iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** VIII **Month of publication:** August 2024

DOI: https://doi.org/10.22214/ijraset.2024.64096

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VIII Aug 2024- Available at www.ijraset.com

Advanced Threat Recognition: Harnessing Machine Learning for Intrusion Detection

Ambalika Dey[1], Sumit Bhowmik[2]
[1]*Senior Technical Officer, Computer Science Department,* [2]*Faculty, Computer Science Department, NIELIT Agartala*

Abstract: ***Conventional intrusion detection systems (IDS) are frequently inadequate to combat complex cyber threats in the changing cybersecurity landscape. To create a reliable intrusion detection system, this work investigates the use of machine learning, more especially the RandomForestClassifier method. The study highlights that in order to maximize model training, careful data preprocessing—including feature selection, label encoding, and scaling—is essential. It also uses Optuna for hyper-parameter adjustment to improve the classifier's performance. As a scalable and dependable response to contemporary cybersecurity issues, the findings show how effective machine learning is in identifying sophisticated attacks. The outcomes show a high detection accuracy and resilience to complex intrusions, confirming machine learning's potential as a scalable and reliable answer to today's cybersecurity problems.***

Keywords: Machine Learning, Intrusion Detection System, Cybersecurity, RandomForestClassifier, Hyperparameter Tuning

I. INTRODUCTION

The security and integrity of computer networks are always at danger in the current digital era due to more complex cyberattacks. These attacks threaten the security of vital infrastructures and confidential data, rendering conventional intrusion detection techniques inadequate. Advanced, effective solutions that can successfully adapt to and combat growing cyber threats are desperately needed. In cybersecurity, machine learning (ML) has become a powerful tool due to its ability to analyze large amounts of network traffic data and identify patterns and anomalies that could be signs of an intrusion.

The capacity of traditional IDS to identify new or developing threats is hampered by their heavy reliance on predefined rules and signatures. On the other hand, machine learning (ML)-based methods make use of statistical models and algorithms to learn from past data, which increases their ability to identify attack patterns that were previously undetected. Machine learning can detect minute signs of compromise that human analysts overlook by automating the examination of large and intricate information. Furthermore, real-time network traffic processing is made possible by the scalability of machine learning models, which is essential for today's high-speed networks. Integrating machine learning (ML) with intrusion detection systems (IDS) is a noteworthy advancement in improving network security and safeguarding confidential data, given the ongoing evolution of complex and frequent cyber threats. This paper investigates using the robust machine-learning algorithm RandomForestClassifier to create a scalable and dependable intrusion detection system (IDS) that can handle the difficulties presented by contemporary cyber threats.

This work aims to create an advanced threat recognition system for intrusion detection by utilizing machine learning. The main goal is to produce highly accurate IDS using the RandomForestClassifier method. To guarantee that the dataset is best prepared for model training, the methodology includes extensive data pre-processing, such as feature selection, label encoding, and scaling. After the initial model training, hyper-parameter adjustment is made using Optuna, a cutting-edge optimization framework, to improve the classifier's performance. By combining these cutting-edge methods, such as data pre-processing, which includes cleaning the data, dealing with missing values, and converting raw data into a format that is appropriate for modeling. Important pre-processing actions consist of feature selection, which is used to find the most pertinent features that have a substantial correlation with the target variable by reducing the number of dimensions in the dataset; this phase increases the accuracy and efficiency of the model. Label encoding includes transforming categorical data into numerical values to facilitate efficient data processing by machine learning algorithms. Scaling is used to find standardising the data to guarantee that each feature adds the same amount to the model's learning. When it comes to distance-based algorithms, this phase is especially crucial, and hyper-parameter tuning with Optuna is used to find the most advanced hyper-parameter optimization framework, Optuna automates the process of finding the ideal hyper-parameters. Implementing a comprehensive pre-processing step, feature selection, and Optuna-driven hyperparameter tuning led to a notable enhancement in the model's performance, attaining a detection accuracy of over 99% in many classes. This indicates how well the suggested methodology improves intrusion detection systems' performance in practical settings. This research aims to offer a scalable and dependable solution for practical cybersecurity applications, showcasing machine learning's potential to transform intrusion detection and provide a strong defence against constantly changing cyber threats.

©IJRASET: All Rights are Reserved

II. BACKGROUND AND RELATED WORK

An important topic of study in network security is intrusion detection systems, which attract much attention from scientists looking to improve and optimize their technology. A detecting system needs to be steady and efficient in order to be effective. Now, many researchers use machine-learning techniques on the KDD CUP 99 dataset. This data set was used for The Third International Knowledge Discovery and Data Mining Tools Competition, in conjunction with KDD-99, The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model distinguishing between “bad” connections, called intrusions or attacks, and “good” normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment and made available to the public to enhance the identification of hostile network activity and, hence, enhance the capabilities of intrusion detection systems.

In paper [1], the author has presented a survey of the various data mining techniques proposed for enhancing anomaly intrusion detection systems. The author also applied the classification methods for classifying the attacks (intrusions) on the DARPA dataset. The results show that the Random Forest's performance is better than other classifiers. However, the time taken for Random Forest is longer than that for other classifiers. In paper [2], Artificial Intelligence methods are gaining the most attention regarding their ability to learn and evolve, making them more precise and efficient in facing the huge number of unpredictable attacks. Hence, a methodology based on a Genetic Algorithm for detecting Probing, Denial of Service, and Remote user attacks is proposed. The proposed approach aims to gain maximum detection of probing, R2L, and DoS attacks with a minimum false positive rate. Out of the total intrusions in the testing dataset, this approach expects to detect more than 97% of the intrusions. This approach will be very useful for detecting today’s changing attack methodologies. If the rules are updated dynamically with the firewall’s log data, this method will be very effective against new attacks. The paper's authors [3] describe three frameworks for network intrusion detection based on data mining. The author uses the random forests algorithm to detect hybrids, anomalies, and abuse. The author used the random forests technique to create patterns of invasions in order to address the issues with rule-based systems. Rather than requiring manual rule coding, the random forests algorithm may automatically construct patterns by learning over training data. The WEKA environment and the Fortran 77 software are used to implement the suggested ways in a Java program. The experimental results demonstrate that our approaches outperform the best KDD'99 results. The author assessed the implementations over several datasets derived from the KDD'99 datasets. The system may automatically detect intrusions in real time by utilizing the built-in intrusions patterns created during the offline phase of the author's abuse detection framework. The author optimizes the random forests algorithm's settings and applies the feature selection algorithm to increase the system's accuracy. To improve the likelihood of detecting minority intrusions in the framework, the author additionally employs sampling techniques. The author suggested a new method for unsupervised anomaly detection because misuse detection cannot identify fresh invasions.

In the paper [4], the authors presented a Random Forest model for Intrusion Detection Systems (IDS) focusing on improving the intrusion detection performance by reducing the input features. From the context of real-world applications, smaller numbers of features are always advantageous in terms of both data management and processing time. The results indicate that the ability of the RF classification with reduced features (25 features) produces more accurate results than that found from Random Forest classification with all features (41 features). Moreover, the time required to process 25 features with RF is smaller than the processing time of RF with 41 features. The research in intrusion detection and feature selection using the RF approach is still ongoing due to its good performance. The findings of this paper will be very useful for research on feature selection and classification. These findings can also be applied to use RF more meaningfully to maximize the performance rate and minimize the false positive rate. In the paper [5], the author explains the Random Forest Algorithm, the number of trees in the forest, and the results from them are directly related, i.e., the more trees, the more accurate the result. It is important to note that decisionmaking using the gain or gain approach is not the same as creating a random forest. This paper presented an overview of the random forest algorithm, and a survey of various techniques proposed by several researchers was summarized. In the paper [6], the author deals with the Random Forest (RF) algorithm to detect four types of attack: DOS (Denial of Service), probe, U2R (User to Root), and R2L (Remote to Local). Here, the author adopted 10 cross-validations that were applied for classification. Feature selection is applied to the data set to reduce dimensionality and remove redundant and irrelevant features. The author applied symmetrical uncertainty of attributes, which overcomes the problems of information gain. The proposed approach is evaluated using the NSL KDD data set. The author also compared random forest modeling with j48 classier in terms of accuracy, DR (Detection Rate), FAR (False Alarm Rate), and MCC (Matthews Correlation Coefficient). Our experimental result proves that accuracy, DR, and MCC for four types of attacks are increased by our proposed method. For future work, we will apply evolutionary computation as a feature selection measure to further improve the accuracy of the classifier.

III. PROPOSED METHODOLOGY

This research aims to create and assess a sophisticated threat recognition system that uses machine-learning methods to detect intrusions. By developing and optimizing machine learning models—Random Forest classifiers in particular—to detect a variety of network intrusions, including both known and unknown threats, the research seeks to increase intrusion detection accuracy.
An ensemble learning technique called Random Forest is mainly applied to regression and classification problems. During training, it constructs several decision trees and combines them to generate a more reliable and accurate forecast. The fundamental principle is that a collection of decision trees, or weak learners, can combine to become a strong learner.

A. *Random Forest Process*

1) *Bootstrap Sampling:* To generate several subsets, randomly select samples from the training dataset using replacement. We will utilize each subset to develop a separate decision tree.
2) *Feature Selection:* A random subset of the features is considered when dividing a node in each tree rather than all of the characteristics. This guarantees that the trees are diverse.
3) *Construction of Decision Trees:* A bootstrap sample is used for each tree. Trees develop to their maximum potential unpruned.
4) *Voting/Averaging:* In terms of classification, every tree votes for a class, and the class that receives the most votes is the one that is predicted at the end. The final output for regression is the mean of all the trees' predictions.

B. *Random Forest's advantages over other techniques*

1) *Minimises Overfitting:* Random Forest minimizes the possibility of overfitting that single decision trees may experience by averaging several decision trees.
2) *Robustness:* It is less susceptible to dataset noise. Overall performance will be improved even if some trees are incorrect because many others will be correct.
3) *Handles High-Dimensional Data:* Random Forest is appropriate for complex tasks because it can handle many input features without requiring feature selection.
4) *Handles Missing Values:* It can naturally handle missing values by utilizing subsets of the data.
5) *Feature Importance:* It ranks the features' relative importance, facilitating a deeper dataset comprehension.

C. *Equations*

1) *Classification Prediction Function*

The majority vote among all n trees determines the final prediction (y) for an input (x) in classification.

$$y = \text{mode}\{h_1(x), h_2(x), \ldots, h_n(x)\}$$

Where the i^{th} tree's prediction is $h_i(x)$.

2) *Prediction Function (Regression)*

The average of all n trees' predictions is the final prediction (y) in regression:

$$y = \frac{1}{n}\sum_{i=1}^{n} h_i(x)$$

3) *Gini Impurity (for node splitting):*

For each node, the Gini impurity G is computed as follows:

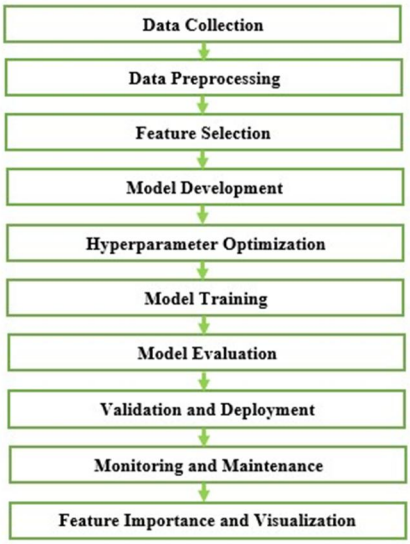
$$G = 1 - \sum_{i=1}^{C} p_i^2$$

Where c is the total number of classes and p_i is the probability of selecting an element from class i.

4) *Information Gain (used in some tree splits)*

$$IG(T, a) = H(T) - \sum_{v \in \text{Values}(a)} \frac{|T_v|}{|T|} \times H(T_v)$$

Where H(T) is the entropy of the set T_1 and T_v is the subset for which attribute a has value v.
In order to determine which network qualities are most suggestive of malicious activity, it will concentrate on finding and sifting the most important features using methods like feature importance analysis and recursive feature elimination (RFE).



The dataset will be ready for reliable model training with the application of efficient data pre-processing techniques, such as handling missing values, encoding categorical variables, and scaling features.

The dataset's structure, including the column names, non-null count, and data types, is outlined in this table.

Table I:
Intrusion Detection Dataset Schema

Sl No.	Column	Non-Null Count	Data Type
0	duration	145585	int64
1	protocol_type	145585	object
2	service	145585	object
3	flag	145585	object
4	src_bytes	145585	int64
5	dst_bytes	145585	int64
6	land	145585	int64
7	wrong_fragment	145585	int64
8	urgent	145585	int64
9	hot	145585	int64
10	num_failed_logins	145585	int64
11	logged_in	145585	int64
12	lnum_compromised	145585	int64
13	lroot_shell	145585	int64
14	lsu_attempted	145585	int64
15	lnum_root	145585	int64
16	lnum_file_creations	145585	int64
17	lnum_shells	145585	int64
18	lnum_access_files	145585	int64

19	lnum_outbound_cmds	145585	int64
20	is_host_login	145585	int64
21	is_guest_login	145585	int64
22	count	145585	int64
23	srv_count	145585	int64
24	serror_rate	145585	float64
25	srv_serror_rate	145585	float64
26	rerror_rate	145585	float64
27	srv_rerror_rate	145585	float64
28	same_srv_rate	145585	float64
29	diff_srv_rate	145585	float64
30	srv_diff_host_rate	145585	float64
31	dst_host_count	145585	int64
32	dst_host_srv_count	145585	int64
33	dst_host_same_srv_rate	145585	float64
34	dst_host_diff_srv_rate	145585	float64
35	dst_host_same_src_port_rate	145585	float64
36	dst_host_srv_diff_host_rate	145585	float64
37	dst_host_serror_rate	145585	float64
38	dst_host_srv_serror_rate	145585	float64
39	dst_host_rerror_rate	145585	float64
40	dst_host_srv_rerror_rate	145585	float64
41	label	145585	object

Here in the dataset, we have 145585 numbers of rows and each row usually corresponds to a single observation, sample, or record. It contains 42 numbers of columns and each column represents a feature or variable in the dataset, including the target variable (label).

This work uses thorough data pre-processing approaches, such as addressing missing values, encoding categorical variables, and scaling features, to guarantee the dataset is ready for dependable model training. Resolving class imbalances is a crucial part of the pre-processing, since it can greatly affect the model's capacity to identify less common forms of incursion. In order to counteract this, fair representation is ensured by creating synthetic instances for minority classes through the use of the Synthetic Minority Over-Sampling Technique (SMOTE). SMOTE improves the Random Forest Classifier's ability to spot imbalances in data by helping it learn from it more successfully. Furthermore, Optuna—a sophisticated framework that automates the search for ideal model parameters—is used for hyperparameter optimisation. In order to optimise the model's performance and guarantee that it can precisely identify different kinds of intrusions, this procedure is essential. A variety of metrics, such as accuracy, confusion matrix, classification report, and ROC-AUC score, are used to assess the model's performance. Additionally, the model's performance is examined on a variety of datasets to guarantee its resilience and generalisability.

In order to make sure that the results are not over fit to a single dataset, the paragraph's last sentence stresses the significance of testing the model on several test datasets. This stage is essential to establishing the model's dependability in practical applications by displaying how well it generalises to fresh, untested data. Feature importance visualization is essential for comprehending their respective roles in identifying various forms of network intrusions. We can determine which elements are most helpful in spotting possible dangers by examining these visualisations. This realisation enables us to focus in on and concentrate on the most significant features, improving the model's detection accuracy. An actual intrusion detection system (IDS) will use the best model, which will have been determined by extensive testing and feature analysis. Evaluating how well it performs in real-world settings where it can assess network data instantly and spot possible dangers as they materialise is the aim. The model's usefulness will be illustrated by this real-world application, which will explain how to incorporate it into current cybersecurity frameworks to offer a strong defence against both current and new threats.

IV. DATA PROCESSING

In this stage, categorical features are transformed using the Encoding approach to create a matrix that includes binary values and category features. Subsequently, we allocate categories inside services. Subsequently, we allocate categories inside services. Currently, Label Encoder is being used to convert the features, turning each category into a number. Next, we divided the dataset label into 23 different attacks. Here, we give each attack type a fictitious name in the following table:

TABLE II
NETWORK EVENT LABEL MAPPING

Label	Network Event
0	Normal
1	Buffer Overflow
2	Load Module
3	Perl
4	Neptune
5	Smurf
6	Guess Password
7	Pod
8	Teardrop
9	Port Sweep
10	Ipsweep
11	Land
12	FTP Write
13	Back
14	IMAP
15	Satan
16	PHF
17	Nmap
18	Multihop
19	Warezmaster
20	Warezclient
21	Spy
22	Rootkit

Following this, features are scaled by dividing them into two groups, X and Y, respectively, consisting of a data frame of features and a set of output variables.

V. FEATURE SELECTION

A critical step in the pre-processing stage of the data is feature selection. In order to improve the model's performance and interpretability, it entails locating and keeping only the most pertinent elements that substantially aid in the prediction of the target variable.

The following phases provide a comprehensive definition of the Recursive Feature Elimination (RFE) procedure that is applied to the dataset:

1) Step 1: First, separate the features and labels.

Take the labels and features out of the dataset. The dataset is used to extract the feature set (X_train) and target labels (Y_train). The column marked "label" contains the target variable, which frequently denotes the result or class.

2) Step 2: Initialize the Estimator

Select the feature-ranking model that RFE uses. A classifier such as RandomForestClassifier is usually chosen as the base estimator. This model will be applied to assess feature importance.

3) Step 3: Apply Recursive Feature Elimination (RFE)

Iteratively identify and remove less significant traits.

- Model Fitting: Every feature is originally fitted to the model.
- Ranking: The model's coefficients or feature importance scores are used to rank the relative relevance of each feature.
- Feature Elimination: In accordance with the rankings, the least significant features are eliminated.
- Repetition: Until a predetermined number of features is reached, steps 1-3 are repeated on the remaining features.

4) Step 4: Complete the Feature Set

Choose the best subset of attributes. The most crucial features are kept when RFE completes its cycles. It is anticipated that these characteristics will have the greatest predictive power and the least amount of redundancy, which will improve model performance.

5) Step 5: Selective Features for Model Training

Utilising the improved feature set, train the model. To make sure the classifier is performance-optimized with the most pertinent data, the model is retrained using only the chosen features.

The RandomForestClassifier is employed alongside Recursive Feature Elimination (RFE) to identify the top ten features from the training set. This approach involves building the model multiple times and recursively removing the least significant features until the optimal set is achieved. Once the top features are selected, StandardScaler is applied to standardize the features, ensuring they have a mean of zero and a variance of one. This step is crucial for improving the model's performance and consistency. To evaluate the effectiveness of the model, the dataset is split, with 30% allocated for training and 70% for validation. This division helps in assessing how well the model generalizes to new, unseen data.

VI. BUILDING MODEL

A high-performing model selection is necessary to obtain a highly accurate classifier that processes real-time data. Using the scikit-learn module in Python, we constructed the intrusion detection model. After isolating the features (X_train) and labels (Y_train), we used a RandomForestClassifier in conjunction with Recursive Feature Elimination (RFE) to determine the top 10 features. In order to ensure zero mean and unit variance, we used StandardScaler to standardise these features.

The RandomForestClassifier is doing incredibly well on both the training and validation sets, as evidenced by these high accuracy numbers. On both the training and validation sets, the model's accuracy is computed. Training Accuracy: 0.9996, or around 99.96 percent Test Accuracy: 0.9983, or almost 99.83 percent. With great precision, the model has learnt to predict the target variable. High accuracy, however, can also indicate overfitting, a condition in which the model performs remarkably well on training data but poorly on fresh, untested data. To make sure the model is resilient, it is crucial to further validate it using methods like cross-validation, looking at confusion matrices, and taking additional metrics (precision, recall, F1 score) into account.

Further, we implemented Optuna for hyperparameter adjustment; the RandomForestClassifier's performance is enhanced. The hyperparameters (n_estimators, max_depth, min_samples_split, and min_samples_leaf) and their corresponding ranges are defined by the objective function. Using these hyperparameters, it trains a RandomForestClassifier and yields the accuracy on the validation set. The goal of a study is to maximise accuracy, or the objective function. The research conducts one hundred trials, experimenting with various hyperparameter values to determine the optimal combination. The best trial's hyper parameters that were discovered throughout the optimisation procedure. Here: Ideal trial: {'max_depth': 32,'min_samples_split': 9,'min_samples_leaf': 1, 'n_estimators': 138}. On the validation set, the final classifier makes predictions, and the accuracy is computed. Maximum Test Precision: 0.9982828097811155. The optimisation procedure revealed that the hyperparameters {'n_estimators': 138,'max_depth': 32,'min_samples_split': 9,'min_samples_leaf': 1} produced the best results on the validation set. Top Test Precision: On the validation set with optimal hyperparameters, the accuracy is roughly 99.83%. This indicates that the RandomForestClassifier keeps a high degree of accuracy after adjusting the hyperparameters, proving that the model is well optimized and performs admirably on the validation data. Given that the optimised model still performs at a comparable level, this further implies that the initial high accuracy was not exclusively the result of overfitting.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

VI. FEATURE IMPORTANCE

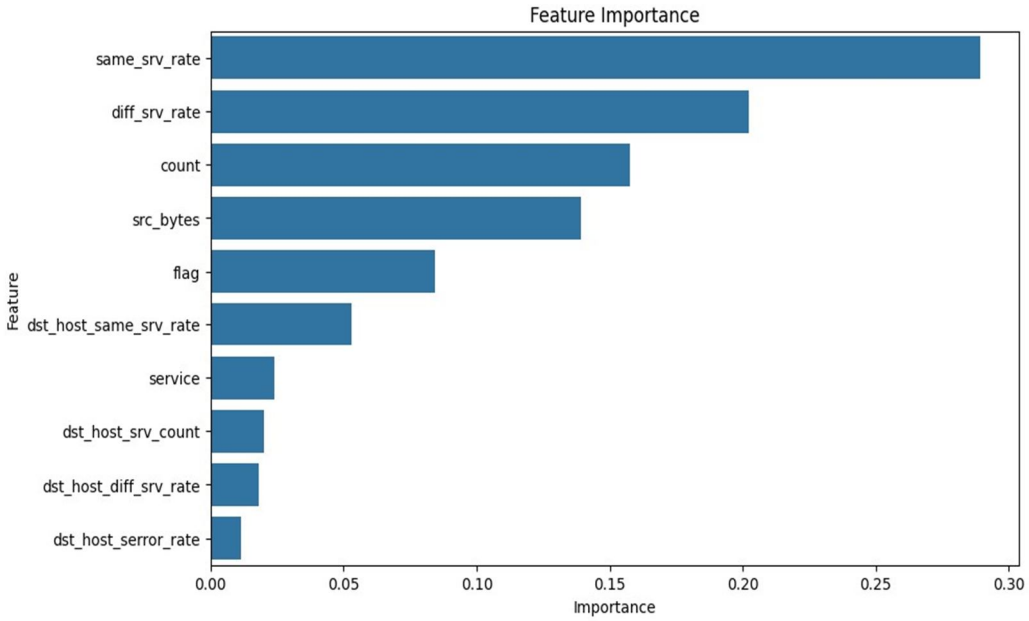
The contribution of each feature to the model's predictions is indicated by the feature significance values. The following summarises the meaning of the output:

1) Feature: The feature's name.

2) Importance: A number that expresses how significant a characteristic is in relation to other features in the model. Greater values indicate that the attribute influences the forecasts more.

Table III
Feature Importance

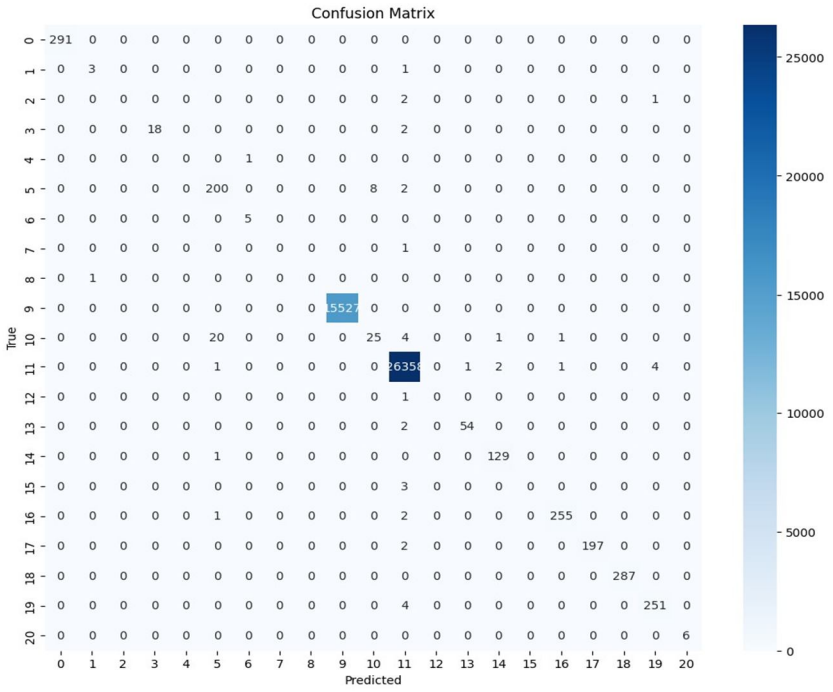
Index	feature	Importance	Percentage Contribution
4	same_srv_rate	0.289317	28.93%
5	diff_srv_rate	0.202333	22.33%
3	count	0.157761	15.77%
2	src_bytes	0.139157	13.92%
1	flag	0.084418	8.44%
7	dst_host_same_srv_rate	0.053143	5.31%
0	service	0.023987	5.31%
6	dst_host_srv_count	0.020161	2.01%
8	dst_host_diff_srv_rate	0.018173	1.81%
9	dst_host_serror_rate	0.011550	1.15%



VII. EXPERIMENTAL RESULT

The experiment was conducted utilising the KDD CUP 99 data set. The KDD CUP 99 dataset has 42 attributes total, with class label making up the final attribute. We experimented with different numbers of Random forest trees. The classification report and confusion matrix offer comprehensive insights into how well the RandomForestClassifier performs in various classifications.

The table used to find the performance of confusion matrix in this classification model. The matrix given below



A. Description of Confusion Matrix Analysis as follow

The dataset's true labels, or actual classes, are represented as rows.
The anticipated classes (the model's output) are shown in columns.
From top-left to bottom-right, the diagonal elements display the number of correctly identified instances for each class.

- Analysis of Class 0

There are 291 samples in total of class 0
It is accurate to anticipate that all 291 samples are Class 0.
For Class 0, the model displays perfect accuracy.

- Analysis of Class 1

There are 15,527 samples in total of class 9
It is accurate to anticipate that all 15527 samples are Class 9.
For Class 9, the model displays perfect accuracy.

- Analysis of Class 11

There are 26,367 samples in total of class 9
With only a few misclassifications, nearly all 26,367 cases are accurately predicted as Class 11. The model has very few faults and is accurate for Class 11.

- Overall Summary

The model performs exceptionally well with larger classes (such as Class 9 and Class 11), where nearly all cases are correctly identified, as the confusion matrix demonstrates.
The model's accuracy declines for smaller classes (such as Class 1), which can result in occasional misclassifications.

B. Explanation of Classification Report

The categorisation report offers important indicators.

1) Precision: The ratio of accurately predicted positive observations to the total number of predicted positives is known as precision. It measures how well the model predicts the positive results.

Formula:

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

Where:

TP: True Positives (Correctly predict the positive observations)

FP: False Positive (incorrectly predict the positive observations)

High precision indicates that the model does not make many false positive predictions

2) *Recall (Sensitivity):* The ratio of accurately predicted positive observations to all observations made during the actual class is known as recall. It gauges how well the model is able to recognise every positive example.

Formula

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

Where:

FP: False Negatives (actual positive observations that were incorrectly classified as negative)

When a model has a high recall, it means that it can catch the majority of positive cases and hardly ever misses a positive class.

3) *F1-Score:* The harmonic mean of Precision and Recall is known as the F1-Score. It offers a fair assessment that takes into consideration both false positives and false negatives.

Formula:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

When you need to strike a compromise between precision and recall or when the class distribution is unbalanced, the F1-Score comes in handy.

c) Support

The amount of real instances in each class is referred to as support. It shows the total number of samples that are a part of the class as a whole. Understanding the number of observations for each class is important for assessing model performance measures, and support aids in this process. For example, if there are relatively few instances, a class with poor support could have an F1-Score that is misleading.

TABLE IIIV
CLASSIFICATION REPORT

Class	Precision	Recall	F1-Score	Support
0	1.00	1.00	1.00	291
1	0.67	0.50	0.57	4
2	0.00	0.00	0.00	3
3	1.00	0.90	0.95	20
4	1.00	1.00	1.00	1
5	0.89	0.97	0.93	210
6	1.00	0.80	0.89	5
7	0.00	0.00	0.00	1
8	0.00	0.00	0.00	1
9	1.00	1.00	1.00	15527
10	0.74	0.39	0.51	51
11	1.00	1.00	1.00	26367
12	0.00	0.00	0.00	1
13	0.98	0.98	0.98	56
14	0.99	0.98	0.99	130

15	0.99	0.98	0.99	130
16	0.00	0.00	0.00	3
17	0.99	0.99	0.99	258
18	1.00	0.99	0.99	199
20	1.00	1.00	1.00	287
21	0.98	0.98	0.98	255
22	0.67	1.00	0.80	6
accuracy			1.00	43676
macro average	0.71	0.69	0.69	43676
weighted average	1.00	1.00	1.00	43676

d) Observation

The resilience of the model in detecting these categories is demonstrated by its good accuracy in most classes, particularly in classes 0, 9, 11, 17, 18, and 20. Nonetheless, some classes—2, 7, 8, 12, and 16—show low recall and precision at zero, indicating difficulties in identifying these particular kinds of intrusions. In spite of these problems, the model performs well overall, handling most classes with little error, according to the high weighted average metrics.

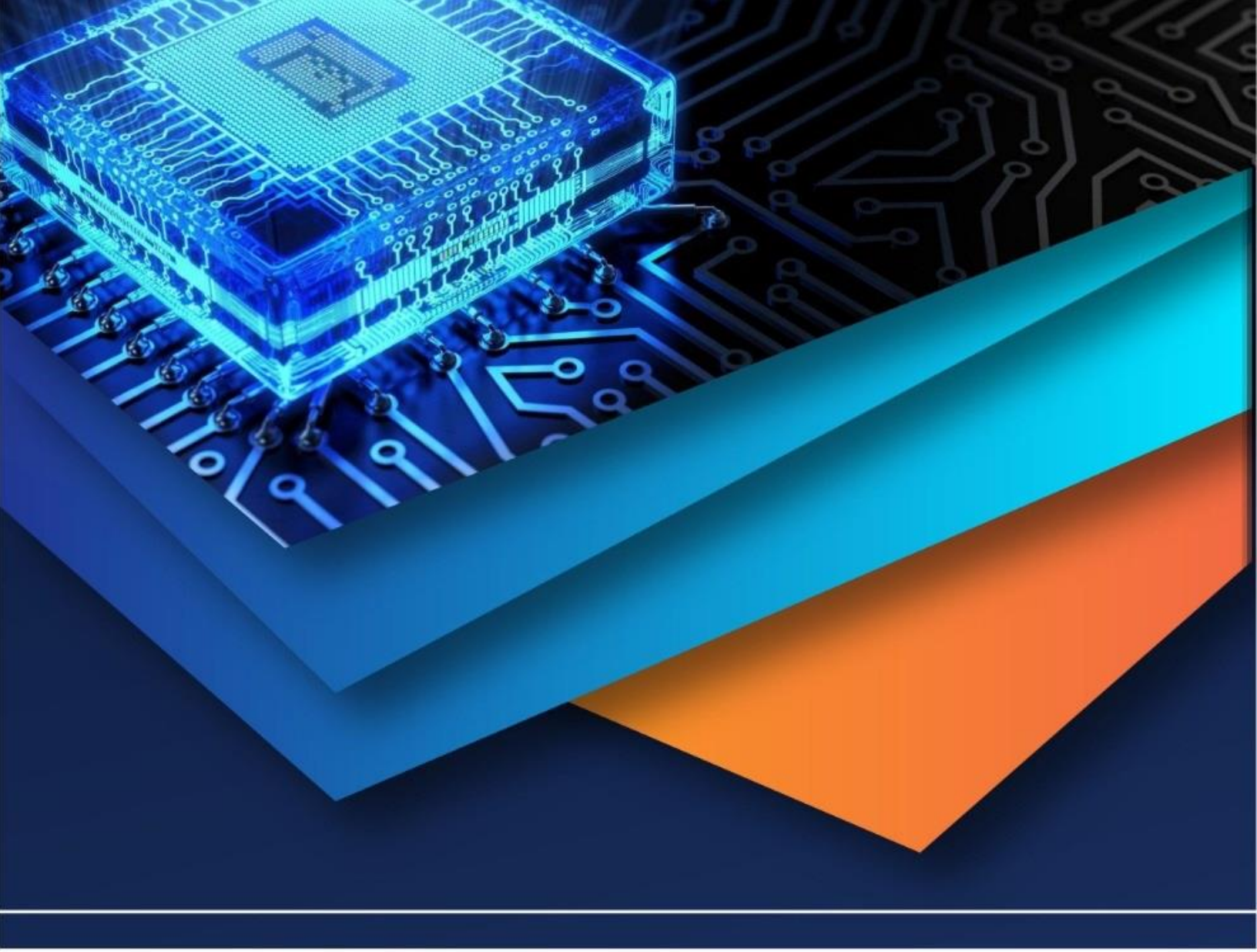
VIII. CONCLUSION

In order to forecast network intrusion detection with a high degree of accuracy, we successfully constructed and optimised a Random Forest classifier for this project. Achieving a stunning 99.83% test accuracy required careful feature selection, careful data preparation, and hyperparameter optimisation with Optuna. This high degree of accuracy highlights the model's stability and dependability in terms of identifying network intrusions with a small margin of error. This project's success depends on figuring out the essential elements that have a big impact on the model's decision-making process. We made sure the model was reliable and effective at processing data and generating predictions by identifying and limiting its features to those that had the greatest influence. In cybersecurity applications, where accuracy and speed are critical, this phase is essential. The model's performance was further improved by the Optuna-powered hyperparameter optimisation method, which adjusted the model's parameters to produce the optimum outcomes. By combining these approaches, we were able to create a model that performs well on a variety of datasets and is both accurate and generalizable. Using sophisticated evaluation criteria, including as accuracy, precision, recall, and F1-score, to gauge the model's performance was another crucial component of this endeavour. With the use of these measurements, the model's advantages and disadvantages could be thoroughly understood, facilitating ongoing development and modification. The model's capacity to accurately detect intrusions while the high scores demonstrate minimising false positives and false negatives obtained across several criteria. The model functioned well across all classes, including those that were under-represented in the dataset, thanks in large part to the application of SMOTE to address class imbalances. In conclusion, this study demonstrates how machine learning may be used to improve network security. Through the use of feature selection, hyperparameter tuning, and sophisticated evaluation procedures, we have optimised the Random Forest classifier and produced a highly accurate and dependable model that can identify a variety of network intrusions. Organisations can use the project's findings to protect their networks from potential threats by applying them to real-world cybersecurity scenarios. It is impossible to overestimate the significance of having resilient and flexible intrusion detection systems given the ongoing evolution of cyber threats. This project demonstrates how machine learning can be essential to reaching that objective.

REFERENCES

[1] Phyu Thi Htun, Kyaw Thet Khaing, “Anomaly Intrusion Detection System using Random Forests and k-Nearest Neighbor”, International Journal of P2P Network Trends and Technology (IJPTT) - Volume 3 Issue 1 January to February 2013
[2] Swati Paliwal, Ravindra Gupta, “Denial-of-Service,Probing &Remote to User (R2L) AttackDetection using Genetic Algorithm” , International Journal of Computer Applications (0975 – 8887) Volume 60– No.19, December 2012
[3] Jiong Zhang, Mohammad Zulkernine, and Anwar Haque, “Random-Forests-Based Network Intrusion Detection Systems”, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. 38, NO.5, SEPTEMBER 2008

[4] Md. Al Mehedi Hasan, Mohammed Nasser, Shamim Ahmad, Khademul Islam Molla, "Feature Selection for Intrusion DetectionUsing Random Forest", Journal of Information Security, 2016, 7, 129-140
[5] Kritika Singh, Bharti Nagpal, "Random Forest Algorithm in Intrusion Detection System: A Survey", International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2018 IJSRCSEIT | ISSN: 2456-3307
[6] Nabila Farnaaz, M. A. Jabbar, "Random Forest Modeling for Network Intrusion Detection System", Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016)
[7] Abdulla Amin Aburomman, Mamun Bin Ibne Reaz, A survey of intrusion detection systems based on ensemble and hybrid classifiers, Computers & Security, Volume 65, 2017, Pages 135-152, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2016.11.004.
[8] Max Landauer, Sebastian Onder, Florian Skopik, Markus Wurzenberger, Deep learning for anomaly detection in log data: A survey, Machine Learning with Applications, Volume 12, 2023, 100470, ISSN 2666-8270, https://doi.org/10.1016/j.mlwa.2023.100470.
[9] A. M. S. Ngo Bibinbe, M. F. Mbouopda, G. R. Mbiadou Saleu and E. Mephu Nguifo, "A survey on unsupervised learning algorithms for detecting abnormal points in streaming data," 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 2022, pp. 1-8, doi: 10.1109/IJCNN55064.2022.9892195.
[10] Ranjeethapriya K, Susila N, Granty Regina Elwin, Balakrishnan S, "Raspberry Pi Based Intrusion Detection System", International Journal of Pure and Applied Mathematics, Volume 119, No. 12, 2018, pp.1197-1205.
[11] S. Balakrishnan, B. Persis Urbana Ivy and S. Sudhakar Ilango, "A Novel and Secured Intrusion Detection System for Wireless Sensor Networks Using Identity Based Online/Offline Signature", ARPN Journal of Engineering and Applied Sciences. November 2018, Vol. 13 No. 21, pp. 8544-8547.
[12] J.P.Ananth, S.Balakrishnan, S.P.Premnath, (2018). "Logo Based Pattern Matching Algorithm for Intrusion Detection System in Wireless Sensor Network", International Journal of Pure and Applied Mathematics, Volume 119, No. 12, 2018, pp. 753-762.
[13] Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM Comput. Surv. 41 (3), 1–58. http://dx.doi.org/10.1145/1541880.1541882.
[14] Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C. and Samatova, N.F. (2015), Anomaly detection in dynamic networks: a survey. WIREs Comput Stat, 7: 223-247. https://doi.org/10.1002/wics.1347
[15] Patcha A, Park J-M. An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput Netw. 2007;51(12):3448– 70.
[16] Oswal, S., Shinde, S., Vijayalakshmi, M. (2023). A Survey of Statistical, Machine Learning, and Deep Learning-Based Anomaly Detection Techniques for Time Series. In: Garg, D., Narayana, V.A., Suganthan, P.N., Anguera, J., Koppula, V.K., Gupta, S.K. (eds) Advanced Computing. IACC 2022. Communications in Computer and Information Science, vol 1782. Springer, Cham. https://doi.org/10.1007/978-3-031-35644-5_17
[17] Heard, Nicholas A., David J. Weston, Kiriaki Platanioti, and David J. Hand. "BAYESIAN ANOMALY DETECTION METHODS FOR SOCIAL NETWORKS." The Annals of Applied Statistics 4, no. 2 (2010): 645–62. http://www.jstor.org/stable/29765524.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)