Vol.15, Issue No 2, 2025

DETECTING UNAUTHORIZED ACCESS IN CYBER NETWORKS:A MACHINE LEARNING APPROACH

Veeraboina Mounika,
UG Student,
Department of CSE,
St. Martin's Engineering College,
Secunderabad, Telangana, India
harikamattaparthi2003@gmail.com

Mrs. A Rajeshwari,
Assistant Professor,
Department of CSE,
St. Martin's Engineering College,
Secunderabad, Telangana, India
rajeshwaricse@smec.ac.in

Abstract- Intrusion Detection Systems (IDS) have been essential in cyber security since the 1980s, when the concept of monitoring network traffic and system activities to detect malicious activities was introduced. Early IDS systems were primarily signature-based, relying on predefined rules and known attack patterns to identify threats. The primary objective of this study is to evaluate the performance of machine learning classifiers in detecting and mitigating cyber intrusions. The title refers to the assessment of machine learning algorithms used to identify unauthorized or malicious activities within a network. It emphasizes the focus on evaluating the effectiveness and accuracy of these algorithms in detecting cyber intrusions. Before the advent of machine learning, traditional IDS relied on signature-based and rule-based detection methods. These systems would compare incoming data against a database of known attack signatures or predefined rules to detect anomalies. While effective for known threats, these methods were limited in detecting new or evolving attacks, often resulting in a high rate of false positives and missed intrusions. Traditional intrusion detection systems faced significant challenges in keeping up with the rapidly evolving landscape of cyber threats. Their reliance on static rules and known attack signatures made them inadequate for detecting sophisticated, zero-day attacks and adaptive adversaries. The growing complexity and frequency of cyber-attacks have highlighted the limitations of traditional IDS. The proposed system leverages machine learning models to enhance the detection of cyber intrusions. By training classifiers on large datasets of network traffic and system activity, these models can identify patterns and anomalies indicative of malicious behavior. Machine learning offers the advantage of adapting to new threats, improving detection accuracy, and reducing false positives compared to traditional methods. This approach provides a dynamic and scalable solution to modern cyber security challenges, making it a vital tool in protecting against emerging threats.

Keywords: Intrusion Detection Systems, IDS, cybersecurity, network traffic, system activities, malicious activities, signature-based detection, rule-based detection, machine learning classifiers.

I. INTRODUCTION

Intrusion Detection Systems (IDS) are vital in safeguarding networks from unauthorized access and cyber threats. Machine learning enhances IDS by enabling the detection of sophisticated attacks, reducing false positives, and improving overall system resilience. Applications include securing financial systems, government networks, healthcare infrastructures, and critical industrial control systems. Early IDS solutions were primarily signature-based, relying on predefined attack patterns to detect threats. However, as cyber threats evolved, these systems struggled to keep up, often resulting in high false-positive rates and missed detections. In India, the growing digital footprint and increasing reliance on online services have led to a surge in cyber-attacks, with incidents rising by over 300% from 2018 to 2022. This alarming trend underscores the need for advanced IDS solutions that can adapt to the dynamic nature of cyber threats, making machine learning-based approaches increasingly critical. Intrusion Detection Systems (IDS) are vital in safeguarding networks from unauthorized access and cyber threats. Machine learning enhances IDS by enabling the detection of sophisticated attacks, reducing false positives, and improving overall system resilience. Applications include securing financial systems, government networks, healthcare infrastructures, and critical industrial control systems.

The proposed system has wide-ranging applications across various sectors. In finance, it can protect against fraud and unauthorized access to financial systems. In healthcare, it ensures the security of patient data and medical devices. Government agencies can use it to safeguard national security by detecting cyber espionage and attacks on critical infrastructure. In industrial control systems, it prevents unauthorized access and potential sabotage of critical processes. The system is also applicable in

protecting online retail platforms from cyber-attacks, ensuring the safety of customer data and transactions. Educational institutions can use it to secure their networks and intellectual property.

II. RELATED WORK

The rapid increase in cyber-attacks, particularly in India, coupled with the limitations of traditional IDS, highlights the urgent need for more adaptive and accurate detection mechanisms. Machine learning offers a promising solution by enabling IDS to learn from vast datasets and detect previously unseen attack patterns. The motivation for this research is to explore and evaluate the potential of machine learning classifiers to enhance the detection of cyber intrusions, reduce false positives, and provide a more robust defense against emerging threats.

In [1], A. Verma et al. explored machine learning-based intrusion detection systems specifically tailored for IoT applications. The study presents various machine learning techniques and their applicability in detecting intrusions in IoT environments. The authors highlight the challenges posed by the resource-constrained nature of IoT devices and propose solutions to enhance detection accuracy while maintaining efficiency. A. Thakkar et al. [2] conducted a comprehensive review of advancements in intrusion detection datasets. The paper discusses the evolution of datasets used for evaluating intrusion detection systems, emphasizing the importance of realistic and diverse datasets to improve the performance and reliability of intrusion detection models. The authors also examine the limitations of existing datasets and suggest future research directions for dataset enhancement.

A. Khraisat et al. [3] provided an extensive survey of intrusion detection systems, covering techniques, datasets, and challenges faced in the field. The paper categorizes various intrusion detection techniques, including signature-based, anomaly-based, and hybrid approaches, and analyzes their effectiveness. Additionally, the authors discuss the challenges related to dataset quality, real-time detection, and the adaptability of intrusion detection systems in dynamic network environments. R. Bace et al. [4] contributed to the field with their NIST Special Publication on Intrusion Detection Systems. This foundational work offers a comprehensive overview of intrusion detection concepts, methodologies, and practical implementation guidelines. It serves as a critical reference for both researchers and practitioners in the cybersecurity domain, providing a baseline for the development and evaluation of intrusion detection systems.

As cyber threats became more complex, these traditional systems proved increasingly inadequate, highlighting the need for more advanced and adaptive approaches like machine learning. Those limitations are mentioned as follows:

- Inability to Detect Unknown Threats
- High False Positives in Anomaly-Based Systems
- Static and Rigid Detection Mechanisms
- Limited Scalability
- Maintenance Overhead
- Resource Intensive

III. PROPOSED WORK

This research uses Random Forest Classifier(RFC) which is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes (classification) of the individual trees. It is widely used for classification tasks because of its robustness and ability to handle large datasets with high-dimensional feature spaces. During prediction, each tree gives a classification, and the forest chooses the class that has the most votes among the trees. This process reduces the risk of overfitting and improves generalization.

Architecture of RFC

1. **Input Layer:** The input layer consists of the entire dataset, which is split into multiple subsets.
2. **Decision Trees:** The model builds several decision trees, each trained on a random subset of the data. These trees are created using a process called "bagging," which involves random sampling with replacement.
3. **Voting Mechanism:** During the prediction phase, each tree makes its prediction, and the final output is determined by majority voting among all the trees.
4. **Output Layer:** The output is the class label that received the majority of votes from the individual decision trees.

Advantages of RFC:

- Robustness: RFC is highly robust to noise and overfitting, as the ensemble approach mitigates the impact of any single noisy tree.

- Scalability**:** RFC can handle large datasets with high-dimensional feature spaces efficiently.
- Feature Importance**:** RFC provides a measure of feature importance, helping in feature selection and model interpretation.
- Versatility: RFC can be used for both classification and regression tasks and is adaptable to various types of data and problem domains.

The project provides a graphic depiction of the model's functionality. This application also visualizes the distribution of the dataset, showing the count of different attack types.

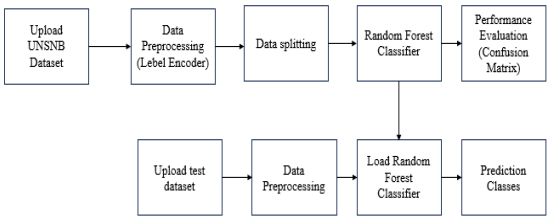

Fig 1: Overall design of proposed methodology

3.1 Data Preprocessing

Data preprocessing is the process of preparing raw data and making it suitable for machine learning models. This is the first important step when creating a machine learning model. When creating a machine learning project, you can't always find clean, formatted data. Also, when working with data, it is essential to clean it and save it in a formatted format. To do this, use data preprocessing tasks. Real world data typically contains noise, missing values, and may be in an unusable format that cannot be directly used in machine learning models. Data preprocessing is a necessary task to clean up data and make it suitable for machine learning models, which also improves the accuracy and efficiency of machine learning models.

- Uploading the dataset
- Importing libraries
- Importing datasets
- Finding missing data
- Removing null values
- Encoding Categorical data
- Splitting data into training and test set

Importing Libraries: To perform data preprocessing using Python, we need to import some predefined 3 Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

Scikit-learn: It provides a comprehensive, user-friendly interface to a wide range of common algorithms. When installing the library, it's typically done using the full name, scikit-learn, but with python code, it's imported and used as sklearn.

This includes efficient tools for machine learning and statistical modelling including classification, regression, clustering, and dimensionality reduction.

NumPy: The NumPy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to addition of large, multidimensional arrays and matrices. So, in Python, we can import it as: import NumPy as nm.

Here we have used nm, which is a short name for NumPy, and it will be used in the whole program.

Matplotlib: The third library is matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code. It will be imported as below: import matplotlib.pyplot as mpt

Here we have used mpt as a short name for this library.

Pandas: The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library. Here, we have used pd as a short name for this library. Consider the below image:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Fig 2: Library command

Handling Missing data: The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

• By deleting the particular row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

• By calculating the mean: In this way, we will calculate the mean of that column or row which

contains any missing value and will put it on the place of missing value.

```
dataset.isnull().sum()
```

Fig 3: Handle missing data

Encoding Categorical data: Categorical data is data which has some categories such as, in our dataset; there are two categorical variables, Country, and Purchased. Since machine learning model completely works on mathematics and numbers, but if our dataset would have a categorical variable, then it may create trouble while building the model. So, it is necessary to encode these categorical variables into numbers. Feature Scaling: Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no variable dominates the other variable. A machine learning model is based on Euclidean distance, and if we do not scale the variable, then it will cause some issue in our machine learning model. Euclidean distance is given as:

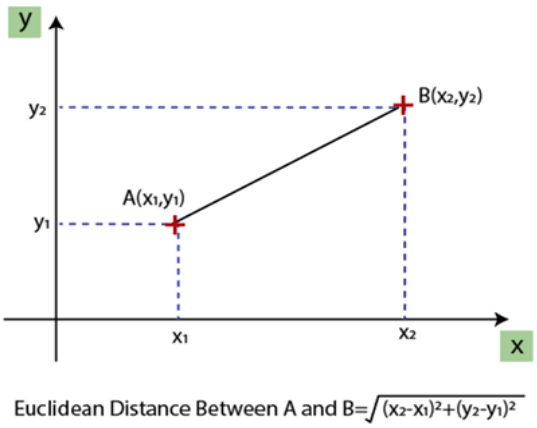

Fig 4: Feature scaling

If we compute any two values from age and salary, then salary values will dominate the age values, and it will produce an incorrect result. So, to remove this issue, we need to perform feature scaling for machine learning.

3.2 Splitting the Dataset

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models. If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set 4 and also with the test dataset. Here, we can define these datasets as:

Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

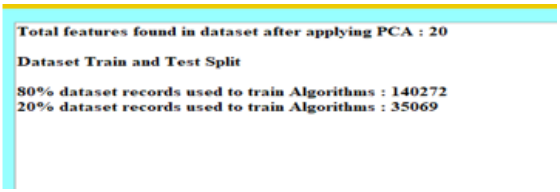

Figure 5: Splitting the dataset

1V. RESULTS & DISCUSSIONS

Results Description

Fig 6: GUI

This figure showcases the graphical user interface (GUI) designed for analyzing instruction detection. It includes interactive elements for data visualization and analysis.

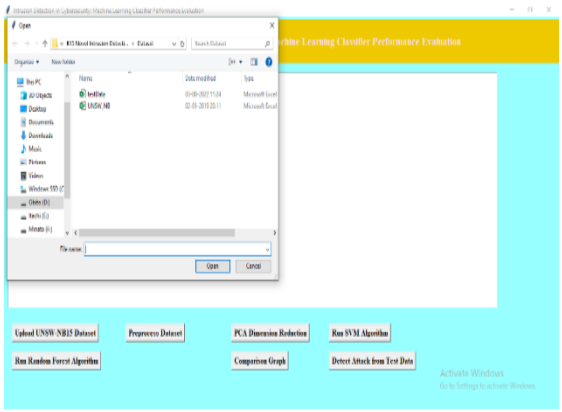

Fig 7: After clicking upload button

Here, the before dataset uploading process is illustrated, indicating how users can import instruction detection data into the GUI for analysis. This step is crucial for accessing the dataset and preparing it for further processing.

Fig 8: After the data is uploaded

	Protocol	Flow Duration	Total Fwd Packets	Total Backward Packets	Fwd Packets Length Total	Bwd Packets Length Total	Fwd Packet Length Max	Fwd Packet Length Min	Fwd Packet Length Mean	Fwd Packet Length Std	...	Active Mean	Active Std	Active Max	Active Min
0	6	9015834	6	0	0	0	0	0	0.00000	0.00000	...	3.009701e+06	0.000000	3009701	3009701
1	0	113498356	52	0	0	0	0	0	0.00000	0.00000	...	1.533333e+01	26.810772	106	5
2	6	56630650	26	24	16880	2680	2797	6	649.23080	962.66284	...	7.457816e+04	107279.790000	209179	20605
3	17	21234	2	2	64	256	32	32	32.00000	0.00000	...	0.000000e+00	0.000000	0	0
4	6	56653775	14	12	84	0	6	6	6.00000	0.00000	...	2.015900e+04	188.881970	28417	27941
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
244301	6	550740	2	1	0	31	0	0	0.00000	0.00000	...	0.000000e+00	0.000000	0	0
244302	6	337766	1	3	31	62	31	31	31.00000	0.00000	...	0.000000e+00	0.000000	0	0
244303	6	228	1	2	0	0	0	0	0.00000	0.00000	...	0.000000e+00	0.000000	0	0
244304	17	47709	2	2	86	194	43	43	43.00000	0.00000	...	0.000000e+00	0.000000	0	0
244305	6	322010	30	33	5902	7444	1073	0	196.73334	353.03930	...	0.000000e+00	0.000000	0	0

Fig 9: Sample dataset

Figure shows the bar plot of output column in that class 0 is very less than the output column 1.

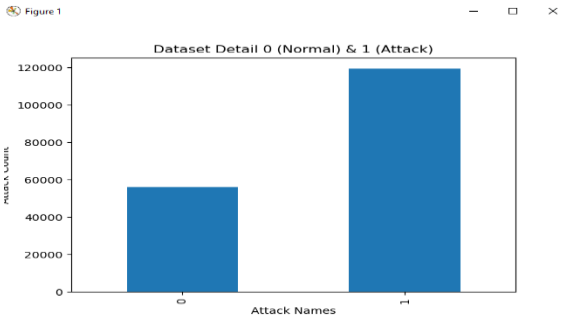

Fig 10: Bar plot of output column

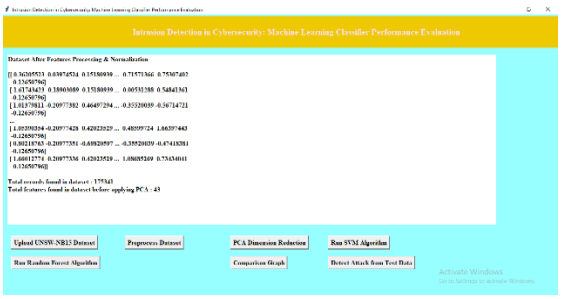

Fig 11: After preprocessing the dataset

Figure shows the data after processing. In the preprocessing label encoding and standard scaling techniques are involved.

Fig 12: After applying PCA

Figure 6 shows the after applying PCA The table shows that the original dataset had a total of 20 features. After applying PCA, the number of features has been reduced to 100%. The text says "100%" but this likely refers to the fact that all 20 original features are captured in the 20 principal components.

The table also shows that the dataset was split into a training set and a test set. The training set contains 80% of the data (140,272 records) and the test set contains 20% of the data (35,069 records). This is a common way to split data for machine learning tasks. The training set is used to train the model, and the test set is

used to evaluate the performance of the model on unseen data.

Fig 13: RFC performance

Figure shows the Random Forest model demonstrates a high level of performance with an accuracy of 91.0%, indicating that it correctly classifies 91% of instances in the dataset.

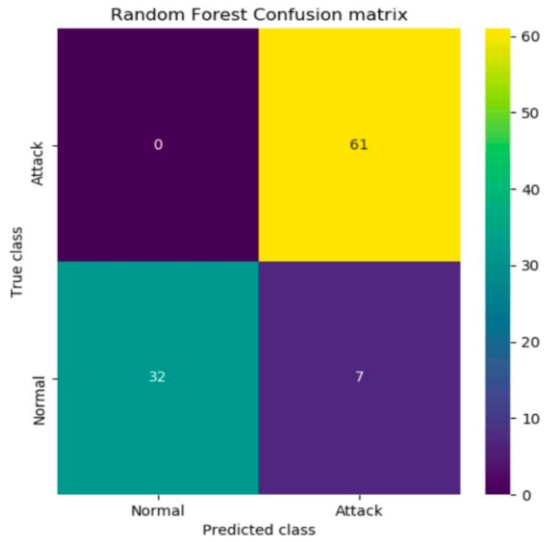

Fig 14: RFC Confusion matrix

Figure shows the

- **True Positives (TP):** The model correctly predicted "Attack" for instances that were actually "Attack." (72)
- **True Negatives (TN):** The model correctly predicted "Normal" for instances that were actually "Normal." (18)
- **False Positives (FP):** The model incorrectly predicted "Attack" for instances that were actually "Normal." (7)
- **False Negatives (FN):** The model incorrectly predicted "Normal" for instances that were actually "Attack." (3)

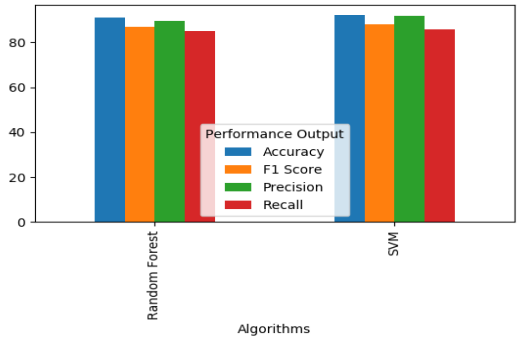

Fig 15: Performance Evaluation

V. CONCLUSION

The evaluation of machine learning classifiers for intrusion detection presents a promising advancement in the field of cybersecurity. Machine learning introduces a dynamic approach to intrusion detection, leveraging large datasets and advanced algorithms to identify patterns and anomalies indicative of malicious behavior. Unlike traditional systems, machine learning-based IDS can learn from data, adapt to new threats, and continuously improve their detection accuracy. This adaptability makes them particularly effective in identifying unknown or zero-day attacks, significantly reducing the risk of undetected intrusions. Moreover, the ability of machine learning models to analyze vast amounts of data in real-time offers a scalable and efficient solution to the growing challenges in cybersecurity. The performance evaluation of these classifiers reveals that while machine learning offers significant advantages, there are also challenges to be addressed. Factors such as model selection, feature engineering, and the quality of training data play crucial roles in the effectiveness of the system. False positives and negatives remain a concern, as well as the potential for adversarial attacks targeting the machine learning models themselves. Nonetheless, the shift towards machine learning in intrusion detection represents a critical step forward, providing a more proactive and robust defense against cyber threats.

The challenge of adversarial attacks, where attackers manipulate inputs to deceive machine learning models, must also be addressed. Developing robust models that can withstand such attacks will be critical in ensuring the reliability of machine learning- based IDS. Furthermore, ongoing research into the ethical implications and privacy concerns associated with

machine learning in cybersecurity will be essential in guiding the development of these systems.

REFERENCES

[1] A. Verma, V. Ranga Machine learning based intrusion detection systems for IoT applications Wirel. Person. Commun., 111 (4) (2020), pp. 2287-2310

[2] A. Thakkar, R. Lohiya A review of the advancement in intrusion detection datasets Procedia Comput. Sci., 167 (2020), pp. 636-645

[3] Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman Survey of intrusion detection systems: techniques, datasets and challenges Cyber Secur., 2 (1) (2019), pp. 1-22

[4] R. Bace, P Mell NIST Special Publication On Intrusion Detection Systems Booz-Allen And Hamilton Inc, Mclean VA (2001)

[5] H. Liu, B. Lang Machine learning and deep learning methods for intrusion detection systems: a survey Appl. Sci., 9 (20) (2019), p. 4396

[6] M.C. Belavagi, B. Muniyal Performance evaluation of supervised machine learning algorithms for intrusion detection Procedia Comput. Sci., 89 (2016), pp. 117-123

[7] K. Kumar, J.S. Batth Network intrusion detection with feature selection techniques using machine-learning algorithms Int. J. Comput. Appl., 150 (12) (2016).

[8] I. Kumar, N. Mohd, C. Bhatt, S.K. Sharma Development of IDS using supervised machine learning Soft computing: Theories and Applications, Springer, Singapore (2020), pp. 565-577

[9] P. Nskh, M.N. Varma, R.R. Naik Principle component analysis based intrusion detection system using support vector machine 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE (2016), pp. 1344-1350

[10] N. Aboueata, S. Alrasbi, A. Erbad, A. Kassler, D. Bhamare Supervised machine learning techniques for efficient network intrusion detection 2019 28th International Conference on Computer Communication and Networks (ICCCN), IEEE (2019)

[11] A. Devarakonda, N. Sharma, P. Saha, S. Ramya Network intrusion detection: a comparative study of four classifiers using the NSL-KDD and KDD'99 datasets Journal of Physics: Conference Series, 2161, IOP Publishing (2022), Article 012043

[12] S. Anita, S.M. Hadi, N.H. Nosrati Network intrusion detection using data dimensions reduction techniques J. Big Data, 10 (1) (2023)