IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 23, Issue 3, Ser. II (May – June 2021), PP 18-30
www.iosrjournals.org

Design of Intrusion Detection System using Machine Learning Techniques

R.Rajakumar[1,] S. Sathiya Devi[2]
**Research Scholar, University College of Engineering, BIT campus, Anna University, Tiruchirappalli*
**Assistant Professor, Department of Information Technology, University College of Engineering, BIT Campus, Anna University, Tiruchirappalli*

Abstract: *Network Intrusion Detection System is a software system used to detect security threats, vulnerabilities and anomaly activities in the computer networks and host-based environments. In the past decade, several researches have been proposed by various authors. However, due to the rapid advancement of technology, still it is being a challenging research area. In this paper, we have analyzed the performance of NSL-KDD 20% dataset in different standard mechanisms. We found the existing techniques are biased due to the imbalanced datasets. We have selected the ensemble techniques in order to enhance the efficiency of intrusion detection. We have proposed ensemble models like bagging, boosting and stacking classifiers to improve the efficiency of intrusion detection techniques. Finally, we have achieved better results for NSL-KDD Dataset using a stacking classifier with cross-validation.*
Key Words: *Attacks, Data Sets, Ensemble Approach, Intrusion Detection System, Machine Learning*

Date of Submission: 14-06-2021 Date of Acceptance: 28-06-2021

I. Introduction

An Intrusion Detection System (IDS) plays a vital role in the field of cyber security for realizing a hard line of protection against cyber threats. The digital technology has become the important infrastructure of the real world because of the dominant use of computers, smart devices and network systems in our day to day activities at low cost. Since the use of Information and Communication Technology (ICT) are rapidly increasing globally, it is very much required for securing network resources against network adversaries. Even though several security mechanisms exist, they are not robust enough for protecting our systems against the new way of attacks. Therefore it is the need of the hour for reconsidering the security by designing alternate architectures for protecting our network resources.

A network system is considered to be secure if the following three principles are realized: Confidentiality, Integrity and Availability (CIA) [1, 2, 3]. There are varieties of techniques followed by each attacker, which cause dangerous threats to computer networks. Every attacker collects significant information about a system in order to violate the confidentiality of the system and when it disturbs the genuine operations, it compromises the availability and integrity of the system. For instance, Denial of Service (DoS) attack interrupts the availability of the system to clients, whereas malware code steals the implementation of the program which damages the integrity principle [2, 4, 5].

An IDS is a mechanism for monitoring and studying the functioning of a computer or network system to discover the possible threats by assessing the violations of CIA principles [6, 7, 8]. The conventional architecture of a Network IDS (NIDS) includes four modules [9] as shown in figure 1, namely, a Packet Decoder (PD), pre-processor, Decision Engine sensor and Defense Response (DR)/alert module, as described below.

The Packet Decoder (PD) collects data packets using audit data collection tools, such as cpdump and Libpcap, which transfer portion of network traffic into the pre-processor for processing. The pre-processor catches a set of features from the raw audit data which can be used after in the DE sensor. A classical pre-processor is the TCP handler which investigates TCP protocols in session flows; for instance, Argus, Bro-IDS and Netflow tools which inspect different protocols, such as HTTP, SMTP and UDP.

The responsibility of the DE sensor is to accept the mined features from the pre-processor and to construct a framework that differentiates the behavior of the attacker from the normal operations. Whenever an attack is detected, it demands the DR module for raising an alarm. The Defense Response (DR) is responsible for the following activities: (i) Triggers alarms and record them in a database and (ii) Alert the security administrator for carrying out an immediate action against the attack.

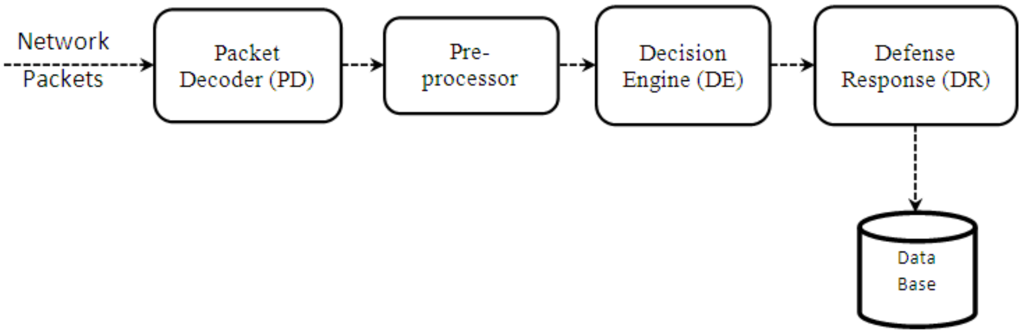

Figure 1. Architecture of classical IDS

Many surveys have been conducted for reviewing the IDS technology. In [10] Ahmed et al. explained the various methods of anomaly detection systems and discussed some of the challenges of IDS datasets. In [11] the authors Aburomman and Reaz have discussed about constructing a hybrid IDSs by integrating feature selection and detection methods for enhancing the accuracy of the detection. However, it requires huge computing resources.

In [12] Peng et al. explained about the intrusion detection and prevention techniques by planning the profiles of users and realizing anomalies by variations. As a recent development researchers surveyed the implementation of IDSs in different application platforms such as Internet of Things (IoT)-based IDS [13] and Cloud-based IDS [14]. For example, in [15] Zarpelao et al. have presented a study of IDSs in IoT networks. The authors have discussed the detection methods, IDS deployments, and security threats. Sharma and Kaul [16] have proposed the methodologies of implementing IDSs in VANET and VANET Cloud. Recently Resende and Drummond [17] have proposed a comprehensive study of IDS using Random Forest methods for evolving a reliable IDS.

At the earlier stage many researchers focused on statistical approaches and developed rule-based expert systems. These methods are capable of efficiently identifying attacks and offer small datasets. However, when we are using large benchmark datasets such as KDD 1999, NSL-KDD, UNBW-NV15 we need to consider the time and space complexity of the algorithms when we are performing very large rules.

Several researchers proposed various Machine Learning (ML) techniques and tools for developing IDS which include Decision Tree (DT), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Naïve Bayes (NB), Logistic Regression (LR), Stochastic Gradient Descent (SGD), Linear Discriminant Analysis (LDA) and many more. However, there is a big challenge with the conventional classifiers in efficiently detecting the attack types due to the highly imbalanced class natures for KDD 1999 dataset (Majority and Minority class samples).

An imbalanced dataset typically refers to an issue with the data set classification problems where the classes are not equally represented or distributed. As an example, KDD 1999 datasets consist of a large number of 67343 major class samples out of 125973 Records. However, other attack classes like Dos, Probing, U2R, and R2L classes are having very fewer numbers of samples. Especially User to Root (U2R) and Remote to Local (R2L) attack classes contains only 0.04% and 0.79% records for attack classes. Therefore standard classifiers fail to identify attack labels. Hence, we need specialized techniques to give importance to minority class samples [18, 19].

1.1 Objectives

The main objectives of this research have two folds:

1. To achieve a high detection rate on the network traffics and to produce a very low false positive alarm rate with minimum cost.
2. To build an ensemble model of the IDS that can efficiently combines each weak learner from a strong learner.

This article focuses on the following:

- Categorizing various classes of IDS with the major types of attacks based on intrusion methods.
- Providing a classification of anomaly based IDS performance metrics
- Explaining the significances of the feature selection methods.

II. Intrusion Detection Systems

Intrusion can be defined as an activity that causes damage to an information system by unauthorized accessing. In other words, an intrusion is an action of any unauthorized person or system that could create a possible threat to the CIA properties of an information system. For instance, activities that would cause the computer or network system insensitive to the legitimate users are said to be intrusion and the person who originate this activity is called as an intruder.

An IDS is a system that recognizes malicious activities on computer systems in order to maintain the CIA properties to keep the system a secured one [20]. The objective of IDS is to recognize various kinds of unauthorized usage of computers and networks, which a traditional firewall cannot detect. In general the IDS systems can be widely classified into two branches: (i) Signature-based Intrusion Detection System (SIDS) and (ii) Anomaly-based Intrusion Detection System (AIDS).

2.1 Signature-based intrusion detection systems (SIDS)

The working principle of SIDS is on pattern matching techniques to detect a known attack. Several matching techniques are used to find the methods of previously used attacks. In other words, if the signature of the intrusion matches with the existing signature of a previous intrusion, an alarm is raised. For SIDS, system's logs are examined to find the sequences of actions which have formerly been recognized as malware. SIDS are also known as Knowledge-based Detection or Misuse Detection [21, 22]. Figure 2 illustrates the concept of SIDS approaches. The key idea behind this is to create a database of intrusion signatures and to compare the activities of the newly present attack against the recorded signatures and trigger an alarm if a match is found.

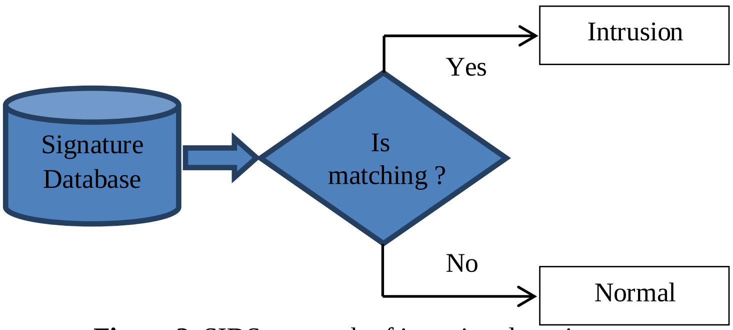

Figure 2. SIDS approach of intrusion detection

However, the major drawback of SIDS is, it is not capable of detecting zero-day attack, since these attacks generate a new form of threats and there could not be found any matching signature in the database. Once the signature of the new attack is extracted and recorded then it could be used for detecting similar attacks in future.

2.2 Anomaly-based intrusion detection system (AIDS)

Most of the scholars are interested in AIDS due to its capability to overcome the drawbacks of SIDS. In AIDS, a regular working behavior model of a computer system is created using knowledge-based, machine learning or statistical-based approaches. The IDS is implemented and the computer system is put under the observation of the IDS. Whenever any deviation between the observed behavior and the modeled behavior is notified, then it is regarded as an anomaly, which can be considered as an intrusion. The abnormal usage of the system which causes the deviation from the normal behavior is classified as an intrusion. The implementation of AIDS includes two phases: (i) The training phase and (ii) The testing phase.

In the training phase, the regular or normal packet transmission configuration is used to train the model of normal behavior, and then in the testing phase, a new data set is used to ascertain the capability of the system in identifying new intrusions. AIDS can be classified into a number of categories based on the method used for training such as knowledge-based, statistical based and machine learning based [23]. The main advantage of AIDS is the ability to detect the zero-day attacks since it is able to recognize the abnormal activity of the system and it does not depend on a signature database [24]. AIDS raises a warning alarm when the observed behavior deviates from the normal behavior. Table 1 presents the differences between SIDS and AIDS.

Table 1. Differences between SIDS and AIDS

IDS Type	Advantages	Disadvantages
SIDS	Simple design	Needs to be updated frequently with new signatures
	Powerful for detecting well known attacks.	Not designed to detect new attack signatures
	Promptly identifies the intrusions	Unable to detect the zero-day attack.
	False detection is very low	Not suitable for detecting multi-step attacks
AIDS	Could be used to detect new attacks	High false positive alarms
	Could be used to create intrusion signature	Needs initial training
	Able to detect zero-day attacks	May create Unclassified alerts
	It is very difficult for an intruder to recognize normal user behavior and to access the system since the system is built from customized profiles.	It is very difficult to create the normal behavior pattern for a dynamic computer system

2.3 Data sources based classification

In the previous section we have discussed about the types of IDS based on the methods used to detect the intrusions. However, IDS can also be categorized based on the input data sources used to detect abnormal activities, namely Host-based IDS (HIDS) and Network-based IDS (NIDS). HIDS functions on the input data that originates from the host system and audit sources, such as operating system, window server logs, firewalls logs, application system audits, or database logs. HIDS can detect insider attacks that do not involve network traffic [25].

NIDS monitors the network traffic that is extracted from a network through packet capture, NetFlow, and other network data sources. Network-based IDS can be used to monitor the computers that are connected in a network. NIDS is capable of monitoring the external threats that could be initiated from an external source at an earlier phase, before the threats spread to another computer system. NIDS can be deployed at several positions inside a particular network topology. HIDS together with NIDS and firewalls, can provide a robust and multi-tier protection against both external and insider attacks.

III. Description Of Ids Data Sets

Data sets are used to assess the performance of the IDS, whether it can be used to sense the attack exactly or not. The outcome of any IDS is based on the quality of the data set. Here we are providing three famous data sets namely KDD Cup'99, NSL-KDD and, UNSW-NB15. Detailed descriptions about the features of these datasets are discussed in the following sections.

3.1. KDD Cup'99 Data set

KDD'99 data set was created by DARPA in 1999 by using recorded network traffic from 1998 dataset. Pre-processing is carried out on this data set to extract 41 features per network connection. The set of features in KDD'99 data set are classified into four groups such as basic features which includes the features from 1 to 9, content features which consists of features from 10 to 22, Time based traffic features contains features from 23 to 31 and Host based traffic features which comprises features from 32 to 41 [26]. KDD'99 [27] contains 4,898,430 records which is larger than other data sets.

There are four different types of attacks dealt in this data set; they are DoS attack, R2L, U2R and Probe. Many data mining techniques are applied to the KDD'99 data set to detect the intrusions in network packet transmission. KDD Cup'99 is the mostly used data set to build IDS. It is found that KDD data set have two critical issues, that greatly affect the performance of the IDS: (i) Most noteworthy issue is that KDD data contains 78% of replicated records and (ii) It is found that 75% records in train and test data set are duplicated. This kind of replicated records causes the learning algorithms to be partial. These records may be destructive to the networks like U2R, R2L etc.

3.2. NSL-KDD Data set

In order to address the issues of KDD Cup data set, they have proposed a new data set, called NSL-KDD, which consists of selected records of the complete KDD Cup'99 data set. There are several advantages of NSL-KDD data set over the KDD Cup'99 data set:

- It doesn't include irrelevant records in the training set, so the classifiers will not be partial towards more repeated records.
- From each difficulty record, number of chosen records is inversely proportional to the percentage of the records in the KDD data set.

So, this results that classification percentage of the different ML techniques differ in a wide range. This make structured comprehensive evaluation of ML approaches [28] [6]. In training and testing data set number of records are logical that make it more comfortable to do the experiments on entire data set without any need to choose random small segments. NSL-KDD gives various advantages to researchers to apply various machine learning techniques and methods and compares their results in this benchmark dataset with an efficient way in the intrusion detection domain. Table 2 gives the number and percentage of class wise distribution of records in the NSL-KDD dataset. There are several advantages in NSL-KDD Dataset

i. Training Dataset Set does not include any redundant records so the classifiers will not be produced unbiased results.
ii. Training Set does not contain any duplicate records; therefore, the performance of the classifier will provide more accurate results.
iii. The number of designated records from each difficulty level is inversely proportional to the percentage of records in the original KDD Dataset.
iv. Not complex Level data

Table 2 NSL KDD 20% Dataset Class Distribution

NSL-KDD 20% -Training Set				
Normal Class	DoS Class	Probe Class	R2L Class	U2R
13449	9234	2289	209	11
53.38%	36.65%	9.08%	0.82%	0.04%
NSL-KDD 20% -Test Set				
Normal Class	DoS Class	Probe Class	R2L Class	U2R Class
2152	4344	2402	2885	67
18.16%	36.65%	20.27%	24.34%	0.56%

Therefore, the classification accuracy rates of dissimilar machine learning methods differ in a wider range, which makes it more capable to have an accurate evaluation of different machine learning models.

3.2.1 Training Dataset

In NSL-KDD dataset, a training set consists of 25192 Records. Each record consists of 41 Feature (independent variable x) and the predicted label (y) consists of the attack label category. There are 39 attacks put under four attack categories namely DOS, Probe, R2L and U2R.

3.2.2. Test Dataset

NSL-KDD Test Set consists of 11850 records with 42 features. First 41 features are x (Independent variable) and 42nd attribute is predicted labels (dependent variables). Based on these attributes we can classify these features into four subgroups of networks like connection vector, content related, Time Related traffic features and Host Based Traffic Features. The attack classes present in NSL KDD dataset are grouped into four categories: (i) Denial-of-Service (DoS) attack (ii) Probing attack (iii) Unofficial access to local super user /Root (U2R) attack (iv) Remote to Local login (R2L). Nowadays, numerous researchers working in IDS domain do experiments with the NSL KDD dataset even though it consists of highly imbalance nature of features and unequal distribution of attack class.

3.3 UNSW-NB15 Data set

The UNSW-NB15 [29] is new dataset published in 2015. It includes moderns attack (nine attack types compared to 14 attack types in KDD'99 dataset). It has 49 features and a variety of normal and attacked activities including with class labels of total 25,40,044 records. There are 2,21,876 normal records and 3,21,283 attacked records in the total number of records. Features of UNSW-NB15 data set are categorized into six groups namely Basic Features, Flow Features, Time Features, Content Features, Additional Generated Features, and Labeled Features. Features counting from 36-40 are known as General Purpose Features. Features counting from 41-47 are known as connection features. Further, UNSW-NB15 dataset has nine type of attacks category known as the Analysis, Fuzzers, Backdoors, DoS Exploits, Reconnaissance, Generic, Shellcode, and Worms.

IV. Preprocessing

In this section, we consider our entire NSL KDD dataset for further analysis. In this preprocessing step we have performed various operations like outlier detection, missing value estimation, encoding, and decoding mechanism for type conversions, features importance calculation, and best feature selection for improving our results.

4.1 Dataset Features Analysis

As indicated in section 3.2 NSL KDD dataset consists of 41 features. There are three nominal features such as protocol type, service and flag and six binary features like land, logged_in, root_shell, su_attempted, is_host_login and is_guest_login. The Remaining 32 features are numerical labels describing various network parameters.

4.2 Feature Selection

Feature engineering is a major role for classification tasks. It reduces computational time and increases classification accuracy. The most used types of feature selection techniques are: (a) Filter methods and (b) Wrapper methods. In general, Filter methods select the subset of the features based on the characteristics of the given training population. It is suitable method for performing very large datasets because it takes very less computational time to compare with wrapper methods [30]. On the other hand, wrapper methods also called optimizer, optimizes the model or classifiers for the feature selection process. Sometimes it produces very good results for the filter method. It takes a lot of computation time to compare with the filter method and it's not suitable for big data processing and large datasets.

In this connection, various researchers proposed various feature selection techniques for classifiers like Information Gain (IG), Gini Ratio (GR), Correlation base feature selection (CFS), etc. In our experiments we have used the hybrid feature selection techniques [31] in the NSL-KDD data sets.

4.3 Train Test Split

After the dataset analysis, some encoding techniques can be applied like label encoder, count encoder and one hot encoding for the categorical variable to dummy variables creation. We have created the function for outlier detection and missing value estimation for our given dataset. After the preprocessing steps, we have split the training set in 70% and 30% ratio using hyper-parameter selection. Finally, we have given input to the classifiers and calculate training set accuracy.

V. Machine Learning Techniques

Nowadays, ML techniques play a vital role to solve the inherent classification problems for a much-advanced domain. ML techniques are computer algorithms that can learn without being explicitly programmed. In this section we discuss about some of the ML techniques for constructing our IDS. We have implemented some famous ML techniques such as Decision Tree (DT), Naïve Bayes (NB), K-Nearest Neighbour (KNN), Support Vector Machine (SVM), Logistic Regression (LR), Linear Discernment Analysis (LDA), Stochastic Gradient Decent (SGD) algorithms. We have calculated the efficiency our IDS by using further ensemble models [32].

5.1 Decision Tree (DT)

A decision tree is a commonly used classification algorithm for the supervised learning domain in ML techniques. DT algorithm with a predefined target variable is commonly used in binary and multiclass classification problems. There are several domains such as pattern recognition, medical domain, security domain, etc have employed DTs for classification analysis. The DT classifiers consist of nodes, edges, and leaves. Root Node represents the entire population or sample and this further gets divided into two or more homogeneous sets. However, the main issues are going to construct the split value of the node. Figure 3 gives the structure of a DT commonly used for classifier algorithms.

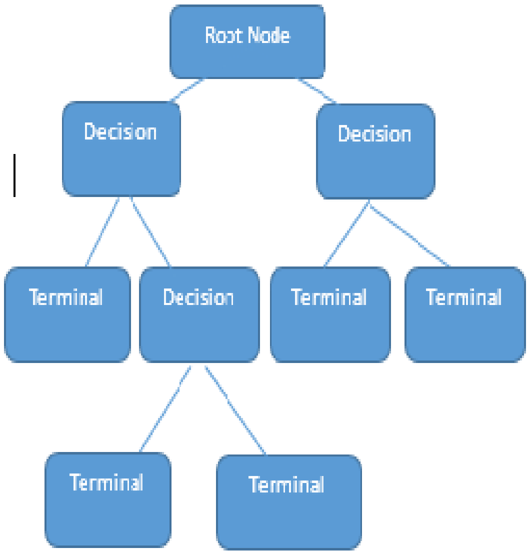

Figure 3. The structure of a Decision Tree

5.1.1 Experiment Results

We have conducted the experiments using the DT technique with NSL-KDD data set. The experiment results are shown in figure 4. The graph shows the percentage of accuracy while using all the 41 features of training and test data set features. These results are compared with the results of using only the most significant 20 features.

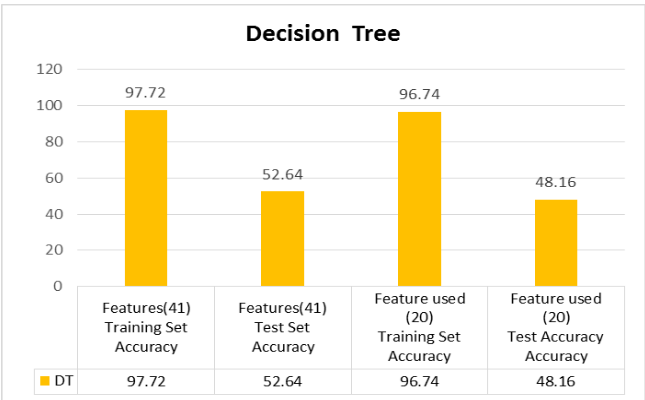

Figure 4. Results of DT for training and test data set with 41 and 20 features

5.2 Naive Bayes (NB)

Naive Bayes (NB) is another standard classifier used in many domains. This technique is based on Bayes' Theorem with an assumption of independence among predictors. It means that the NB classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. NB model is very easy to perform experiments with very large number of datasets. It produces highly accurate results while comparing with other classifiers. Bayes Theorem gives a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c) the formula for Bayes Theorem is shown in equation (1).

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \quad (1)$$

$$P(c|x) = P(x_i|c)*P(x_2|c)*\ldots\ldots..P(x_n|c)*P(c)$$

Where P(c|x) is the posterior probability of independent variables (x) and class variable or dependent variable (y) prediction P(c) is the prior probability of target features(y), P(x|c) is the likelihood which is the probability of predictor given input class(x) and P(x) is the prior probability that is going to use as a predictor(y). The experimental results of NB approach is given in figure 5.

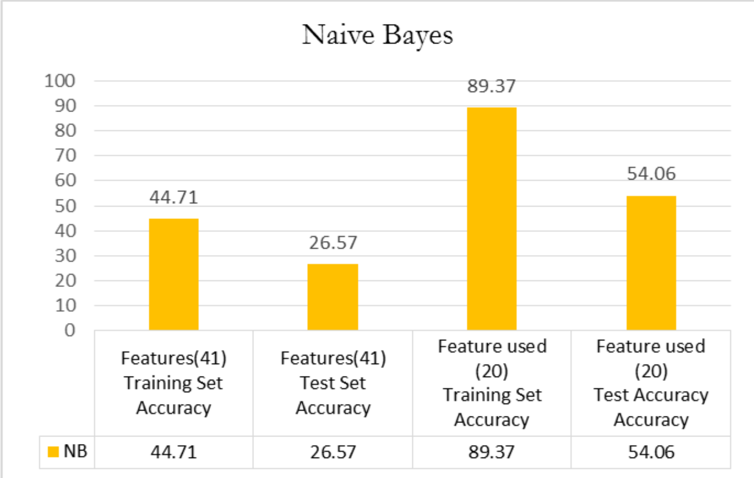

Figure 5 Results of NB for training and test data set with 41 and 20 features

5.3 K Nearest Neighbor (KNN)

The K-Nearest Neighbor method is another familiar classifier for supervised learning problems. It is a standard classifier and it works across various domains. It is a lazy learning model because classification is calculated from simply gathering votes of a k number of nearest neighbors of each point. This algorithm is easy one to construct, healthy to noisy training data and can actively work with large training datasets [33] [4].

5.3.1 Experimental Results

In K-NN, we need to decide the value of K and the computation is high as it needs to compute the distance of each instance to all the training samples. In this work, we have used this model and calculate the efficiency of our NSL KDD 20 percent Dataset. Figure 6 demonstrates the implementation results and prediction rate.

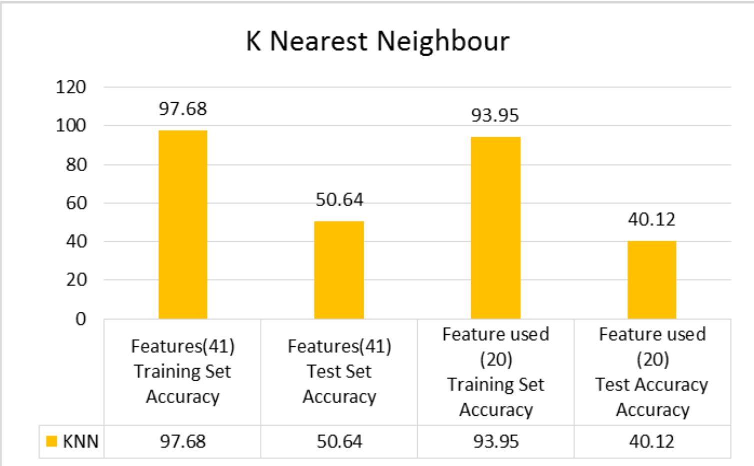

Figure 6. Results of K-NN for training and test data set with 41 and 20 features

5.4 Logistic Regression (LR)

LR is a supervised learning algorithm that can be used to predict a binary outcome (0/1, Yes/No, True/false) when a set of independent variables(x) given.

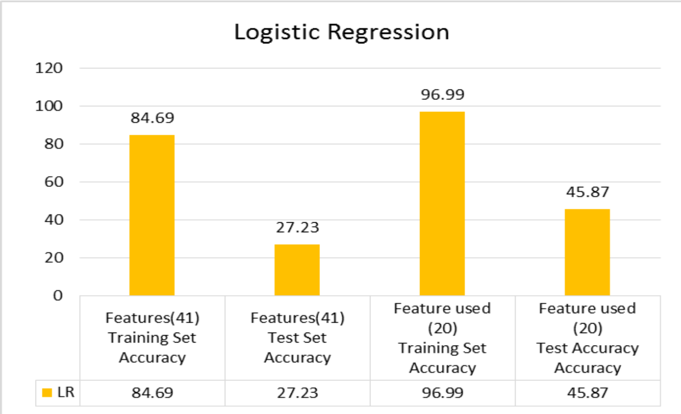

Figure 7. Results of LR for training and test data set with 41 and 20 features

In case any categorical variables present in the independent variables (set ox x) we can use dummy variables or label encoder techniques. In simple terms, it predicts the probability of occurrence of an event by fitting data to a logic function. It is suitable for the binary/logical classification domain. Here we used this model and calculated the efficiency of our NSL KDD 20 percent dataset [34] [5]. Our experimental results are shown in figure 7.

5.5 Support Vector Machine (SVM) classifier

SVM is a standard and modern classifier for binary classification in data mining domain. Generally, the SVM is a statistical pattern learning algorithm by using its own kernel functions. In recent days, pattern recognition, artificial intelligence and neural networks domains used SVM for improving their accuracy. It has become one of the popular techniques for anomaly based IDS, since all training samples are in statistical pattern. Another positive approach of this algorithm is to perform global minimum and risk minimization for high dimensional data patterns. SVMs can study a large dataset of patterns and be able to scale better because the classification complication does not depend on the dimensionality of the feature space.

Here we have used the SVM technique and calculate the efficiency of our NSL KDD Dataset. The experimental results and the prediction rate of this model is shown in figure 8.

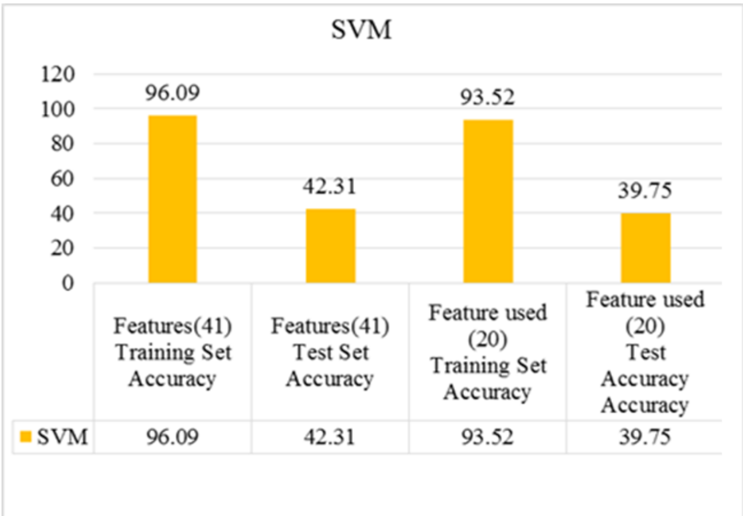

Figure 8. Results of SVM for training and test data set with 41 and 20 features

5.6 Neural Networks

The neural network also called as an Artificial Neural Network (ANN) classifier. It deeply analyzes statistical, categorical data pattern from the given set of input. It works on many domains such as Deep Learning (DL), Image Processing (IP), Signal Processing (SP), Pattern Recognition (PR) and Convolution Neural Networks (CNN) Domains. It consists of three important layers namely input layer, hidden layer, and output layer. Each layer has unique functionalities. The training samples deeply observed by input layers. Hidden layers work as the intermediate layers between the other layers. It helps to find out the complicated relationships involved in the training samples.

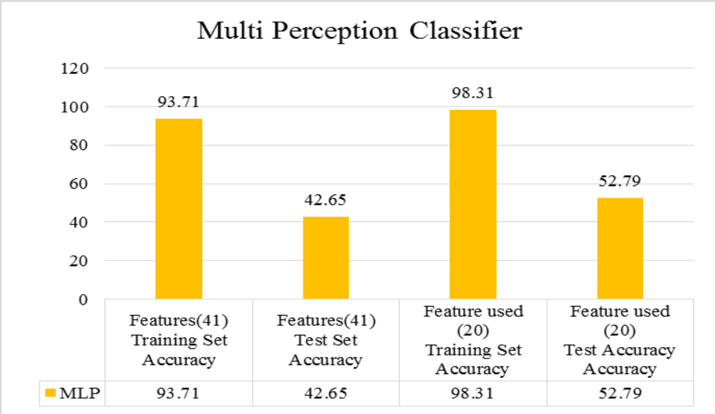

Figure 9. Results of ANN for training and test data set with 41 and 20 features

The output layer is used to extract the final output from the other two layers. Here, due to the nature of high imbalance data and statistical pattern we have used the NSL KDD 20% Dataset for calculating the efficiency. Our experimental results show the performance and the prediction rate of our given dataset. Figure 9 demonstrates the performance of ANN on NSL-KDD dataset.

5.7 Linear Discriminant Analysis (LDA)

In general LR is a classification technique usually limited to handle only two class classification (Yes/No, 0/1) problems. If the given dataset contains more than two labels (target y) then LR will not be an appropriate approach to use further analysis. In this concern, LDA will be the preferable solution for the classification as mentioned in the earlier problems.

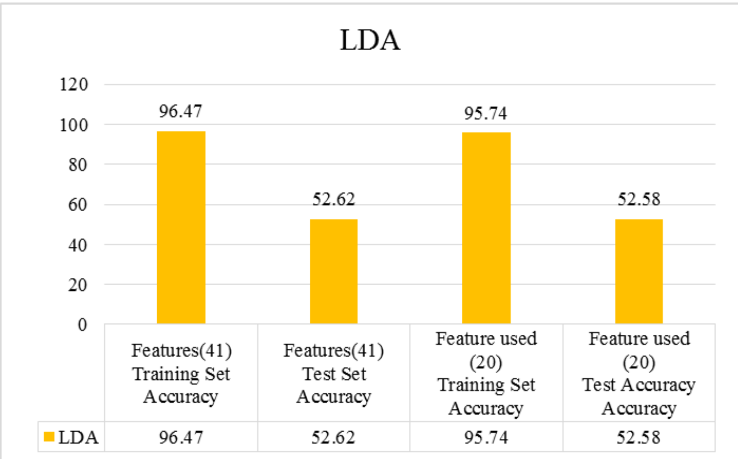

Figure 10. Results of LDA for training and test data set with 41 and 20 features

LDA model fits by Gaussian density to each class, assuming that all classes share the covariance matrix. The fitted LDA model can also be used to decrease the size of the training samples by projecting it to the most discriminative orders. Here, due to the different class labels and nature of our data, we used this model for calculating the efficiency. Figure 10 shows the experimental analysis of LDA model performed on given NSL KDD dataset.

5.8 Stochastic Gradient Decent (SGD)

SGD is a simple and very effective approach for linear models. SGD is widely used for high dimensional training samples datasets. It supports different loss functions and penalties for classification. It reduces training and test set errors like bios and variance trade-off. The training set produces bios trade-off and the test set produces variance trade-off. Therefore we need to implement more generalized or unbiased model where SGD can be a suitable model. This model also performs very well for highly imbalanced datasets. Moreover, there are a number of ways to improve efficiency of this model. Especially, hyper-parameter settings play a major role in this model. There are several hyper-parameters used in this model like learning rate, min-

Leaf, max-Leaf, estimator, train test split rate, random shuffle, criterion, depth and so on. Figure 11 illustrates the performance result of the SGD model with our NSL-KDD data set.

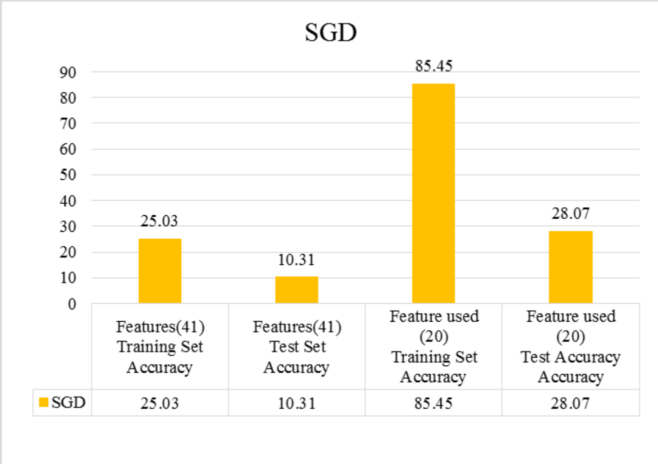

Figure 11. Results of SGD for training and test data set with 41 and 20 features

VI. Ensemble Model

In this section we have proposed an ensemble model for improvisation of our current research in order to reduce the false alarm rate in the intrusive environment. The ML environment gives various advantages to users: (i) we need not write or frame any rules or policies; (ii) the machine can build the rule itself based on the given training samples; (iii) once we construct the model, this model can be used for similar problems in real-world applications. (iv) it solves many real-world problems with good accuracy compared with traditional supervised learning models.

6.1 Ensemble Models

Ensemble learning is a model that can be used for combining more than one weak learner by iteratively grouping the weak learners so as to yield a better learner that can easily be trained the samples from the given dataset. An ensemble model is a classification technique used for a supervised learning algorithm as it can train itself and make prediction more accurately. The ensemble model is the appropriate solution to improve generalization capability and reduce the variance and bios trade-off in training and test samples compared with individual base learners. It also avoids over-fitting and under-fitting to the models.

Ensemble models are having some conditions and policies:

- Base learners must be accurate better than a random guess. It means base leaners must give a minimum percentage of prediction rates in normal binary classification problems. Otherwise, it cannot produce a more generalized model to our current issues.
- We must use various base learners because, every model over-fits with different ways from training samples and produce higher diversity.

Ensemble models produce various choices which are to improve the accuracy rates like heterogeneous and homogeneous patterns via:

1. Different Training Sub-samples
2. Different hyper-parameter settings like (min-leaf, max-leaf, learning rate, estimator, max-depth, in-depth, estimator and criterion.
3. Different training samples
4. Different algorithm selection

The main objective of the overhead options is to improve the more generalization capability in training and test samples and produce a very positive prediction rate. Figure 12 gives the structure of an ensemble model. Ensemble models can be classified in to three subcategories such as bagging, boosting and stacking. In most of the scenarios ensemble models use voting and averaging methods to make better predictions.

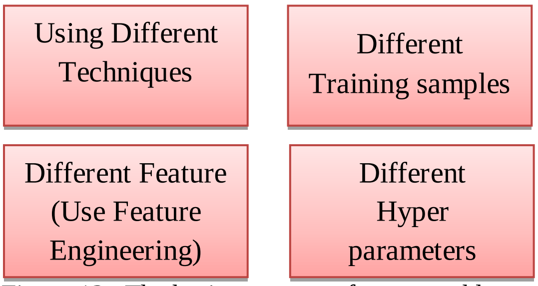

Figure 12. The basic structure of an ensemble model

6.2 Averaging

It is a popular mechanism for ensemble learning. There are two types of averaging mechanism that helps in making a better prediction (i) Simple Average and (ii) Weighted Average. In simple averaging method the actual output and the predicted output are combined to calculate the average by individual learners directly.

$$H(x) = \frac{1}{T}\sum_{i=1}^{T} h_i(x) \tag{2}$$

Weighting average gives the combined output with different weights.

$$H(x) = \sum_{i=1}^{T} w_i * \mathrm{h}_1(\mathrm{x}) \tag{3}$$

6.3 Bagging

We have used the bagging model first for our experimental analysis and obtained a very good result from the given datasets. Initially, the entire training populations are split into various training subsets using different hyper-parameters like training test split ratio and random state. After this process, the split training subset is passed into multiple similar classifiers and builds the models. Finally, we combine the results of the classifier and apply the voting mechanism for better prediction.

We have applied two bagging mechanism for our experimental analysis: (i) Random Forest and (ii) Bagging Classifier. Our experimental section shows how this bagging classifier plays a major role in ensemble paradigm. The Bagging Formula is given in the equation 4:

$$f(x) = 1/M \sum_{m=1}^{n} f_m(x) \tag{4}$$

6.3.1 Random Forest

Random Forest is a very familiar classification model in ensemble approach. It produces enhanced results for many real-world classification problems. It works like decision tree mechanism. Initially, there are several decision trees built independently by this model and each decision tree observed the training samples in different ways like different train samples and different features. Finally the results are combined and the voting mechanism is applied for better prediction. Based on the observation, random forest can be classified into two sub-categories like random subspace (each tree built using a different subset of features) and Bagging Classifiers (each tree built from a different subset of the training set). Bootstrap sampling is another important mechanism for bagging classifiers.

6.4 Boosting

Boosting is an iterative technique in which decision tree is built upon to improve the previous tree. Initially, all training samples have an equal probability of being chosen. If an observation was classified incorrectly, it tries to increase or modify the probabilities of each tree. Here, the weighted mechanism plays a major role such that correctly predicted instance are down-weighted and incorrectly predicted instances are up-weighted for each iteration. Finally, results are calculated by each tree using the vote and weight mechanism. In general boosting reduces the bias errors and tries to build more generalized predictive models. Sometimes it may be over-fitted on the given training samples.

6.4.1 ADA Boost

ADA boost is one of the simplest and more reliable models in boosting. It works on the decision tree mechanism. It is built on the multiple decision tree models sequentially. Each tree is correcting the errors from their previous tree model. In general, it uses a weighted mechanism for giving high priority weight to incorrectly predicted instance and subsequent build the next level of the tree to predict these values correctly [35].

6.4.2 Gradient Boost Classifier

Gradient boosting classifier (GBT) is another model for the ensemble approach and it works on both regression and classification problems. The main objective of this algorithm is to find the best split to minimize the error. It is a boosting technique and it combines weak learners in to form a strong learner using error minimization function. GBT uses a regression tree in base level learner and subsequent tree builds by error calculated on their previous tree. New errors are calculated by error function until the maximum number of iterations is reached. (Error function does not change).our experimental results show how the algorithm works on my given NSL KDD dataset.

6.4.3 XG Boost

The XG Boost is an advanced version of the gradient boosting algorithm. Nowadays, several ML domains like pattern recognition, medical domain, and security domain have used this algorithm. It has a consistent record for solving ML problems and it helps to reduce over-fitting and under-fitting in train test models. It has 10 times faster than the gradient boosting technique so it is suitable to perform parallel, big data, and Hadoop environment [35].

XG Boost has its own technique for missing values imputation so the user need not write any special function for identifying missing values. Moreover, it supports pruning and cross-validation functions. XG Boost permits a user to run cross-validation at each iteration of the boosting process and thus it is easy to get the precise best number of boosting iterations in a single run [36, 37].

VII. Stacking

In this section, we explained our proposed ensemble model called as stacking. Stacking is an ensemble approach also called as Meta Ensembling which can be used to combine information from multiple predictive models to generate a new model. Stacking is most effective when the base models are significantly different. Stacking consists of various stages like first level prediction (base learners) and second level prediction (Meta classifier).

Initially, we have trained the entire training samples by multiple machine learning algorithms. After successful completion of first level training, the predicted output (new dataset) is combined and given to the second stages of prediction algorithm (Meta-Classifier). The meta-classifier trains the results and builds the new model for future test set prediction. Finally, the test set samples trained by base learners and predicted output verified by the strong learners (Meta learners). It gives better results for standard classifiers. Figure 13 demonstrates how our proposed model achieves good results against the previous one.

In the last decade, the stacking model has been successfully used on a wide variety of predictive modeling problems to boost the models' prediction accuracy beyond the level obtained by any of the individual models.

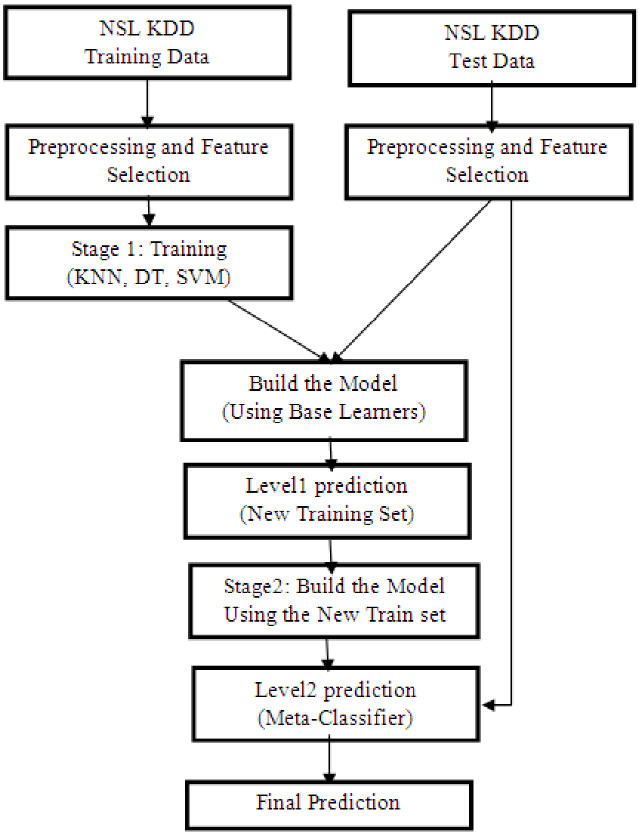

Figure 13. Proposed Ensemble Model Process Diagram

7.1 Experimental Results

We have analyzed NSL KDD Dataset by various machine learning techniques.

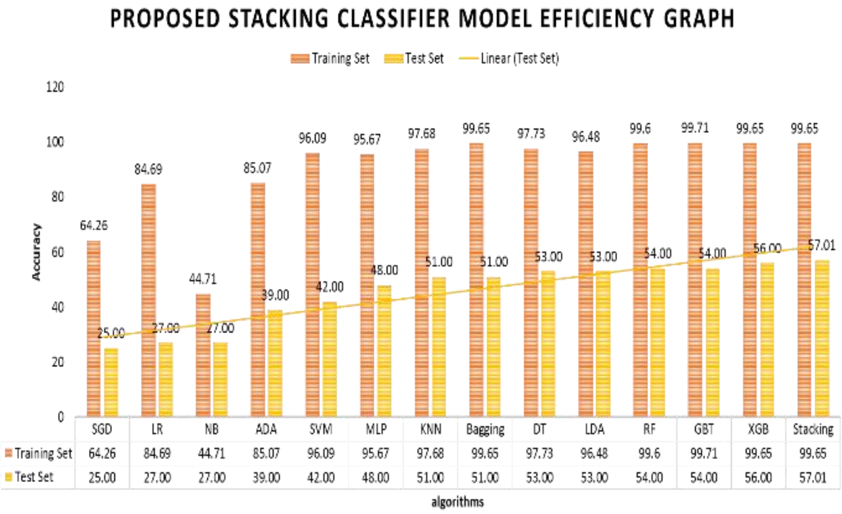

Figure 14. Comparison of ensemble results

Finally, we have achieved good result for ensemble classifiers as shown in table 4 and the comparison results are shown in figure 14.

Table 4 Result of ensemble classifier

Algorithm	Training Set Accuracy	Test Set Accuracy
SGD	64.26	25.00
LR	84.69	27.00
NB	44.71	27.00
ADA	85.07	39.00
SVM	96.09	42.00
MLP	95.67	48.00
KNN	97.68	51.00
Bagging	99.65	51.00
DT	97.73	53.00
LDA	96.48	53.00
RF	99.06	54.00
GBT	99.71	54.00
XGB	99.65	56.00
Stacking	**99.65**	**57.01**

VIII. Conclusions And Feature Work

In this paper, we have analyzed NSL KDD dataset by using various machine learning models. This research is helpful for identifying the better model for constructing IDS in a networked environment. Initially, we have worked with standard classifiers with the NSL KDD dataset. The experimental results show that those standard models are not capable to work with imbalanced datasets. So we have extended our work by using ensemble models like bagging, boosting and stacking. Our experimental results proved that the results are enhanced while comparing with the previous one. Especially our proposed stacking model achieved the expected results for other classification models. Current work only identifies a capable model for detecting intrusions and reducing false alarm rates. These results still could be improved by using the multiclass dataset. In the Future, we have planned to improve multiclass dataset accuracy by using feature selection techniques, label encoding, and cross-validation techniques.

References

[1]. Shameli-Sendi, A., Cheriet, M., Hamou-Lhadj, A., 2014. Taxonomy of intrusion risk assessment and response system. Comput. Secur. 45, 1–16.

[2]. Moustafa, N., Slay, J., Creech, G., 2017. Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. IEEE Transactions on Big Data 1–14.

[3]. Moustafa, N., Creech, G., Slay, J., 2018a. Anomaly detection system using beta mixture models and outlier detection. In: Progress in Computing, Analytics and Networking. Springer, pp. 125–135.

[4]. Pontarelli, S., Bianchi, G., Teofili, S., 2013. Traffic-aware design of a high-speed fpga network intrusion detection system. IEEE Trans. Comput. 62 (11), 2322–2334.

[5]. Wang, L., Jones, R., 2017. Big data analytics for network intrusion detection: a survey. Int. J. Network. Commun. 7 (1), 24–31.

[6]. Inayat, Z., Gani, A., Anuar, N.B., Khan, M.K., Anwar, S., 2016. Intrusion response systems: foundations, design, and challenges. J. Netw. Comput. Appl. 62, 53–74.

[7]. Anwar, S., Mohamad Zain, J., Zolkipli, M.F., Inayat, Z., Khan, S., Anthony, B., Chang, V.,2017. From intrusion detection to an intrusion response system: fundamentals, requirements, and future directions. Algorithms 10 (2), 39.

[8]. Zarpelao, B.B., Miani, R.S., Kawakani, C.T., de Alvarenga, S.C., 2017. A survey of intrusion detection in internet of things. J. Netw. Comput. Appl. 84, 25–37.

[9]. Corona, I., Giacinto, G., Roli, F., 2013. Adversarial attacks against intrusion detection systems: taxonomy, solutions and open issues. Inf. Sci. 239, 201–225.

[10]. Ahmed, M., Mahmood, A.N., Hu, J., 2016. A survey of network anomaly detection techniques. J. Netw. Comput. Appl. 60, 19–31.

[11]. Aburomman, A.A., Reaz, M.B.I., 2017. A survey of intrusion detection systems based on ensemble and hybrid classifiers. Comput. Secur. 65, 135–152.

[12]. Peng, J., Choo, K.-K.R., Ashman, H., 2016. User profiling in intrusion detection: a review. J. Netw. Comput. Appl. 72, 14–27.

[13]. Moustafa, N., Turnbull, B., Choo, K.R., 2018b. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things. IEEE Internet of Things Journal 1, https://doi.org/10.1109/ JIOT.2018.2871719.

[14]. Moustafa, N., Creech, G., Sitnikova, E., Keshk, M., 2017c. Collaborative anomaly detection framework for handling big data of cloud computing. In: Military Communications and Information Systems Conference (MilCIS), 2017. IEEE, pp. 1–6.
[15]. Zarpelao, B.B., Miani, R.S., Kawakani, C.T., de Alvarenga, S.C., 2017. A survey of intrusion detection in internet of things. J. Netw. Comput. Appl. 84, 25–37.
[16]. Sharma, S., Kaul, A., 2018. A survey on intrusion detection systems and honeypot based proactive security mechanisms in vanets and vanet cloud. Vehicular Communications 138–164.
[17]. Resende, P.A.A., Drummond, A.C., 2018. A survey of random forest based methods for intrusion detection systems. ACM Comput. Surv. 51 (3), 48.
[18]. S.Revathi, Dr A.Malathi,"A detailed analysis on NSL-KDD data set using various machine learning techniques for intrusion detection", Vol 2(12),pp. 1848-1853,Dec 2013.
[19]. He H, Garcia E. Learning from imbalanced data. IEEE Trans Knowl Data Eng. 2009;21(9):1263–84.
[20]. H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: a comprehensive review," J Netw Comput Appl, vol. 36, no. 1, pp. 16–24, 2013a/01/01/ 2013
[21]. Khraisat A, Gondal I, Vamplew P (2018) An anomaly intrusion detection system using C5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining. Springer International Publishing, Cham, pp 149–155
[22]. C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," J Netw Comput Appl, vol. 36, no. 1, pp. 42–57, 2013/01/01/ 2013
[23]. Butun I, Morgera SD, Sankar R (2014) A survey of intrusion detection systems in wireless sensor networks. IEEE Communications Surveys & Tutorials 16(1):266–282
[24]. Alazab. A, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in 2012 international symposium on communications and information technologies (ISCIT), 2012, pp. 296–301
[25]. Creech G, Hu J (2014a) A semantic approach to host-based intrusion detection systems using Contiguous and Dis-contiguous system call patterns. IEEE Trans Comput 63(4):807–819
[26]. Choudhary, Sarika, and Nishtha Kesswani. "Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT." *Procedia Computer Science* 167 (2020): 1561-1573
[27]. Tavallaee,M.,Bagheri,E.,Lu,W.,Ghorbani,A.A.,2009.A detailed analysis of the KDD cup 99 dataset, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, IEEE. pp.1–6.
[28]. Meena, G. , Choudhary, R.R., 2017. A review paper on ids classification using kdd 99 and nsl-kdd data set in weka, in: 2017 International Conference on Computer, Communications and Electronics (Comptelix), IEEE. pp.553–558.
[29]. Moustafa,N.,Slay,J.,2015.Unswnb 15: a comprehensive data set for network intrusion detection systems (unsw-nb 15 network data set), in: 2015 military communications and information systems conference (MilCIS), IEEE. pp.1–6.
[30]. A. G. García and M. J. Muñoz-Bouzo. Sampling-related frames in finite U-invariant subspaces. Appl. Comput. Harmon. Anal., 39:173-184, 2015
[31]. W. Lee "Margin and Boosting", Machine Learning proceeding of 14th international conference, pp. 1-9, 1997
[32]. Beant Kaur, Williamjeet Singh," Review on Heart Disease Prediction System using Data Mining Techniques", International Journal on Recent and Innovation Trends in Computing and Communication, Volume: 2 Issue: 10 ,2014
[33]. M.Akhil Jabbar, B.L Deekshatulu, Priti Chandra," Classification of Heart Disease Using K- Nearest Neighbor and Genetic Algorithm", ScienceDirect Procedia Technology 10 85 – 94,2013
[34]. Urvesh Bhowan, Mark Johnston and Mengije Zhang "Developing New Fitness Functions in Genetic Programming for Classification With Unbalanced Data" IEEE Transaction on the system, man and cybernetics—part b, volume 42, pp 406-421 (2012).
[35]. W. Lee "Margin and Boosting", Machine Learning proceeding of 14th international conference, pp. 1-9, 1997.
[36]. Chris, "RUSBoost: A Hybrid Approach to Alleviating Class Imbalance Problem", IEEE transactions on systems, man, and cybernetics part a: systems and humans, vol. 40, pp.185,197, 2010.
[37]. V. KrishnaVeni,T. Sobha Rani," On the Classification of Imbalanced Datasets" IJCST Vol. 2, SP 1, December 2011

R.Rajakumar, et. al. "Design of Intrusion Detection System using Machine Learning Techniques." *IOSR Journal of Computer Engineering (IOSR-JCE)*, 23(3), 2021, pp. 18-30.