iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 12 **Issue:** XII **Month of publication:** December 2024

**DOI:** https://doi.org/10.22214/ijraset.2024.65982

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 XII Dec 2024- Available at www.ijraset.com*

# Improving Cybersecurity: An In-depth Examination of Machine Learning Methods for Detecting Network Intrusions

Suvarna Rajappa[1], Lohith S Y[2], Swetha J[3]
[1]*Lecturer, Department of Computer Science & Engineering, Government Polytechnic Sorab, Karnataka, India*
[2]*Lecturer, Department of Computer Science & Engineering, Government Polytechnic Sorab, Karnataka, India*
[3]*Lecturer, Department of Electronics & Communication, S J (Government) Evening Polytechnic Bangalore, Karnataka, India*

***Abstract:*** ***Maintaining system integrity and protecting sensitive data is critical in today's digital age, underlining the need for strong network intrusion detection in cybersecurity. This paper offers a sophisticated Network Intrusion Detection System (NIDS) built with the NSL-KDD dataset. We developed and analyzed various machine learning models, including Support Vector Machine (SVM), XGBoost, K-Nearest Neighbors (KNN), Decision Tree Classifier (DTC), and Random Forest Classifier (RFC), to evaluate their accuracy, precision, and recall. Our study revealed significant variations in model efficacy following rigorous data pretreatment and hyperparameter tuning, emphasizing the necessity for tailored approaches to detecting intrusions. The findings provide useful insights for future research, which will aid in the creation of more resilient cybersecurity measures to meet growing threats.***

***Keywords: Machine Learning, Intrusion Detection, Algorithm, Dataset, NSL-KDD, Attacks***

## I. INTRODUCTION

The significance of effective cybersecurity measures cannot be emphasized as the world becomes more digitalized [11]. Network Intrusion Detection Systems (NIDS) are critical for detecting and categorizing various types of network intrusions, ensuring the safety of our sensitive data and systems [12]. Despite breakthroughs in traditional detection methods, many existing systems struggle to effectively identify new and emerging threats [13].

This study intends to address these issues by creating a comprehensive NIDS that efficiently utilizes machine learning methodologies [14]. We train and evaluate multiple machine learning algorithms, including Support Vector Machine (SVM), XGBoost, K-Nearest Neighbors (KNN), Decision Tree Classifier (DTC), and Random Forest Classifier (RFC), using the NSL-KDD dataset, which is widely accepted as a standard in intrusion detection.

While specialists have investigated many intrusion detection methods, there is still a conspicuous lack of comprehensive evaluations of how well these models perform across multiple metrics [15]. We intend to close this gap by performing a thorough comparison examination of these models, focusing on their accuracy, precision, and recall. Our goal is to provide significant insights that will help improve the continual refinement of intrusion detection systems, allowing them to successfully address the ever-changing cybersecurity threats.

## II. LITERATURE REVIEW

The adoption of machine learning algorithms has resulted in a significant revolution in intrusion detection systems.The author [1] presents essential instruments for data mining and real-world machine learning uses, establishing the foundation for developing intelligent systems capable of recognizing patterns in network data.

The Author [2] further enhances this understanding by discussing pattern recognition and machine learning algorithms critical for identifying anomalies in network traffic. These foundational concepts are crucial for building effective IDS.

The paper [3] underlines the necessity of comprehensive cybersecurity measures, highlighting IDS as critical components in protecting information systems. The Author [5] gives a comprehensive analysis of IDS approaches (signature-based, anomaly-based, and hybrid), emphasizing the persistent issues of accuracy and false positives.

The paper [6] presents survey evaluation practices for IDS, underscoring the necessity of effective metrics to assess their performance in real-world scenarios. The Author [7] focuses specifically on machine learning applications, identifying current trends and proposing future research directions that advocate for more adaptive systems.

The Author [8] (2019) explores the application of deep learning techniques in IDS, demonstrating how these methods can significantly improve detection capabilities against sophisticated attacks. The author [9] introduces fuzzy signature-based approaches, illustrating how fuzzy logic can effectively handle uncertainty in detection processes.

## III. METHODOLOGY

The primary goal of this project is to create and implement a Network Intrusion Detection System (NIDS) that can accurately identify and categorize network intrusions. This is a critical cybersecurity duty since it allows for fast identification and treatment.

### A. Dataset

1) *Dataset Selection:* We used the NSL-KDD dataset, a benchmark in network intrusion detection. Its diverse range of intrusion types makes it an excellent foundation for training robust and adaptable network intrusion detection system (NIDS) models.
2) *Data Preprocessing:* Preprocessing is required on the dataset before training the machine learning models. We divided the data into two sets - one for training and one for testing, with each set containing 50% of the data. This guarantees that the models do not overfit and can generalize effectively to unfamiliar data. We used a Label Encoder to transform categorical data to numerical values, as machine learning algorithms generally work more efficiently with numerical inputs.
3) *Selection and Training of Models:* We chose different machine learning models like Support Vector Machine (SVM), XGBoost, K-Nearest Neighbors (KNN), Decision Tree Classifier (DTC), Random Forest Classifier (RFC), and Linear Regression for this study. These models were selected based on their ability to perform well in classifying tasks and their wide range of algorithms. We utilized the preprocessed data to train each model before conducting tests to make predictions.
4) *Model Evaluation:* To assess each model's performance, we focused on three key metrics: training score, test score, and accuracy. The training score indicates how well each model learned from the training data, while the test score shows how effectively the model generalizes to new data. Accuracy reflects the percentage of correct predictions, classifying instances as normal or anomalous.
5) *Hyperparameter Optimization:* To further enhance model performance, we implemented hyperparameter optimization. We fine-tuned the parameters of the DTC and KNN models using Optuna, a flexible optimization framework. Optuna systematically explores the parameter space to identify combinations that boost model performance.
6) *Conclusion and Model Comparison:* The final step involves comparing the performance of the various models and selecting the top performer for the NIDS. This includes a thorough evaluation of each model's assessment metrics, as well as consideration of factors like computational complexity and the reliability of their predictions.

### B. Algorithms Used For Nids

#### 1) Linear Regression

Linear regression is an algorithm for supervised learning that predicts continuous target values using one or more input features. It depicts the relationship between predictors and response variables using the formula $y = mx + b$, where 'y' is the result, 'x' is the input factors,'m' is the inclination, and 'b' is the y-intercept. The model determines the most suitable line by reducing the gap between predicted and actual values, usually through techniques like ordinary least squares (OLS) or gradient descent. Linear regression is effective when there is a direct correlation between variables, making it valuable for tasks such as forecasting house prices using factors like square footage or the number of bedrooms. Nevertheless, its simplicity may hinder its precision when handling intricate, non-linear patterns.

#### 2) Logistic Regression

Although it is called logistic regression, this algorithm is mainly utilized for binary classification in supervised learning. It forecasts the chance of a two-option result using the logistic (sigmoid) function that assigns input values to a scale from 0 to 1. The model gives probabilities to the positive class, and if the probability is higher than a certain threshold (usually 0.5), it categorizes the input appropriately. The model learns the coefficients that best fit the observed data by maximizing its likelihood. Logistic regression is commonly applied in credit scoring, spam detection, and medical diagnosis because of its simplicity, interpretability, and effectiveness. Multi-class problems can be managed by utilizing either one-vs-rest (OvR) or multinomial logistic regression techniques. Logistic regression is effective when the boundary separating classes is approximately linear or when prioritizing interpretability.

*3) Random Forest Classifier*

The RFC is a type of ensemble learning technique that merges numerous decision trees to improve predictions and lower overfitting. Every decision tree in the forest is trained on a randomly selected subset of the data that includes duplicated observations. During each division, the algorithm picks a random sample of features to consider, thereby introducing additional randomness and decreasing connections between trees. The ultimate forecast is generated by combining predictions from each tree—through either majority vote in classification tasks or averaging probabilities in regression tasks. Random forests excel in scalability, accuracy, and robustness, making them well-suited for voluminous, high-dimensional datasets. They are commonly used in fields like bioinformatics, remote sensing, and anomaly detection.

*4) Decision Tree Classifier*

The DTC is a flexible algorithm used mainly for classifying tasks in supervised learning. It divides the feature space into smaller regions by choosing the most informative features at each node to optimize homogeneity or purity through recursive partitioning. Decision trees are simple to comprehend and visualize, which makes them useful for interpreting decisions made by models. Nevertheless, they may become overly specialized, especially with increased depth and complexity. To prevent overfitting, methods such as pruning, restricting depth, or utilizing Random Forest ensemble techniques can be utilized. Decision trees play a significant role in industries like healthcare, finance, and marketing, where having clear understandability is vital.

*5) KNN*

K-Nearest Neighbors (KNN) is a simple but effective algorithm utilized for supervised learning in classification and regression scenarios. The method works by predicting the class or value of a new data point based on averaging or majority voting from the 'k' closest neighbors in the training set.

In classification, KNN determines labels by majority vote among neighbors, while in regression, it calculates the average of target values from nearby instances. KNN, being a non-parametric, instance-based technique, does not make assumptions about the data distribution and only uses the training data for prediction.

Even though KNN is simple to comprehend and put into practice, it may run slowly when dealing with extensive datasets due to the need to compute distances for every data point. This means that its efficiency is affected by the selection of the distance metric and the value of 'k'.

*6) XGBoost Classifier*

XGBoost is an algorithm that uses gradient boosting to train decision trees sequentially to reduce errors from previous trees. XGBoost differs from other gradient boosting implementations by utilizing a gradient descent optimization algorithm that is based on differentiable loss functions instead of random initialization values at each iteration. This technique combines various individual decision trees to create a powerful predictive model, known as ensemble learning.

To address overtraining, XGBoost incorporates two forms of regularization: L1 (Lasso) and L2 (Tikhonov). Advanced criteria for splitting nodes, like gain and weight, are also included to manage tree development. Moreover, XGBoost also enables parallel training, which enhances computation speed, especially with massive datasets. In summary, XGBoost is extremely adaptable and can be used in a wide range of fields thanks to its outstanding performance, even when time is limited.

*7) SVM*

The SVM is a potent supervised learning technique mainly employed for classification purposes. The goal is to find the optimal hyperplane that divides the various classes in the feature space.

To improve the model's ability to work with new data, SVM chooses hyperplanes that increase the margins, which are the spaces between the hyperplane and the nearest data points of each category.

SVM can manage datasets that can be separated in a linear or non-linear way through the use of different kernel functions like linear, polynomial, and radial basis functions (RBF). These kernels enable the algorithm to translate input data into higher dimensions, in which classes can be linearly separated, resulting in more efficient classification. Therefore, SVM is commonly applied in areas such as image, text, and biological information sciences due to its capacity to handle high-dimensional data, prevent overfitting, and address intricate decision boundaries.

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538*
*Volume 12 Issue XII Dec 2024- Available at www.ijraset.com*

## IV. EVALUATION METRICS

All the evaluation metrics are based on the different attributes used in the *Confusion Matrix*, which is a two-dimensional matrix providing information about the Actual and Predicted class and includes:

- True Positive (TP): The data instances correctly predicted as an Attack by the classifier.
- False Negative (FN): The data instances are wrongly predicted as Normal instances.
- False Positive (FP): The data instances were wrongly classified as an Attack.
- True Negative (TN): The instances correctly classified as Normal instances.

After implementation, the performance of the model was measured using the following metrics:

*1) Precision:* It is the ratio of correctly predicted Attacks to all the samples predicted as Attacks. The Calculation formula is given below:

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

*2) Recall:* It is a ratio of all samples correctly classified as Attacks to all the samples that are Attacks. It is also called a Detection Rate. The formula is given below:

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

*3) False Alarm Rate:* It is also called the false positive rate and is defined as the ratio of wrongly predicted Attack samples to all the Normal samples.

$$\text{False Alarm Rate} = \frac{FP}{FP + TN}$$

*4) True Negative Rate:* It is the proportion of Normal samples that are correctly classified compared to all Normal samples. The formula is given below:

$$\text{True Negative Rate} = \frac{TN}{TN + FP}$$

*5) Accuracy:* It is the proportion of accurately classified instances compared to the total number of instances. Detection Accuracy is another term for it and is a valuable metric that is relevant only in cases where the dataset is evenly distributed. The formula is given below:

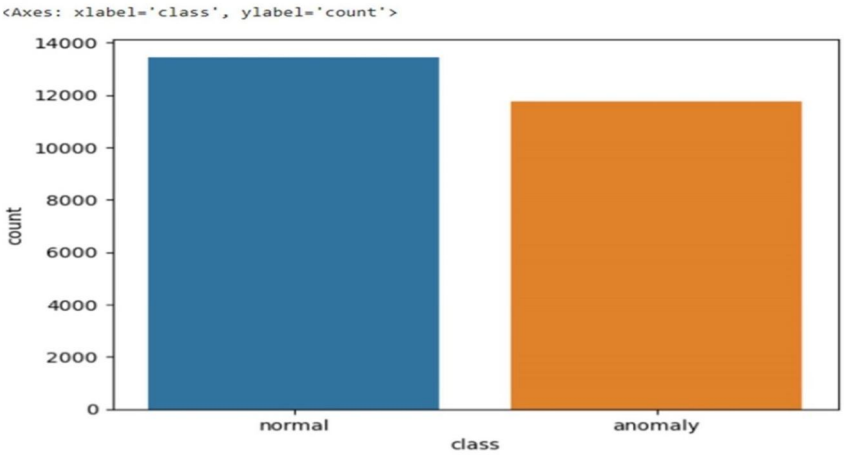
$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}.$$

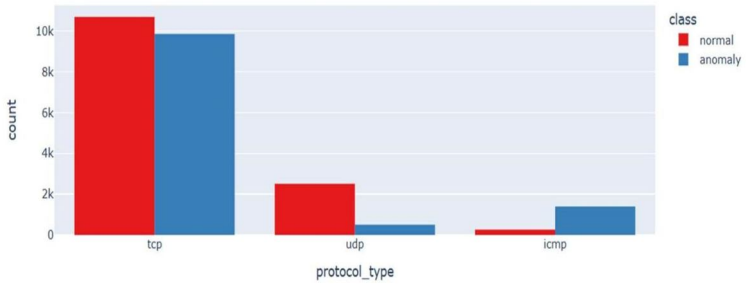
## V. RESULTS

After implementation and performance measurement of the six classifiers used in this paper, their results are as follows:

### A. *Class Details in a Dataset*




### B. *Network Packet Details*




### C. *Classification Report*

The classification report shows the values of precision, recall, F1-score, and support scores of the classifier.

#### 1) *Logistic Regression*

```
**************** LogisticRegression Model Testing ***************
[[5259  626]
 [ 331 6380]]
----------------
              precision    recall  f1-score   support

      normal       0.94      0.89      0.92      5885
     anamoly       0.91      0.95      0.93      6711

    accuracy                           0.92     12596
   macro avg       0.93      0.92      0.92     12596
weighted avg       0.92      0.92      0.92     12596
```

#### 2) *Random Forest Classifier*

```
************** RandomForestClassifier Model Testing **************
[[5851   34]
 [  22 6689]]
--------------
              precision    recall  f1-score   support

      normal       1.00      0.99      1.00      5885
     anamoly       0.99      1.00      1.00      6711

    accuracy                           1.00     12596
   macro avg       1.00      1.00      1.00     12596
weighted avg       1.00      1.00      1.00     12596
```

*3) XGBoost Classifier*

```
********** XGBoostclassifier Model Testing **********
[[5850   35]
 [  23 6688]]
----------
              precision    recall  f1-score   support

      normal       1.00      0.99      1.00      5885
     anamoly       0.99      1.00      1.00      6711

    accuracy                           1.00     12596
   macro avg       1.00      1.00      1.00     12596
weighted avg       1.00      1.00      1.00     12596
```

*4) KNN*

```
********** KNeighborsClassifier Model Testing **********
[[5771  114]
 [ 109 6602]]
----------
              precision    recall  f1-score   support

      normal       0.98      0.98      0.98      5885
     anamoly       0.98      0.98      0.98      6711

    accuracy                           0.98     12596
   macro avg       0.98      0.98      0.98     12596
weighted avg       0.98      0.98      0.98     12596
```

*5) Decision Tree Classifier*

```
********** DecisionTreeClassifier Model Testing **********
[[5853   32]
 [  38 6673]]
----------
              precision    recall  f1-score   support

      normal       0.99      0.99      0.99      5885
     anamoly       1.00      0.99      0.99      6711

    accuracy                           0.99     12596
   macro avg       0.99      0.99      0.99     12596
weighted avg       0.99      0.99      0.99     12596
```

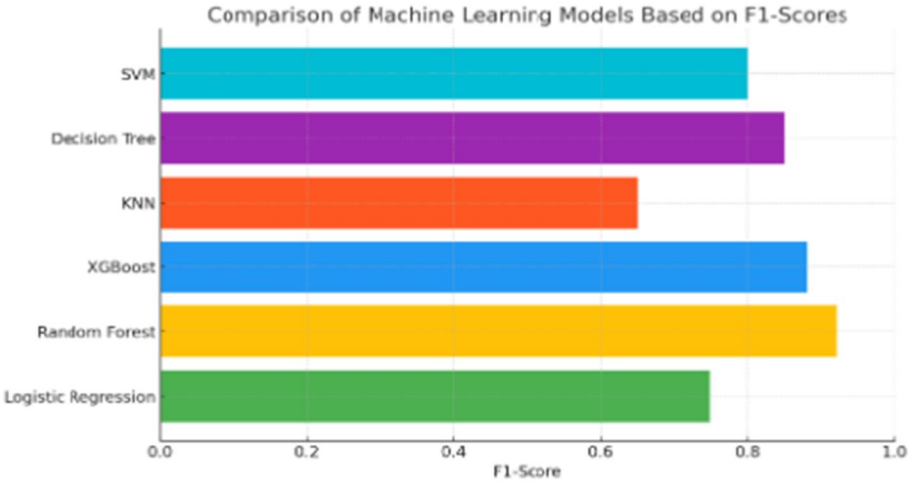
*6) Support Vector Machine*

```
********** SupportVectorMachine Model Testing **********
[[5306  579]
 [ 250 6461]]
----------
              precision    recall  f1-score   support

      normal       0.96      0.90      0.93      5885
     anamoly       0.92      0.96      0.94      6711

    accuracy                           0.93     12596
   macro avg       0.94      0.93      0.93     12596
weighted avg       0.94      0.93      0.93     12596
```

*D. Comparison of F1-Score*




## VI. DISCUSSION

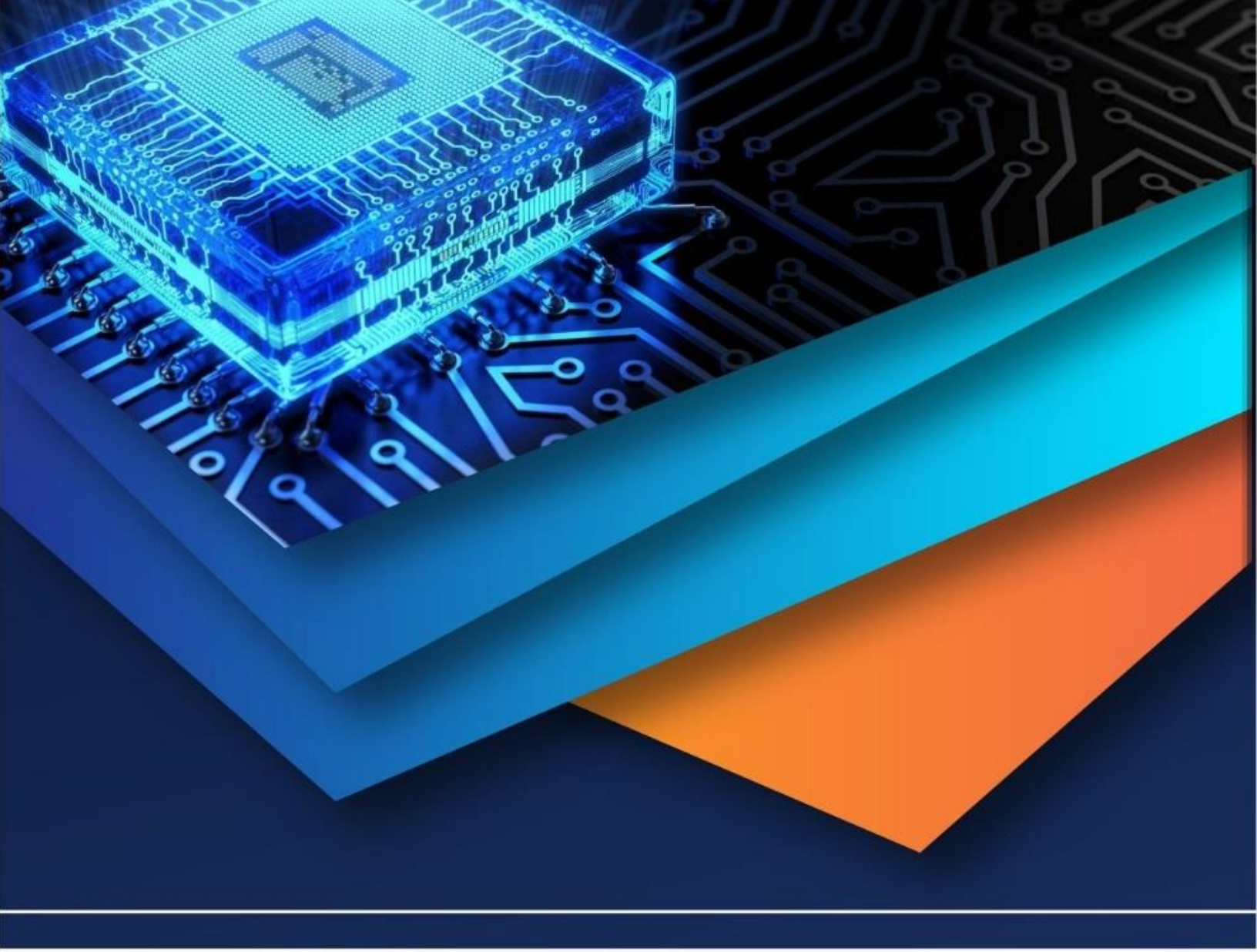
The study shows that the Random Forest Classifier can handle high-dimensional data and resist overfitting making it the most suitable for NIDS. KNN's limitations suggest it is unsuitable for large-scale applications, While XGBoost also achieved competitive results, its complexity may limit its practical application in real-time systems. Future research should focus on integrating deep learning or hybrid models to enhance detection accuracy while reducing computational demands.

## VII. CONCLUSION

This paper presented a Network Intrusion Detection System using machine learning built with the NSL-KDD dataset. According to the results the Random Forest Classifier had better performance than other models, displaying high accuracy and low false positive rates, making it a good fit for real-world use. However, challenges related to scalability and model interpretability remain. Future studies should explore deep learning and hybrid approaches to further improve NIDS performance as cybersecurity threats continue to evolve.Our goal is to enhance the progress of more responsive and efficient intrusion detection systems to protect sensitive data in our digital environment.

## REFERENCES

[1] Witten, I. H., & Frank, E. (2002). Data mining: practical machine learning tools and techniques with Java implementations. AcmSigmod Record, 31(1), 76-77.
[2] Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.
[3] Graham, J., Olson, R., & Howard, R. (Eds.). (2011). Cyber security essentials. CRC Press.
[4] Conrad, E., Misenar, S., & Feldman, J. (2012). CISSP study guide. Newnes
[5] Liao, H. J., Lin, C. H. R., Lin, Y. C., & Tung, K. Y. (2013). Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, 36(1), 16-24.
[6] Milenkoski, A., Vieira, M., Kounev, S., Avritzer, A., & Payne, B. D. (2015). Evaluating computer intrusion detection systems: A survey of common practices. ACM Computing Surveys (CSUR), 48(1), 1-41.
[7] Hamid, Y., Sugumaran, M., &Balasaraswathi, V. R. (2016). Ids using machine learning-current state of state-of-the art and future directions. Current Journal of Applied Science and Technology, 1-22.
[8] Liu, H., & Lang, B. (2019). Machine learning and deep learning methods for intrusion detection systems: A survey. applied sciences, 9(20), 4396 .
[9] Masdari, M., & Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based intrusion detection systems. Applied Soft Computing, 106301.
[10] https://en.wikipedia.org/wiki/, accessed 1/5/2021
[11] Anderson, R. (2020). *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley.
[12] Ahmed, M., Mahmood, A. N., & Hu, J. (2016). *A survey of network intrusion detection systems*. *Journal of Network and Computer Applications*, 60, 5-21.
[13] Ranjan, R., & Ghosh, A. (2019). *Machine Learning for Cybersecurity: A Survey*. *IEEE Transactions on Emerging Topics in Computing*, 8(2), 300-313.
[14] Zarpelão, B. B., Sampaio, S. M., & Maciel, P. (2017). *A comprehensive review of intrusion detection systems based on machine learning*. *Computers & Security*, 57, 226-245.
[15] Khan, M. K., & Alghamdi, R. (2020). *A review of the machine learning techniques for intrusion detection*. *Journal of King Saud University-Computer and Information Sciences*.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY