Performance Comparison of Supervised Classifiers in Intrusion Detection Systems

Onesinus Saut Parulian Tamba[1]
Department of Information Technology, Universitas Nusa Mandiri, Jakarta, Indonesia

Abstract. Intrusion Detection System (IDS) is an essential component in network security to detect and respond to cyber attacks. This study explores the use of several supervised classifier algorithms to classify attacks using the KDDTest-21 dataset from NSL-KDD. This dataset was chosen because of its improvement over the KDD'99 dataset that reduces bias due to duplication and uneven data distribution. The algorithms used include Logistic Regression, K-Nearest Neighbors (KNN), Gaussian Naive Bayes, Support Vector Machine (SVM), Decision Treem, Random Forest and Deep Feedforward Neural Network. Each algorithm is applied to a normalized dataset using RobustScaler. Performance assessment is carried out based on metrics such as accuracy, precision, recall, and F1-score to determine the best algorithm in detecting various types of attacks. The results show that several algorithms, especially Random Forest, have better performance in detecting attacks with high accuracy with 97.68% and other merics score are more than 98%. This finding is important for optimizing a more effective IDS in protecting network infrastructure from increasingly complex cyber attacks.

Keywords: Machine Learning; Supervised Learning; Classification; Intrusion Detection System
Received Desember 2025 / **Revised** February 2026 / **Accepted** February 2026

INTRODUCTION

Intrusion detection systems (IDS) play a crucial role in safeguarding network security by identifying and mitigating potential cyber-threats [1]. With the increasing volume and sophistication of cyber attacks, traditional IDS techniques face significant challenges in effectively detecting diverse and evolving intrusion patterns [2]. The widely used KDD'99 dataset suffers from issues such as redundant records and skewed class distributions, which can lead to biased results and over-fitting [3]. To address these limitations, the NSL-KDD dataset was used as an improved benchmark, removing duplicate entries and balancing instance difficulty to enable more reliable assessments of machine-learning approaches for IDS [3][4]. Numerous studies have applied supervised classifiers including Decision Trees, Support Vector Machines (SVM), Naive Bayes, and Random Forests to network intrusion detection and demonstrated their promise [4]. However, only a limited number of investigations have undertaken a systematic comparison of multiple supervised classifiers on the refined NSL-KDD benchmark [5][6][7], particularly using subtasks such as the KDDTest-21 subset. Accordingly, this research implements and compares seven supervised learning algorithms: Logistic Regression, K-Nearest Neighbors (KNN), Gaussian Naive Bayes, SVM, Decision Tree, and Random Forest for intrusion classification on the KDDTest-21 subset of the NSL-KDD dataset.

Our objective is to identify the most effective classifier by evaluating them on metrics of accuracy, precision, recall, and F1-score. By doing so, we aim to enhance IDS detection capabilities and to inform future IDS research and deployment by revealing the strengths and weaknesses of each algorithm under consistent dataset and evaluation conditions. Additionally, we apply a feature-scaling approach to improve evaluation robustness [8], acknowledging the importance of preprocessing in ML-based IDS systems [9]. The findings of this study have significant practical implications for organizations seeking optimal IDS configurations, and for researchers designing machine-learning-based security solutions in an increasingly digital and interconnected world [10].

Dataset

The dataset used for this study is the KDDTest-21 subset of the NSL-KDD dataset. The NSL-KDD dataset is an improved version of the KDD'99 dataset, designed to address issues of redundancy and data imbalance.

[1] * Corresponding author.
Email addresses: onesinus231@gmail.com (Tamba)

The KDDTest-21 subset specifically excludes instances with a difficulty level of 21, providing a more balanced set of normal and attack data for evaluation purposes. The dataset represents network traffic data characterized by several features capturing different aspects of the network connections. It includes features such as duration, protocol type, service, source and destination bytes, and various indicators of abnormal activities (e.g., wrong fragments, failed login attempts). Dataset can be accessed on Kaggle at https://www.kaggle.com/datasets/hassan06/nslkdd/data.

The dataset also contains statistical metrics like the count of connections to the same host or service, and error rates for different types of connections. Each record is labeled with an outcome representing the type of network activity (e.g., normal or a specific type of attack such as guess_passwd, snmpguess, processtable, nmap). The dataset is used for training and evaluating intrusion detection models by distinguishing between normal and malicious network activities. Table 1 and Table 2 shows the dataset sample to be processed in this research.

Table 1. Dataset Sample

#	Feature Name	Data Type	Data Example				
			1	2	3	4	5
1	duration	int64	13	0	0	0	0
2	protocol_type	object	tcp	udp	tcp	udp	tcp
3	service	object	telnet	private	telnet	private	private
4	flag	object	SF	SF	S3	SF	SH
5	src_bytes	int64	118	44	0	53	0
6	dst_bytes	int64	2425	0	44	55	0
7	land	int64	0	0	0	0	0
8	wrong_fragment	int64	0	0	0	0	0
9	urgent	int64	0	0	0	0	0
10	hot	int64	0	0	0	0	0
41	outcome	object	guess_passwd	snmpguess	processtable	normal	nmap
42	level	int64	2	12	18	17	17

Table 2. List of other features

#	Feature Name	#	Feature Name	#	Feature Name
11	logged_in	12	num_compromised	13	root_shell
14	su_attempted	15	num_root	16	num_file_creations
17	num_shells	18	num_access_files	19	num_outbound_cmds
20	is_host_login	21	is_guest_login	22	count
23	srv_count	24	serror_rate	25	srv_serror_rate
26	rerror_rate	27	srv_rerror_rate	28	same_srv_rate
29	diff_srv_rate	30	srv_diff_host_rate	31	dst_host_count
32	dst_host_srv_count	33	dst_host_same_srv_rate	34	dst_host_diff_srv_rate
35	dst_host_same_src_port_rate	36	dst_host_srv_diff_host_rate	37	dst_host_serror_rate
38	dst_host_srv_serror_rate	39	dst_host_rerror_rate	40	dst_host_srv_rerror_rate

Data distribution for feature protocol type and outcome is shown as Figure 1.

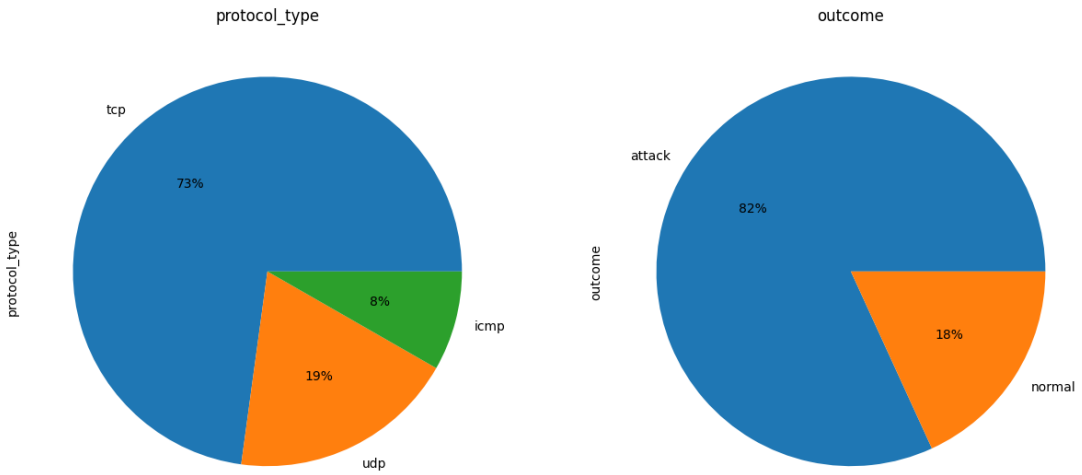

Figure 1. Features and Target

METHODS

This section outlines the methodology used in this study, including dataset description, data preprocessing techniques, implementation of multiple supervised classifiers, and the evaluation metrics used to assess their performance. All experiments were conducted on a standard computing environment with Python 3.8 and Scikit-Learn. The models were implemented using default settings of the Scikit-Learn library. The results were analyzed and compared to identify the best-performing model for classifying network intrusions. This research flow is outlined as Figure 2.

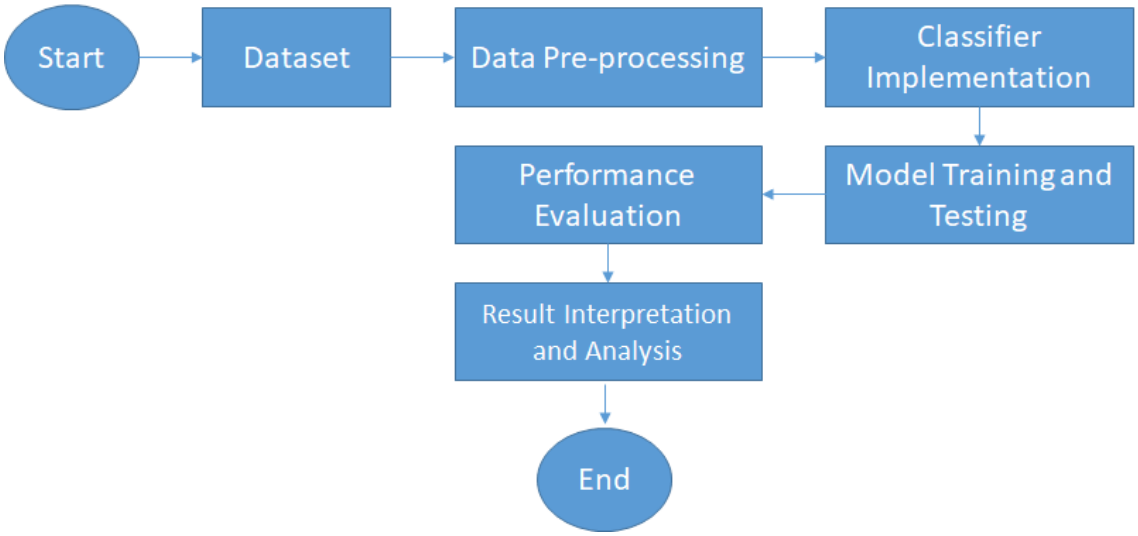

Figure 2. Research Method Flow

Data Preprocessing

Data preprocessing is crucial for ensuring the effectiveness of machine learning models. The following steps were undertaken:

1. Feature Selection and Encoding: The dataset initially contains both numeric and categorical features. Categorical features were encoded using one-hot encoding to convert them into a format suitable for machine learning algorithms. PCA has been implemented for feature processing.
2. Data Scaling: A scaling techniques RobustScaler, were applied to normalize the feature set and mitigate the impact of outliers.

Classifier Implementation

Seven supervised classifiers were implemented using Python and the Scikit-Learn library:

1. Logistic Regression: A linear model was employed to establish a baseline performance for binary classification tasks.
2. K-Nearest Neighbors (KNN): This algorithm was used to assess the importance of local data points in classification.
3. Gaussian Naive Bayes: A probabilistic classifier was tested to evaluate its performance on the assumption of feature independence.
4. Support Vector Machine (SVM): A kernel-based method was utilized to handle high-dimensional data and classify complex patterns.
5. Decision Tree: A tree-based model was used to understand the hierarchical structure of decision-making in detecting intrusions.
6. Random Forest: An ensemble method was applied to enhance classification performance by combining multiple decision trees.
7. Deep Feedforward Neural Network (DNN): Deep Feedforward Neural Network (Fully Connected Network).

Model Training and Testing:
The dataset was split into training and testing sets using an 80-20 ratio. The training set was used to train the models, while the testing set was used to evaluate their performance.

Evaluation Metrics:
The dataset was split into training and testing sets using an 80-20 ratio. The training set was used to train the models, while the testing set was used to evaluate their performance. To evaluate the performance of each classifier, multiple metrics were employed. Equation 1, 2, 3, 4 are the metrics formula.

- Accuracy: The proportion of correctly predicted instances out of the total instances.

$$Accuracy = \frac{TN+TP}{TN+FP+TP+FN} \quad (1)$$

- Precision and Recall: Metrics used to evaluate the classifier's ability to correctly identify attack instances (positive class).

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

- F1-Score: The harmonic mean of precision and recall, providing a balanced measure of a classifier's performance

$$F1\ Score = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (4)$$

RESULT AND DISCUSSION

The performance of each classifier is evaluated using accuracy, precision, recall, and F1-score for both training and test datasets. The results highlight the effectiveness and limitations of each model in detecting network intrusions. The performance metrics for each classifier are summarized in Table 3. The metrics include training and test accuracy, precision, recall, and F1-score.

Table 3. Classifiers Performance Comparison

Classifier	**Accuracy**		**Precision**		**Recall**		**F1 Score**	
	Training	Testing	Training	Testing	Training	Testing	Training	Testing
Logistic Regression	86.33	85.91	88.35	87.74	95.97	96.11	92.00	91.73
K-Nearest Neighbors (KNN)	94.08	93.42	94.75	93.77	98.22	98.44	96.45	96.05
Gaussian Naive Bayes	**79.63**	**79.11**	97.23	96.08	**77.35**	**77.49**	86.16	85.79
SVM	92.68	92.36	92.62	92.30	**98.96**	98.86	95.68	95.47
Decision Tree	99.63	97.13	**100.00**	98.29	99.55	98.18	99.77	98.24
Random Forest	**99.63**	**97.68**	99.68	**98.35**	99.87	98.81	**99.77**	**98.58**
Deep Feedforward Neural Network	97.10	95.91	96.86	95.94	99.69	**99.17**	98.26	97.53

Comparison of Classifier Performance

This section presents and discusses the results obtained from the implementation of multiple supervised classifiers.

1. The results indicate that the Random Forest classifier achieved the highest performance among all classifiers, with a training accuracy of 99.63% and a test accuracy of 97.68%. It also exhibited high precision, recall, and F1-score, indicating its ability to accurately detect intrusions with minimal false positives and false negatives. This can be attributed to the ensemble nature of Random Forest, which combines multiple decision trees to enhance predictive accuracy and control overfitting.
2. The Decision Tree classifier also performed well, with a test accuracy of 97.13%.
3. Linear SVC (SVM) and K-Nearest Neighbors (KNN) also demonstrated strong performance, with test accuracies of 92.36% and 93.42%, respectively. Linear SVC had the advantage of handling high-dimensional data well, reflected in its high recall of 98.86%. However, its slightly lower precision suggests it may have produced more false positives compared to other classifiers. KNN, while showing competitive results, is more computationally intensive during prediction, which could be a limitation in real-time IDS applications.
4. Logistic Regression, while achieving moderate results (test accuracy of 85.91%), showed a high recall (96.11%) but comparatively lower precision, indicating it is more sensitive to detecting attacks but may produce more false positives. Gaussian Naive Bayes had the lowest accuracy (79.11%) among all classifiers, which can be attributed to its strong assumption of feature independence that may not hold true for complex network intrusion data.
5. Deep Feedforward Neural Network is achieving better result as well with overall more than 95% score.

Qualitative Analysis of Classifier Suitability

From a qualitative perspective, the choice of classifier depends on the specific requirements of the IDS. For environments where the cost of false negatives is high (e.g., critical infrastructure), classifiers like Random Forest or Decision Tree are preferred due to their high recall and precision. In contrast, for applications

where computational efficiency is crucial, and some degree of false positives is acceptable, classifiers like Logistic Regression or Linear SVC may be more suitable.

Summary of Findings

Random Forest classifier outperformed the other models in all evaluated metrics, demonstrating its robustness in handling the diverse types of network intrusions presented in the KDDTest-21 dataset. The Decision Tree classifier also showed strong performance but with a tendency toward overfitting. The results highlight that while simpler models like Logistic Regression and Gaussian Naive Bayes are less effective for complex datasets, they can still offer insights when computational simplicity is a priority. The findings suggest that ensemble methods like Random Forest should be considered for implementation in IDS due to their ability to handle complex intrusion patterns with high accuracy and robustness. Future research could focus on optimizing these models further, considering real-time constraints and resource limitations commonly encountered in practical deployment scenarios. To provide a clearer understanding of the performance comparison, Figure 3, Figure 4, Figure 5, and Figure 6 are showing the results of experiment.

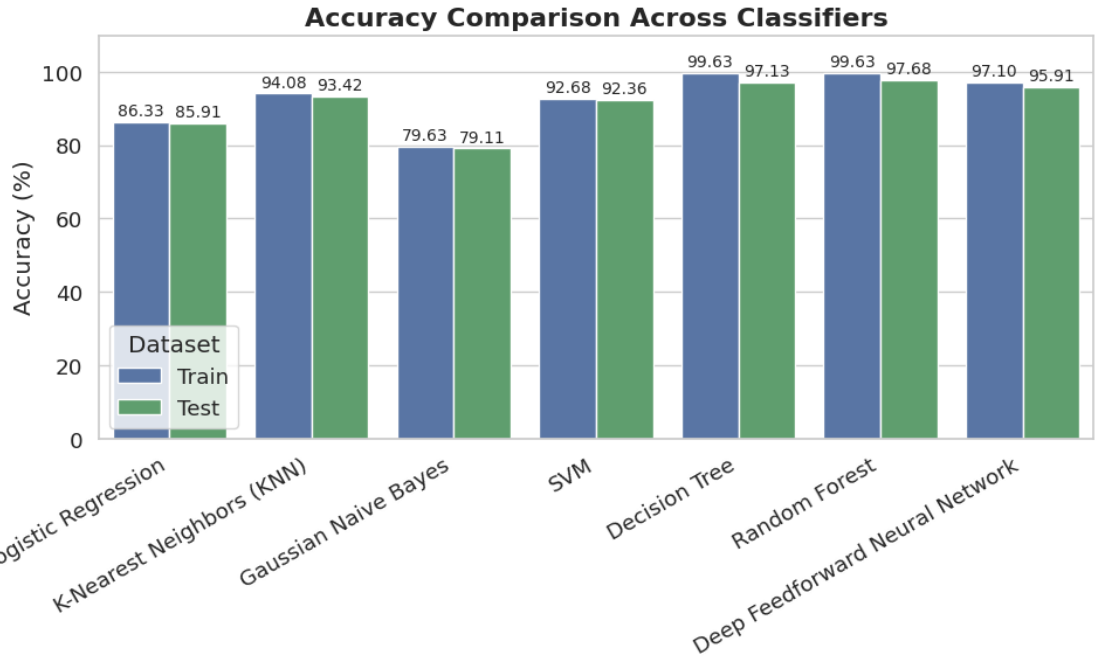

Figure 3. Accuracy Result

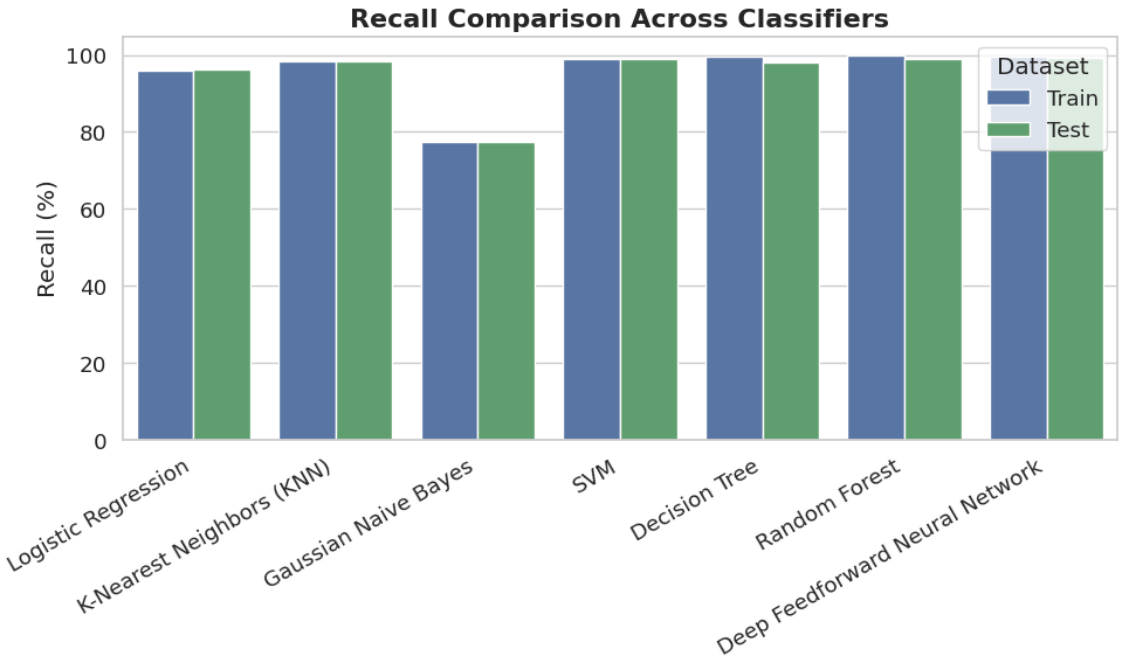

Figure 4. Recall Result

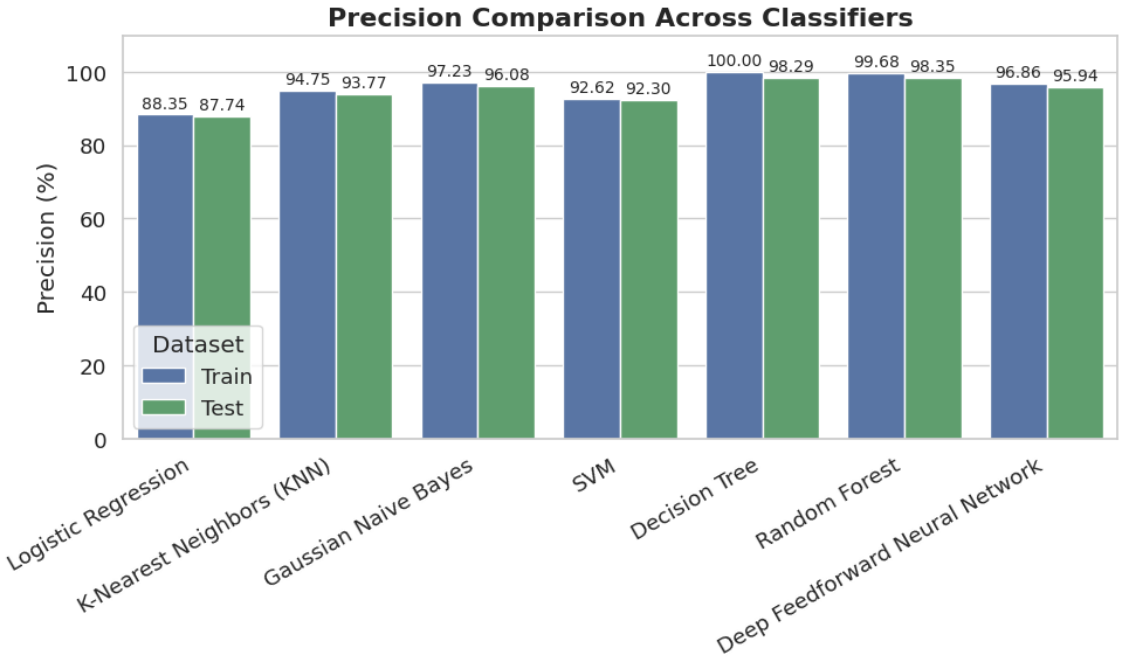

Figure 5. Precision Result

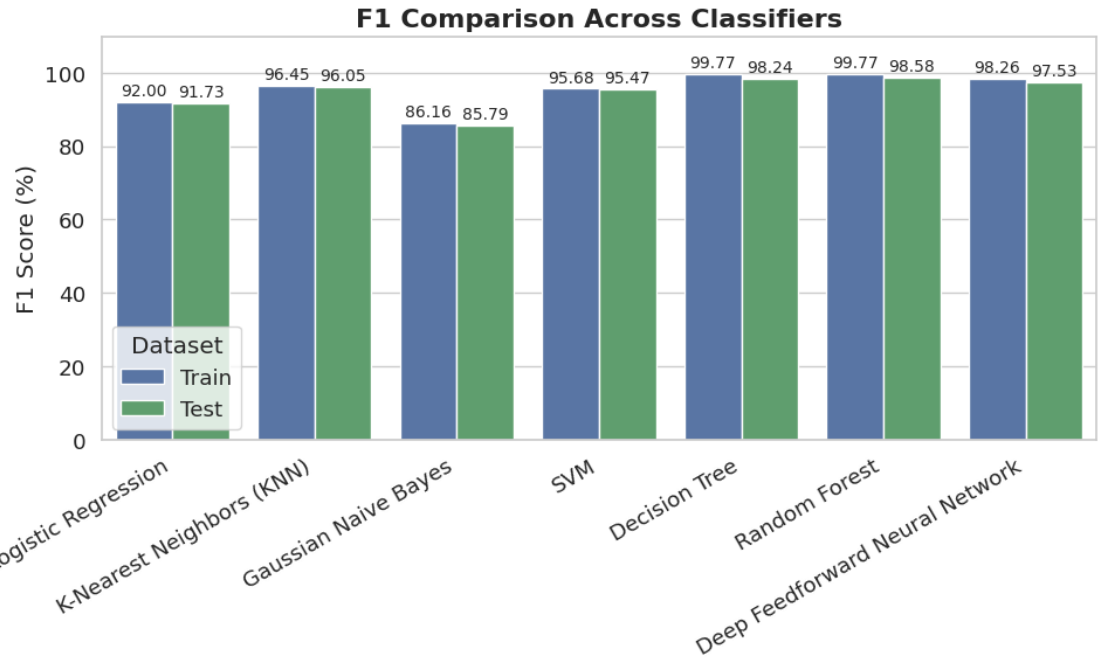

Figure 6. F1 Score Result

CONCLUSION

This study shows that the Random Forest classifier is the most effective for detecting network intrusions in the KDDTest-21 dataset, achieving the highest accuracy, precision, recall, and F1-score. The Decision Tree classifier also performed well but tended to overfit, suggesting the need for further optimization. Linear SVC (SVM) and K-Nearest Neighbors provided competitive results, but SVC had slightly lower precision, and KNN was computationally intensive. Logistic Regression and Gaussian Naive Bayes were less effective but could be useful in low-resource environments. Future research could focus on optimizing parameters, combining classifiers, and testing on newer datasets to improve model generalizability. It is considerable to use Random Forest for high-accuracy intrusion detection systems, also Decision Trees with regularization for better generalization, and use simpler models like Logistic Regression or Naive Bayes when resources are limited. Further exploration of ensemble and hybrid methods is suggested to enhance detection performance.

REFERENCES

[1] Akoh Atadoga, Enoch Oluwademilade Sodiya, Uchenna Joseph Umoga, and Olukunle Oladipupo Amoo, “A comprehensive review of machine learning’s role in enhancing network security and threat detection,” *World Journal of Advanced Research and Reviews*, vol. 21, no. 2, pp. 877–886, Feb. 2024, doi: 10.30574/wjarr.2024.21.2.0501.

[2] E. Benkhelifa, T. Welsh, and W. Hamouda, “A Critical Review of Practices and Challenges in Intrusion Detection Systems for IoT: Toward Universal and Resilient Systems,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3496–3509, 2018, doi: 10.1109/COMST.2018.2844742.

[3] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the KDD CUP 99 data set,” in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, IEEE, Jul. 2009, pp. 1–6. doi: 10.1109/CISDA.2009.5356528.

[4] L. Dhanabal and S. P. Shantharajah, “A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms,” *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, 2015, doi: 10.17148/IJARCCE.2015.4696.

[5] L. Tianyao, H. Huadong, and L. Run, “An Intrusion Detection Framework with Optimized Feature Selection and Classification Combination Using Support Vector Machine,” in *2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI)*, IEEE, May 2023, pp. 182–186. doi: 10.1109/ICETCI57876.2023.10176950.

[6] M. I. Ragab and K. M. Fouad, “Intelligent System for Intrusion Detection Based on Machine Learning,” in *2024 6th Novel Intelligent and Leading Emerging Sciences Conference (NILES)*, IEEE, Oct. 2024, pp. 135–138. doi: 10.1109/NILES63360.2024.10753257.

[7] J. Kevric, S. Jukic, and A. Subasi, “An effective combining classifier approach using tree algorithms for network intrusion detection,” *Neural Comput Appl*, vol. 28, no. S1, pp. 1051–1058, Dec. 2017, doi: 10.1007/s00521-016-2418-1.

[8] V. Asha, M. T. Vasumathi, A. Prasad, Y. V, Y. A. P, and M. Sivani, “Evaluation of ML Models using SMOTE and Feature Scaling for Intrusion Detection System(IDS),” in *2025 International Conference on Visual Analytics and Data Visualization (ICVADV)*, IEEE, Mar. 2025, pp. 243–249. doi: 10.1109/ICVADV63329.2025.10961506.

[9] K. C. Santos, R. S. Miani, and F. de Oliveira Silva, “Evaluating the Impact of Data Preprocessing Techniques on the Performance of Intrusion Detection Systems,” *Journal of Network and Systems Management*, vol. 32, no. 2, p. 36, Apr. 2024, doi: 10.1007/s10922-024-09813-z.

[10] S. Kim and K.-J. Park, “A Survey on Machine-Learning Based Security Design for Cyber-Physical Systems,” *Applied Sciences*, vol. 11, no. 12, p. 5458, Jun. 2021, doi: 10.3390/app11125458.