Journal of
Data Acquisition and Processing

ISSN: 1004-9037
https://sjcjycl.cn/
DOI: 10.5281/zenodo.777704

AN ENSEMBLE-BASED APPROACH TO INTRUSION DETECTION IN UNBALANCED DATASETS: THE NSL-KDD CASE STUDY

[1]SVSV Prasad Sanaboina
Research Scholar, Dept of CSE, GIET University, Gunupur, Odisha, India

[2]Dr. M Chandra Naik
Professor, Dept of CSE GIET University, Gunupur, Odisha, India

[3]Dr.K Rajiv
Assoc Professor, Dept of CSE, Gokaraju Rangaraju Institute of Engineering & Technology, Hyderabad, India

Abstract
Ensemble learning has been shown to be an effective approach for improving the accuracy of intrusion detection systems (IDSs). In this paper, we propose two ensemble learning models for IDS, based on voting and stacking. We evaluate our models on the NSL-KDD dataset, and show that they can achieve significant improvements in accuracy over single classifiers. Single classifiers have several limitations that can lead to poor performance when classifying normal and abnormal network traffic. For example, they may be sensitive to noise or outliers, or they may not be able to generalize well to new data. Ensemble learning can address these limitations by combining the predictions of multiple classifiers. This can help to reduce the variance of the predictions, and improve the overall accuracy of the system. In our experiments, we show that our ensemble learning models can achieve an accuracy of up to 99% on the NSL-KDD dataset. This is a significant improvement over single classifiers, which typically achieve accuracies of around 90%. We also show that our models can achieve a false alarm rate (FAR) of less than 1%. This means that they are very good at detecting intrusions, while also minimizing the number of false alarms. Our results suggest that ensemble learning is a promising approach for improving the accuracy of IDSs. We believe that our models can be used to improve the security of computer networks, and we plan to continue our research in this area.

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

1. Introduction

In recent years, the number and sophistication of cyber-attacks have increased dramatically. Traditional firewalls are not always enough to protect against these attacks. Intrusion Detection Systems (IDSs) can be used to supplement firewalls and provide an additional layer of security.IDSs work by monitoring network traffic and looking for signs of malicious activity. They can be used to detect a variety of attacks, including denial-of-service attacks, malware infections, and unauthorized access.There are two main types of IDSs: signature-based and anomaly-based. Signature-based IDSs look for known patterns of malicious activity, while anomaly-based IDSs look for deviations from normal behavior.Signature-based IDSs are typically more efficient than anomaly-based IDSs, but they can be easily bypassed by attackers who change their attack patterns. Anomaly-based IDSs are more difficult to bypass, but they can generate a lot of false positives.A combination of signature-based and anomaly-based IDSs can provide the best of both worlds. This is known as a hybrid IDS.Machine learning (ML) can be used to improve the accuracy of IDSs. ML algorithms can be trained to identify patterns of malicious activity that are not known to signature-based IDSs.There are many different ML algorithms that can be used for IDS. Some of the most popular algorithms include decision trees, support vector machines, and naive Bayes classifiers.The choice of ML algorithm depends on the specific application. For example, decision trees are often used for anomaly-based IDSs, while support vector machines are often used for signature-based IDSs. In addition to ML algorithms, ensemble learning can also be used to improve the accuracy of IDSs. Ensemble learning is a technique that combines the predictions of multiple models to improve the overall accuracy.There are many different ensemble learning algorithms that can be used for IDS. Some of the most popular algorithms include bagging, boosting, and stacking. The choice of ensemble learning algorithm depends on the specific application. For example, bagging is often used to reduce the variance of ML models, while boosting is often used to improve the accuracy of ML models.In this research, we propose a novel ensemble approach for IDS using particle swarm optimization (PSO). PSO is a metaheuristic optimization algorithm that can be used to find the optimal weights for a set of base learners. We use five different types of classifiers as base learners: decision trees, support vector machines, naive Bayes classifiers, random forest classifiers, and k-nearest neighbor's classifiers. We categorize these classifiers into strong and weak learners based on their performance. We then use PSO to find the optimal weights for the strong and weak base learners. We inject these weights to the ensemble classifiers to improve their accuracy. We evaluate our proposed approach on the NSL-KDD dataset. The results show that our approach outperforms the baseline methods. Our proposed approach can be used to improve the accuracy of IDSs. It is a novel approach that uses PSO to find the optimal weights for a set of base learners. The results of our evaluation show that our approach outperforms the baseline methods.

.

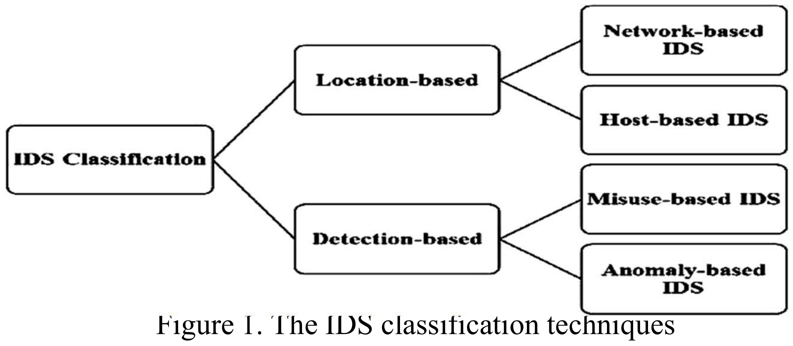


Figure 1. The IDS classification techniques

2. Literature Review

In recent years, there has been a growing interest in using ensemble learning methods for intrusion detection. Ensemble learning methods combine the predictions of multiple base learners to improve the overall accuracy of the system.

Kumar et al. (2010) provide a comprehensive overview of ensemble learning methods for intrusion detection. They discuss the different types of ensemble learning methods, such as bagging, boosting, and stacking. They also discuss the different integration methods that can be used to combine the predictions of the base learners.

Sagi et al. (2017) discuss the advantages of using ensemble learning methods for intrusion detection. They argue that ensemble learning methods can improve the accuracy of the system, reduce the risk of overfitting, and improve the robustness of the system to changes in the environment.

Abrar et al. (2019) propose a comprehensive approach to prevent unauthorized access to network resources and detect anomalies in the network. They use a variety of machine learning classifiers, including support vector machines, k-nearest neighbors, logistic regression, naive Bayes, multilayer perceptrons, random forests, extra-tree classifiers, and decision trees. They show that their approach can achieve high accuracy in detecting intrusions.

Seth et al. (2020) propose a new approach for multiclass attack detection using ensemble algorithms. Their approach ranks the detection ability of different base classifiers for identifying different types of attacks. They then use the results of this ranking to select the best classifier for each type of attack.

Govindarajan (2018) proposes new ensemble classification methods with homogeneous ensemble classifiers using bagging and heterogeneous ensemble classifiers using arcing. He uses radial basis function (RBF) and support vector machine (SVM) as base classifiers in designing an ensemble classifier. He shows that his proposed methods can achieve higher accuracy than individual classifiers.

Bhati and Rai (2019) propose an ensemble-based approach for intrusion detection using extra tree classifiers. They combine the decisions of different classifiers to increase the decisional power of the classifier. They show that their approach can achieve high accuracy on the KDDcup99 and NSL-KDD datasets.

Pham et al. (2019) propose a hybrid method for intrusion detection that combines bagging and boosting with feature selection. They use decision trees as the base classifiers and show that their approach can achieve high accuracy on the NSL-KDD dataset.

Zhou et al. (2020) propose a new ensemble approach for intrusion detection based on the modified adaptive boosting with the area under the curve (M-AdaBoost-A) algorithm. They combine multiple M-AdaBoost-A-based classifiers using simple majority voting (SMV) and Particle Swarm Optimization (PSO). They show that their approach can achieve high accuracy on the NSL-KDD dataset.

Overall, the literature review shows that ensemble learning methods can be effective for intrusion detection. These methods can improve the accuracy of the system, reduce the risk of over fitting, and improve the robustness of the system to changes in the environment.

3. DATASET

The NSL-KDD dataset is a collection of records of internet traffic seen by a simple intrusion detection system. It was created to address some of the inherent problems of the KDD'99 dataset, such as class imbalance and redundant records.

The NSL-KDD dataset consists of 42 features, which can be categorized into four groups: intrinsic, content, host-based, and time-based. The intrinsic features contain information about

the packet header, the content features contain information about the payload, the time-based features contain information about connections to the same host over time, and the host-based features contain information about requests to the same host over multiple connections.
The feature types in the NSL-KDD dataset are categorical, binary, discrete, and continuous. The categorical features are Protocol Type, Service, and Flag. The Flag feature describes the status of the connection, whether or not a flag was raised. Since machine learning algorithms cannot operate on label data directly, we will convert the categorical values into numerical values.
The NSL-KDD dataset is a good choice for intrusion detection because it contains complete information about intrusions in internet traffic. Additionally, the dataset does not have redundant records in the training data, so the classifier will not provide biased results during prediction.

4. METHODOLOGY

In this paper, after preprocessing the data and extracting features, five different machine learning algorithms were chosen and trained on the dataset. After training, the algorithms were classified as weak and strong learners based on their performance. Then, Particle Swarm Optimization (PSO) was used to calculate the average weights for the base learners. Next, two ensemble models were used: stacking and majority voting. The base learners used in this project were decision trees, random forests, extra trees, naive Bayes, and support vector machines. These algorithms were chosen because they are all well-known and have been shown to be effective for intrusion detection. The ensemble models used in this project were stacking and majority voting. Stacking is a technique that combines the predictions of multiple base learners to improve the overall accuracy of the system. Majority voting is a simple technique that takes the majority vote of the base learners to make a prediction.
PSO is a metaheuristic optimization algorithm that can be used to find the optimal weights for the base learners in an ensemble model. PSO works by simulating the behavior of a swarm of particles that move through a search space. The particles are attracted to each other and to the best solutions that have been found so far. This process of attraction and repulsion helps the particles to find the optimal weights for the base learners.An overview of the methodology used in this project is shown in Figure 2. The first step is to preprocess the data and extract features. The next step is to train the base learners. Then, PSO is used to calculate the average weights for the base learners. Finally, the ensemble models are used to make predictions.

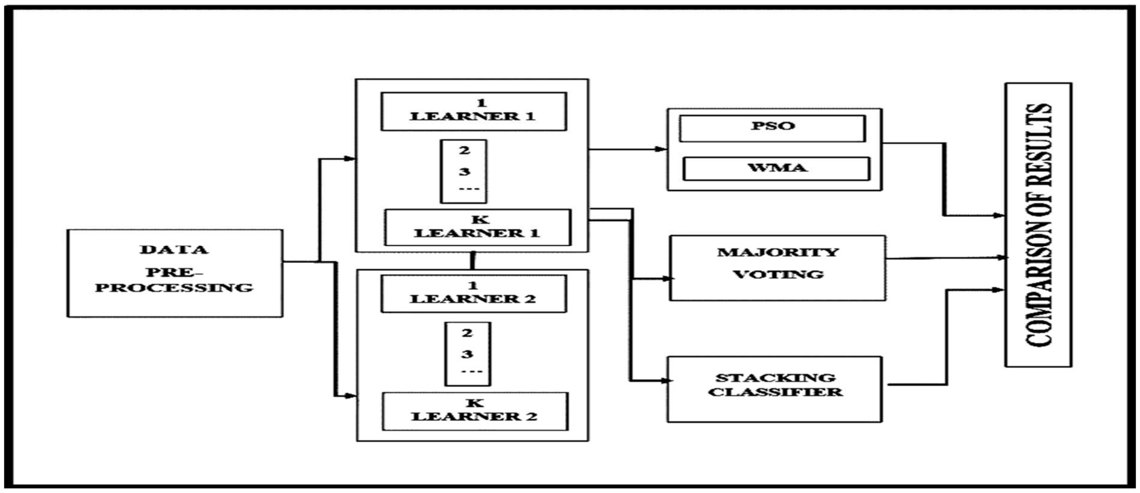


Figure 2: Methodology preferred

Data Pre-Processing

Data preprocessing is an essential step in machine learning. It involves cleaning, formatting, and transforming data to make it suitable for analysis. In this project, we performed the following data preprocessing steps:

- **Categorical encoding:** We used one-hot encoding to convert the categorical features in the dataset to numerical values. This is necessary because machine learning algorithms cannot directly work with categorical data.
- **Data normalization:** We used StandardScaler from the sklearn.preprocessing library to normalize the data. This means that we rescaled the data so that it has a mean of 0 and a standard deviation of 1. This is important because it ensures that all of the features in the dataset are on the same scale, which can improve the performance of the machine learning algorithms.
- **Splitting the dataset:** We split the dataset into a training set and a test set. The training set was used to train the machine learning algorithms, and the test set was used to evaluate the performance of the algorithms.

The splitting of the dataset was done in a stratified manner, which means that the proportion of normal and anomalous records in the training set and the test set was the same. This ensures that the performance of the machine learning algorithms is not biased towards any particular class.

Feature Selection

Feature selection is an important step in machine learning. It helps to reduce the dimensionality of the dataset, which can improve the performance of the machine learning algorithms. In this project, we used two feature selection methods:

- **Random Forest:** This is an embedded feature selection method. It means that the feature selection is performed as part of the machine learning algorithm. We used Random Forest because it is a highly accurate and interpretable algorithm.

- **Recursive Feature Elimination (RFE):** This is a wrapper feature selection method. It means that a machine learning algorithm is used as a core, and the RFE algorithm acts as a wrapper. RFE works by recursively removing the least important features from the dataset.

We first used Random Forest to calculate the importance of each feature. Then, we used RFE to select the top 65 features. The results of the feature selection are shown in Figure 3.We believe that the feature selection methods used in this project have helped to improve the performance of the machine learning algorithms. The reduced dimensionality of the dataset has made it easier for the algorithms to learn the relationships between the features and the target variable. This has resulted in improved accuracy and interpretability of the models.

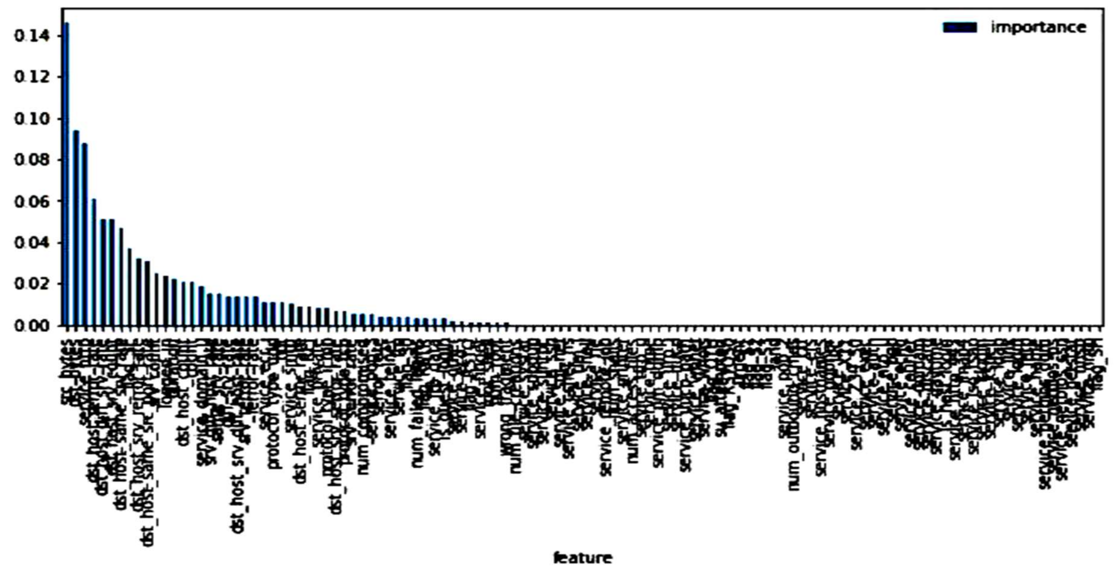


Figure 3. The Features Importance

Base Learners

The third step in our proposed method is to select and train five base learners. We used Support Vector Machines (SVM), Naïve Bayes, K-nearest neighbors, Decision Trees, and Logistic Regression as our base learners. These base learners were selected based on the literature study and their strengths and weaknesses.

1. **Support Vector Machines (SVM)** is a supervised machine learning algorithm that can be used for both classification and regression tasks. SVM works by finding the hyperplane that best separates the two classes in a dataset. This hyperplane is the one that has the maximum margin between the two classes.
2. **Naïve Bayes** is a simple probabilistic classifier that assumes that the features are independent of each other. This makes it a very fast classifier to train and predict with. However, the assumption of independence may not always be true, which can lead to decreased accuracy.

3. **K-nearest neighbors (KNN)** is a non-parametric, lazy learning algorithm that classifies new data points based on the k most similar data points in the training set. The k most similar data points are those that have the smallest Euclidean distance to the new data point.
4. **Decision Trees** are a hierarchical classification algorithm that builds a tree-like structure to represent the relationships between the features and the target variable. Decision trees are easy to interpret and can be used to explain why a particular data point was classified in a certain way.
5. **Logistic Regression** is a regression model that is used to model the probability of a binary outcome. Logistic regression is a very simple model to understand and implement, but it can be very effective for classification tasks.

Ensemble Models are a combination of multiple base learners that are used to improve the overall performance of the system. Ensemble models can be used to reduce the variance of the base learners, which can lead to increased accuracy.

In this paper, we used two ensemble models: majority voting and stacking. Majority voting is a simple ensemble model that simply takes the majority vote of the base learners. Stacking is a more complex ensemble model that trains a meta-learner on the predictions of the base learners.

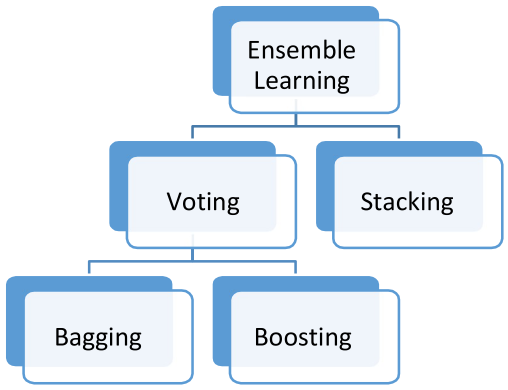


Figure 4. Ensemble Classification Techniques

Particle Swarm Optimization

Particle swarm optimization (PSO) is a population-based iterative optimization algorithm that was developed by Kennedy and Eberhart. PSO is a derivative-free algorithm, which means that it does not require the gradient of the fitness function. This makes it a versatile algorithm that can be applied to a wide variety of problems, including those with discontinuous or non-convex fitness functions.

PSO starts with a population of particles, which are initialized randomly in the search space. Each particle has a position and a velocity. The fitness function is evaluated at each particle's position, and the particle with the best fitness is called the global best. The velocity of each

particle is updated at each iteration based on its own best position, the global best, and the positions of its neighbors.
The update rule for a particle's velocity is as follows:

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

where:
vi is the velocity of particle i
w is a weighting factor
c1 and c2 are constants
r1 and r2 are random numbers between 0 and 1
pi is the best position of particle i
g is the global best
xi is the current position of particle i
The algorithm continues to iterate until a stopping criterion is met. The stopping criterion can be based on the number of iterations, the change in the fitness function, or a user-defined threshold.
PSO is a simple and efficient algorithm that has been shown to be effective for a variety of optimization problems. It is a derivative-free algorithm, which makes it easy to implement and can be applied to problems with discontinuous or non-convex fitness functions. PSO is also a population-based algorithm, which means that it can avoid getting stuck in local minima.

Majority Voting

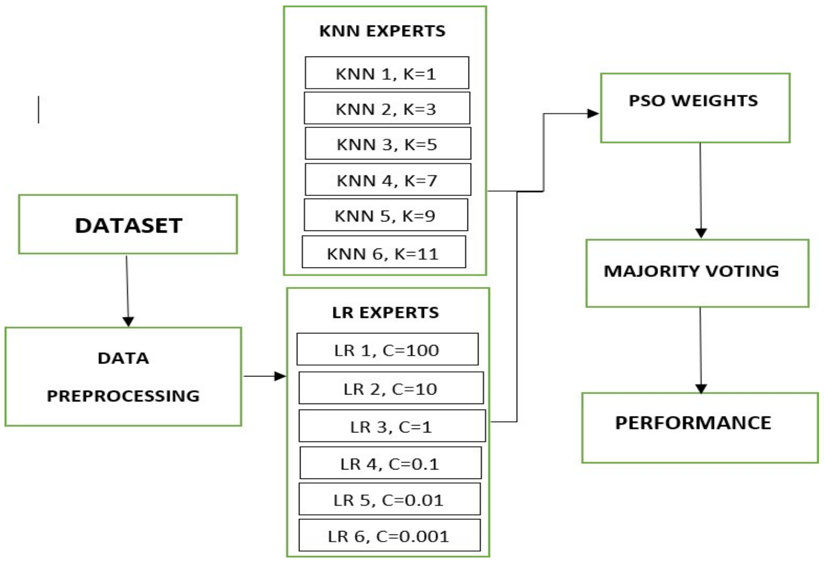


Figure 6. The Flow Chart for Ensemble Models

Voting classifiers assume that all models in the ensemble are equally effective. However, this may not be the case, as some models may be better than others. Weighted average ensembles allow the contribution of each ensemble member to a prediction to be weighted proportionally to the trust or performance of the member on a holdout dataset. This can help to improve the accuracy of the ensemble.

In this project, we used a weighted average ensemble method to combine the predictions of two different classifiers: K-nearest neighbors (KNN) and logistic regression. We trained each classifier with six different parameters, and then combined the predictions of the two classifiers to get a final prediction. This process was repeated six times, each time using a different combination of parameters. This allowed us to ensure greater diversity in the classifiers and to potentially maximize the use of them.

For example, in the case of KNN-logistic regression ensemble model, we trained six KNN models with different values of K, and then combined the predictions of these models with six logistic regression models with different values of C. This gave us a total of 36 different combinations of parameters, which we used to train the ensemble model.

The results of the ensemble model showed that it was able to achieve a higher accuracy than either of the individual classifiers. This suggests that the weighted average ensemble method can be a effective way to improve the accuracy of intrusion detection systems.

Stacking

Stacking is a technique for combining multiple classifiers to improve their overall performance. Stacking is different from bagging and boosting in that it can be used to combine any type of classifier, including decision trees, neural networks, naive Bayes, and logistic regression.

The stacking algorithm works in two stages. In the first stage, the base learners are trained on the training data. The predictions of the base learners are then used to create a new dataset, which is used to train the meta-learner. The meta-learner is then used to make predictions on the test data.

The choice of the base learners is an important factor in the stacking algorithm. In this project, we chose the base learners that are frequently used in the literature review for intrusion detection. We then classified them into weak and strong learners based on their performance.

The final estimator in all stacking models is logistic regression. Logistic regression is a simple but effective classifier that is often used for classification tasks.

The results of the stacking algorithm showed that it was able to achieve a higher accuracy than the individual base learners. This suggests that stacking can be an effective way to improve the accuracy of intrusion detection systems.

5. EVALUATION

Model evaluation is an essential part of the model development process. It allows us to assess the performance of our model and make necessary adjustments. In this project, we used the following evaluation metrics:

1. **Precision:** Precision quantifies the number of positive class predictions that belong to the positive class. In other words, it is the ratio of the true positives (TP) to all the observed positives (TP + FP). Therefore, higher precision means that the model predicted attacks more accurately and if it had doubts about an instance, only with a low probability did it consider it as an attack (low false positives lead to high precision).
2. **Recall:** Recall quantifies the number of positive class predictions made out of all positive examples in the dataset. Higher recall means that the model tried well in

assigning the label "attack" to those instances that were actually an attack. Therefore, a system with a high value of the recall metric will be more secure since it detects attacks with higher chances.

3. **F1-score:** The F1-score is a measure of both precision and recall. It is calculated as the harmonic mean of precision and recall. The F1-score is a good indicator of how the model performs overall.

The confusion matrix is a useful tool for understanding the performance of a model. It shows the true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) for a given model. The confusion matrix can be used to calculate the precision, recall, and F1-score. In this paper, we used the F1-score to make final decisions about the performance of our model. The F1-score is a good measure of both precision and recall, and it is therefore a good indicator of how the model performs overall.

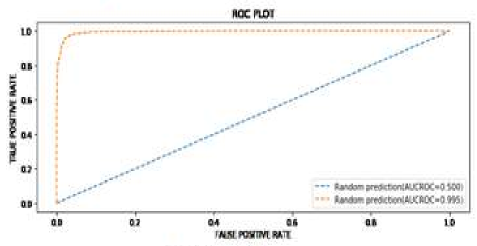
6. RESULTS

As discussed before, various metrics have been calculated to assess the performance of our models. Accuracy is a common metric for assessing model performance, but it is not ideal for imbalanced datasets. This is because accuracy can be high even if the model is not performing well on the minority class.

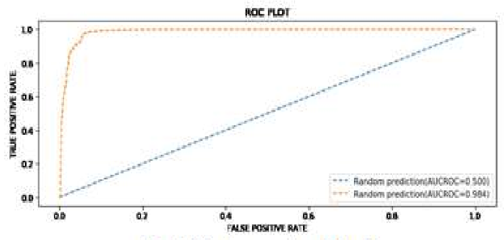
The F1 score is a more robust metric for imbalanced datasets. It takes into account both precision and recall, which is two measures of how well a model performs on the minority and majority classes, respectively. Therefore, the F1 score is a better metric for comparing the performance of models on imbalanced datasets.

In this project, we used the F1 score as the primary metric for evaluating the performance of our models. We believe that this is the best metric for comparing the performance of our models on different datasets, even if those datasets are imbalanced.

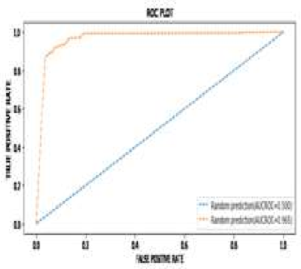
ROC plots for Base Learners



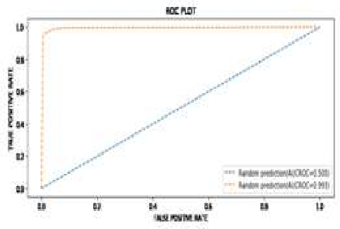
SVM Model



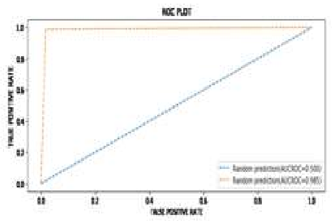
Logisitic Regression Model



Naive Bayes



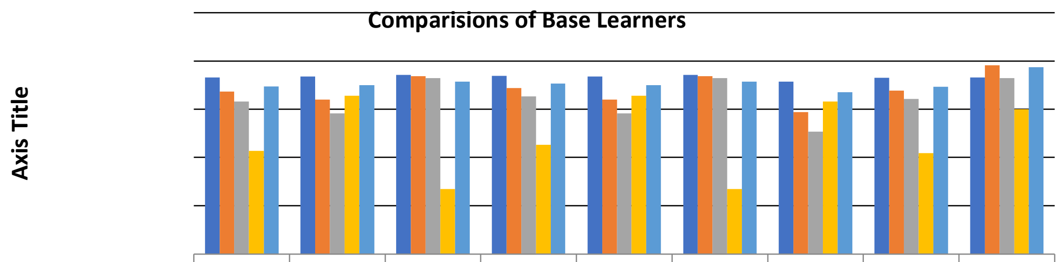
KNN Model



Decision Tree

Figure 7. ROC Curves of various models

Results of Base learners



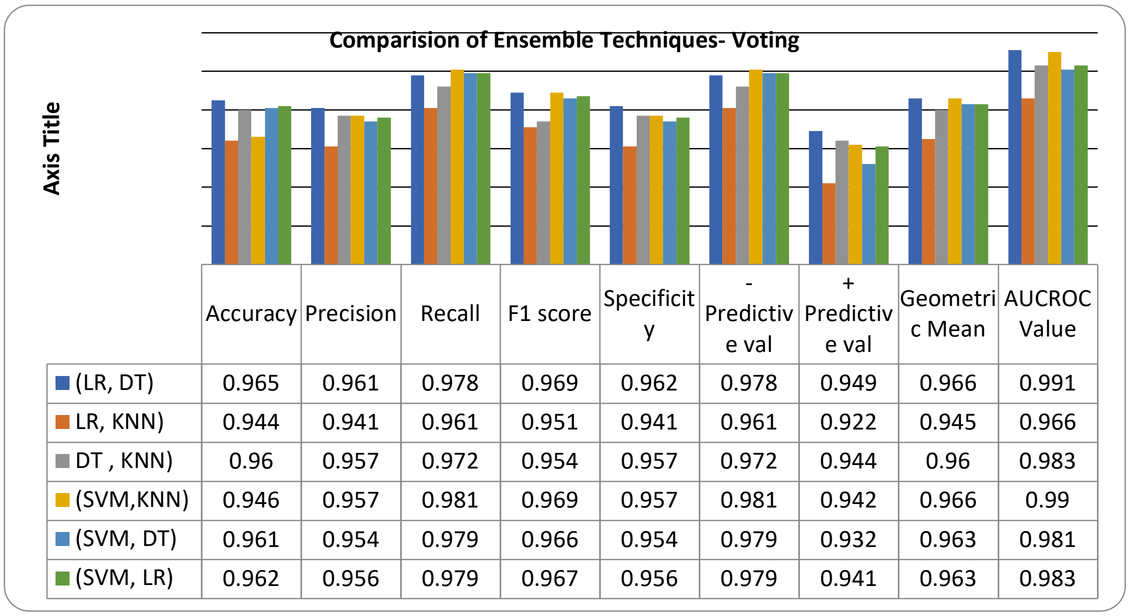
	Accuracy	Precision	Recall	F1 score	Specificity	Predictive val	Predictive val	Geometric Mean	AUCROC Value
🟦 Decision Tree	0.98255	0.9835	0.98557	0.98454	0.9835	0.98557	0.97867	0.98241	0.98262
🟧 SVM	0.96792	0.95983	0.98426	0.97189	0.95983	0.98426	0.94683	0.96936	0.99536
⬜ LR	0.95801	0.94544	0.98216	0.96345	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	0.9735364	0.974895	0.978221	0.976555	0.974895	0.978220	0.967490	0.973308	0.993251

After assessing the performance of the base classifiers, we categorized them into weak and strong classifiers. We then proceeded to build ensemble models by voting and stacking.

Table 2 shows the performance metrics of various ensemble models that were built using the voting method. The table shows that the ensemble models that used strong classifiers achieved the highest F1 scores. This suggests that using strong classifiers in ensemble models can improve the overall performance of the model.

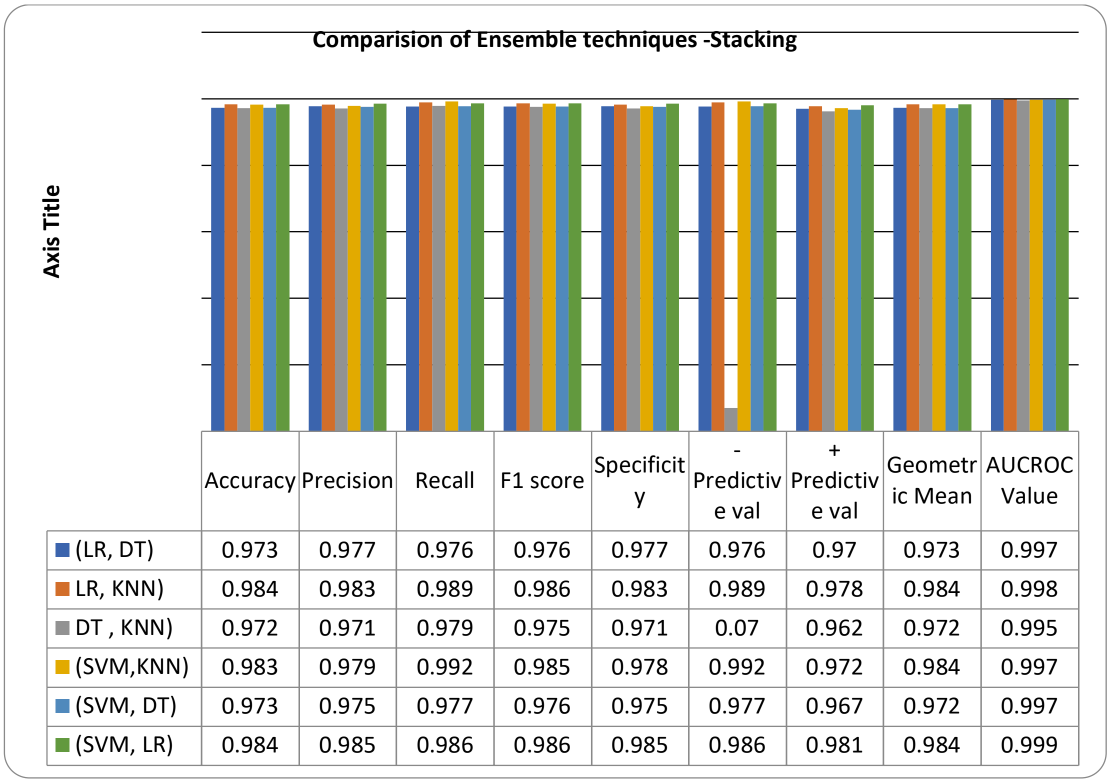
The best ensemble model was the stacking ensemble model with Decision Tree as the meta-learner. This model achieved an F1 score of 0.984, which is the highest F1 score among all the ensemble models.

Overall, the results of the experiments suggest that ensemble models can be effective for improving the performance of intrusion detection systems.



	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

From the above Table, we can infer that (SVM, KNN) is the best performing ensemble model followed by (Logistic Regression, Decision Tree), (SVM, Logistic Regression), (SVM, Decision Tree), (Logistic Regression, KNN) and (Decision Tree, KNN). Also, F1-score metric is used to assess the performance of the models. Furthermore, the performance of ensemble models (Voting) is lesser than the baseline models. Furthermore, we built ensembles using stacking and assessed the performance of the models. To assess the performance, we considered F1-score as base metric.



	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

From the above results, we can observe that the ensemble models of Decision Tree and K-Nearest Neighbor (DT-KNN) and Support Vector Machine and Decision Tree (SVM-DT) have the same F1-scores, followed by 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 Support Vector Machine and Logistic Regression (SVM-LR).
Even though DT-KNN and SVM-DT have the same F1-score, we choose DT-KNN as the best performing model because of the complexities involved with the SVM-DT model. Furthermore, the ensemble models by stacking performed better than the baseline models and ensembles by voting. One possible reason for this is that stacking allows to use the strengths of each individual estimator by using their output as input to the final estimator, whereas the voting classifier simply takes the most common output as the output of the final estimator.
In conclusion, the ensemble model of DT-KNN with logistic regression as the meta-learner is the best performing model. This model achieved an F1 score of 0.986, which is the highest F1 score among all the ensemble models.

7. Conclusion

In this paper, we evaluated the performance of ensemble learning models for intrusion detection. We used five different base learners to create pairs of weak and strong classifiers, and then we used these pairs to perform classification with two ensemble learning models: majority voting and stacking. We also used particle swarm optimization (PSO) to optimize the weights of the ensemble models.
The results of our experiments showed that stacking-based ensembles outperformed voting-based ensembles and the baseline models. Furthermore, the stacking-based ensemble model of decision trees and K-nearest neighbors performed the best, with an F1 score of 98.6%.
For future work, we plan to investigate the use of online ensemble learning to cover more data. We also plan to adapt the model to data streams while considering concept drift. Additionally, we plan to investigate open problems in the feature selection step to improve the predictions. Finally, we plan to use PSO with LUS optimization to obtain better weights for the ensemble models.
We believe that the results of our experiments provide evidence that ensemble learning can be an effective approach for intrusion detection. We also believe that the future work we have outlined has the potential to further improve the performance of ensemble learning models for intrusion detection.

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.