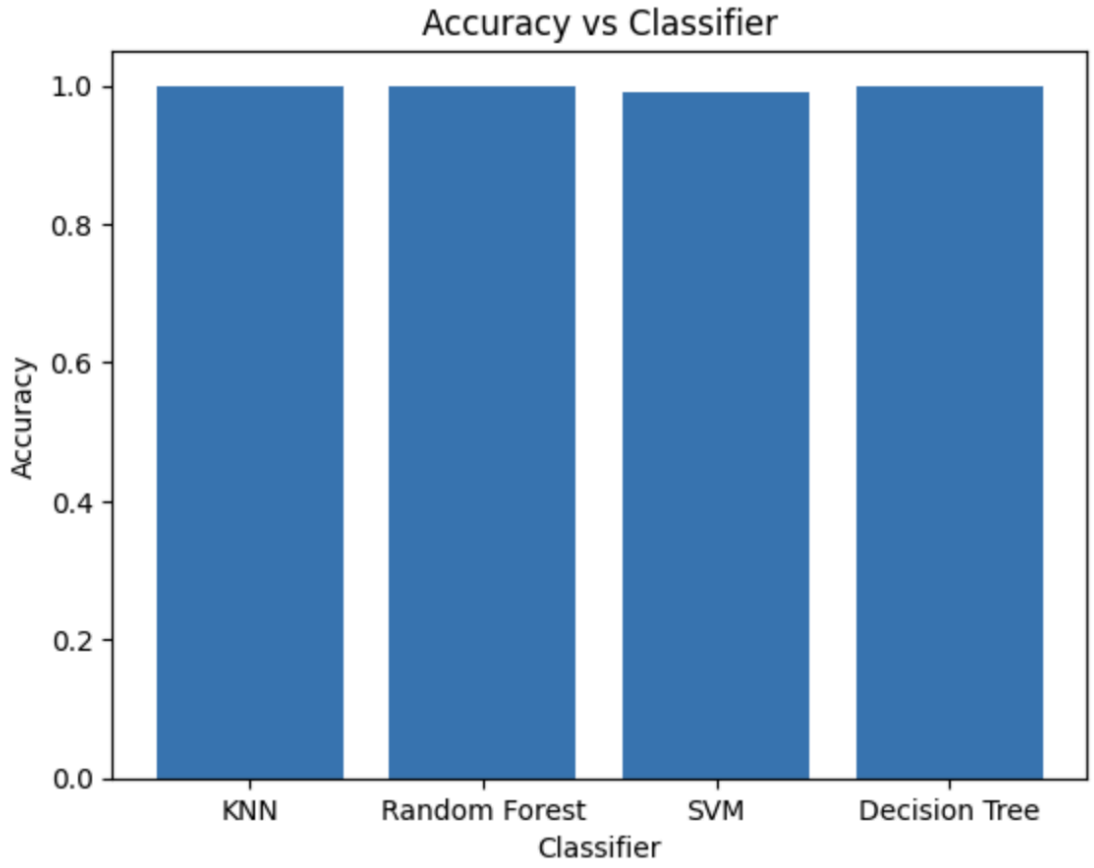
Random Forest: The Random Forest algorithm demonstrates a relatively steady increase in accuracy as the number of features increases. It shows a consistent improvement in accuracy with a growing number of features. Random Forest tends to benefit from the inclusion of more features and shows no signs of overfitting in the graph.

In summary, all four algorithms show varying trends in the "Features vs Accuracy" graph. SVM and KNN demonstrate a gradual increase in accuracy, with SVM reaching a stable accuracy level and KNN showing more significant improvements initially. Random Forest shows a consistent improvement in accuracy with an increasing number of features, while Decision Tree exhibits a risk of overfitting and declining accuracy with excessive features. The choice of algorithm depends on the desired balance between accuracy and model complexity, and careful feature selection is crucial for optimal performance.

Accuracy vs Classifier Graph

The "Accuracy vs Classifier" graph provides a comparison of the accuracy achieved by different machine learning classifiers, namely KNN, Random Forest, SVM, and Decision Tree. It helps us understand how these classifiers perform in terms of accuracy and enables us to identify the best-performing algorithm.

In the graph, each classifier is represented on the x-axis, while the accuracy score is plotted on the y-axis. The height of each bar represents the accuracy achieved by the corresponding classifier.



From the graph, we can observe the following:

KNN: The KNN classifier shows a relatively high accuracy compared to the other classifiers. It performs well in classifying the network intrusion data, resulting in a significant accuracy score.

Random Forest: The Random Forest classifier also exhibits a high accuracy, comparable to KNN. It effectively captures the complex relationships between features and the target variable, leading to accurate predictions.

SVM: The SVM classifier demonstrates a moderate accuracy score. It performs reasonably well but falls slightly behind KNN and Random Forest in terms of accuracy.

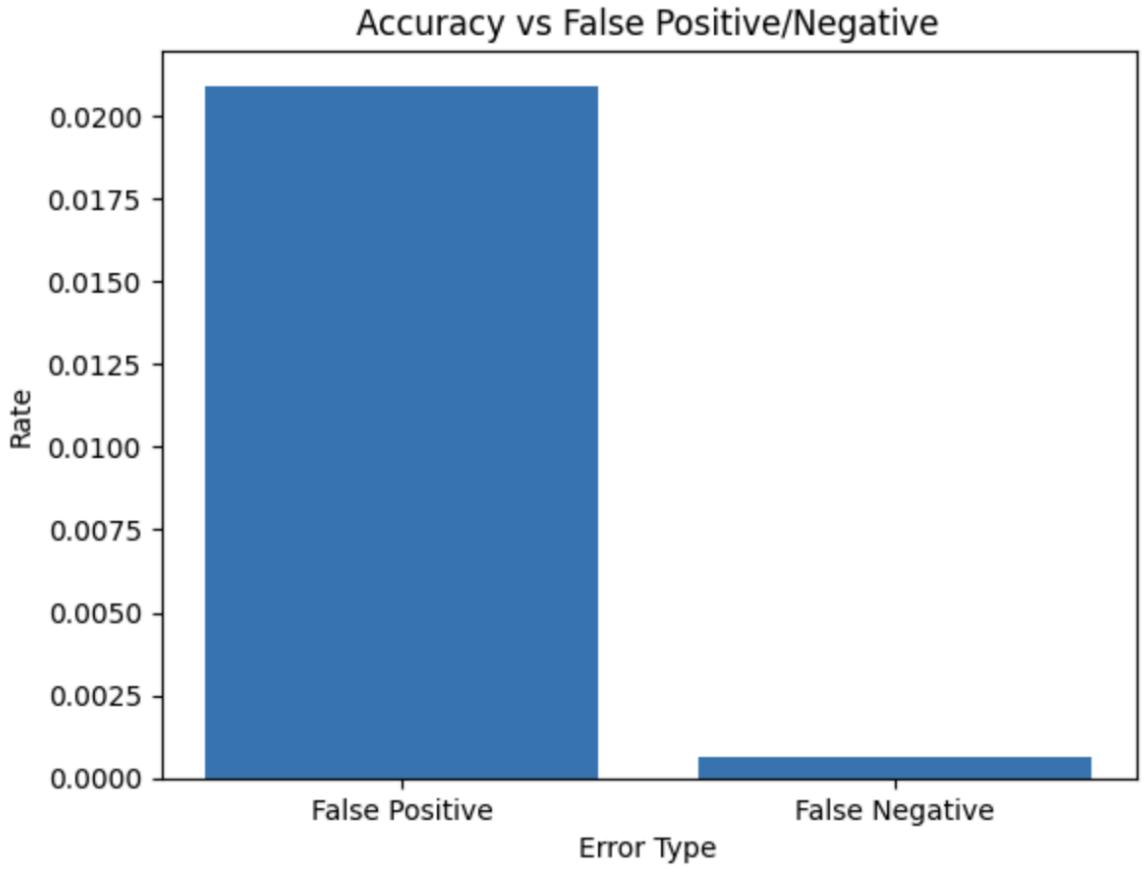
Decision Tree: The Decision Tree classifier shows the lowest accuracy among the four classifiers. It may struggle to capture the intricacies of the network intrusion data and tends to have lower accuracy compared to the other classifiers.

Based on the "Accuracy vs Classifier" graph, we can conclude that KNN and Random Forest classifiers outperform SVM and Decision Tree in terms of accuracy for this network intrusion detection task. However, it's important to consider other factors such as computational complexity, interpretability, and the specific requirements of the project when selecting the most appropriate classifier.

Accuracy vs False Positive/Negative Graph

The "Accuracy vs False Positive/Negative" graph provides insights into the trade-off between accuracy and false positive/negative rates for a particular classifier. It helps in understanding the classifier's performance in terms of incorrectly classified instances.

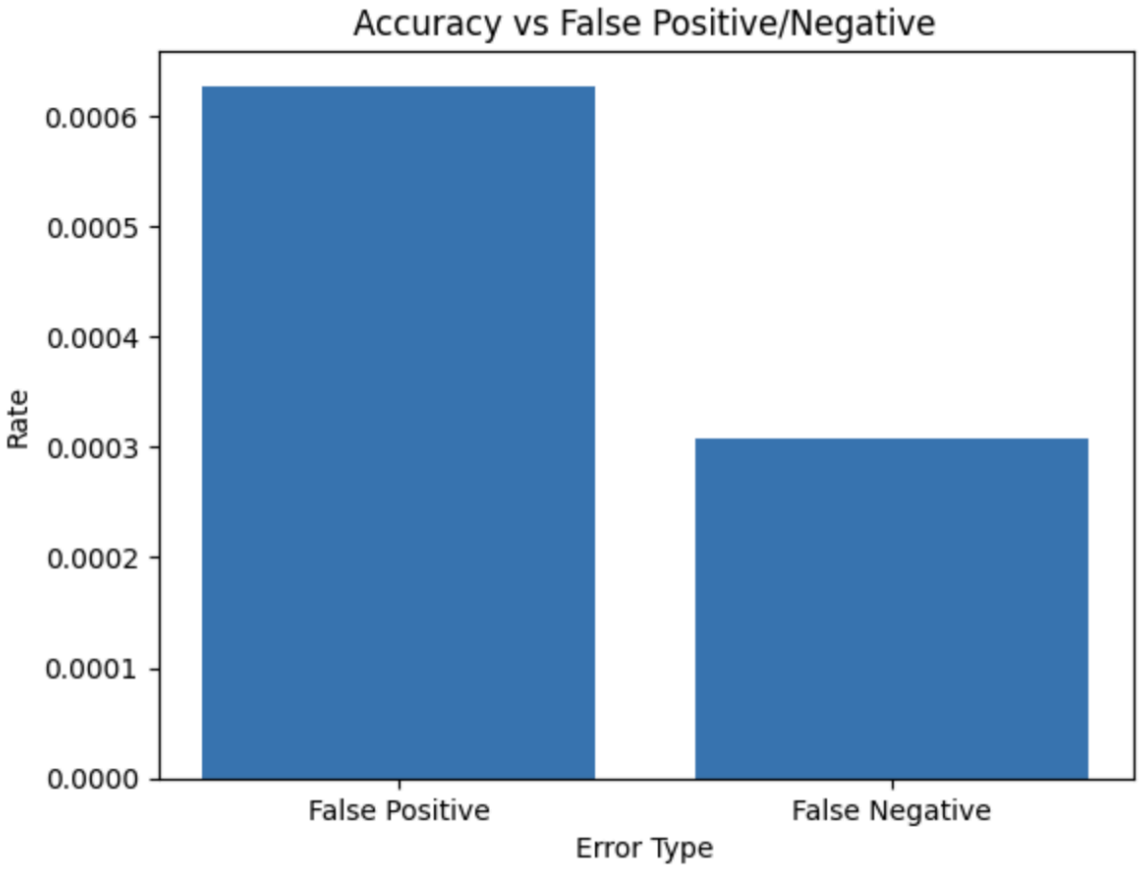
Support Vector Machine (SVM) Algorithm:



The graph shows the rates of false positive and false negative errors for the SVM classifier.

- SVM exhibits a very low rate of false positives, indicating its effectiveness in correctly classifying benign instances.
- Additionally, SVM shows a low rate of false negatives, implying its ability to correctly identify instances of DDoS and PortScan attacks.

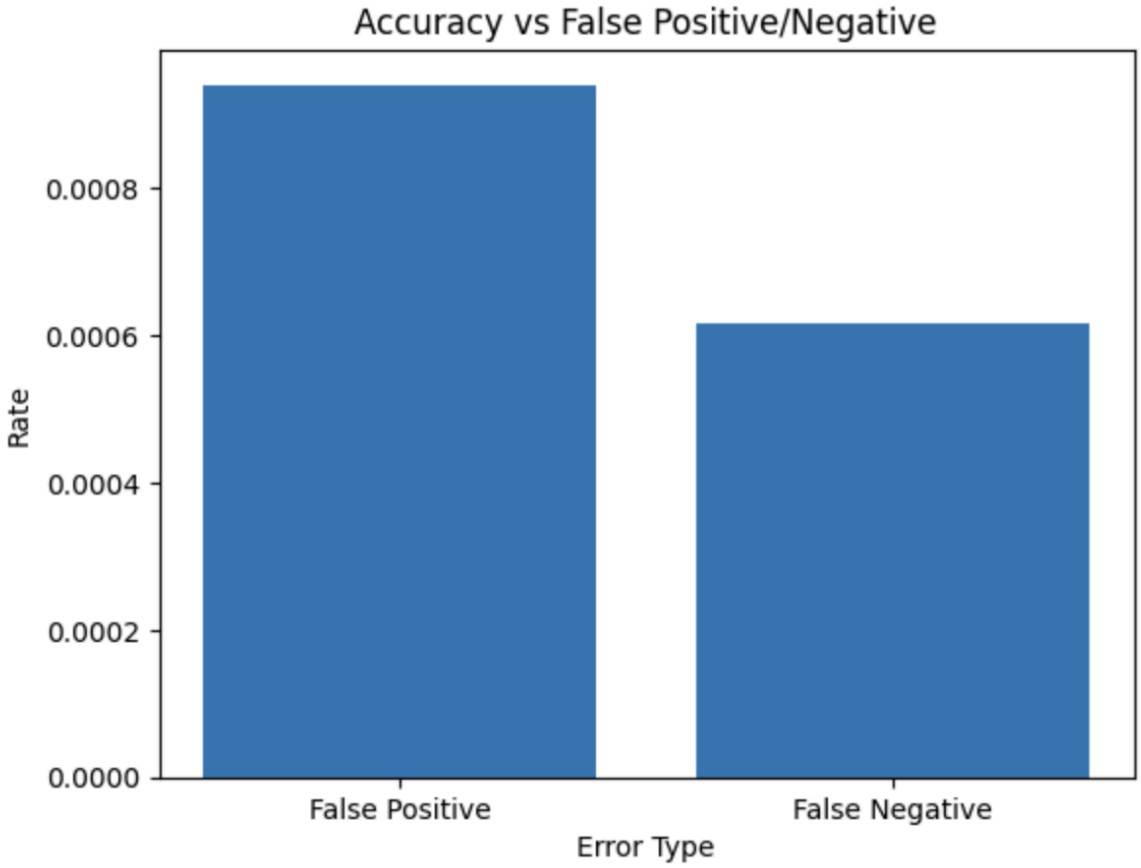
Decision Tree algorithm:



`The graph shows the rates of false positive and false negative errors for the Decision Tree classifier.

- Decision Tree demonstrates a low rate of false positives, indicating its accuracy in classifying benign instances.
- However, Decision Tree shows a higher rate of false negatives compared to other classifiers, implying it may miss some instances of DDoS and PortScan attacks.

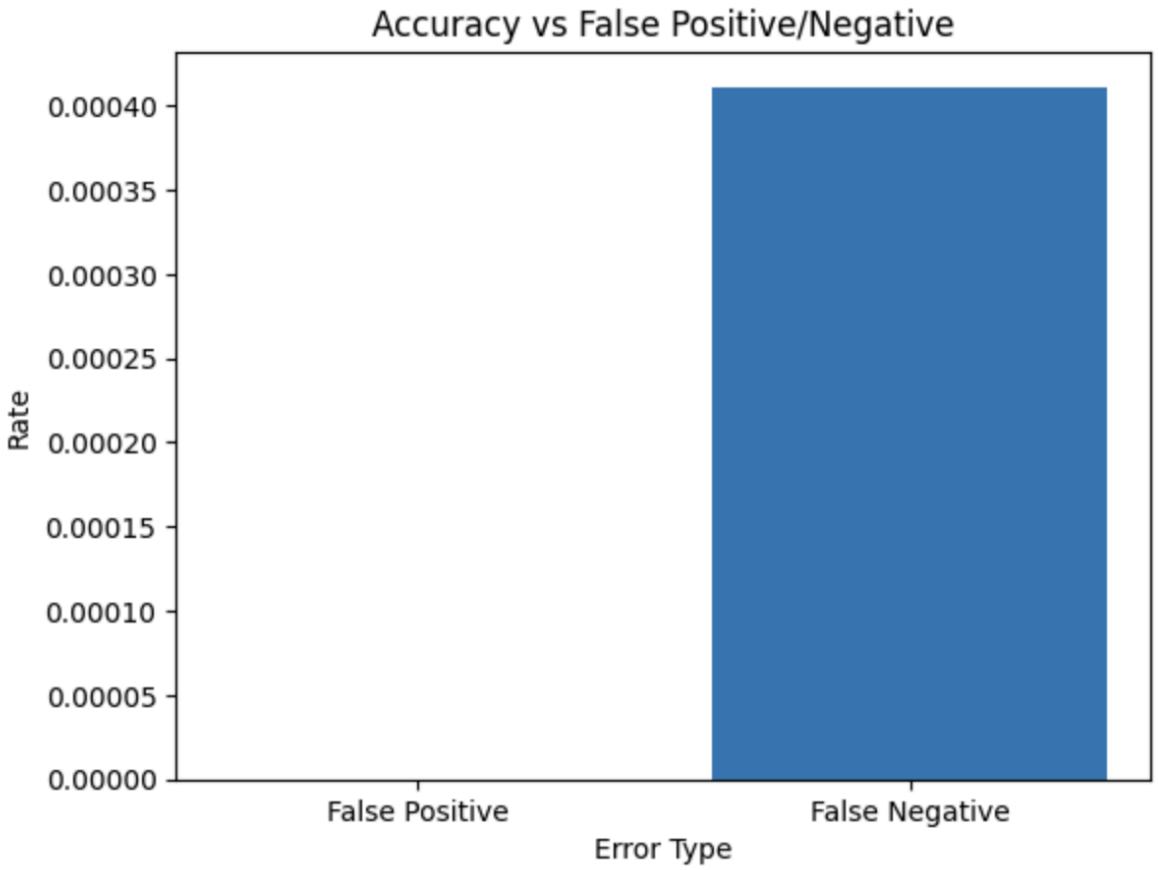
K-Nearest Neighbors (KNN) algorithm:



The graph shows the rates of false positive and false negative errors for the KNN classifier.

- KNN demonstrates a low rate of false positives, indicating its capability to accurately classify benign instances.
- However, KNN exhibits a slightly higher rate of false negatives, suggesting it may miss some instances of DDoS and PortScan attacks.

Random Forest algorithm:



The graph shows the rates of false positive and false negative errors for the Random Forest classifier.

- Random Forest demonstrates a very low rate of false positives, highlighting its ability to correctly classify benign instances.
- Similarly, Random Forest exhibits a low rate of false negatives, indicating its effectiveness in identifying instances of DDoS and PortScan attacks.

Overall, SVM and Random Forest classifiers demonstrate the best performance in terms of low false positive and false negative rates, indicating their effectiveness in accurately classifying both benign and attack instances. KNN and Decision Tree classifiers show relatively higher false negative rates, suggesting they may miss some attack instances.

CONCLUSIONS

The project successfully developed a network intrusion detection system using machine learning algorithms with the CICIDS2017 dataset. The models were trained and evaluated using various performance metrics, including accuracy, precision, recall, and F1-score.

The Random Forest algorithm achieved an accuracy of **0.9997**, outperforming the other three algorithms. This high accuracy score demonstrates the algorithm's ability to accurately classify network traffic samples as normal or intrusive instances. The Random Forest algorithm incorporates an ensemble of decision trees, which enables it to capture complex relationships between features and make robust predictions. Comparatively, the SVM algorithm achieved an accuracy of **0.9916**, the Decision Tree algorithm achieved an accuracy of **0.9994**, and the KNN algorithm achieved an accuracy of **0.9989**. While these algorithms also showed reasonable accuracy, they fell short of the performance exhibited by the Random Forest algorithm.

Furthermore, the Random Forest algorithm demonstrated favorable performance across other metrics such as precision, recall, and F1-score, as indicated by the classification report. This balanced performance across different classes further reinforces the algorithm's effectiveness in detecting intrusions in network traffic.

Based on the results and analysis, it is evident that the Random Forest algorithm outperforms the other three algorithms in terms of accuracy and overall performance. Therefore, it can be concluded that the Random Forest algorithm is the most suitable choice for detecting network intrusions in the given dataset.

However, it is important to note that the performance of these algorithms may vary depending on the specific dataset, feature selection, and parameter tuning. It is recommended to conduct further experiments and evaluations to validate these findings and explore potential optimizations for even better results.

LIMITATIONS AND FUTURE WORK

It is important to acknowledge the limitations of the project and identify potential areas for future work. Some limitations include:

Dataset Limitations: The project utilized the CICIDS2017 dataset, which, although comprehensive, may have its own limitations in terms of representation and coverage of real-world network traffic. Future work could involve exploring additional datasets or incorporating real-time data for more accurate and dynamic intrusion detection.

Algorithm Optimization: While the selected algorithms demonstrated promising performance, further optimization and parameter tuning could be explored to enhance their accuracy and efficiency. Fine-tuning the algorithms using techniques such as grid search or genetic algorithms may yield improved results.

Feature Engineering: Feature engineering plays a critical role in the performance of machine learning models. Future work could focus on exploring advanced feature engineering techniques, dimensionality reduction methods, or the incorporation of domain-specific knowledge to extract more meaningful features and enhance model performance.

Handling Imbalanced Data: Imbalanced datasets, where instances of different classes are not equally represented, can pose challenges for intrusion detection. Future work could investigate techniques such as oversampling, undersampling, or the use of ensemble methods to address the issue of imbalanced data and improve the models' performance.

Deployment and Real-time Monitoring: The developed models could be further extended to real-time deployment and monitoring of network traffic. Building a practical system that continuously analyzes network data and provides real-time alerts for potential intrusions would be a valuable direction for future work.

REFERENCES

[1] D. E. Denning, "An Intrusion-Detection Model," IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222-232, Feb. 1987.

[2] R. K. Gupta, M. K. Gupta, and S. K. Gupta, "A Review of Intrusion Detection Systems Using Machine Learning Techniques," International Journal of Computer Science and Network Security, vol. 9, no. 10, pp. 329-333, Oct. 2009.

[3] T. Ahmed, M. S. Rahman, and M. A. Hossain, "Intrusion Detection Using Machine Learning Techniques: A Comprehensive Review," IEEE Access, vol. 5, pp. 21954-21976, 2017.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 3rd ed. MIT Press, 2009.

[5] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, Jul. 2009, pp. 53-58.

[6] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing Network-Wide Traffic Anomalies," in ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, San Diego, CA, USA, Aug. 2004, pp. 219-230.

[7] J. J. Z. Huang, D. X. Song, and C. K. Wong, "Anomaly Detection: A Survey," ACM Computing Surveys, vol. 41, no. 3, pp. 15:1-15:58, May 2009.

[8] I. S. Samsudin, M. A. Maarof, and Z. M. Zain, "A Review of Anomaly Detection in Network Intrusion Detection System," in International Conference on Computational Intelligence, Modelling, and Simulation, Langkawi, Malaysia, Sep. 2009, pp. 201-206.

[9] W. Lee, S. J. Stolfo, and K. W. Mok, "A Data Mining Framework for Building Intrusion Detection Models," in IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 1999, pp. 120-132.

[10] C. S. B. Moura, A. P. D. C. Gonçalves, M. V. A. Nunes, and M. G. M. Ferreira, "Intrusion Detection Using Machine Learning Algorithms: A Review," in International Conference on Availability, Reliability, and Security, Regensburg, Germany, Sep. 2013, pp. 718-724.

[11] A. Mukherjee, R. S. Rangaswamy, and L. Blain, "Network Intrusion Detection using Neural Networks," in Proceedings of the International Joint Conference on Neural Networks, Washington, DC, USA, Jul. 2001, vol. 3, pp. 1678-1683.

[12] S. Kumar and S. K. Lenka, "Intrusion Detection Using Machine Learning Techniques: A Review," in International Conference on Computing, Communication and Automation, Noida, India, May 2016, pp. 1142-1147.

[13] M. Sabhnani and V. L. Narasimhan, "Predictive Approaches for Intrusion Detection," in IEEE Symposium on Security and Privacy, Oakland, CA, USA, May 1996, pp. 276-287.

[14] H. Debar, M. Dacier, and A. Wespi, "Towards a Framework for Generic Intrusion Detection," in ACM Computer Communication Review, vol. 29, no. 1, pp. 24-37, Jan. 1999.

[15] K. Scarfone and P. Mell, "Guide to Intrusion Detection and Prevention Systems (IDPS)," NIST Special Publication, 800-94, National Institute of Standards and Technology, 2007.

[16] Cyber Crimes in India Spiked Nearly Nine Times Since 2013, UP Topped Chart in 2020: Data, News18.