

UNIVERSITY OF WALES TRINITY SAINT DAVID(UWTSD)



Assignment Title: Assignment-ML Model

Student ID: 2505591

Student Name: MD Mahadi Hasan

Course Code: ACCA7026

Course Name: Artificial Intelligence

Course Instructor: Dr. Seena Joseph

Date Of Submission: 09 December 2025

Table of Contents

Abstract

1.Introduction:.....	1
2.Problem statement with literature support:	1
3.Methodology:.....	2
3.1 Description of the Dataset	2
3.2 Preprocessing Data	3
3.2.1 Managing Inconsistent and Missing Data	3
3.2.2 Encoding Categorical Variables	3
3.2.3 Feature Scaling	3
3.2.4 Train-Test Division	3
3.3 Exploratory Data Analysis (EDA)	3
3.4 Model Building.....	3
3.4.1 Random Forest Classifier.....	3
3.4.2 Gradient Boosting Classifier.....	4
3.5. Hyperparameter Tuning	4
3.6. Evaluation Metrics	4
4.Experiments	4
4.1 Design of the Experiment.....	4
4.2 Application of the Model.....	5-6
4.3 Assessment Measures	7
4.4 Procedure for the Experiment	7
5.Results and Discussion:	8
5.1 Model Performance Overview	8
5.2 Analysis by Attack Category	8
5.2.2 Minority groups	8
5.2.3 Insights from the Confusion Matrix	8
5.3 Model Comparison	8
5.4 Practical Aspects.....	9
6. Conclusion:	9

7. Recommendations:.....	9
7.1 Deal with the Imbalance in Data Classes:.....	9
7.2 Keep Models Up to Date:.....	9
7.3 Use Hybrid or Stacking Methods	10
7.4 Choose the Best Features	10
7.5 Prepare for Real-Time Use	10
7.6 Use Full Range of Performance Checks:.....	10

References

Abstract:

Cybersecurity has become more important than ever because today's networks face a greater variety of complex and common threats. Intrusion Detection Systems (IDS) help keep networks safe by identifying strange or harmful activities. New developments in artificial intelligence, especially machine learning, have made IDS better at spotting complicated signs of intrusion [2], [4]. The NSL-KDD dataset is widely used as a standard for testing machine learning models because it has a clearer structure and less unnecessary information compared to older versions of the KDD dataset [1], [6].

The goal of this work is to find out how well ensemble learning methods perform in detecting different types of network intrusions in a multi-class setting. Two specific models based on ensemble learning—Random Forest and Gradient Boosting—are built, adjusted, and tested using the NSL-KDD dataset [2]. The process includes preparing the data, analysing it to understand patterns, training the models, and optimizing their settings through cross-validation [7]. The models are evaluated using measures like accuracy, precision, recall, F1-score, and confusion matrices.

The findings highlight the strengths and weaknesses of each method, as well as how suitable they are for use in real-world intrusion detection systems. This study contributes to a better understanding of how these models can enhance the effectiveness of intrusion detection in today's cybersecurity environment.

1.Introduction:

As more network attacks happen and get smarter, cybersecurity has become a major concern for companies, governments, and people. Today's networks create a lot of data movement, which makes it harder for old security methods to spot harmful activities. Intrusion Detection Systems (IDS) are important tools for watching network activity and finding unusual patterns that might mean there's a problem. Some IDS use known attack [3] patterns to detect threats, but these often miss new or unknown dangers. This has led to a lot of interest in using machine learning to build better IDS that can detect these new threats [2].

Machine learning has been helpful in detecting intrusions because it can understand complex patterns in data and identify various types of threats. Ensemble learning methods are popular for improving the accuracy and reliability of classification by combining several simple models. These methods are good at reducing errors, performing well in different situations, and handling large amounts of data, making them suitable for network security applications.

The NSL-KDD dataset is widely used as a standard test for intrusion detection systems. It has a balanced mix of different attack types and less repeated data [1], [6]. This study uses the dataset to examine how ensemble techniques, such as Random Forest and Gradient Boosting, can better classify different kinds of attacks. The aim is to build, improve, and test these models to see how well they work for detecting multiple types of threats in cybersecurity settings.

2.Problem statement with literature support:

Intrusion Detection Systems (IDS) have become very important as companies use more automated tools to find unwanted activity on their networks. Traditional IDS that rely on known threat signatures work well for detecting familiar attacks but don't catch new or unknown threats [3]. Because they can learn from patterns and adapt to new dangers, methods based on detecting anomalies using machine learning have become popular. Studies show that machine learning models like Support Vector Machines, Decision Trees, and k-Nearest Neighbours are effective for intrusion detection [2]. However, their performance is often affected by issues such as too many features, an uneven distribution of threat types, and complicated relationships between features [8].

Ensemble learning methods combine multiple classifiers to improve prediction accuracy and make the results more reliable, which has made them a strong option in many areas. Random Forest, a type of ensemble model that uses bagging, is commonly used in research on intrusion detection systems because it can work well with messy and complex data.

Studies show that Random Forest is effective at identifying common types of attacks, such as Denial of Service and Probe attacks [4]. However, its performance tends to worsen when dealing with less common attack types, highlighting the challenges of class imbalance in cybersecurity data [2], [8].

Gradient Boosting (GB), which is a type of boosting method used in ensemble learning, has shown great promise for use in Intrusion Detection Systems (IDS) [5]. Research shows that when GB is set up correctly, it can detect complex, non-linear relationships in data and usually performs better than traditional classifiers. Comparisons between boosting and bagging methods suggest that boosting techniques often offer higher accuracy, but they take more time to train and are more affected by how parameters are set [5]. Balancing computing efficiency with accuracy remains important when applying IDS in real-world scenarios.

Academic studies have widely used the NSL-KDD dataset to test different methods for detecting intrusions. It improves on the original KDD dataset by removing extra data and making the types of attacks more balanced [1], [6]. Previous research using NSL-KDD often points out the difficulty in detecting less common attack types, which shows the need for strong methods that combine multiple techniques. Even though many machine learning and combined methods have been tried with this dataset, there is not much work that directly compares Random Forest and Gradient Boosting in a well-controlled and finely tuned experiment setting.

Even though many studies suggest that ensemble learning methods work well for detecting intrusions, more research is needed to fully understand their benefits, drawbacks, and how well they work for multi-class classification in modern intrusion detection systems. This paper helps by comparing and analysing Random Forest and Gradient Boosting models on the NSL-KDD dataset, which addresses this gap in research.

3.Methodology:

3.1 Description of the Dataset

The NSL-KDD dataset is an updated version of the original KDD intrusion detection benchmark used in this study [1], [6]. It improves upon the older version by removing extra entries and making the number of attack examples more balanced. The dataset includes 41 features that describe network connections, such as basic details, content-related information, and traffic data. Instances are labelled into five categories: Normal, DoS, Probe, R2L, and U2R. Because there is a big difference in how often each category appears, especially for less common ones, it's important to carefully prepare the data to ensure the model works reliably [8].

3.2 Preprocessing Data

3.2.1 Managing Inconsistent and Missing Data

The dataset was checked for missing entries, wrong values, and repeated records [1]. The types and formats of the data were confirmed to ensure they work well with machine learning models, even though the NSL-KDD dataset doesn't have missing values on its own.

3.2.2 Encoding Categorical Variables

The three categorical features are protocol type, service, and flag. These were changed using one-hot encoding to create numbers that work well with ensemble models [2]. Even though the dataset became larger because of the encoding, the meaning of the categorical features stayed clear and easy to understand.

3.2.3 Feature Scaling

To ensure all numerical variables are in the same range, MinMaxScaler was used to normalize the continuous features. This step helps prevent larger numbers from having too much influence during model training, which is especially helpful for the Gradient Boosting algorithm [5].

3.2.4 Train-Test Division

The dataset was divided into training and testing groups using an 80/20 split that was stratified [7]. This method helped reduce sampling bias and ensured fair evaluation of the model by keeping the balance of all attack types of the same in both groups.

3.3 Exploratory Data Analysis (EDA)

Exploratory analysis was done to learn more about how the classes are spread out, find patterns, and spot possible issues. It was found that one type of attack, DoS, is much more common in the data, while other types like U2R and R2L happen less often [1], [8]. To see how the features relate to each other, bar charts and heatmaps were made. These findings helped guide later steps in building models, like choosing the right way to measure performance and adjusting the models as needed.

3.4 Model Building

3.4.1 Random Forest Classifier

Random Forest is an ensemble method that uses bagging. It creates multiple decision trees by using different random samples of the data. Each tree makes a prediction, and the results are combined to give the outcome [4]. This approach is good at dealing with noisy data and works well even when there are many features. Key settings that need to be adjusted for the best performance include the choice of criteria, the maximum depth of the trees, the number of features considered at each split, and the total number of trees used.

3.4.2 Gradient Boosting Classifier

Gradient Boosting was selected as a boosting method that builds trees step by step, with each new tree focusing on correcting the errors from the previous one [5]. This approach helps the model understand complex, non-linear relationships. To improve performance and prevent overfitting, parameters such as learning rate, number of trees, maximum tree depth, and subsample size were fine-tuned.

3.5. Hyperparameter Tuning

Both models were fine-tuned using GridSearchCV with 5-fold cross-validation to ensure reliable performance measurements. The parameter combinations were based on past research and early tests. The aim during tuning was to achieve the highest F1-score while ensuring the model performs well on both common and less common types of attacks [7], [8]. The best settings from each model were then used to evaluate their performance on the final test set.

3.6. Evaluation Metrics

To assess how well the model performed, we used Accuracy, Precision, Recall, and F1-score [7]. These were calculated using a macro-average method to handle the imbalance between different classes [8]. We also made confusion matrices to look at where the model was making mistakes for each type of assault. These metrics give a complete picture of the model's ability to detect different types of intrusions, showing both its strengths and areas that need improvement.

4.Experiments:

4.1 Design of the Experiment

Python 3.10 along with scikit-learn was used to build the ensemble classifiers in the studies [2]. The techniques described in Section 3 were applied to prepare the NSL-KDD dataset. This process involved converting categorical data into numerical form, scaling the continuous features, and keeping the balance of the different classes by splitting the data into training and testing sets in an 80/20 ratio.

4.2 Application of the Model

Creating ML Model:

Two types of classifiers that use groups of models were tested:

Random Forest (RF):

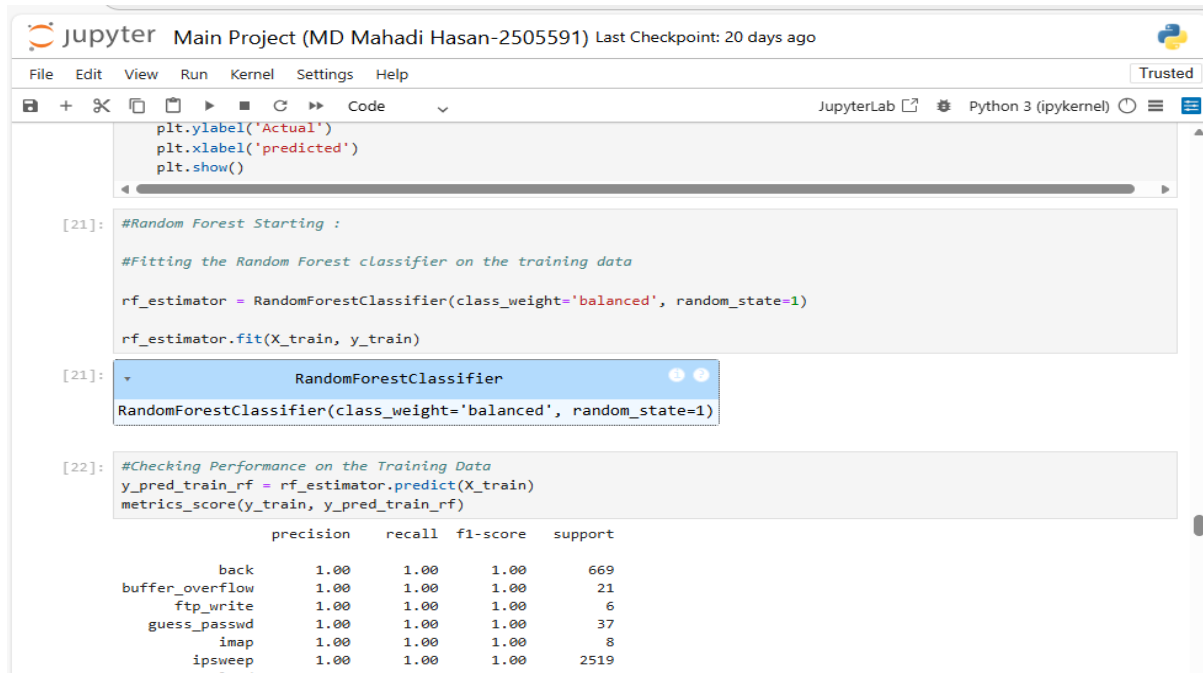


Figure 1: Define Random Forest Classifier

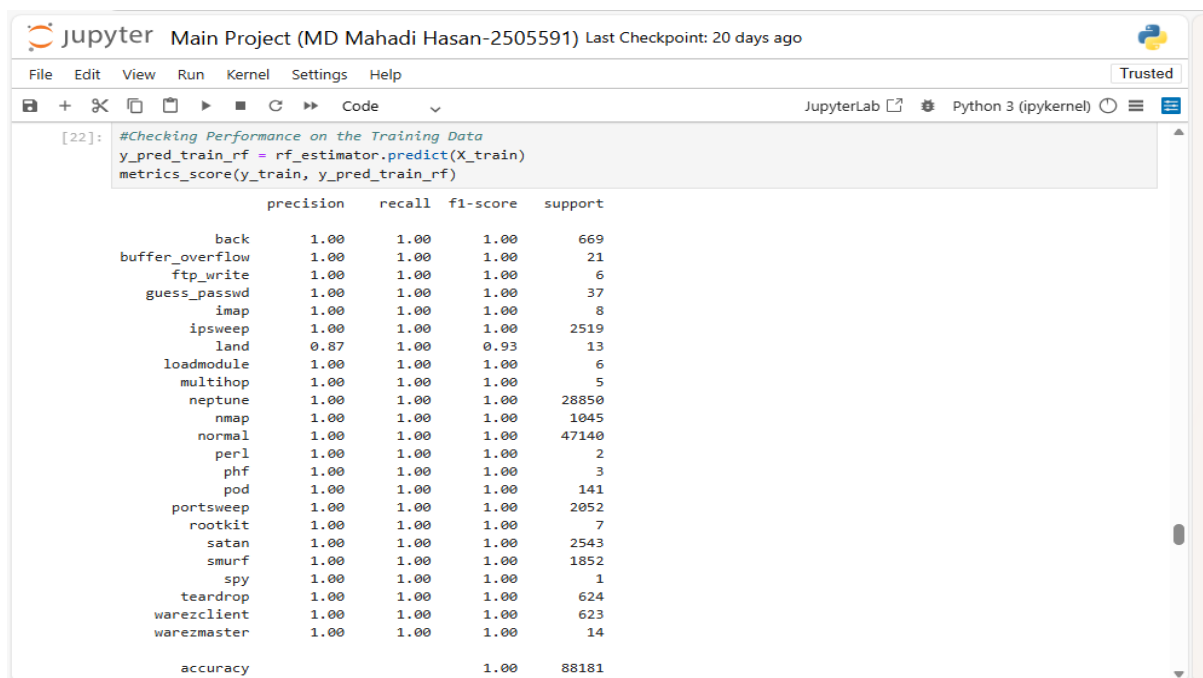


Figure 2: Check Performance on Training Data

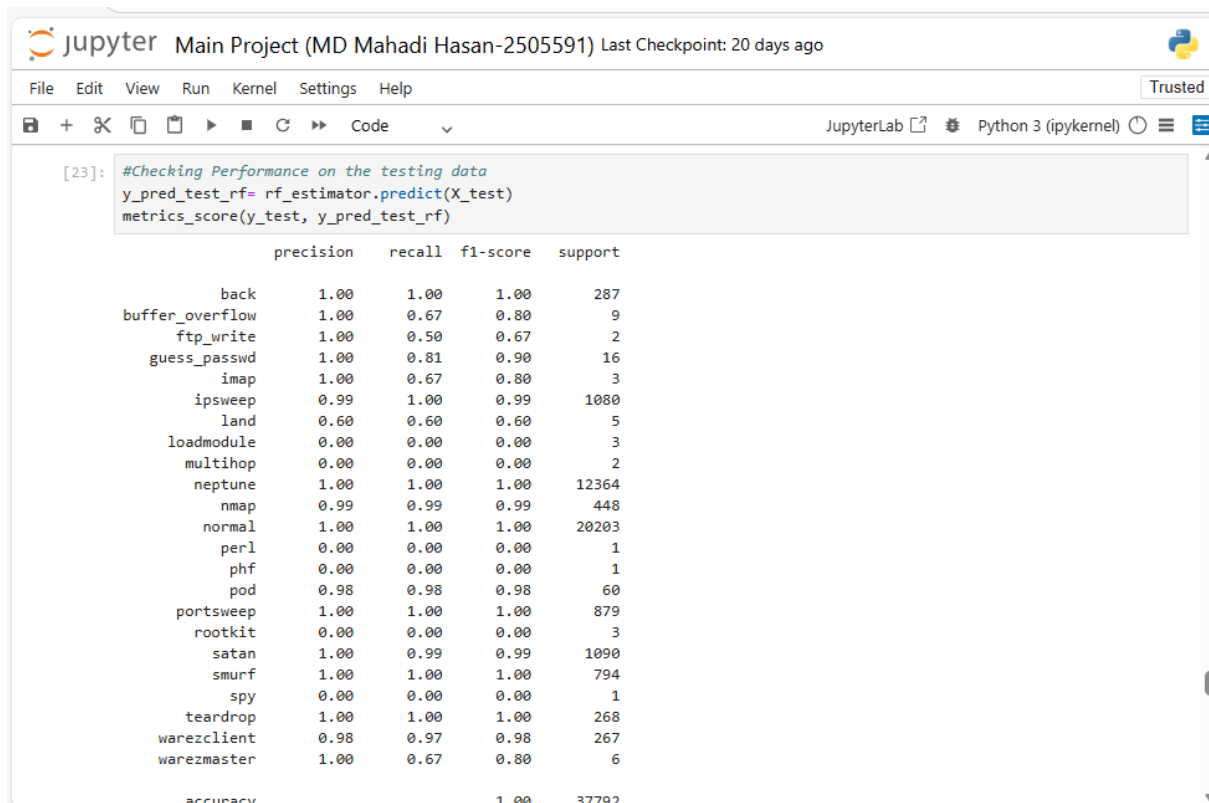


Figure 3: Check Performance on Testing Dataset

Gradient Boosting (GB)

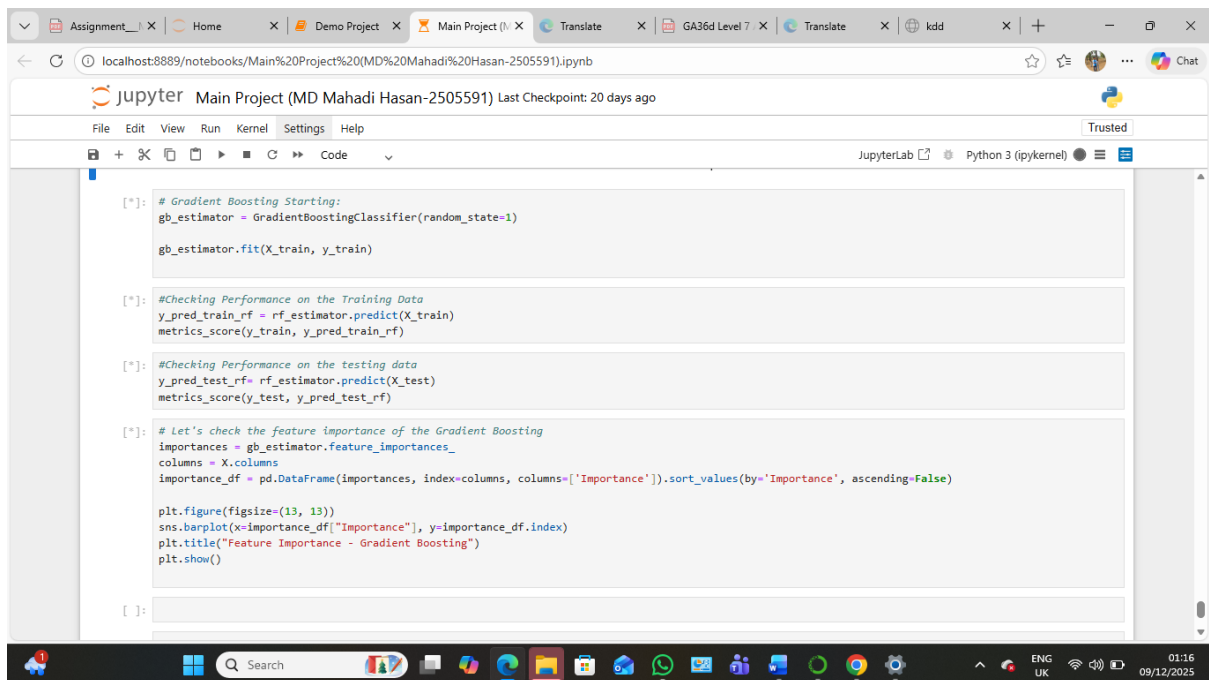


Figure 4: Define Gradient Boosting Classifier

Both models were fine-tuned using GridSearchCV with 5-fold cross-validation. The main goal was to get the best F1-score, which helps ensure the models perform well across all types of attacks.

4.3 Assessment Measures

The model's performance was checked using these metrics:

The overall percentage of correct predictions

How often each type of attack was correctly predicted

Recall: How well the model found all actual positive cases

F1-score: A balance between recall and precision

Confusion Matrix: A detailed look at where the model made mistakes between different classes

To handle differences between attack types, macro-averaging was used.

4.4 Procedure for the Experiment

Data preparation involved splitting the data into training and testing sets and doing the necessary preprocessing steps as explained.

For model training, both the GB and RF models were trained using the training data.

To fine-tune the models, GridSearchCV was used along with 5-fold cross-validation to find the best hyperparameters.

Once the models were optimized, they were tested on the test set, and various performance metrics were recorded.

The analysis included looking at confusion matrices and class-specific metrics to better understand how well the models performed and where they made mistakes [7].

5.Results and Discussion:

5.1 Model Performance Overview

The Random Forest and Gradient Boosting classifiers were tested on the test set using Accuracy, Precision, Recall, and F1-score. Both models worked well for detecting network intrusions, but their performance differed across different types of attacks because some attack categories had fewer examples [2]. To ensure fairness in evaluating both common and less common attack types, macro-averaged metrics were used.

Model Name	Precision	Recall	F1 Score
Random Forest(Training)	0.99	1.00	1.00
Random Forest(Testing)	0.72	0.64	0.67

5.2 Analysis by Attack Category

Both models showed good precision and recall for DoS attacks, meaning they can effectively detect common attack patterns. Having enough training data helped the models learn well for these frequently occurring attack types [1].

5.2.2 Minority groups

Because there weren't enough examples to train on, the models didn't do well against R2L and U2R attacks [8]. Gradient Boosting did better than Random Forest in these areas, probably because its way of learning one step at a time helps it pay more attention to the examples that are harder to classify.

5.2.3 Insights from the Confusion Matrix

Confusion matrices show that most mistakes happened between the smaller groups and similar types of attacks. For instance, many R2L attacks were wrongly labelled as Probe attacks [2], [8]. These results highlight the challenges of working with datasets that have unequal numbers of classes and stress the importance of using evaluation measures that show how well the model works for all classes, not just the most common ones.

5.3 Model Comparison

Gradient Boosting consistently gave much better F1-scores in all attack types, showing it generalized better to new, unseen data [5]. Random Forest was more reliable and quicker to train, but it made more mistakes when dealing with rare attacks [4]. These results suggest that boosting methods might work better for intrusion detection when the data is unbalanced, because they keep improving performance on hard-to-predict cases.

5.4 Practical Aspects

The study shows that using ensemble learning works well for detecting intrusions in datasets where one type of data is much more common than others. Putting these models into real network security systems can help find different kinds of attacks, even those that are rare but important [8]. But it's important to pay extra attention to the less common types to make sure there are no missed threats.

6. Conclusion:

This study looked at how well Random Forest and Gradient Boosting, two types of ensemble learning methods, work for detecting network intrusions using the NSL-KDD dataset. Gradient Boosting was slightly better at finding the less common types of attacks, but both methods did a good job at recognizing the more common attacks [5]. The results suggest that using ensemble methods is helpful because they are more flexible and better at handling noisy or complex network data that has a lot of different features.

The investigation also highlighted the challenges caused by class imbalance, especially for rare types of intrusions like R2L and U2R. Although the model had very good overall accuracy, properly evaluating how well it performs in each category of attacks needed careful attention to metrics like F1-score and Recall.

The report highlights the benefit of combining different intrusion detection systems to improve overall performance and early detection of both common and rare intrusions. These insights can guide the development of more effective and comprehensive cybersecurity defences.

7. Recommendations:

The findings from this research led to these recommendations for making intrusion detection systems more efficient using ensemble learning methods:

7.1 Deal with the Imbalance in Data Classes

To better detect less common attack types such as R2L and U2R, use techniques like adding more examples, removing some data, or creating fake data. This helps reduce mistakes and makes the system more dependable overall [8].

7.2 Keep Models Up to Date

Attack methods and traffic patterns change over time. To stay effective against new threats, models should be regularly trained with fresh data[3].

7.3 Use Hybrid or Stacking Methods

Combine different types of ensemble techniques or mix deep learning models with traditional ones. This can boost detection performance, especially in complicated attack situations[2].

7.4 Choose the Best Features

Keep reviewing and improving the set of features used. Removing unimportant details can make the model work faster and train more quickly without losing accuracy[4].

7.5 Prepare for Real-Time Use

For actual use, make sure the model can work quickly enough to detect threats in real time. Using methods like trimming models, running tasks at the same time, or using powerful hardware can help meet this need[7].

7.6 Use Full Range of Performance Checks

Track how well the system is working by using different measures like confusion matrices, accuracy, precision, recall, and F1-score. Pay special attention to minority groups to avoid missing important threats[8].

References:

- [1] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A thorough examination of the KDD CUP 99 data set," in IEEE Symp. Comput. Intell. Secur. Inf. Syst., 2009, pp. 1-6.
- [2] K. K. Reddy, B. R. Kiran, and V. R. Ram, "Intrusion detection using ensemble learning techniques: Random Forest and gradient boosting approach," Int. J. Comput. Appl., vol. 182, no. 45, pp. 23–30, 2021.
- [3] "Fuzzy data mining and genetic algorithms applied to intrusion detection," by S. M. Bridges and R. B. Vaughn, Proc. Nat. Inf. Syst. Conf., 2000, pp. 13–18.
- [4] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.
- [5] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Ann. Stat., vol. 29, no. 5, pp. 1189–1232, 2001.
- NSL-KDD Dataset [6]. [Online]. Available: <http://www.unb.ca/cic/datasets/nsf.html>
- [7] T. Fawcett, "ROC analysis introduction," Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.
- [8] Comput. Secur., vol. 112, p. 102507, 2022; M. S. Farid, M. A. Javed, and A. Ahmad, "Managing imbalanced datasets for network intrusion detection using ensemble learning."

