International Journal of Engineering Trends and Technology Volume 70 Issue 3, 37-47, March, 2022
ISSN: 2231 – 5381 /doi:10.14445/22315381/IJETT-V70I3P205 © 2022 Seventh Sense Research Group®

Original Article

A Hybrid Machine Learning-Based Approach for Dynamic Identification of Network Attacks in Cloud Environment

Madhura S. Mulimani[1], Rashmi R. Rachh[2], Shridhar Allagi[3]

[1]*Research Scholar, Department of Computer Science, Visvesvaraya om Technological University, Belagavi, Karnataka, India*
[2]*Associate Professor, Department of Computer Science, Visvesvaraya Technological University, Belagavi, Karnataka, India*
[3]*Associate Professor, Department of Computer Science, KLE Institute of Technology, Visvesvaraya Technological University, Belagavi, Karnataka, India*

[1]madhurasm@gmail.com, [2]rashmirachh@gmail.com, [3]shridharallagi1@gmail.com

Abstract — *The exponential evolution of the network and development of various technologies has led to an exponential increase in connected devices and users. While users can conveniently leverage the network services and resources, the ever-increasing number of users and resources has escalated the extent of various types of attacks which may have detrimental consequences in the network when left undetected or untreated. Traditional cybersecurity systems that use static methods are inadequate in coming up with rules for emerging threats or zero-day attacks and also lack the scalability with the increasingly complex cyberspace landscape. In recent times, machine learning techniques have gained a lot of attraction due to their potential in learning and making accurate predictions. In the considered work, numerous machine learning techniques have been comprehensively explored to recognize different types of attacks as well as to classify them. In the experiments, base classifiers such as Logistic Regression, Decision-Tree, Neural Network, Naïve-Bayes, and Support Vector Machine have been used to identify the attacks. To further boost the model's performance, ensemble learning models have been used. Additionally, the unsupervised method has been used. The experiment has been conducted using the NSL-KDD dataset to provide a detailed performance analysis of various machine learning techniques.*

Keywords – *Cybersecurity, Ensemble Machine Learning, K-means, Machine learning, NSL-KDD Dataset.*

I. INTRODUCTION

With the tremendous growth and evolution of the Internet over the past several years, there have been many different technologies such as time-sharing computing, client-server computing, distributed computing, ubiquitous computing, grid computing, etc. Each of these technologies has its own advantages. As a result, there has been a tremendous upsurge in the count of computers and other devices in the networking domain. On the other hand, it has also led to a rise in the number of threats or outbreaks that violate the security of the network and damage the system resources. Though there are several antivirus tools to block or restrict the attacks from affecting the system, however, they lack the capability in providing security to the system, and as a consequence, millions of hosts on the network are hacked and compromised[1]. In any technology, security is the prime concern, as users are concerned about the security of their sensitive and confidential data such as credit card information, social security number, personal health records, etc. There is a high probability that attackers attempt to find opportunities to sneak into the network to access information about the users and hence, cause an attack on the system [2]. To counter these attacks and provide security against such attacks in the network is quite a challenging task, and it is gaining a lot of attention [3]. Hence, network-intrusion-detection systems (IDS) are being developed, which often make use of signature-based methods to search whether the attack encountered has a signature in the database. Depending on the detection method that the IDS uses, it is classified into signature-based detection and anomaly detection. Since there are new attacks being created very often, they may not be found in the database. In such a case, another type of method that detects intrusions, called anomaly detection, is used, which identifies and detects all those attacks that deviate from the normal behaviour and do not exist in the database. Based on where the attack detection occurs, the IDS are of two kinds, host-intrusion-detection systems (HIDS) and network-intrusion-detection systems (NIDS). The HIDS observes the activities of the operating system, and parallelly the incoming network traffic is analyzed by the NIDS[4].

Traditional defence tools like firewalls, antivirus, and gateways, fail to detect these attacks as they normally use signature-based techniques that can detect only the known attacks and not the previously unseen attacks. These attacks are the major security threats that Internet users face today as they grow in volume and variety at a high velocity [5]. To overcome this drawback and to detect and discover these attacks, static and dynamic methods of analysis are used [1].

A. Static Analysis

In this type of analysis, the file is not executed, and also, the source code of the malware can only be viewed or read to interpret the behavioural properties of the files. Though the static analysis can read all the behavioural malware, it is used rarely as it is time-consuming [6].

B. Dynamic Analysis

This analysis monitors the file when it is executed and interprets the behavioural properties of the file. It is much faster when compared to the static analysis and is usually run in the virtual sandboxed environment for several minutes, and its behaviour is traced. This analysis frequently uses system-call level monitoring. Dynamic analysis is time-consuming as the malware needs to observe for several minutes [5], [6].

Irrespective of whether the static or dynamic analysis is used to detect the malware, either the signature-based or behaviour-based analysis is used, depending on whether the analysis uses the attack signature database or not.

C. Signature-based Analysis

It is a static method that depends on predefined signatures like fingerprint-like MD5 and SHA1 hashes. The file is first analyzed by antivirus and then compared between the sample malware. If the signatures between the comparisons match, then the file is targeted as suspicious. The sample malware can just use the hash values to detect, and hence, this analysis is easier to use. However, the attackers gain a benefit from this issue as they can modify the signature, and once it is modified, the attack cannot be analyzed. Then, the analysis cannot detect more using the pure signature-based detection unless the signature was created [6]. In signature-based detection, the pattern is exposed, and this reduces the system expenses and also the performance time for malware prediction. However, these signature-based detection techniques overlook the feature collection [5].

D. Behaviour-based analysis: This type of analysis uses empirical data and does not require a signature to detect the attack. It observes the attack behaviour during execution and looks for deviations in the normal behaviour. The combination of the action increases the files' level of suspicion. In other words, it can detect events whose behaviour is different from normal behaviour [6].

It is imperative to detect these attacks and provide security to the systems because even with a single attack, the information and financial loss can be enormous. To facilitate the identification of the attack as malicious or not, the network transactions need to be analyzed. Since there is an enormous number of such network transactions, their analysis can incur too much time and effort and can as well be error-prone. With the usherance of machine learning techniques, various tasks of analysis have been made easy as they train the computer to learn the pattern of the activities in the network, and thus, can be trained to identify and detect the attacks in the network. In addition, the machine learning techniques can as well classify them as benign files or malicious [7].

Machine learning techniques used can be unsupervised, semi-supervised or even supervised. The supervised technique will allow an algorithm to learn through a supervisor/moderator and, thus, trains the model to classify the instances using the class membership of each training sample. The algorithm studies a function that correlates an input entry to an output label based on the given pairs of input-output combinations. It uses the training data, which is labelled and consists of a numerous set of training samples and infers the function from it. Supervised machine learning algorithms can be classification or regression types.

The unsupervised learning technique will analyze the raw datasets and generate the analytic insights or draw deductions from datasets that contain unlabeled data and create groups of instances. Unsupervised learning techniques group objects based upon the distance or similarity such that each instance in the group is highly similar to another instance within the same group but highly dissimilar to an instance in another group. Principal component analysis, K-means clustering, etc. are the most widely used unsupervised learning algorithms

Since the traditional machine learning methods do not successfully capture several characteristics and the fundamental structure of the imbalanced, noisy, high-dimensional data, ML problems take into account a single model to predict the outcome. However, there is no guarantee that the model would be the best predictor. Rather than relying on a single model and hoping it to yield the best prediction, ensemble methods are used, which use a combination of models and average the models to produce one final model. So, the better approach is to use Ensemble learning models, which integrates the fusion of data, model and mining for creating a unified framework. It first extracts the valid subset of features with varying transformations and then imparts the knowledge in multiple machine learning algorithms that produce weak predictive results. It uses the informative knowledge from the results thus obtained to discover knowledge and improve the prediction performance for which they use voting schemes in an adaptive way [8].

The paper has been structured as follows: Section II lists the related works in the network security and intrusion detection domains. Section III gives a description of the proposed methodology used for handling intrusions. Section IV provides the analysis of the results of experimentation carried out. Section V gives the conclusion of the experiment.

II. RELATED WORK

Several research experiments were carried out in intrusion detection systems and considered it a classification problem in which the network traffic is classified as an anomaly or normal. They have proposed various methodologies like Decision Tree, k-nearest neighbour, Support Vector Machine, etc. Other researchers have shown that ensemble models provide better performance when compared to a single classifier. Some others have used unsupervised learning techniques to create groups of attacks or normal network transactions.

Seraphim et al. [4] have experimented with Decision Tree, k-nearest neighbour, Logistic Regression, and Artificial Neural Network machine learning techniques to implement the intrusion detection system. In their experiment, the NSL-KDD dataset has been used for training. They have applied a two-level approach wherein they first compare the different supervised/unsupervised learning algorithms and then augment the results of level one for deep learning with an artificial neural network.

Ludwig [9] has used classification methods to analyze the activity of intrusive patterns within a computer network. He has employed the neural network ensemble method consisting of a deep neural network, an autoencoder, and extreme learning machine models for categorizing different attack types in the NSL-KDD dataset.

Gu et al. [10] have employed feature augmentation with SVM ensemble to propose a framework for intrusion detection. They implement logarithmic density with marginal ratios for transformation in the original feature set for obtaining the new subset of training data with enhanced quality. They use the SVM thus obtained to construct an intrusion detection model. Their framework achieves good and robust performance using ensemble learning rather than using a single model to improve the performance.

Hariharan et al. [11]designed the intrusion detection system that uses a stacked ensemble learning which combines Random Forest, SVM, and CARTmachine learning algorithms to classify the attacks in the NSL-KDD dataset. The proposed intrusion detection system achieves better accuracy compared to others.

Gao et al. [12] have constructed an adaptive ensemble learning model that employs the base classifiers like kNN, Decision-Tree, Random Forest, and DNN to improve the overall detection effect and their proposed method proved that the quality of data feature set plays a crucial role in identifying the detection effect.

Halim et al. [13] have addressed the problem of selecting features in intrusion detection and network security domains and hence, have proposed a genetic-algorithm-based feature selection method, using which they increase the classifiers' accuracy. They have tested the proposed method on the network traffic datasets such as UNSW-NB15, Bot-IoT, and CIRA-CIC-DOHBrw-2020.

Gu and Lu [14] have proposed an intrusion detection model in which they generate new, high-quality data by employing the Naïve Bayes technique for feature transformation. They used a support-vector-machine classifier on generated transformed data and tested for the proposed intrusion detection model on UNSW-NB15, NSL-KDD, CICIDS2017, and Kyoto 2006+ datasets and obtained promising detection rate with higher accuracy and lesser number of false alarm rates.

Yerriswamy and Murtugudde[15] have proposed a genetic-based enhanced grey wolf optimization algorithm to detect intrusions and experimented with the NSL-KDD dataset and have used the feature selection method to boost the proposed model's performance.

Kunal and Dua[16] have reduced the size of the feature vector using the ranker-based attribute evaluation method and built a machine learning model that is trained on attack patterns of intrusions. They have used an ensemble of Random Tree, j48graft, R.E.P Tree, IBk (kNN), and Random-Forest classifiers to evaluate the proposed intrusion detection model.

Shukla [17] has combined several homogeneous methods for feature selection and using an ensemble technique and selected the optimum subgroup of non-redundant and appropriate features. He has proposed an SVM-based intrusion detection system that uses the selected features and obtains good accuracy when tested with KDD-CUP 99 and NSL-KDD datasets.

Maniriho et al. [18] have used the most appropriate feature subset generated by the Gain Ratio Feature Evaluator (GRFE). They evaluated and compared the performance of the single machine learning technique such as kNN over an ensemble technique, as well as to detect intrusions in a computer network. They have analyzed the performance using NSL-KDD and UNSW-NB15datasets. In their previous work, they detected the intrusions in network traffic using the NSL-KDD dataset, Correlation Ranking Filter (CRF) feature selection method, and GRFE. They have extended the previous work and combined it with several other machine learning techniques.

Masoodi et al. [19] have employed machine learning classifiers for classifying the data in the NSL-KDD dataset also determined the most suitable algorithm for detecting each type of attacks, such as DoS, Probe, R2L, and U2R, which can then be used to avoid unauthorized access to the network resources.

Allagi and Rachh[20] have proposed a two-level framework and availed the NSL-KDD dataset to detect unknown and unseen attacks as well as to analyze them. In the first level, they have detected the known attacks using supervised learning techniques such as Neural Network and Support Vector Machine. While, in the second level, they group the data as normal or anomaly using k-means clustering.

Rajagopal et al. [21] have suggested a hybrid multimodel solution to overcome the ineffectiveness of traditional machine learning algorithms used for building intrusion detection systems. Hence, they have developed an ensemble model that uses a meta classification approach with stacked generalization. They have carried out the experiment on two different datasets captured in an emulated and real-time network traffic environment, namely the packet-based dataset, UNSW-NB15 and flow-based dataset, UGR'16. Their experiment shows that the stacking ensemble performs better predictions with the real-time dataset than with the emulated one.

Vinutha and Poornima [22] have leveraged canopy clusters to overcome the drawbacks of K-means clustering as they work like pre-clusters for the K-means algorithm. This approach improves the accuracy as well as detection rate since they use Manhattan and Euclidean distance metrics. This decreases the number of erroneous classification instances when the K-means clustering is applied.

Abdulhammed et al. [23] have devised a machine learning intrusion detection system in which they reduce the feature dimensionality using the autoencoders and principal component analysis and then used the reduced number of features to build different classifiers using the CICIDS2017 dataset. Their experiment shows that using features with low dimensions for binary with multi-class classification aids in obtaining a higher detection rate, accuracy, F-measure, and false alarm rate.

Abirami et al. [24] have reduced the dimensions of the features using the feature selection method and then built an intrusion detection system that they have tested on KDD CUP 99, Kyoto 2006, and NSL-KDDdatasets. To keep the intrusion detection system adaptable to novel attacks and make it less costly, they have applied the ensemble learning algorithm on the UNSW-NB-15 dataset using the logistic regression stacking classifier as the meta-classifier that combines algorithms like Support Vector Machine, Random Forest, and Naïve Bayes.

III. PROPOSED METHODOLOGY

In this part, the proposed framework used in the experiment has been discussed. The network interaction activities occurring in the network may be normal or anomalous. The anomalous activities can have detrimental effects on the system or the network when left undetected or untreated. Hence, it is important to identify and classify the activities as normal or anomaly and thereby protect the system against such malicious attacks. The experiment uses the NSL-KDD dataset to comprehensively explore different machine learning algorithms, and the proposed framework is as depicted in Fig. 1.

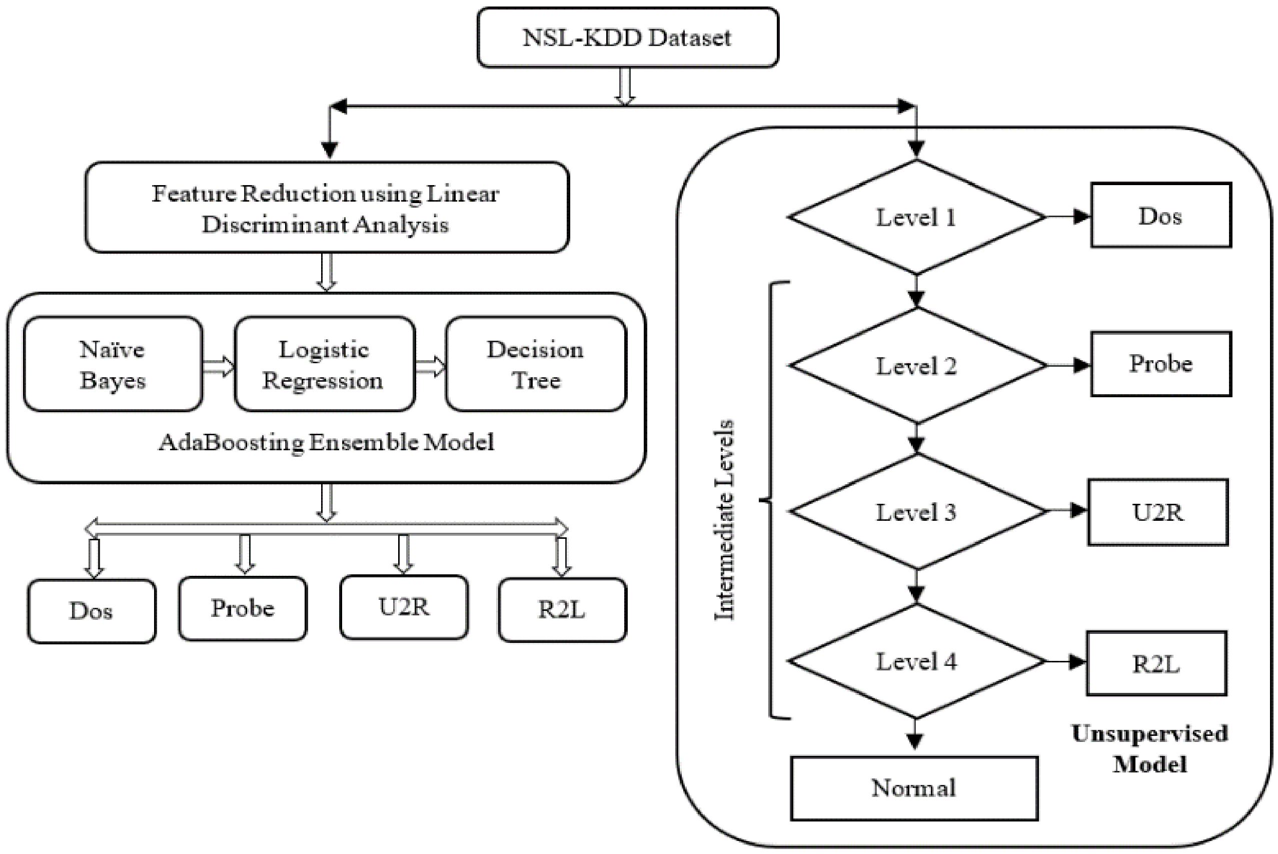

Fig. 1 Proposed framework

The proposed model has been experimented with the NSL-KDD dataset considering 43 features. Since it contains some redundant and irrelevant features, feature reduction has been performed on it using the linear discriminant analysis and obtaining a reduced dataset containing 21 trivial features. The base classifiers may be biased towards a particular attack and prone to misclassification, and hence, to address this, the ensemble model has been used. This aids in improving the performance of the proposed model. Hence, the reduced dataset has been considered in the Ada-Boosting ensemble model, which uses a combination of the base classifiers, Decision Tree, Naïve Bayes, and Logistic Regression. The ensemble model classifies the network transaction records in the reduced NSL-KDD dataset into distinct classes of attacks such as Probe, DoS, U2R, and R2L. Further, detection of attacks has been explored using an unsupervised learning algorithm and cluster the network transaction records into four clusters, with each cluster representing one of the four attack types. The class-by-class approach has been used wherein two clusters have been created at each level for improving the performance of unsupervised learning algorithms.

A. Dataset

In the experimentation, the dataset NSL-KDD, which contains 44 features, has been used. Its training set has 125973 records, while its testing set has 25195 records. The NSL-KDD training dataset has 4 major types of attack classes, Probe, DoS, Remote-to-Local (R2L), and User-to-Root (U2R).

a) Probe

An attacker examines the network and collects the information to make some breaches in the future, leading to a probing attack.

b) Denial of Service (DoS)

It happens at the time an attacker avoids the genuine and legitimate users from giving access to system resources by restricting the processing time of resources which is its main objective.

c) Remote-to-Local (R2L)

When an attacker transmits a message to a server and makes a few modifications to the server to illegally access the system resources from a remote system, it results in a root to local attack

d) User-to-Root (U2R)

An attacker makes an effort to acquire the secret code of a legitimate user and gain access to the user host as a valid user for retrieving the information. This leads the user to root the attack. It is quite likely that the attacker may try to get control of the server too and have full access[4].

Table 1 lists the different attributes in the NSL-KDD dataset.

Table 1. Feature set of NSL-KDD dataset [25]

No.	**Feature**	**Description**
1	Dst_host_rerror rate	% of connections to the current host with RST errors
2	Dest_host_serro r rate	% of connections to the current host with S0 errors
3	Dst_host_diff_s rv_ rate	% of different services on the current host
4	SRV count	Count of connections to the same service as the current connection in the past two seconds
5	Dst_host_same_ src_port_rate	% of connections to the current host with same src port
6	Dest_host_srv_ count	Count of connections with same destination host and that identical use service
7	Srv_rerror_rate	% of connections with REJ errors
8	Wrong_fragmen t	Count of wrong fragments
9	Diff_srv_rate	% of connections to different services
10	Srv_serror_rate	% of connections with "SYN" errors
11	Failed_logins	Count of logins failed
12	Dst_host_count	Count of connections with same destination host
13	compromised	Count of "compromised" conditions
14	count	Count of connections to the same host as the current connection in the past two seconds
15	Su_attempted	1 – "su root" command attempted; 0 – otherwise
16	Access_files	Number of operations on access control files
17	Dst_host_srv_se rror_rate	% of connections to the current host and specified service having S0 error
18	shells	Count of shell prompts
19	Same_srv_rate	% of connections to the same service
20	Outbound_cmds	Count of outbound commands belonging to sessions of FTP
21	Dst_host_srv_di ff_host_rate	% of connections to the same service arriving from discrete hosts
22	land	1 – connection from/to the same host/port; 0 – otherwise

23	Dst_host_srv_rerror_rate	% of connections to the current host and indicated service with RSTerrors
24	Serror_rate	% of connections with “SYN” errors
25	root	Count of “root” accesses
26	Rerror_rate	% of connections with REJ errors
27	Destination_bytes	Bytes leaving from destination towards the source
28	Is_hot_login	1 – login belongs to the “hot” list; 0 – otherwise
29	Logged_in	1 – successful login; 0 – otherwise
30	srvdiff_host rate	% of connections to distinct hosts
31	file creations	Count of file creation operations
32	Source_bytes	Bytes leaving from source
33	Dst_host_samesrv_rate	% of connections with same destination host and using the same service
34	flag	Status flag
35	urgent	Count of urgent packets
36	Is_guest_login	1 – “guest” login; 0 – otherwise
37	service	Type of service
38	hot	Count of “hot” indicators
39	Protocol_type	Protocol used for connection (e.g., UDP, TCP, ICMP)
40	Root_shell	1 –obtained; 0 – not obtained
41	duration	Connection duration
42	Class	Attack class
43	No	The number assigned to the attack

B. Algorithms used in the experimentation

a) Naïve Bayes

It is a probability-based method that solves problems of detection and prognostic nature. It uses conditional probability and assumes that each variable in the dataset is independent of the other [4].

b) Logistic Regression

It is a predictive analysis technique used for binary classification problems and analyzes a dataset having one or more independent variables and a dependent variable that has only two possible outcomes [26].

c) Neural Network

It consists of different layers like input, hidden, and output. Between the input and output, layers lie one or more hidden layers. The basic neural network has only two or three layers. Each layer comprises neurons.

d) Support Vector Machine

It is a supervised learning algorithm that uses kernels for performing non-linear classification. It is used for classification and regression but most often is used for classification [27].

e) Decision Tree

It uses a tree-like structure to solve the classification problem and comprises nodes and branches. The nodes signify the attributes or the features and have branches emanating from them, indicating the possible values of the attributes [28].

f) K-means

This unsupervised learning algorithm is used for clustering problems. To begin with, it randomly selects the centroids as the starting points for every cluster. It then iteratively performs computations to optimize the positions of the centroids. This continues until all the instances belong to one or the other cluster [29].

C. Performance metrics

The performance of various models used in the experiment is assessed based on recall, precision, F-score, and accuracy performance metrics. These metrics use false positive (FP), true positive (TP), false negative (FN), and true negative (TN) values from the confusion matrix.

a) Precision

It is expressed as the total number of true positive instances divided by the sum of true positive and false positive instances. It is given by the formula:

$$\boldsymbol{Precision = \frac{TP}{TP + FP}}$$

b) Recall

It is the ratio of the number of correct positive predictions to the sum of positive instances. It is also known as true positive rate (TPR) or sensitivity. Its maximum value is 1, and 0 is its least or worst value.

$$\boldsymbol{Recall = \frac{TP}{TP + FN}}$$

c) F-Score

It measures the harmonic mean of recall and precision and gives the derived effectiveness. It is computed as follows:

$$\boldsymbol{F - Score = \frac{2\,.\,Precision\,.\,Recall}{Precision + Recall}}$$

d) Accuracy

It measures the classifier’s performance and is often the number of correctly recognized instances divided by the total number of instances. A high accuracy indicates better results [12][13].

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

D. Experimental Setup

To perform the experiment, Ubuntu 18, Nvidia Titan V Graphics with Python 3.6 and Keras APIs have been used.

The server (Ubuntu 18.04 Linux) is used to conduct the experiments of algorithms, whereas one more additional server is used to inject malicious and non-malicious programs for the network. One important thing worth noting is that, in terms of time consumption and speed, both servers have exhibited similar performance for all experiments executed in this work.

IV. RESULTS AND DISCUSSION

The experimentation has been performed using all 44 features in the NSL-KDD dataset and using logistic regression, Naïve Bayes, Neural Network, Support Vector Machine, and Decision Treesmachine learning algorithms. Table 2. shows the performance metrics of these algorithms.

Table 2. Performance metrics of algorithms with all features in the dataset

Classifier	Precision	Recall	F-Score
Naïve Bayes (NB)	0.81	0.83	0.79
Logistic Regression (LR)	0.86	0.79	0.65
Neural Network (NN)	0.89	0.73	0.81
Support Vector Machine (SVM)	0.74	0.86	0.91
Decision Tree (DT)	0.91	0.85	0.93

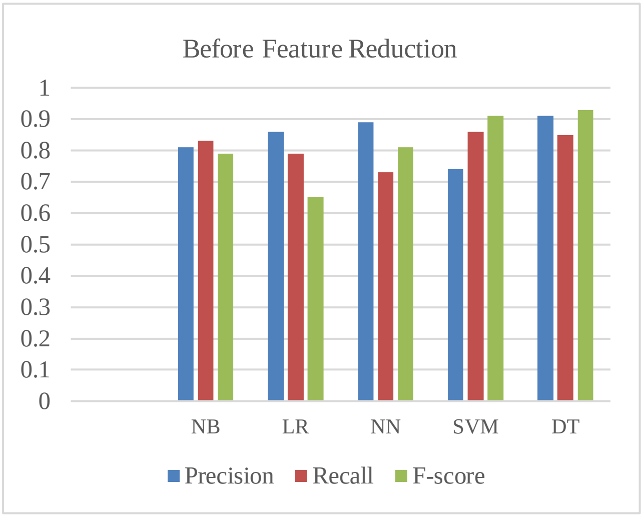

Fig. 2 Performance Metrics of the classifiers with the original dataset containing 44 features

Fig. 2 exhibits the performance metrics of the different machine learning techniques. When all 44 features in the dataset are used for the experiment, it is seen that logistic regression, neural network, and decision tree have a superior precision in comparison to the remaining algorithms used in the experiment. This means that these algorithms correctly predict the true positives more than 85% of the time. Naïve Bayes, Decision Tree, and Support Vector Machine are able to correctly identify true positives from more than 85% of relevant data, indicating that they have a better recall value than that of algorithms like Neural Network and Logistic Regression. Support Vector Machine and Decision Tree have much better F-score than other algorithms.

A. Feature Reduction Algorithm

The dataset contains 44 features, some of which may be irrelevant and redundant. These can affect the model performance. Hence, to improve it, the following linear discriminant analysis algorithm has been used as a feature reduction algorithm that takes as input the 44 features of the original dataset and gives as output 21 features. The steps of the linear discriminant algorithm are as follows:

a) Compute the class between variances

$$\mathbf{X_b} = \sum_{x=1}^{y} \mathbf{N_i}(\bar{\mathbf{y}}_i - \bar{\mathbf{y}})(\bar{\mathbf{y}}_i - \bar{\mathbf{y}})^T \quad \ldots\ldots (1)$$

Where $\mathbf{X_b}$ **determinesthevarianceforclassb**. N is a number of classes.

b) Compute within-class variance

$$X_w = \sum_{x=1}^{y}(N_i - 1)X_i = \sum_{i=1}^{g}\sum_{x=1}^{N_i} N_i(\bar{y}_i - \bar{y})(\bar{y}_i - \bar{y})^T \quad \ldots\ldots (2)$$

X_w **represents the class variance computed within classes**

c) Determine Fishers creation indices

$$F_{id} = max(argumant\frac{P^T S_x P}{P^T S_w P}) \quad \ldots\ldots.(3)$$

F_{id} ***representstheFishersindices***

As a result of the feature reduction algorithm, a dataset with a reduced number of features is obtained, which now contains 21 features. The same machine learning algorithms used earlier are applied to the reduced dataset to build the models. Table 3. presents the performance metrics of these algorithms.

Table 3. Performance metrics of algorithms with reduced dataset

Classifier	Precision	Recall	F-Score
Naïve Bayes (NB)	0.95	0.91	0.81
Logistic Regression (LR)	0.89	0.92	0.91
Neural Network (NN)	0.78	0.69	0.72
Support Vector Machine (SVM)	0.84	0.81	0.85
Decision Tree (DT)	0.93	0.78	0.82

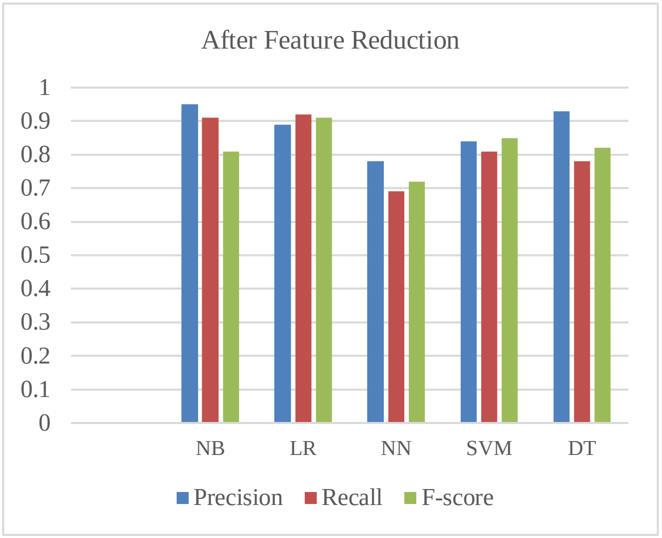

Fig. 3 Performance Metrics of the classifiers with the reduced dataset containing 21 features

Fig. 3 shows the performance metrics obtained using the machine learning algorithms on the reduced dataset, and it is observed that the precision of Naïve Bayes has increased from 0.81 for all features to 0.95 with the reduced dataset, that of support vector machine from 0.74 to 0.84, and for Decision Tree from 0.91 to 0.93. This means that the Decision Tree, Naïve Bayes, and Support Vector Machine algorithms detect attacks more precisely even with reduced datasets compared to other algorithms.

The comparison of Fig. 2 with Fig. 3 reveals that the performance of Naïve Bayes increases with the reduced dataset containing 21 features. At the same time, the performance of Logistic Regression is better for the original dataset containing 44 features. From Fig. 2 and Fig. 3, it can be witnessed that the performance of Neural Networks is superior with a huge amount of data than that with a smaller dataset. The performance of the Support Vector Machine is better for a large dataset but reduces for a small dataset, as can be noticed in the comparison between Fig. 2 and Fig. 3. Decision trees, on the other hand, have better performance on large as well as small datasets.

Table 4 compares the accuracy obtained before feature reduction with different models using all 44 features and after feature reduction with 21 features.

Table 4. Accuracy comparison of the different classifiers

Classifier	Accuracy with all 44 features (Before Feature Reduction) (in %)	Accuracy with 21 features (After Feature Reduction) (in %)
Naïve Bayes (NB)	74.52	92.66
Logistic Regression (LR)	80.68	93.54
Neural Network (NN)	83.82	81.57
Support Vector Machine (SVM)	65.26	79.39
Decision Tree (DT)	84.36	91.65

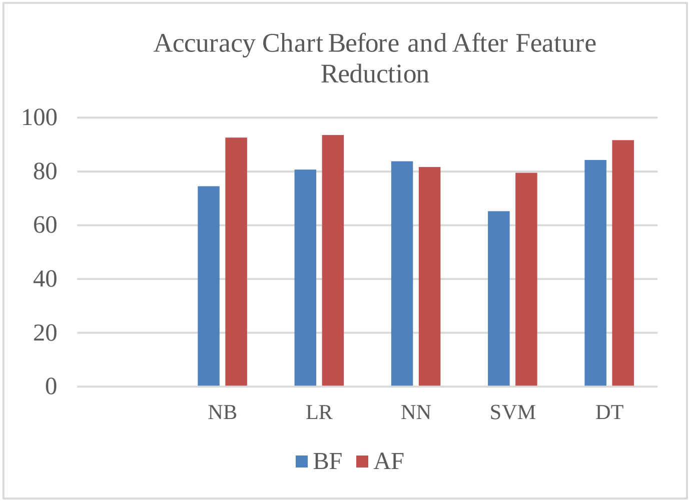

Fig. 4 Accuracy Comparison of different classifier before (BF) and after (AF) feature reduction

Fig. 4 displays the comparison of accuracies of different machine learning algorithms when all features in the dataset are used and when the reduced dataset is used. The accuracy of Decision Tree, Naïve Bayes, Logistic Regression, and Support Vector Machine has increased significantly. However, Neural Network shows a slight degradation in the accuracy with the reduced dataset. This shows that the Neural Network algorithm has better accuracy with a larger dataset than with a smaller dataset. Support Vector Machine shows an improvement with the reduced dataset. However, it is not as good as that of Naïve Bayes, Logistic Regression, or Decision Tree.

Table 5. Class-wise accuracy comparison of different models

Classifier	Class	Accuracy Before Feature Reduction(in %)	Accuracy After Feature Reduction(in %)
Naïve Bayes	DoS	72.36	91.45
	U2R	71.01	92.88
	R2L	73.68	93.54
	Probe	71.52	90.23
Logistic Regression	DoS	79.52	89.63
	U2R	83.78	95.21
	R2L	82.69	93.27
	Probe	83.55	93.19
Neural Network	DoS	81.65	78.25
	U2R	80.25	83.45
	R2L	87.33	82.1
	Probe	85.95	81.13
Support Vector Machine	DoS	58.23	78.33
	U2R	63.82	81.36
	R2L	70.12	80.14
	Probe	72.36	79.36
Decision Tree	DoS	83.3	88.45
	U2R	85.45	91.35
	R2L	88.35	89.44
	Probe	80.75	91.66

Table 5 shows the accuracy comparison of various models used in the experiment for each of the classes like DoS, Probe, R2L, and U2R. From Table 5, it is perceived that the performances of Naïve Bayes, Logistic Regression, and Decision Tree algorithms show significant improvement with reduced dataset compared to those with primary dataset. The performance of Neural Network has degraded for the reduced dataset when compared to the primary dataset. The performance of the Support Vector Machine improves significantly with the reduced dataset. However, the performances of Support Vector Machine and Neural Network are less when assessed against the accuracies of algorithms like Logistic Regression, Naïve Bayes, and Decision Tree algorithms used in the experimentation. Hence, the algorithms, Decision Tree, Logistic Regression, and Naïve Bayes, are employed in the AdaBoosting ensemble model. This model combines the decisions of these individual classifiers and boosts the model's performance by classifying the records in the reduced dataset as DoS, Probe, U2R, and R2L with better accuracy compared to the base classifiers.

B. Use of Unsupervised Learning algorithm

The supervised learning algorithms discussed in the previous section used the labelled data to train the different classifiers. To improve the performance of those classifiers, an ensemble learning model was used. To further explore the attack detection, an unsupervised learning algorithm like K-means was used. In this part of the work, all 44 features of the original dataset were used and based on the guideline that there are four different attack types, the value of *k*as 4has been used to create a cluster for each attack types DoS, R2L, Probe, and U2R. The performance metrics obtained with K-means clustering is shown in Table 6.

Table 6. Performance metrics for k-means clustering

Attack	Precision	Recall	F-score
DoS	65.23	0.45	0.12
U2R	59.17	0.39	0.35
R2L	69.36	0.56	0.44
Probe	58.14	0.22	0.36

As can be seen from Table 6, the precision with which the attacks are classified into the different groups using K-means clustering is not on par with the supervised learning algorithms' results. To ensure that none of the transaction records goes unidentified, the transaction records have been further explored using the class-by-class approach with unsupervised learning algorithms for each of the attack types. Hence, for improving the performance of k-means, the class-by-class approach is used, which clusters the transaction records in the original NSL-KDD dataset into two groups. For example, the transaction records in level 1 are clustered into two groups, DoS or not DoS. If it is not DoS, then in level 2, they are clustered into two groups, Probe or not Probe. Similarly, the other two attacks further go through the intermediate levels. As a final output, the result obtained is one of the attacks (DoS, Probe, R2L, or U2R) or normal. The performance metrics for each of these clusters are tabulated in Table 7.

Table 7. Class-wise performance metrics for different attack types

Performance metrics for K-means clustering for DoS attack			
Attack	Precision	Recall	F-score
DoS	89.65	0.88	0.79
Performance metrics for K-means clustering for Probe attack			
Attack	Precision	Recall	F-score
Probe	96.24	0.88	0.76
Performance metrics for K-means clustering for R2L attack			
Attack	Precision	Recall	F-score
R2L	98.24	0.88	0.79
Performance metrics for K-means clustering for U2R attack			
Attack	Precision	Recall	F-score
U2R	93.44	0.96	0.88

From Table 7, it can be observed that by employing the class-by-class approach, the performance metrics of K-means, such as recall, precision, and F-score, have enhanced performance.

The accuracies of several supervised learning algorithms like Neural Network, Logistic Regression, Decision Tree, Naïve Bayes, and Support Vector Machine and of unsupervised learning algorithms like K-means with 4 and 2 clusters ($k = 4$, $k = 2$) have been tabulated in Table 8 below.

Table 8. Accuracy comparison supervised and unsupervised learning algorithms

	Classifier	Class	Accuracy (in %)
Supervised Learning	Naïve Bayes	DoS	72.36
		U2R	71.01
		R2L	73.68
		Probe	71.52
	Logistic Regression	DoS	79.52
		U2R	83.78
		R2L	82.69
		Probe	83.55
	Neural Network	DoS	81.65
		U2R	80.25
		R2L	87.33
		Probe	85.95
	Support Vector Machine	DoS	58.23
		U2R	63.82
		R2L	70.12
		Probe	72.36
	Decision Tree	DoS	83.3
		U2R	85.45
		R2L	88.35
		Probe	80.75
Unsupervised Learning	K-Means (k=4)	DoS	64.22
		U2R	55.39
		R2L	73.41
		Probe	62.36
	K-Means (k=2)	DoS	94.56
		U2R	90.14
		R2L	99.15
		Probe	97.10

From Table 8, it is noticeable that the accuracy of the supervised learning algorithms in classifying the different attacks is around 75%. Comparing the performance of the supervised learning algorithms with that of the unsupervised algorithm, it is obvious that the accuracy of K-means unsupervised learning algorithm with 4 clusters ($k = 4$) is in the range of 62% – 73%. However, with the class-by-class approach in unsupervised learning, the accuracy is in the range of 90% – 99%, which is extremely good.

V. CONCLUSION

In work considered, a framework based on machine learning has been implemented for identifying and classifying the attacks. Different machine learning algorithms have been extensively employed in the work that spans from supervised to ensemble to unsupervised. Neural Network, Logistic Regression, Naïve Bayes, Decision Tree, and Support Vector Machine models, along with K-means, have been used in work. Experimentation has been carried out on NSL-KDD raw dataset as well as on the reduced dataset obtained from feature reduction.

Unsupervised learning is randomly applied with a k value as 4, and in the class-by-class approach, it uses the value of k as 2. Comprehensive performance evaluation and analysis of all these algorithms has been carried out. The performance of ensemble learning algorithms outperforms that of base classifiers. From the detailed analysis, it is seen that an unsupervised learning algorithm with a class-by-class approach has superior performance than other algorithms. With the use of a proactive approach, the system can be saved from the catastrophic effects of undetected attacks.

REFERENCES

[1] S. Choudhary and A. Sharma, Malware Detection Classification using Machine Learning, Proc. - 2020 Int. Conf. Emerg. Trends Commun. Control Comput. ICONC3 ,(2020) 20–23.

[2] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, Building an Efficient Intrusion Detection System based on Feature Selection and Ensemble Classifier, Comput. Networks,174(8) (2020) 1–12.

[3] R. Patil, H. Dudeja, and C. Modi, Designing in-VM-assisted Lightweight Agent-Based Malware Detection Framework for Securing Virtual Machines in Cloud Computing, Int. J. Inf. Secur., 19(2) (2020) 147–162.

[4] B. I. Seraphim, S. Palit, K. Srivastava, and E. Poovammal, Implementation of Machine Learning Techniques Applied to the Network Intrusion Detection System, Int. J. Eng. Adv. Technol., 8(5) (2019) 2721–2726.

[5] S. Talukder, Tools and Techniques for Malware Detection and Analysis, (2020). [Online]. Available: http://arxiv.org/abs/2002.06819.

[6] S. Agarkar and S. Ghosh, Malware Detection and Classification using Machine Learning, Proc. - 2020 IEEE Int. Symp. Sustain. Energy, Signal Process. Cyber Secur. iSSSC 2020 (2020).

[7] A. Meryem and B. EL Ouahidi, Hybrid Intrusion Detection System using Machine Learning, Netw. Secur., 5 (2020) 8–19.

[8] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, A Survey on Ensemble Learning, Front. Comput. Sci., 14(2) (2020) 241–258.

[9] S. A. Ludwig, Applying a Neural Network Ensemble to Intrusion Detection, J. Artif. Intell. Soft Comput. Res., 9(3) (2019) 177–188.

[10] J. Gu, L. Wang, H. Wang, and S. Wang, A Novel Approach to Intrusion Detection using SVM Ensemble with Feature Augmentation, Comput. Secur., 86 (2019) 53–62.

[11] H. Rajadurai and U. D. Gandhi, A Stacked Ensemble Learning Model for Intrusion Detection in Wireless Network, Neural Comput. Appl., 5 (2020).

[12] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, An Adaptive Ensemble Machine Learning Model for Intrusion Detection, IEEE Access, 7 (2019) 82512–82521.

[13] Z. Halim et al., An Effective Genetic Algorithm-Based Feature Selection Method for Intrusion Detection Systems, Comput. Secur., 110 (2021)102448.

[14] J. Gu and S. Lu, An Effective Intrusion Detection Approach using SVM with Naïve Bayes Feature Embedding, Comput. Secur., 103 (2021) 102158.

[15] Y. T and G. Murtugudde, An Efficient Algorithm for Anomaly Intrusion Detection in a Network, Glob. Transitions Proc., 2(2) (2021) 255–260.

[16] Kunal and M. Dua, Attribute Selection and Ensemble Classifier based Novel Approach to Intrusion Detection System, Procedia Comput. Sci., 167(2019) (2020) 2191–2199.

[17] A. K. Shukla, Building an Effective Approach toward Intrusion Detection using Ensemble Feature Selection, Int. J. Inf. Secur. Priv., 13(3) (2019) 31–47.

[18] P. Maniriho, L. J. Mahoro, E. Niyigaba, Z. Bizimana, and T. Ahmad, Detecting Intrusions in Computer Network Traffic with Machine Learning Approaches, Int. J. Intell. Eng. Syst., 13(3) (2020) 433–445.

[19] F. Masoodi and others, Machine Learning for Classification analysis of Intrusion Detection on NSL-KDD Dataset, Turkish J. Comput. Math. Educ., 12(10) (2021) 2286–2293.

[20] S. Allagi and R. Rachh, Framework for Providing Security in Private Cloud using Machine Learning Techniques, Int. J. Eng. Adv. Technol., 9(1) (2019) 7641–7645.

[21] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, A Stacking Ensemble for Network Intrusion Detection Using Heterogeneous Datasets, Secur. Commun. Networks, 2020 (2020).

[22] H. P. Vinutha and B. Poornima, Analysis of Feature Selection Algorithms for Naïve Bayes Classifier using NSL-KDD, Int. J. Eng. Manuf. Sci.,8(1) (2018) 167–175.

[23] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, Features dimensionality reduction approaches for machine learning-based network intrusion detection, Electron., 8(3) (2019).

[24] M. S. Abirami, U. Yash, and S. Singh, Building an Ensemble Learning Based Algorithm for Improving Intrusion Detection System, Adv. Intell. Syst. Comput., 1056 (2020) 635–649.

[25] P. G. Jeya, M. Ravichandran, and C. S. Ravichandran, Efficient Classifier for R2L and U2R Attacks, Int. J. Comput. Appl., vol. 45, 21 (2012) 29.

[26] S. Ware, S. Rakesh, and B. Choudhary, Heart Attack Prediction by using Machine Learning Techniques, Int. J. Recent Technol. Eng., 8(5) (2020) 1577–1580.

[27] R. Geetha and T. Thilagam, A Review on the Effectiveness of Machine Learning and Deep Learning Algorithms for Cyber Security, Arch. Comput. Methods Eng., 28(4) (2021) 2861–2879.

[28] I. Abrar, Z. Ayub, F. Masoodi, and A. M. Bamhdi, A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset, Proc. - Int. Conf. Smart Electron. Commun. ICOSEC 2020, no. Icosec, (2020) 919–924.

[29] O. I. Al-Sanjary, M. A. Bin Roslan, R. A. A. Helmi, and A. A. Ahmed, Comparison and Detection Analysis of Network Traffic Datasets Using K-Means Clustering Algorithm, J. Inf. Knowl. Manag., 19(3) (2020) 1–22.