INTRUSION DETECTION SYSTEM CLASSIFICATION USING DIFFERENT MACHINE LEARNING ALGORITHMS ON KDD-99 AND NSL-KDD DATASETS - A REVIEW PAPER

Ravipati Rama Devi[1] and Munther Abualkibash[2]

[1]Department of Computer Science, Eastern Michigan University,
Ypsilanti, Michigan, USA
[2]School of Information Security and Applied Computing, Eastern Michigan University,
Ypsilanti, Michigan, USA

ABSTRACT

Intrusion Detection System (IDS) has been an effective way to achieve higher security in detecting malicious activities for the past couple of years. Anomaly detection is an intrusion detection system. Current anomaly detection is often associated with high false alarm rates and only moderate accuracy and detection rates because it's unable to detect all types of attacks correctly. An experiment is carried out to evaluate the performance of the different machine learning algorithms using KDD-99 Cup and NSL-KDD datasets. Results show which approach has performed better in term of accuracy, detection rate with reasonable false alarm rate.

KEYWORDS

Intrusion Detection System, KDD-99 cup, NSL-KDD, Machine learning algorithms.

1. INTRODUCTION

Intrusion is a set of actions that attempts to compromise the integrity, confidentiality, or availability of any resource on a computing platform. An intrusion detection system (IDS) is a system that monitors network traffic for doubtful activity and matters alert when such activity is exposed. While anomaly discovery and reportage is the primary function, some interruption detection systems are capable of taking actions when malicious activity or irregular traffic is detected, including blocking traffic sent from suspicious IP addresses. An IDS is a combination of hardware and software that detects intrusions in the network. It is used to track, identify and detect the intruders. It is a combination of both hardware and software that detects intrusions in the network. IDS is used to detect unauthorized intrusions that occur in computer systems and networks. Feature selection for intrusion detection is most important factor for the success of intrusion detection system. The objectives of IDS are confidentiality, integrity, availability and accountability. Intrusion Prevention System is classified into four types:

a. Network Based Intrusion Prevention System –Monitors the entire network for suspicious traffic by analysing the protocol activity.
b. Wireless Intrusion Prevention System – Monitors the entire network for suspicious traffic by analysing the wireless networking protocols.
c. Network Behaviour Analysis – Examines the network to identify the threats such as distributed denial of service.

DOI: 10.5121/ijcsit.2019.11306

d. Host Based Intrusion Prevention System – An installed software which monitors a host for a suspicious activity by analysing the host.

Intrusions can be defined as the set of actions that attempt to compromise the confidential harmony, integrity or availability of a computer resource. When intruders deliberately gain unauthorized access of the resource, they try to access information, manipulate data, or render information in a system to make unreliable or unusable. An IDS is a union of hardware and software components that detect harmful or malicious attempts in the network. IDS can monitor all the network activities and hence can detect the signs of intrusions. The main aim of IDS is to inform the system administrator that any doubtful activity happened. There are two kinds of intrusion detection techniques:

A) **Anomaly Detection:** Recognizes malicious activities based on deviations from the normal conduct and considers these deviations as attacks.

B) **Misuse Detection:** Recognizes intrusions based on a standard pattern of the malicious activity. It can be very helpful for known attack patterns. Also, the rate of misplaced report is high. One disadvantage of Misuse Detection over Anomaly Detection is that it can only notice intrusions which contain known patterns of attack. An IDS monitors the activities of a given environment and decides whether these activities are malicious or normal based on system integrity, confidentiality and the availability of information resources. When building IDS, one needs to consider many issues, such as data collection, data preprocessing, intrusion recognition, reporting, and response.

Working of Intrusion Detection System: There are 4 steps in the working of IDS. They are Collecting Data, Selecting Features, Analyzing the Data, and last the Actions to be Performed.

a. **Collecting Data**: In order to do IDS, we need to collect the information about the network traffic like kinds of traffic, hosts and protocol details.
b. **Selecting Features**: From the large amount of data collected we need to extract the features which we need.
c. **Analyzing the Data**: The selected features data is analyzed to find whether the data is anomalous or not.
d. **Actions to be Performed**: IDS alarm or alert is made by the system administrator when an attack has occurred, and it tells about the type of the attack. IDS also participates in controlling the attacks by closing the network port and killing the processes.

2. DATASET DESCRIPTION

2.1. KDD-99 Cup Dataset

The KDD-99 cup dataset is the most used dataset for training the algorithms. It is the subset of DARPA-98 dataset. KDD-99 dataset is a multi-variety dataset. The dataset contains 4.8 million instances. The characteristics of the attributes used in the dataset are categorical and integer in nature. It has forty-two attributes. TheKDD-99 cup dataset consists of the following:

a. This dataset constitutes the TCP dump data of simulated network traffic captured in 1998 at Lincoln Labs. Seven weeks of traffic resulted in five million connection records used for a training set. A further two weeks of network traffic generated a test set with two million examples. The complete schedule can be found at the MIT website [1]. KDD-99 is a filtered version of this data.

b. KDD-99 has five classes of patterns like Normal, DoS (Denial of Service), U2R (User to Root), R2L (Remote to Local) and Probe (Probing Attack). Each intrusion category is further subclassified by the specific procedure used to execute that attack. We listed those attacks in Table 2.

Table 1:KDD-99 CUP train and test data distribution

Class	Training Set	Percentage	Test Set	Percentage
Normal	812,814	75.611%	60,593	19.481%
DoS	247,267	23.002%	229,853	73.901%
Probe	13,860	1.289%	4,166	1.339%
R2L	999	0.093%	16,189	5.205%
U2R	52	0.005%	228	0.073%
Total	1,074,992	100%	311,029	100%

c. As provided, KDD's training dataset contains 4,898,431 data points. However, due to high redundancy (78%), there exist only 1,074,992 unique data points. Similarly, the test dataset is highly redundant (89.5%); it was pruned from 2,984,154 to 311,029 patterns. We consider these reduced datasets in this paper.
d. Each pattern has 41 features, allocated to one of three categories: Basic, Traffic, and Content. However, analysis of the Mean Decrease Impurity metric [2] found 17to be irrelevant.
e. The skewed nature of the dataset is evident from Table I: 98.61% of the data belongs to either the Normal or DoS categories. As we see in later sections, this hampers the performance of classifiers on the remaining classes.
f. The non-stationary nature of the KDD-99 dataset is clearly visible from train and test distributions in Table I. The training set has 23% of its data as DoS examples versus 73.9% in the test set; Normal is 75.61% of the train set but only 19.48% of the test set. It has been demonstrated that such divergence negatively affects performance [3]. Fugate and Gattiker [4] investigates anomaly-detection techniques on KDD-99, accounting for this problem by segmenting the dataset and using a stationary partition for training.

The KDD dataset mainly consists of four types of attacks:

1. **Denial of Service (DOS)**: DOS is an attack category, which depletes the victim's resources thereby making it unable to handle legitimate requests – e.g., syn flooding.
2. **Remote to Local (R2L)**: Unauthorized access from a remote machine, the attacker intrudes into a remote machine and gains local access of the victim machine – e.g., password guessing.
3. **User to Root (U2R)**: Unauthorized access to local super user (root) privileges is an attack type, by which an attacker uses a normal account to login into a victim system and tries to gain root/administrator privileges by exploiting some vulnerability in the victim – e.g., buffer overflow attacks.
4. **Probing**: Surveillance and other probing attack's objective is to gain information about the remote victim – e.g., port scanning.

The training dataset is made up of 21 different attacks out of the 37 present in the test dataset. The known attack types are those present in the training dataset while the novel attacks are the additional attacks in the test dataset, i.e. not available in the training datasets. The attack types are grouped into four categories: DOS, Probe, U2R and R2L as in KDD dataset.

Table 2. Classification of attack types in KDD Dataset.

Classes	Attack types
Normal	Normal: 97277
DOS	Smurf: 280790 Neptune: 107201 Teardrop: 979 Pod: 264 Land: 21
R2L	Warezclient: 1020 Guess_passwd: 53 Warezmaster: 20 Imap: 12 ftp_write: 8 Multihop: 7 Phf: 4 Spy: 2
U2R	Buffer: 30 Rootkit: 10 Loadmodule: 9 Perl: 3
PROBE	Satan: 1589 Ipssweep: 1247 Portsweep: 1040 Nmap: 231

2.2. NSL-KDD Dataset

To solve the issues in KDD-99 cup dataset, researchers proposed a new dataset, NSL-KDD, which consists of only selected records from the complete KDD dataset and does not suffer from any of the issues. NSL-KDD is an effort by Tavallaee et al. [5] to rectifyKDD-99 and overcome its drawbacks. However, as the authors mention, the dataset is still subject to certain problems, such as its non-representation of low footprint attacks [6].

The benefits of using the NSL-KDD dataset are

1) No duplicate records in the test set which has better reduction rates.
2) The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD dataset.
3) NSL-KDD has fewer data points than KDD-99, all of which are unique. It is thus less computationally expensive to use for training machine learningmodels.

Table 3: NSL-KDD train and test data distribution

Class	Training Set	Percentage	Test Set	Percentage
Normal	67,342	53.458%	9,710	43.075%
DoS	45,927	36.458%	7,457	33.080%
Probe	11,656	9.253%	2,421	10.740%
R2L	995	0.790%	2,754	12.217%
U2R	52	0.041%	200	0.887%
Total	125,972	100%	22,542	100%

3. METHODOLOGY

In this section various machine learning techniques have been utilized to measure classification performance on the datasets and highlight their characteristics. After selecting the features from data, we needed to convert text into binary values and then label the data.

3.1. Feature Extraction:

According to Sung and Mukkamala [7], Kayacik, Zincir-Heywood et al. [8] and Lee, Shin et al., [9] most published results observing feature reduction on the KDD Cup '99 datasets are trained and tested on the '10%' training set only. The full training dataset contains 4,898,431 records. It contains 22 different attack types. The dataset is unlabeled whereas the 10% training set is labeled. Neural networks require floating point numbers for the input neurons, in the range of [−1, 1], and for target neurons floating point numbers should be in the range of [0, 1]. All features were pre-processed to fulfill this requirement. For nominal features with distinct values like UDP, ICMP, TCP we used UDP= [0,1], ICMP= [1,0] and UDP= [-1, -1]. For nominal features with large number of distinct values we assigned rank to them according to the number of occurrences in the test set. For the numeric feature's 'duration', 'src bytes' and 'dst bytes', we started with the removal of outliers before scaling the values to the range [−1, 1].

3.2. Data Labeling:

Algorithm uses supervised learning in which sets of data should be labeled for training. +1 is used for normal data and -1 is used for attacked data.

3.3. Analysis Metrics:

The confusion matrix consists of four values: True Positive, False Negative, False Positive, and True Negative.

a. True Positive (TP): It is the test result which detects the condition if the condition is present. The TP value in the confusion matrix is at cm [0][0].
b. False Negative (FN): It is the test result which does not detect the condition even if the condition is not present. The FN value in the confusion matrix is at cm [1][0].
c. False Positive (FP): It is the test result which detects the condition when the condition is absent. It is also called a False Alarm rate. The FP value in the confusion matrix is at cm [0][1].
d. True Negative (TN): It is the test result which does not detect the condition even if the condition is present. The TN value in the confusion matrix is at cm [1][1].

Performance can be measured using the

1. Accuracy: The accuracy (*AC*) is the proportion of the total number of predictions that were correct. It is given by the relation

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

2. False Alarm Rate: False Alarm rate is calculated as the number of all incorrect predictions divided by the number of true positives. It is given by the relation

$$\text{False Alarm rate} = \frac{FP+FN}{TP} \quad (2)$$

3. Error Rate: Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset. The best error rate is 0.0, whereas the worst is 1.0. It is given by the relation

$$\text{Error rate} = \frac{FP+FN}{TP+TN+FP+FN} \quad (3)$$

4. Recall: Recall can be defined as the ratio of the total number of correctly classified positive examples divided by the total number of positive examples. High Recall indicates the class is correctly recognized. It is also called Sensitivity, True Positive Rate or Detection Rate. It is given by the relation

$$\text{Recall} = \frac{TP}{TP+FN} \quad (4)$$

5. Precision: Precision is defined as the number of true positives over the number of true positives plus the number of false positives. It is given by the relation

$$\text{Precision} = \frac{TP}{TP+FP} \quad (5)$$

4. MACHINE LEARNING ALGORITHMS

There are three types of machine learning algorithms like Supervised learning, Semi-Supervised learning and Un-Supervised learning.

4.1. Supervised Learning

In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines which label should be given to new data based on patterns and associating the patterns to the unlabeled new data. Supervised Learning can be divided into 2 categories i.e., Classification and Regression.

Classification predicts the category that the data belongs to whereas Regression predicts a numerical value based on previous observed data.

Classification

Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, based on a training set of data containing observations (or instances) whose category membership is known.

A. Logistic Regression:

The objective of Logistic Regression (LR) is to create the best fitting model to establish a relationship or dependence between the class variable and the features [1]. For a test case with only two classes: 0 and 1, it basically predicts a value between 0 and 1 which is the probability that the class is 1 for an observation. The simple LR model is only suitable for binary classification, but with effort can be extended for a multiclass purpose. We form a linear expression of x:

$$\theta x = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots\ldots\ldots + \theta_n x_n \quad (6)$$

Where θ and x are vectors $[\theta_0, \theta_1, \ldots, \theta_n]$ and $[x_1, x_2, \ldots, x_n]$ respectively. And this fed to the hypothesis function will predict the value of y. Our goal is to determine the values of θ, so that for a given set of values of x, we get the correct y. The hypothesis of our LR is the sigmoid function.

$$h = g(z) = \frac{1}{1+e^{-z}} \quad (7)$$

We take the output(z) of the linear equation and give to the function g(z) which returns a squashed value, the value will lie in the range of 0 to 1. To understand how sigmoid function squashes the values within the range, let's visualize the graph of the sigmoid function.

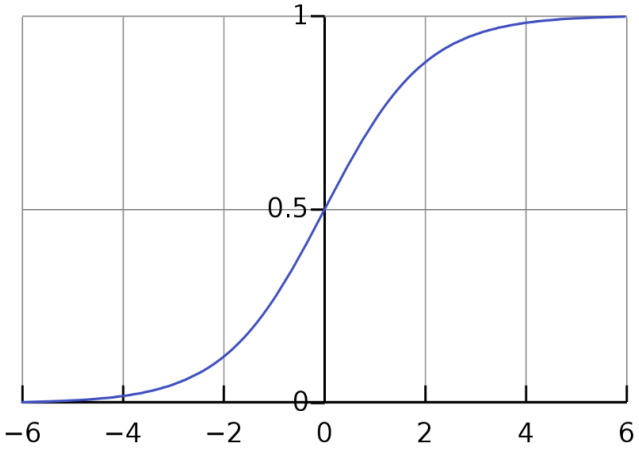

Fig1: Sigmoid Function Graph

The sigmoid function becomes asymptote to y=1 for positive values of z and becomes asymptote to y=0 for negative values of z.

Since, we are trying to predict class values, we cannot use the same cost function used in a linear regression algorithm. So, we use a logarithmic loss function to calculate the cost for misclassifying.

$$\begin{aligned} \text{Cost}(h_\theta(x), y) = &-\log(h_\theta(x)) \text{ if } y=1 \text{ or} \\ &-\log(1-h_\theta(x)) \text{ if } y=0 \end{aligned} \quad (8)$$

B. KNearest Neighbor Algorithm:

K Nearest Neighbor (KNN) is a managed machine learning algorithm useful for classification problems. KNN is a non-parametric and lazy learning algorithm which means there is no supposition for fundamental data distribution. Non-parametric means there is no assumption for underlying data distribution. In KNN, K is the number of nearest neighbors. The number of neighbors is the core deciding factor. It calculates the distance between the test data and the input and gives the prediction accordingly. The distance is calculated using the Euclidean distance formula.

$$\begin{aligned} d(p, q) &= \sqrt{(q1-p1)^2 + (q2-p2)^2 + \cdots + (qn-pn)^2} \\ &= \sqrt{\sum_{i=1}^{n} (qi-pi)^2} \end{aligned} \quad (9)$$

The data point which is located at the minimum distance from the test point is assumed to belong to the same class. The advantage of KNN is simple implementation and makes no prior assumption of the data. The disadvantage of KNN is the prediction time is quite high as it finds the distance between every data point.

KNN algorithm works as follows:

1. Load the data.
2. Initialize the value of k.
3. For getting the predicted class, iterate from 1 to total number of training data points.
    a. Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
    b. Sort the calculated distances in ascending order based on distance values.
    c. Get top k rows from the sorted array.
    d. Get the most frequent class of these rows.
    e. Return the predicted class.

C. Decision Tree:

A decision tree is one of most frequently and widely used supervised machine learning algorithms that can perform both regression and classification tasks. The intuition behind the Decision Tree algorithm is simple, yet also very powerful. It requires relatively less effort for training the algorithm and can be used to classify non-linearly separable data. It is very fast and efficient compared to KNN and other classification algorithms. Entropy and Information Gainare the most commonly used Attribute Selection Measures.

1. Entropy: Entropy is the degree or amount of uncertainty in the randomness of elements or in other words it is a measure of impurity**.**

$$E(S) = \sum_{i=1}^{c} -P_I \, log_2 P_i \qquad (10)$$

Entropy calculates the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero, and if the sample is an equally divided it has an entropy of one.

2. Information Gain: It measures the relative change in entropy with respect to the independent attribute. It tries to estimate the information contained by each attribute. Constructing a Decision Tree is all about finding the attribute that returns the highest information gain.

$$Gain\ (T, X) = Entropy(T) - Entropy\ (T, X) \qquad (11)$$

Where Gain (T, X) is the information gain by applying feature X. Entropy(T) is the entropy of the entire set, while the second term calculates the entropy after applying the feature the disadvantage of a Decision Tree Model is over fitting as it tries to fit the model by going deeper in the training set and thereby reducing test accuracy.

Decision Tree works as follows:

a. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
b. Make that attribute a decision node and breaks the dataset into smaller subsets.

c. It starts building the tree by repeating this process recursively for each child node until any one of the condition matches:

1. All the tuples belong to the same attribute value.
2. There are no more remaining attributes.
3. There are no more instances.

D. Support Vector Machine (SVM):

The SVM has been chosen because it represents a framework both interesting from a machine learning perspective and from an embedded systems perspective. An SVM is a linear and non-linear classifier, which is a mathematical function that can distinguish two different kinds of objects. These objects fall into classes, which is not to be mistaken for a Java class. Training a SVM can be illustrated with the following pseudo code:

Require: X and y loaded with training labeled data, $\alpha <= 0$ or $\alpha <=$ partially trained SVM

1.C<= some value (10 for example)
2. repeat
3. for all $\{x_i, y_i\}$, $\{x_j, y_j\}$ do
4. Optimize α_i and α_j
5. end for
6. until no changes in α or other resource constraint criteria met
Ensure: Retain only the support vectors ($\alpha_i > 0$)

E. Naive Bayes:

The Naive Bayes algorithm is an intuitive method that uses the probabilities of each attribute belonging to each class to make a prediction. It is the supervised learning approach used to model a predictive modeling problem probabilistically. Naive Bayes has become one of the most efficient learning algorithms [10].Naive Bayes simplifies the calculation of probabilities by assuming that the probability of each attribute belonging to a given class value is independent of all other attributes [11]. This is a strong assumption but results in a fast and effective method. The probability of a class value given a value of an attribute is called the conditional probability. By multiplying the conditional probabilities together for each attribute for a given class value, we have a probability of a data instance belonging to that class. To make a prediction, we can calculate probabilities of the instance belonging to each class and select the class value with the highest probability. Naive Bayes is often described using categorical data because it is easy to describe and calculate using ratios. A more useful version of the algorithm for our purposes supports numeric attributes and assumes the values.

The advantages of using Naïve Bayes are

- It is easy to apply and predicts the class of a test data set fast. It also performs well in multi-class predictions.
- When the assumption of independence holds, a Naive Bayes classifier performs better compared to the other models like logistic regression as less training data is needed.
- It performs well in the case of categorical input variables compared to a numerical variable(s). For the numerical variable, a normal distribution is assumed (bell curve, which is a strong assumption).

F. Multi-Layer Perceptron (MLP):

An MLP can be viewed as a logistic regression classifier where the input is first transformed using a learnt non-linear transformation. An MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called back propagation for training.

Multi-layer perceptron works as follows:

1. For all input neurons i do
2. Set $a_i \leftarrow x_i$
3. End for
4. For all hidden and output neurons i in topological order do
5. Set $net_i \leftarrow w_{i0} + \Sigma_{j \epsilon pred(i)} w_{ij} a_j$
6. Set $a_i \leftarrow f_{\log} \quad (net_i)$
7. End for
8. For all output neurons i do
9. assemble a_i in output vector y
10. End for
11. Return y.

G. Random Forest:

Random Forest algorithm is a supervised classification algorithm. We can see it from its name, which is to create a forest by some way and make it random. There is a direct relationship between the number of trees in the forest and the results it can get i.e., the larger the number of trees, the more accurate the result. But one thing to note is that creating the forest is not the same as constructing the decision with information gain or gain index approach. The difference between Random Forest algorithm and the Decision Tree algorithm is that in Random Forest, the processes of finding the root node and splitting the feature nodes will run randomly. There are two stages in Random Forest algorithm, one is random forest creation, the other is to make a prediction from the random forest classifier created in the first stage. For applications in classification problems, Random Forest algorithm will avoid the over fitting problem and for both classification and regression tasks, the same random forest algorithm can be used. The Random Forest algorithm can be used for identifying the most important features from the training dataset.

Random Forest Algorithm Creation Method:

1. Randomly select **"**K**"** features from total **"**m**"** features where k << m
2. Among the **"**K**"** features, calculate the node **"**d**"** using the best split point
3. Split the node into daughter nodes using the best split
4. Repeat the a to c steps until "l" number of nodes has been reached
5. Build forest by repeating steps a tod for "n" number times to create "n" number of trees.

The random forest prediction pseudo code is shown below:

1. Takes the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target)
2. Calculate the votes for each predicted target
3. Consider the high voted predicted target as the final prediction from the random forest algorithm

H. AdaBoost Algorithm:

In this algorithm, we initially extract features and specify data labeling. Later for AdaBoost algorithm we need a group of weak classifiers. In this algorithm, we use decision stump as weak classifiers. Finally, a strong classifier is obtained by combining the weak classifiers and this has the higher classification accuracy when compared to weak classifiers. [12]

AdaBoost Algorithm

1. Initialize weights for each data point as $w(x_i, y_i) = \frac{1}{n}$, I = 1,….,n.
2. For iterations m= 1,,M
   a. Fit weak classifiers to the data set and select the one with the lowest weighted classification error:
      $€_m = E_w[1_{y \neq f(x)}]$ (12)
   b. Calculate the weight of the m classifier:
      $\theta_m = \frac{1}{2}\ln(\frac{1-€_m}{€_m})$ (13)
3. Update the weight for each data point as:
   $w_{m+1}(x_i, y_i) = \frac{w_m(x_i, y_i)\exp[-\theta_m y_i f_m(x_i)]}{z_m}$ (14)
   Where z_m is a normalization factor that ensures the sum of all instance weights is equal to 1.

After M iteration we can get the final prediction by summing up the weighted prediction of each classifier.

Table 4: Number of samples in KDD dataset

Dataset	Normal	Attacks			
		Dos	R2L	U2R	Probe
Training	97278	391458	1126	52	4107
Testing	60593	229853	16189	228	4166

Table 5: Results with uniform initial weights and without over fitting handling.

Training set		Testing set	
Detection Rate	False Alarm Rate	Detection Rate	False Alarm Rate
99.25%	2.766%	90.738%	3.428%

4.2. Unsupervised Learning

The unsupervised machine learning algorithm is used for exploring the structure of the information, extracting valuable insights, detecting patterns and implementing this into its operation to increase efficiency.

Unsupervised learning algorithms apply the following techniques to describe the data:

- **Clustering:** It is an exploration of data used to segment it into meaningful groups (i.e., clusters) based on their internal patterns without prior knowledge of group credentials. The credentials are defined by similarity of individual data objects and aspects of its dissimilarity from the rest (which can also be used to detect anomalies).
- **Dimensionality reduction:** There is a lot of noise in the incoming data. Machine learning algorithms use dimensionality reduction to remove this noise while distilling the relevant information.

Most commonly used algorithms are k-means for clustering problems and Apriority algorithm for association rule learning problems.

A.K-means: K-Means is an unsupervised clustering algorithm that is used to group data into k-clusters. K means it is an iterative clustering algorithm which helps to find the highest value for every iteration. Initially, the desired number of clusters are selected. In this clustering method, the data points are clustered into k groups. A larger k means smaller groups with more granularity in the same way. A lower k means larger groups with less granularity.
Algorithm for K-Means is

Repeat the two steps below until clusters and their mean is stable:

1. For each data item, assign it to the nearest cluster center. Nearest distance can be calculated based on distance algorithms.
2. Calculate mean of the cluster with all data items.

Limitations of K-Means algorithm:

- K-Means algorithm does not work well with missing data.
- It uses a random seed to generate clusters which makes the results undeterministic and random. We can however supply our own random seed number.
- It can get slower with larger data items.
- It does not work well with categorical (textual) data.

B. Apriori Algorithm:

The Apriori algorithm is a categorization algorithm. Some algorithms are used to create binary appraisals of information or find a regression relationship. Others are used to predict trends and patterns that are originally identified. Apriority is a basic machine learning algorithm which is used to sort information into categories. Sorting information can be incredibly helpful with any data management process. It ensures that data users are apprised of new information and can figure out the data that they are working with.

The Apriori algorithm works as follows

Join Step: C_k is generated by joining $L_{k-1}$with itself
Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset• Pseudo-code:

C_k : Candidate itemset of size k
L_k: frequent itemset of size k
L_1= {frequent items};
for (k= 1; L_k! =∅; k++) do begin
    C_{k+1}= candidates generated from L_k;
    for each transaction tin database do
        increment the count of all candidates in C_{k+1}that are contained in t
    L_{k+1}= candidates in C_{k+1} with min_ support
    End
Return $\cup_k L_k$;

4.3. Semi Supervised Learning

Semi-supervised learning algorithms represent a middle ground between supervised and unsupervised algorithms. The semi-supervised model combines some aspects of both into a model of its own.

Semi Supervised learning algorithms work as follows

1. Semi-supervised Machine Learning algorithms use a limited set of labeled sample data to shape the requirements of the operation (i.e., train itself).
2. The limitation results in a partially trained model that later gets the task to label the unlabeled data. Due to the limitations of the sample data set, the results are considered pseudo-labeled data.
3. Finally, labeled and pseudo-labeled data sets are combined, which creates a distinct algorithm that combines descriptive and predictive aspects of supervised and unsupervised learning.

5. RESULTS

Table 6: Accuracy and False alarm rate of supervised machine learning algorithms using KDD Dataset

Algorithms	Accuracy	False Alarm Rate
Logistic Regression	79.7%	2.5
Decision tree	81.05%	2.85
KNN	94.17%	5.82
SVM	83.09%	16.90
Random Forest	99.0%	2.43
Adaboost	90.73%	3.42
Multi-Layer Perceptron	80.5%	1.94
Naïve Bayes	92.4%	3.8

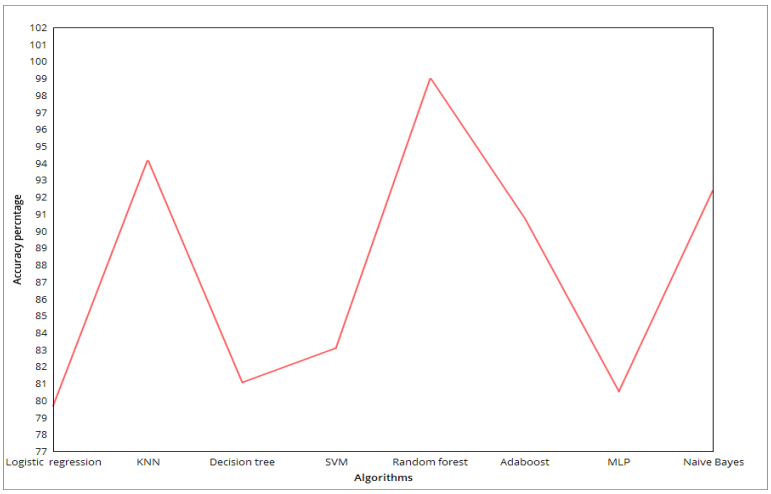

Fig 2: Accuracy percentages of supervised Machine learning algorithms on KDD-99 cup Dataset

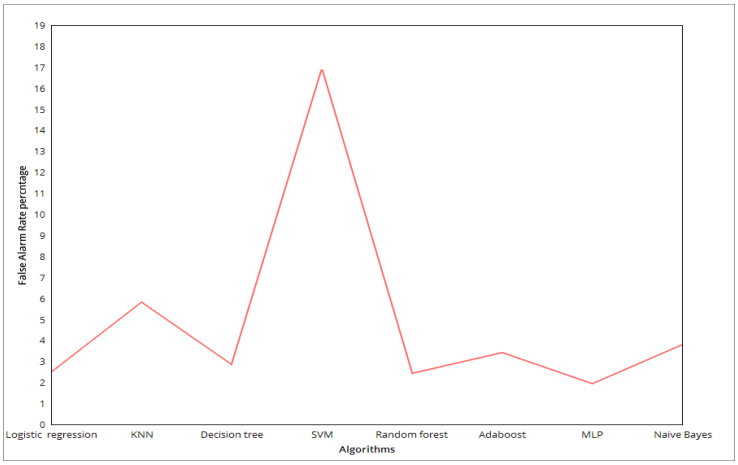

Fig 3: False Alarm Rate of supervised Machine learning algorithms on KDD-99 cup Dataset

Table 7: Accuracy and False Alarm Rate of supervised machine learning algorithms Using NSL-KDD Dataset

Algorithms	Accuracy	False Alarm rate
Logistic Regression	97.4%	4.7
Random Forest	99.7%	3.2
Stochastic Gradient Descent	97.4%	4.8
Naive Bayes	89.5%	2.1
Adaboost	89.3%	4.5
Multi-Layer Perceptron	88.9%	3.9

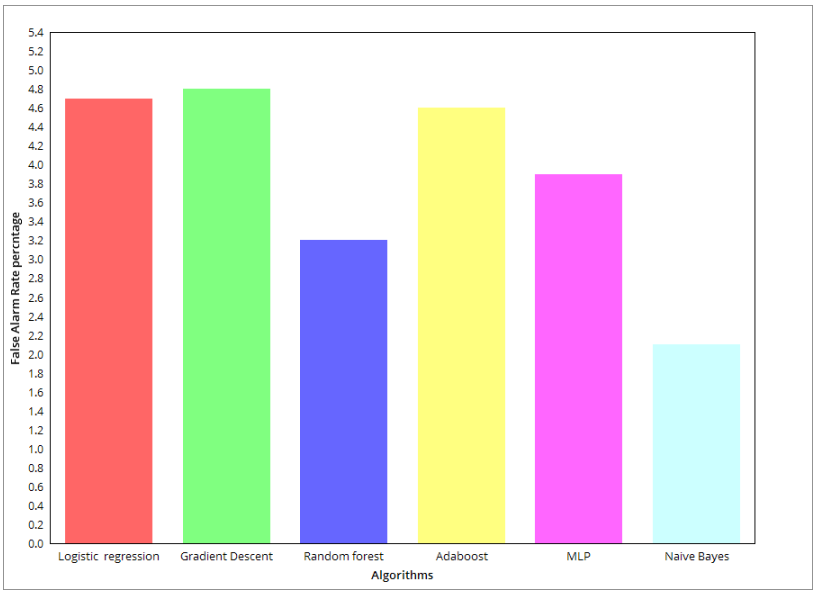

Fig 4: False Alarm rate of supervised Machine learning algorithms on NSL-KDD Dataset

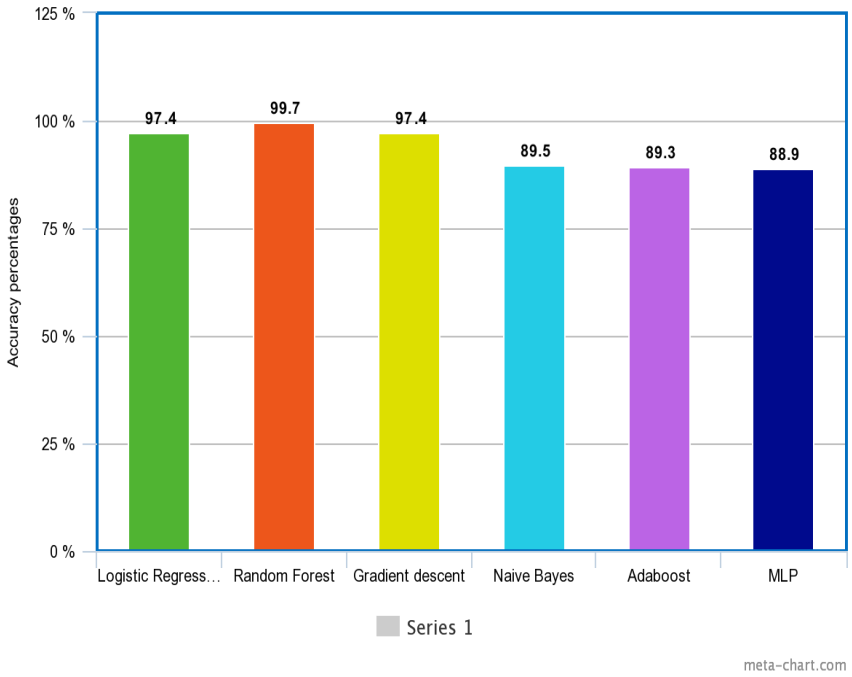

Fig 5: Accuracy percentages of supervised Machine learning algorithms on NSL-KDD Dataset.

6. CONCLUSIONS

In this survey we have introduced an overview of different machine learning algorithms for Intrusion Detection System (IDS) and different detection methodologies, and classifiers for KDD-99 and NSL-KDD dataset. As per the studied of techniques suggested by various authors, the ways it can detect the intruder are presented here. The experiment results show that KNN is having high false rate and detection rate but AdaBoost algorithm has a very low false rate with

high detection rate and run speed of algorithm is faster when compared with other supervised algorithms. Our Further goal is to implement the unsupervised algorithms and find out is there any algorithm which is better than AdaBoost.

REFERENCES

[1] "DARPA98 attack description and schedule," https://www.ll.mit.edu/ideval/docs/attacks.html, 1998, retrieved December 15, 2016.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees, 1984.

[3] D. A. Cieslak and N. V. Chawla, "A framework for monitoring classifiers' performance: when and why failure occurs? "Knowledge and Information Systems, vol. 18, no. 1, pp. 83–108, 2009.

[4] M. Fugate and J. R. Gattiker, "Anomaly detection enhanced classification in computer intrusion detection," in Pattern Recognition with Support Vector Machines, vol. 2388, 2002, pp. 186–197.

[5] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, 2009.

[6] J. McHugh, "Testing Intrusion detection systems: a critique of the 1998and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 262–294, 2000.

[7] S. Sung, A.H. Mukkamala. "Identifying important features for intrusion detection using support vector machines and neural networks". In Proceedings of the Symposium on Applications and the Internet (SAINT), pp. 209–216. IEEE.

[8] H. Kayacik, A. Zincir-Heywood and M. Heywood. "Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets". In Proceedings of the Third Annual Conference on Privacy, Security and Trust (PST). 2005.

[9] C. Lee, S. Shin and J. Chung. "Network intrusion detection through genetic feature selection". In Seventh ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD), pp. 109–114. IEEE Computer Society,2006.

[10] H. Zhang and J.Su., "Naive Bayes for optimal ranking", Journal of Experimental and Theoretical Artificial Intelligence. 2008, 20: 79-93.

[11] Z. Muda, W. Yassin, M.N. Sulaiman, N.I. Udzir, "Intrusion Detection based on K-Means Clustering and Naïve Bayes Classification" 2011 7thInternational Conference on IT in Asia (CITA)".

[12] Weiming Hu, Steve Maybank, "AdaBoost-Based Algorithm for Network Intrusion Detection". In IEEE transaction on systems, MAN, and CYBERNETICS, APRIL 2008.