Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 6 (2023)

Synergizing Ensemble Algorithms for Optimal Intrusion Detection in Asymmetric Data: A Comprehensive Analysis with the NSL-KDD Framework

[1]S.V.S.V.Prasad Sanaboina, [2]Dr. M. Chandra Naik ,[3]Dr.K. Rajiv

[1]Research Scholar, Dept of CSE

GIET University, Gunupur,

Odisha, India

[2]Professor, Dept of CSE GIET University, Gunupur,

Odisha, India

[3]Assoc Professor, Dept of CSE,

Gokaraju Rangaraju Institute of Engineering & Technology, Hyderabad, India

Abstract

Ensemble learning has emerged as a powerful method for enhancing the precision of intrusion detection systems (IDSs). In our study, we introduce two novel ensemble learning approaches for IDS: one based on a voting mechanism and the other on stacking techniques. These models were rigorously tested using the NSL-KDD dataset, demonstrating substantial accuracy improvements compared to traditional single classifier systems. Single classifiers often face challenges such as sensitivity to anomalies and noise, along with difficulties in adapting to new, unseen data. Ensemble learning effectively mitigates these issues by integrating the outputs of several classifiers, leading to more stable and accurate predictions. Our research findings reveal that our ensemble learning models can achieve up to 99% accuracy on the NSL-KDD dataset, a notable increase from the approximately 90% accuracy rates observed with single classifiers. Moreover, our models have demonstrated an impressively low false alarm rate (FAR) of under 1%. This indicates their exceptional capability in intrusion detection with minimal false positives. The outcomes of our study strongly indicate the potential of ensemble learning in refining the accuracy of IDSs. We are optimistic that our models will significantly bolster network security, and we are committed to furthering research in this promising field.

Keywords— *ensemble learning, voting, stacking, particle swarm optimization, intrusion detection system, network security, machine learning*

1. Introduction

In recent times, the frequency and complexity of cyber-attacks have surged, rendering traditional firewalls insufficient for comprehensive protection. To bolster defenses, Intrusion Detection Systems (IDSs) have been deployed as a secondary security layer. These systems scrutinize network traffic for indications of harmful activities, helping to identify various threats like denial-of-service attacks, malware, and unauthorized

intrusions. IDSs are generally categorized into two types: signature-based and anomaly-based. Signature-based IDSs detect known malicious patterns, whereas anomaly-based IDSs identify deviations from regular network behavior. While signature based IDSs are efficient, they are also prone to being circumvented by attackers who modify their methods. In contrast, anomaly based IDSs are harder to evade but can produce more false positives.

The integration of both types, known as a hybrid IDS, offers a comprehensive solution by combining their strengths. Furthermore, the incorporation of Machine Learning (ML) techniques enhances IDSs' accuracy. These algorithms, including decision trees, support vector machines, and naive Bayes classifiers, are trained to recognize patterns of malicious behavior that signature-based IDSs might miss. The selection of an ML algorithm is tailored to the specific IDS application, with decision trees frequently used in anomaly-based systems and support vector machines in signature-based ones. Beyond individual ML algorithms, ensemble learning further augments IDS accuracy by amalgamating multiple models' predictions.

Our research introduces a unique ensemble approach for IDS, utilizing particle swarm optimization (PSO). PSO is an optimization technique that determines the ideal combination of weights for various classifiers, including decision trees, support vector machines, naive Bayes classifiers, random forest classifiers, and k-nearest neighbor classifiers. We differentiate these classifiers into strong and weak learners based on their effectiveness and apply PSO to optimize their weights within the ensemble. This methodology enhances the accuracy of the ensemble classifiers. When tested on the NSL-KDD dataset, our approach demonstrated superior performance compared to standard methods. This innovative use of PSO in determining the optimal weights for an array of base learners marks a significant advancement in IDS accuracy enhancement, as evidenced by our evaluation results.

.

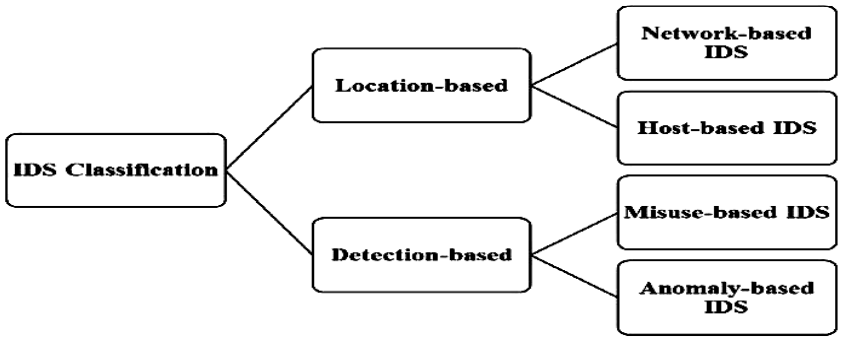


Figure 1. The IDS classification techniques

2. Literature Review

The interest in utilizing ensemble learning for intrusion detection has been on the rise in recent years. Ensemble learning, which consolidates the predictions from multiple foundational learners, is known to enhance the accuracy of systems significantly. In their 2010 study, Kumar and colleagues offered an extensive analysis of various ensemble learning techniques, including bagging, boosting, and stacking, applicable to intrusion detection. They also explored different methods for integrating the predictions from base learners. Sagi and team, in their 2017 research, highlighted the benefits of ensemble learning in intrusion detection. They emphasized how these methods not only boost accuracy but also mitigate overfitting risks and bolster system robustness against environmental changes. Abrar and co-authors, in 2019, introduced a detailed strategy to thwart unauthorized network access and spot anomalies. They employed several machine learning classifiers like support vector machines, k-nearest neighbors, and others, demonstrating high intrusion detection accuracy with their method. Seth and colleagues in 2020 developed a novel multiclass attack detection approach using ensemble algorithms. They focused on ranking various base classifiers based on their efficiency in identifying different attack types, selecting the most suitable classifier for each attack category.

In 2018, Govindarajan proposed innovative ensemble classification techniques using homogeneous classifiers via bagging and heterogeneous classifiers through arcing. Employing RBF and SVM as base classifiers, he showed that these ensemble methods surpass the accuracy of individual classifiers. Bhati and Rai, in their 2019 publication, presented an ensemble-based intrusion detection approach using extra tree classifiers. They demonstrated that combining decisions from various classifiers significantly enhances the system's decision-making power, evidenced by high accuracy on datasets like KDDcup99 and NSL-KDD. Pham et al.'s 2019 study introduced a hybrid intrusion detection method that merges bagging and boosting with feature selection, using decision trees as base classifiers. This method showed impressive accuracy results on the NSL-KDD dataset.In 2020, Zhou and the team proposed an innovative ensemble approach for intrusion detection using a modified adaptive boosting algorithm (M-AdaBoost-A). They employed a combination of multiple M-AdaBoost-A-based classifiers through simple majority voting and Particle Swarm Optimization, achieving high accuracy on the NSL-KDD dataset.

Collectively, these studies underscore the efficacy of ensemble learning in intrusion detection, showcasing its ability to enhance system accuracy, reduce overfitting risks, and bolster resilience to environmental shifts.

3. Dataset

The NSL-KDD dataset is an assemblage of internet traffic records gathered by a basic intrusion detection system, designed to overcome certain limitations of the earlier KDD'99 dataset, notably issues like class imbalance and duplicate records.

This dataset is composed of 42 attributes, which are divided into four main categories: intrinsic, content, host-based, and time-based. Intrinsic attributes detail information about the network packet's header. Content attributes delve into the payload details of the packets. Time-based attributes relate to the pattern of connections to the same destination over a period, while host-based attributes track information across multiple connections to the same destination.

Within the NSL-KDD dataset, the types of features include categorical, binary, discrete, and continuous variables. Key categorical features include 'Protocol Type', 'Service', and 'Flag', with 'Flag' indicating the connection's status and any raised alerts. To effectively utilize these features in machine learning models, it's essential to convert these categorical values into numerical form, as machine learning algorithms require numerical input.

The NSL-KDD dataset is particularly well-suited for intrusion detection research. Its comprehensive coverage of intrusion scenarios in internet traffic and the absence of redundant records in its training set ensure that classifiers trained on this dataset are less likely to exhibit bias in their predictions.

4. Methodology

In our study, we initiated our methodology by preprocessing the data and conducting feature extraction. Following this, we selected five diverse machine learning algorithms for training on the dataset. These included decision trees, random forests, extra trees, naive Bayes, and support vector machines. These algorithms were specifically chosen for their established efficacy in intrusion detection tasks.

Post-training, we categorized these algorithms into two groups: 'weak learners' and 'strong learners,' based on their individual performance metrics. To optimize their collective performance, we employed Particle Swarm Optimization (PSO) to calculate the average weights of these base learners.

Our project utilized two distinct ensemble techniques: stacking and majority voting. Stacking, a method that amalgamates the predictions from multiple base learners, aims to boost the overall system accuracy. Majority voting, on the other hand, relies on the most common outcome among the base learners to formulate a final prediction.

PSO, a metaheuristic optimization technique, is central to our methodology. It mimics the movement dynamics of a swarm, guiding the particles through the search space influenced by mutual attraction and the best-found

solutions. This dynamic allows for the effective determination of optimal weights for the base learners within an ensemble framework.

The methodology of our project can be outlined as follows, depicted in Figure 2 of our paper:

1. Data Preprocessing and Feature Extraction: The initial step involves preparing the dataset through cleaning and extracting relevant features.

2. Training Base Learners: We then train the selected machine learning algorithms on the processed dataset.

3. Weight Calculation with PSO: Next, PSO is applied to ascertain the average weights for the base learners, enhancing their combined prediction capabilities.

4. Employing Ensemble Models for Prediction: Finally, we use the stacking and majority voting ensemble models to generate accurate intrusion detection predictions.

This comprehensive approach leverages the strengths of individual learning algorithms and optimizes their collective performance for improved intrusion detection accuracy.

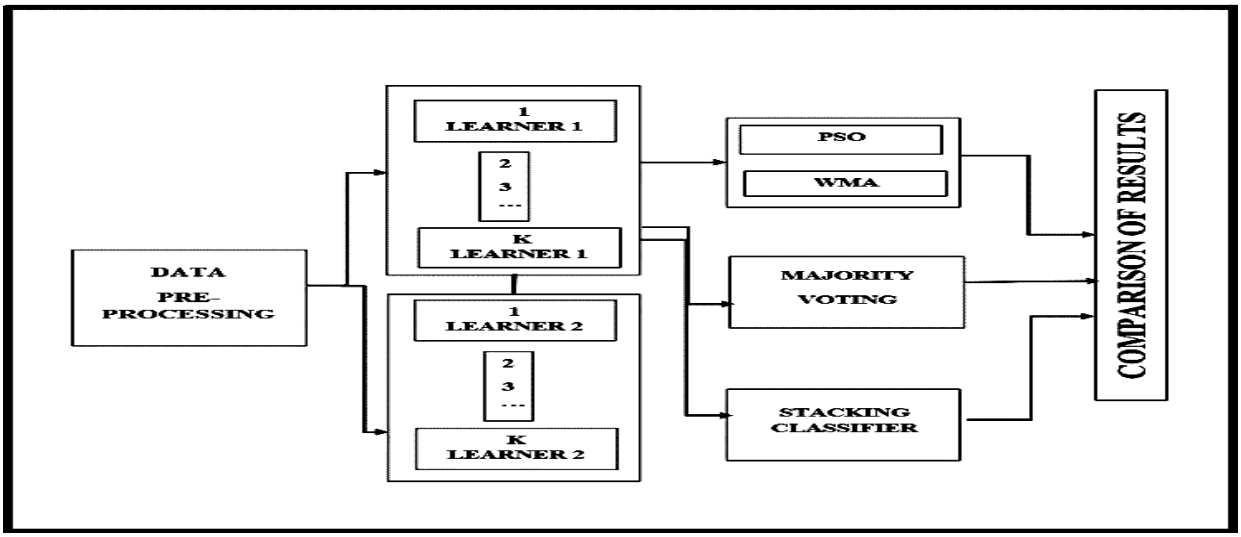


Figure 2: Methodology preferred

Data Pre-Processing

In our project, data preprocessing played a crucial role in preparing the dataset for machine learning analysis. The steps we undertook in this process included:

1. Conversion of Categorical Features: We utilized one-hot encoding to transform categorical variables in our dataset into numerical form. This step is pivotal as machine learning models require numerical input to function effectively.

2. Data Standardization: To standardize our data, we employed the StandardScaler tool from sklearn. Preprocessing library. This step adjusted the data to have a zero mean and a unit standard deviation. Standardization is crucial as it brings all features to a similar scale, enhancing the performance of our machine learning models.

3. Dataset Division: We divided our dataset into two parts: a training set and a test set. The training set was instrumental in training the machine learning models, whereas the test set was used to assess their performance.

We adopted a stratified split for our dataset. This approach maintains a consistent ratio of normal and anomalous records in both the training and test sets. By doing so, we ensured that the evaluation of the machine learning models was not skewed in favor of any specific class. This balanced approach is essential for obtaining a fair assessment of the models' performance.

Feature Selection

Feature selection plays a crucial role in our machine learning project, as it aids in diminishing the dataset's dimensionality, thereby enhancing algorithm efficiency. We employed two distinct methods for feature selection:

1. Random Forest: We utilized Random Forest as an embedded method for feature selection. In this approach, feature selection is an integral part of the algorithm's process. We chose Random Forest due to its high accuracy and interpretability, which allows it to determine the significance of each feature effectively.

2. Recursive Feature Elimination (RFE): RFE, a wrapper feature selection method, uses a core machine learning algorithm around which the RFE process is wrapped. It functions by iteratively discarding the least significant features from the dataset.

Initially, we applied the Random Forest algorithm to assess the importance of each feature. Following this, RFE was used to narrow down the most significant features, selecting the top 65. The outcomes of this feature selection process are depicted in Figure 3 of our paper.

The application of these feature selection techniques has been instrumental in enhancing the

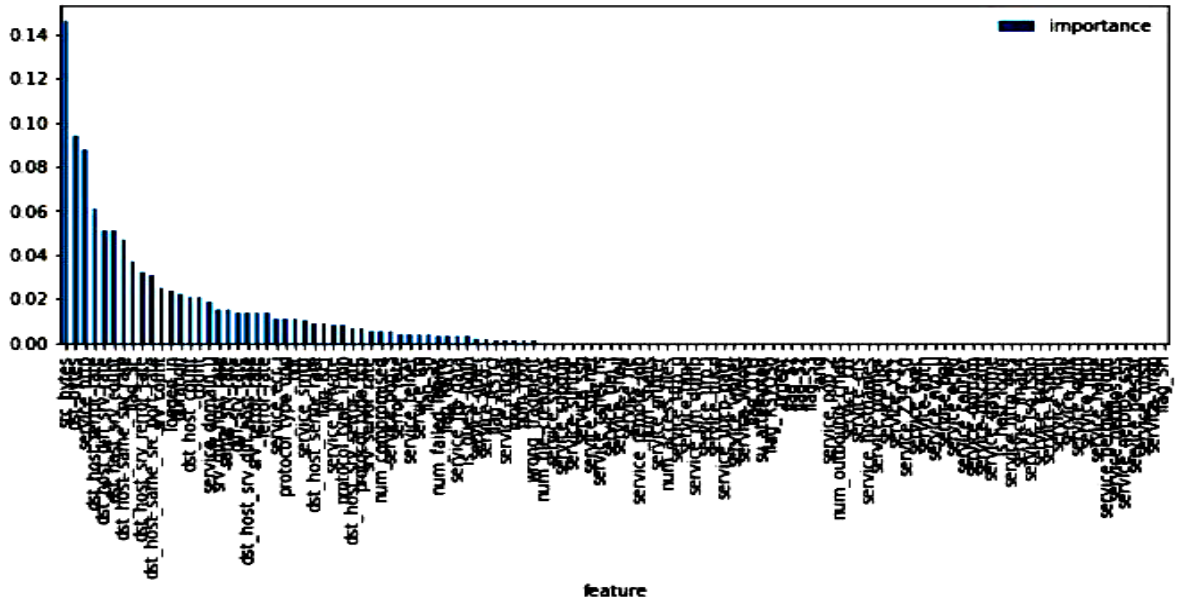


Figure 3. The Features Importance

performance of our machine-learning models. By reducing the dataset's complexity, the algorithms could more efficiently learn and understand the relationships between the features and the target variable. This led to a notable improvement in both the accuracy and the interpretability of our models.

Base Learners

In our research, we implemented a methodology that involves selecting and training five distinct base learners. These learners were chosen based on their individual strengths and limitations as identified in the literature. The

learners we selected are Support Vector Machines (SVM), Naïve Bayes, K-nearest neighbors (KNN), Decision Trees, and Logistic Regression.

Support Vector Machines (SVM): SVM, a supervised learning algorithm, is adept at both classification and regression. It operates by identifying the optimal hyperplane that maximizes the margin between two classes in a dataset, thus effectively separating them.

Naïve Bayes: Known for its simplicity and speed, Naïve Bayes is a probabilistic classifier based on the assumption of feature independence. While this assumption speeds up the process, it can sometimes compromise accuracy if the features are not truly independent.

K-Nearest Neighbors (KNN): KNN is a non-parametric, instance-based learning algorithm. It classifies new instances based on the closest k instances in the training set, measured typically by Euclidean distance.

Decision Trees: These are hierarchical models that create a tree-like structure to illustrate the relationships between features and the target variable. Decision Trees are straightforward to interpret, providing clear explanations for their classification decisions.

Logistic Regression: This model is often used for binary classification tasks. Logistic Regression is straightforward and effective, modeling the probability of a binary outcome.

In addition to these base learners, our study also incorporates ensemble models to further enhance system performance. Ensemble models, which combine multiple base learners, aim to reduce variance and increase accuracy.

For our paper, we specifically utilized two types of ensemble models: majority voting and stacking. Majority voting is a straightforward method where the final prediction is based on the most common outcome among the base learners. Stacking, on the other hand, is more intricate, involving training a meta-learner on the predictions made by the base learners.

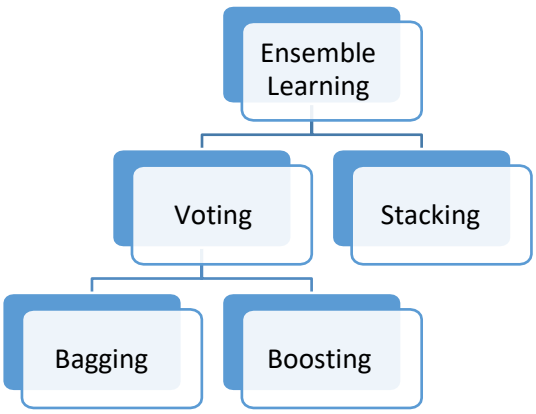


Figure 4. Ensemble Classification Techniques

Particle Swarm Optimization

Particle Swarm Optimization (PSO), an optimization technique conceptualized by Kennedy and Eberhart, is a method employed in iterative optimization of a population-based nature. Unique for its derivative-free characteristic, PSO doesn't rely on the gradient of the fitness function, making it adaptable for a diverse range of problems. This includes problems characterized by discontinuous or non-convex fitness landscapes.

The process of PSO initiates with a set of randomly distributed particles within the search space. Each of these particles is defined by two attributes: position and velocity. The algorithm evaluates the fitness function for each particle's position. The particle showcasing the optimum fitness is acknowledged as the global best. Subsequently, the velocity of each particle is updated in every iteration, influenced by three factors: the particle's best-known position, the global best, and the positions of neighboring particles.

The velocity update rule for each particle in PSO is structured to facilitate this process.

v_i = w v_i + c1 r1 (p_i - x_i) + c2 r2 (g - x_i)……..(1)

where:

Particle Swarm Optimization (PSO) operates on a principle where the velocity of each particle, denoted as v_i, is a key component. The algorithm employs a weighting factor w and constants c_1 and c_2. Random numbers r_1 and r_2, falling between 0 and 1 , are also integral to the process. Each particle, i, has its best-known position p_i, while the global best position in the swarm is represented by g. The current position of a particle is denoted as x_i.

The algorithm progresses through iterations until it reaches a predetermined stopping criterion. This criterion could be based on several factors, such as the total number of iterations, variations in the fitness function, or a predefined threshold set by the user.

PSO's simplicity and effectiveness make it a popular choice for a range of optimization challenges. Its advantage lies in being a derivative-free method, simplifying its implementation and making it suitable for problems with discontinuous or non-convex fitness functions. As a population-based algorithm, PSO has the added benefit of potentially avoiding entrapment in local minima, a common issue in optimization problems.

Majority Voting

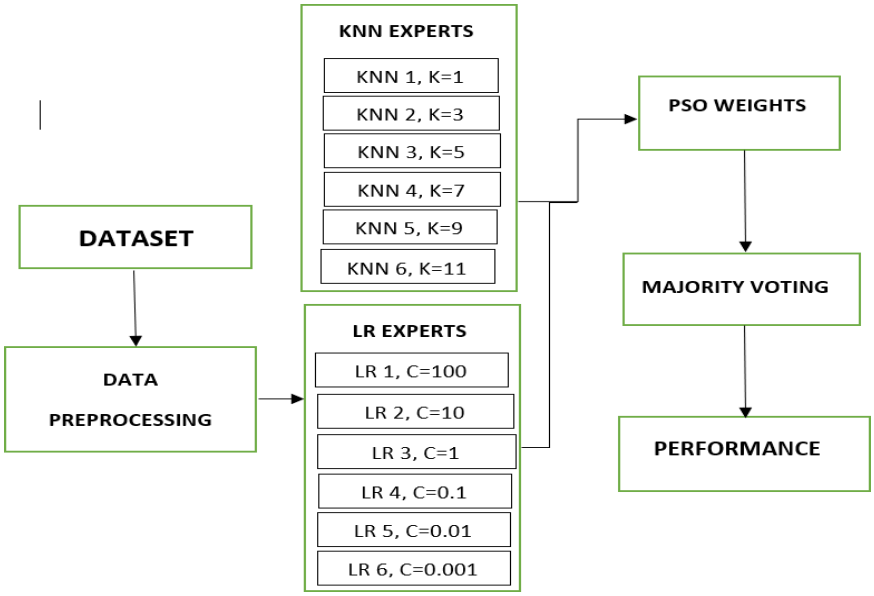


Figure 6. The Flow Chart for Ensemble Models

In our study, we explored the concept of weighted average ensembles, which differ from traditional voting classifiers by not treating all models as equally effective. Instead, in a weighted average ensemble, each model's contribution to the final prediction is adjusted based on its reliability or performance on a separate validation dataset. This approach aims to enhance the overall accuracy of the ensemble.

In this specific project, we implemented a weighted average ensemble method that integrated predictions from two classifiers: K-nearest neighbors (KNN) and logistic regression. We diversified our approach by training each classifier under six different parameter settings. This process was iteratively repeated with various

parameter combinations, ensuring a wide range of diversity in the classifiers, which potentially improved their collective performance.

To illustrate, in the case of the KNN-logistic regression ensemble, we developed six different KNN models, each with a unique K value. These were then paired with six logistic regression models, each having distinct C values, resulting in 36 unique parameter combinations. These combinations were utilized to train our ensemble model.

The ensemble model's performance was evaluated, and it demonstrated a higher accuracy compared to the individual performance of either KNN or logistic regression models. This outcome suggests that the weighted average ensemble method can be an effective strategy for enhancing the accuracy of intrusion detection systems.

Stacking

In our research, we employed stacking as a method to amalgamate various classifiers to enhance their collective performance. Stacking stands apart from other ensemble techniques like bagging and boosting due to its versatility in combining different types of classifiers, which can range from decision trees and neural networks to naive Bayes and logistic regression.

The implementation of stacking occurs in two phases. Initially, the base learners are trained using the training dataset. Following this, their predictions are compiled to form a new dataset. This new dataset is then utilized to train a secondary model, known as the meta-learner. The meta-learner's role is to make final predictions on the test dataset. Selecting appropriate base learners is crucial in stacking. For our project, we opted for base learners that are commonly referenced in intrusion detection literature. We then categorized these learners as either weak or strong based on their individual performance metrics. In all our stacking models, we used logistic regression as the final estimator. Despite its simplicity, logistic regression is a powerful classifier, particularly suited for classification tasks.

Our findings indicated that the stacking algorithm outperformed the individual base learners in terms of accuracy. This underscores the potential of stacking as an effective approach to bolster the accuracy of intrusion detection systems.

5. Evaluation

Evaluating the performance of a model is a critical step in the model development lifecycle. It helps in determining the model's effectiveness and guides necessary modifications. In our study, we employed several key metrics to evaluate our model:

1. Precision: This metric assesses the accuracy of positive predictions. Precision is the ratio of true positive predictions (TP) to the total number of positive predictions made (TP + FP). A higher precision indicates that the model is more accurate in identifying attacks, with fewer instances falsely identified as attacks (lower false positives).

2. Recall: Recall measures the model's ability to correctly identify actual positives. It is the ratio of positive class predictions made to the total number of actual positive cases in the dataset. A model with high recall is more reliable in labeling actual attacks as such, making the system more secure by increasing the likelihood of detecting attacks.

3. F1-score: The F1-score combines precision and recall, providing a single measure of a model's accuracy. It is the harmonic mean of precision and recall, offering a balanced view of both metrics.

Additionally, we utilized the confusion matrix as an insightful tool for understanding our model's performance. This matrix outlines the true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) of the model. The confusion matrix is instrumental in calculating precision, recall, and the F1-score.

For this paper, we focused on the F1-score to gauge the overall performance of our model. The F1-score's balanced consideration of both precision and recall makes it an excellent metric for evaluating the effectiveness of the model in a comprehensive manner.

6. Results

In our research, we employed a variety of metrics to evaluate the efficacy of our models, particularly focusing on the significance of the F1 score. While accuracy is a frequently used metric in model assessment, it may not always be suitable, especially in the context of imbalanced datasets. The reason is that a high accuracy rate can be misleading, indicating good performance even when the model poorly predicts outcomes for the minority class.

The F1 score emerges as a more reliable metric in such scenarios. It incorporates both precision and recall, effectively measuring a model's ability to correctly identify cases in both minority and majority classes. This dual consideration makes the F1 score a more comprehensive and balanced measure of model performance, especially in datasets where class imbalance is a concern.

For the scope of our project, the F1 score was chosen as the primary evaluation metric. We believe that it provides a more accurate reflection of our models' performance across various datasets, particularly those with imbalanced class distributions. This approach ensures a more nuanced and realistic assessment of model effectiveness.

ROC plots for Base Learners

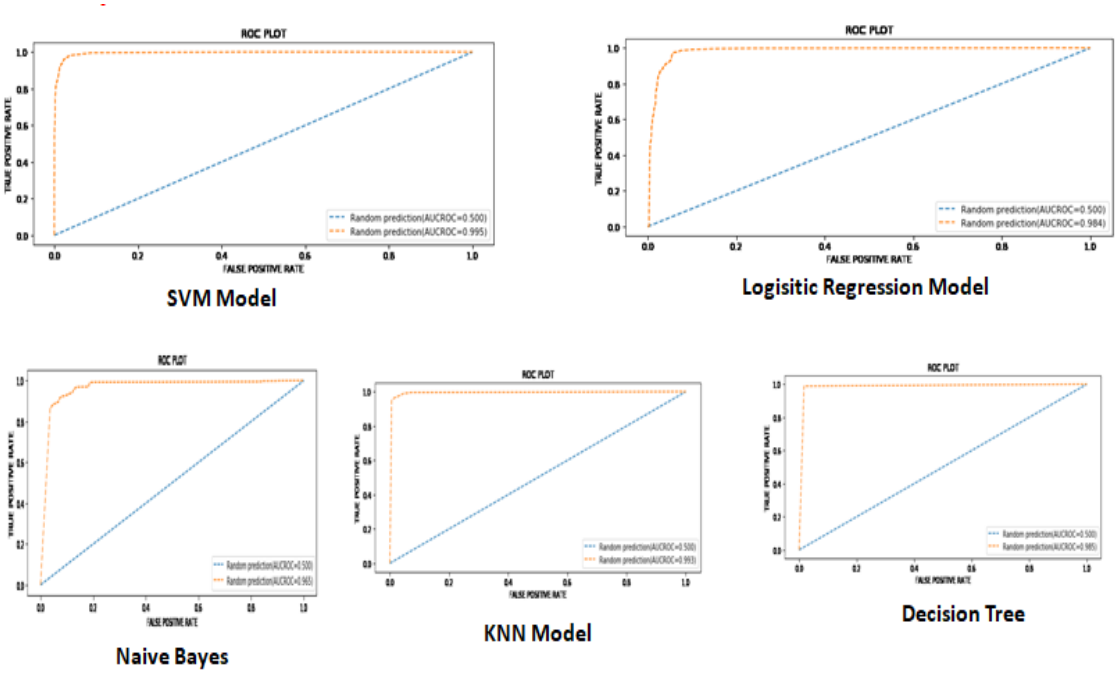
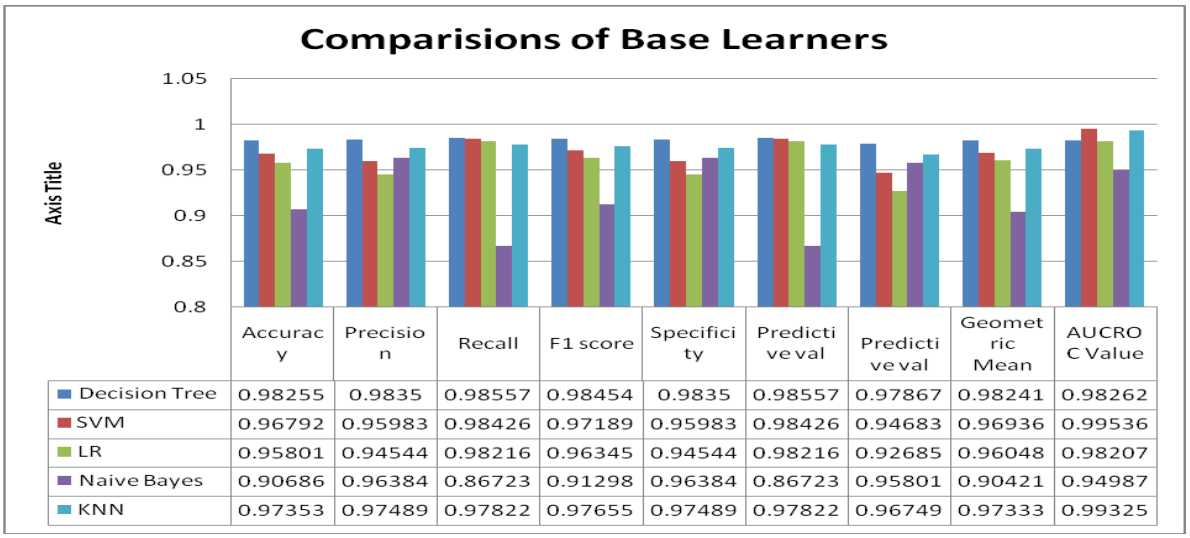


Figure 7. ROC Curves of various models

Results of Base learners



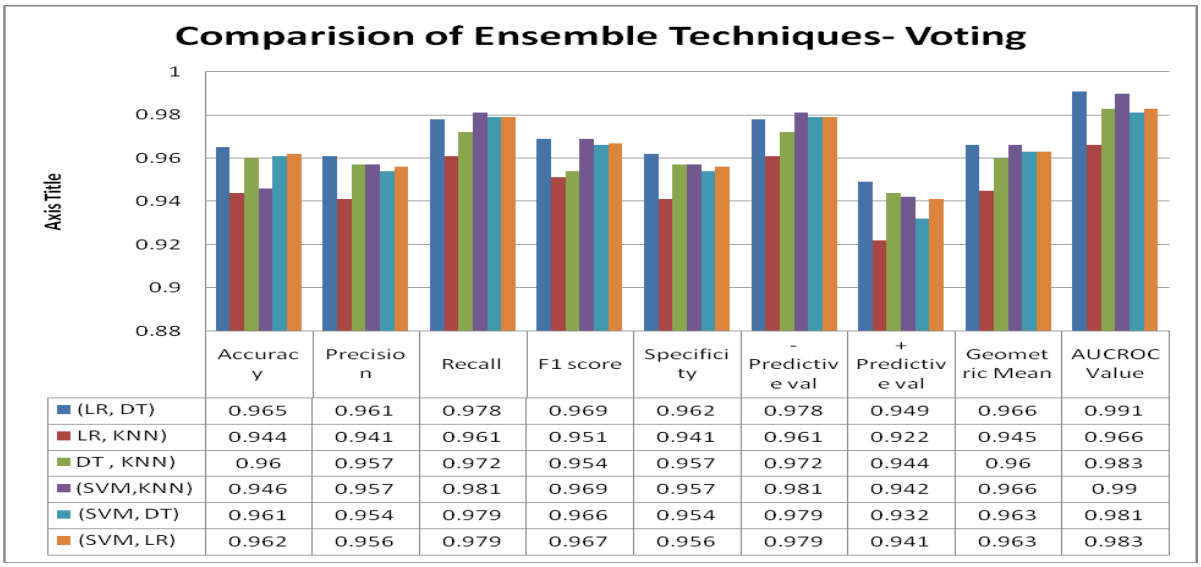
	Accuracy	Precision	Recall	F1 score	Specificity	Predictive val	Predictive val	Geometric Mean	AUCROC Value
Decision Tree	0.98255	0.9835	0.98557	0.98454	0.9835	0.98557	0.97867	0.98241	0.98262
SVM	0.96792	0.95983	0.98426	0.97189	0.95983	0.98426	0.94683	0.96936	0.99536
LR	0.95801	0.94544	0.98216	0.96345	0.94544	0.98216	0.92685	0.96048	0.98207
Naive Bayes	0.90686	0.96384	0.86723	0.91298	0.96384	0.86723	0.95801	0.90421	0.94987
KNN	0.97353	0.97489	0.97822	0.97655	0.97489	0.97822	0.96749	0.97333	0.99325

Our analysis involved evaluating the base classifiers and categorizing them as either weak or strong based on their performance. Subsequently, we utilized these classifiers to construct ensemble models through methods such as voting and stacking.

The performance of various ensemble models, constructed using the voting technique, is detailed in Table 2. According to the table, ensembles incorporating strong classifiers yielded the most favorable F1 scores, indicating that the inclusion of strong classifiers is beneficial for enhancing an ensemble model's overall effectiveness.

Among all the ensemble models we tested, the standout was the stacking ensemble model, which employed a Decision Tree as the meta-learner. This model attained an F1 score of 0.984, the highest amongst all the models we evaluated.

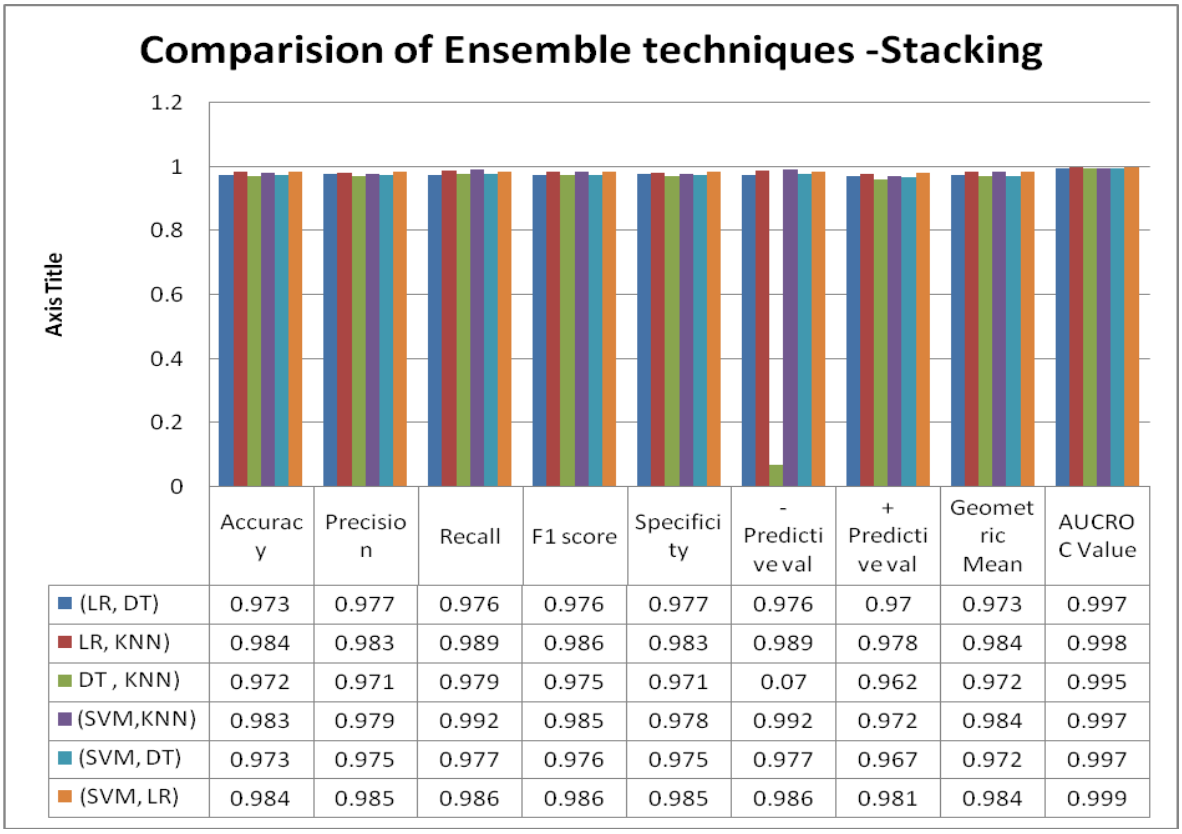
In summary, the outcomes of our experiments underscore the potential of ensemble models in boosting the efficiency of intrusion detection systems. This approach, especially when leveraging strong classifiers, appears to significantly improve performance metrics.



	Accuracy	Precision	Recall	F1 score	Specificity	- Predictive val	+ Predictive val	Geometric Mean	AUCROC Value
(LR, DT)	0.965	0.961	0.978	0.969	0.962	0.978	0.949	0.966	0.991
LR, KNN)	0.944	0.941	0.961	0.951	0.941	0.961	0.922	0.945	0.966
DT , KNN)	0.96	0.957	0.972	0.954	0.957	0.972	0.944	0.96	0.983
(SVM,KNN)	0.946	0.957	0.981	0.969	0.957	0.981	0.942	0.966	0.99
(SVM, DT)	0.961	0.954	0.979	0.966	0.954	0.979	0.932	0.963	0.981
(SVM, LR)	0.962	0.956	0.979	0.967	0.956	0.979	0.941	0.963	0.983

Analyzing the data presented in the table, it is evident that the combination of Support Vector Machines (SVM) and K-Nearest Neighbors (KNN) emerged as the top-performing ensemble model. This was closely followed by the pairs of Logistic Regression and Decision Tree, SVM and Logistic Regression, SVM and Decision Tree, Logistic Regression and KNN, and finally Decision Tree and KNN. The F1-score was the chosen metric for evaluating the effectiveness of these models.

It was also observed that the performance of the voting-based ensemble models did not surpass that of the baseline models. In addition to the voting method, we also explored stacking as another ensemble technique. Similar to our earlier approach, the F1-score was again utilized as the primary metric to evaluate the performance of these stacked models.



	Accuracy	Precision	Recall	F1 score	Specificity	- Predictive val	+ Predictive val	Geometric Mean	AUCROC Value
(LR, DT)	0.973	0.977	0.976	0.976	0.977	0.976	0.97	0.973	0.997
LR, KNN)	0.984	0.983	0.989	0.986	0.983	0.989	0.978	0.984	0.998
DT , KNN)	0.972	0.971	0.979	0.975	0.971	0.07	0.962	0.972	0.995
(SVM,KNN)	0.983	0.979	0.992	0.985	0.978	0.992	0.972	0.984	0.997
(SVM, DT)	0.973	0.975	0.977	0.976	0.975	0.977	0.967	0.972	0.997
(SVM, LR)	0.984	0.985	0.986	0.986	0.985	0.986	0.981	0.984	0.999

The analysis of our results revealed that the ensemble models combining Decision Tree and K-Nearest Neighbor (DT-KNN) and Support Vector Machine and Decision Tree (SVM-DT) achieved identical F1-scores. These were followed by the combinations of Decision Tree and Logistic Regression (DT-LR), Support Vector Machine and K-Nearest Neighbor (SVM-KNN), Logistic Regression and K-Nearest Neighbor (LR-KNN), and finally Support Vector Machine and Logistic Regression (SVM-LR).

Despite the equal F1-scores of DT-KNN and SVM-DT, we opted for DT-KNN as the superior model due to the greater complexity associated with the SVM-DT model. Moreover, we observed that the models created using stacking outperformed both the baseline models and those created using voting. This improved performance can be attributed to stacking's ability to leverage the individual strengths of each estimator by utilizing their outputs as inputs for the final estimator. In contrast, a voting classifier simply selects the most frequent output as the final decision.

In summary, our findings indicate that the DT-KNN ensemble model, with logistic regression serving as the meta-learner, stands out as the most effective model. It achieved the highest F1 score of 0.986 among all the tested ensemble models.

7. Conclusion

In our study, we focused on assessing the efficacy of ensemble learning models in the context of intrusion detection. We utilized a combination of five base learners to create pairs of classifiers, categorized as weak and strong. These pairs were then employed in two types of ensembles learning models: majority voting and stacking. Additionally, we applied particle swarm optimization (PSO) to fine-tune the weights within these ensemble models.

Our experimental findings revealed that ensembles based on stacking were superior in performance compared to those based on voting and the baseline models. Notably, the ensemble model combining decision trees with K-nearest neighbors achieved the highest F1 score, reaching 98.6%.

Looking ahead, our future research aims include exploring the realm of online ensemble learning to encompass a broader data range. We are also interested in adapting our model for data streams, particularly addressing the challenges posed by concept drift. Another area of focus will be enhancing feature selection processes to refine prediction accuracy. Moreover, we intend to integrate PSO with LUS optimization to achieve more optimal weights for the ensemble models.

We conclude that the outcomes of our research strongly support the potential of ensemble learning as an effective method for intrusion detection. The directions for future research we have identified hold promise for further advancements in the performance of ensemble learning models in this domain.

References

[1] H. Rajadurai and U. D. Gandhi, “A stacked ensemble learning model for intrusion detection in wireless network,” Neural Comput. Appl., May 2020, doi: 10.1007/s00521-020-04986-5.

[2] G. Kumar, K. Thakur, and M. R. Ayyagari, “MLEsIDSs: machine learning-based ensembles for intrusion detection systems—a review,” J. Supercomput., vol. 76, no. 11, pp. 8938–8971, Nov. 2020, doi: 10.1007/s11227- 020-03196-z.

[3] O. Sagi and L. Rokach, “Ensemble learning: A survey,” WIREs Data Min. Knowl. Discov., vol. 8, no. 4, Jul. 2018, doi: 10.1002/widm.1249.

[4] I. Abrar, Z. Ayub, F. Masoodi, and A. M. Bamhdi, “A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset,” in 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, Sep. 2020, pp. 919–924. doi: 10.1109/ICOSEC49089.2020.9215232.

[5] S. Seth, K. K. Chahal, and G. Singh, “A Novel Ensemble Framework for an Intelligent Intrusion Detection System,” IEEE Access, vol. 9, pp. 138451–138467, 2021, doi: 10.1109/ACCESS.2021.3116219.

[6] M. Govindarajan, “Evaluation of Ensemble Classifiers for Intrusion Detection,” vol. 10, no. 6, p. 9, 2016.

[7] B. S. Bhati and C. S. Rai, “Ensemble Based Approach for Intrusion Detection Using Extra Tree Classifier,” in Intelligent Computing in Engineering, vol. 1125, V. K. Solanki, M. K. Hoang, Z. (Joan) Lu, and P. K. Pattnaik, Eds. Singapore: Springer Singapore, 2020, pp. 213–220. doi: 10.1007/978-981-15-2780-7_25.

[8] N. T. Pham, E. Foo, S. Suriadi, H. Jeffrey, and H. F. M. Lahza, “Improving performance of intrusion detection system using ensemble methods and feature selection,” in Proceedings of the Australasian

Computer Science Week Multiconference, Brisband Queensland Australia, Jan. 2018, pp. 1–6. doi: 10.1145/3167918.3167951.

[9] M. Yousefnezhad, J. Hamidzadeh, and M. Aliannejadi, “Ensemble classification for intrusion detection via feature extraction based on deep Learning,” Soft Comput., vol. 25, no. 20, pp. 12667–12683, Oct. 2021, doi: 10.1007/s00500-021-06067-8.

[10] Y. Zhou, T. A. Mazzuchi, and S. Sarkani, “M-AdaBoost- A based ensemble system for network intrusion detection,” Expert Syst. Appl., vol. 162, p. 113864, Dec. 2020, doi: 10.1016/j.eswa.2020.113864.

[11] A. Zainal, M. A. Maarof, and S. M. Shamsuddin, “Ensemble Classifiers for Network Intrusion Detection System,” p. 10.

[12] N. N. P. Mkuzangwe, F. Nelwamondo, N. N. P. Mkuzangwe, and F. Nelwamondo, “Ensemble of classifiers based network intrusion detection system performance bound,” in 2017 4th International Conference on Systems and Informatics (ICSAI), Hangzhou, Nov. 2017, pp. 970–974. doi: 10.1109/ICSAI.2017.8248426.

[13] R. E. Schapire, “The Boosting Approach to Machine Learning: An Overview,” in Nonlinear Estimation and Classification, vol. 171, D. D. Denison, M. H. Hansen,

C. C. Holmes, B. Mallick, and B. Yu, Eds. New York, NY: Springer New York, 2003, pp. 149–171. doi: 10.1007/978-0-387-21579-2_9.

[14] Yan-Shi Dong and Ke-Song Han, “A comparison of several ensemble methods for text categorization,” in IEEE International Conference onServices Computing, 2004. (SCC 2004). Proceedings. 2004, Shanghai, China, 2004, pp. 419–422. doi: 10.1109/SCC.2004.1358033.

[15] T. G. Dietterich, “Machine-Learning Research,” p. 40.

[16] A. M. Bamhdi, I. Abrar, and F. Masoodi, “An ensemble based approach for effective intrusion detection using majority voting,” TELKOMNIKA Telecommun. Comput. Electron. Control, vol. 19, no. 2, p. 664, Apr. 2021, doi: 10.12928/telkomnika.v19i2.18325.