****

**Performance Analysis of Network Intrusion Detection Using Classical Supervised Machine Learning Technique**

**Submitted by:**

**Md. Tariqul** Islam   
 151010400040   
 Dept. of IT

**Submitted to:  
Md Mynoddin**Lecturer, Dept. of CSE  
UITS

**Declaration**

This is to declare that the work presented in this thesis is the outcome of the investigations carried out by us under the supervision Md. Mynoddin, Lecturer, Department of Information Technology (IT), University of Information Technology & Sciences, Dhaka. It is further declared that neither this thesis nor any part there of has been submitted anywhere else for the award of any degree or diploma.

**Candidates**

**....………………… …………………..... ..………………….**

Md.Tariqul Islam Shanta Bhuiyan Tapas Biswas

ID- 151010400040 ID- 13310193 ID- 142010400014

Department of IT Department of IT Department of IT

UITS,Dhaka-1212. UITS,Dhaka-1212. UITS,Dhaka-1212.

**Supervised By**

**………………………….**

Md Mynoddin

Lecturer,

Department of Information Technology

UITS,Dhaka-1212.

**Acknowledgement**

First and foremost, I offer my sincerest gratitude to my advisor Md. Mynoddin, who has backed us, all around our thesis with his perseverance and knowledge. This research would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

Secondly, I would like to thank my research colleagues for their encouragement throughout my research work.

Finally, I would like to thank my parents for their unending love, support and understanding during my research work.

**Abstract**

Although KDD99 dataset is more than 15 years old, it is still widely used in academic research. To investigate wide usage of this dataset in Machine Learning Research (MLR) and Intrusion Detection Systems (IDS); this study reviews 149 research articles from 65 journals indexed in Science Citation Index Expanded and Emerging Sources Citation Index during the last six years .If we include papers presented in other indexes and conferences, number of studies would be tripled. The number of published studies shows that KDD99 is the most used dataset in IDS and machine learning areas, and it is the de facto dataset for these research areas. To show recent usage of KDD99 and the related sub-dataset (KDD) in IDS and MLR, the following descriptive statistics about the reviewed studies are given: main contribution of articles, the applied algorithms, compared classification algorithms, software toolbox usage, the size and type of the used dataset for training and testing, and classification output classes (binary, multi-class). In addition to these statistics, a checklist for future researchers that work in this area is provided.

**Keywords:** Machine Learning · KDD99 · SVM · Intrusion Detection · Supervised Learning · KNN .Naive Bayes . Linear Regression . Decision Tree. Adaboost. Random Forest .

**Table of Contents**

|  |  |
| --- | --- |
| Title | Page No |
| Declaration | I |
| Acknowledgement | II |
| Abstract | III |
| Table of Contents | IV |
| List of figure |  |
| List of Table |  |
| List of Abbreviations and Symbols |  |
| Chapter -1 : Introduction  1.1 History of IDS  1.2 Scope of the research  1.3 Importance of the research |  |
| Chapter -2: Types of IDS Presence in Network  2.1 Types of Attacks  2.2 IDS Approaches |  |
| Chapter -3: Description Machine Learning Techniques  3.1 System Model  3.2 Linear Regression Theory  3.3 Naïve Bayes Classifier  3.3.1 Bayes’ Theorem  **3.3.2 Naive assumption**  3.4 K-Nearest Neighbors Algorithm  3.5 Random forests  3.6 Decision Trees  3.7 Support Vector Machine (SVM) |  |
| Chapter – 4: Dataset descriptions, Methodology, Step diagram  4.1 KDDCup99  4.2 Methodology |  |
| Chapter - 5: Experimental Result Analysis and Performance    5.1 Preprocessing ……………………………………………….  5.2 Performance Analysis ………………………………………  5.2.1 Logistic Regression Results……………………….  5.2.2 Gaussian Naïve Bayes Results ……………………  5.2.3 K Nearest Neighbors (KNN) Results ……………..  5.2.4 Decision Tree Classifier Results ………………….  5.2.5 AdaBoost Classifier Results ………………………  5.2.6 Random Forest Classifier Results ………………...  5.2.7 Support Vector Machine (SVM) Classifier Results  5.3 Result Analysis …………………………………………….. |  |
| Chapter – 6: Conclusion |  |

**List of Figure**

|  |  |  |
| --- | --- | --- |
| **Sl No.** | **Title** | **Page No** |
| 01 | Fig 3.1 Proposed supervised machine learning classifier system | 15 |
| 02 | Fig 3.2 Logistic Regression | 17 |
| 03 | Fig 3.3 K-Nearest neighbors algorithm | 20 |
| 04 | Fig 3.4: SVM classifier in two dimensional problem spaces | 24 |
| 05 | Fig 4.1 A portion of KDD Dataset | 33 |
| 07 | Fig 5.1: Performance analysis of different Machine Learning Classifiers | 41 |
| 08 | Fig 5.2 Intrusion Detection accuracy for different technique | 42 |
| 09 | Figure 5.3: Line representation Intrusion Detection accuracy for different technique | 42 |

**List of Table**

|  |  |  |
| --- | --- | --- |
| **Sl No.** | **Title** | **Page No** |
| 01 | Table-4.1 Datasets Analysis | 32 |
| 02 | Table-4.2 Features type in KDD | 32 |
| 03 | Table-5.1 List of classical machine learning technique | 34 |
| 04 | Table-5.2 Confusion Matrix for Logistic Regression Analysis | 35 |
| 05 | Table-5.3 Logistic Regression Analysis results | 35 |
| 06 | Table-5.4 Confusion Matrix for Gaussian Naïve Bayes | 36 |
| 07 | Table-5.5 Gaussian Naïve Bayes results | 36 |
| 08 | Table-5.6 Confusion Matrix for K-Nearest Neighbors | 36 |
| 09 | Table-5.7 K-Nearest Neighbors results | 37 |
| 10 | Table-5.8 Confusion Matrix for Decision Tree Classifier | 37 |
| 11 | Table-5.9 Decision Tree Classifier results | 37 |
| 12 | Table-5.10 Confusion Matrix for Adaboost | 38 |
| 13 | Table-5.11 Adaboost results | 38 |
| 14 | Table-5.12 Confusion Matrix for Random Forest | 38 |
| 15 | Table-5.13 Random Forest results | 39 |
| 16 | Table-5.14 Confusion Matrix SVM Classifier | 39 |
| 17 | Table-5.15 SVM Classifier reults | 39 |
| 18 | Table-5.16 Intrusion Detection accuracy for different technique | 40 |

**Chapter-1: Introduction**

With the wide spreading usages of internet and increases in access to online contents, cybercrime is also happening at an increasing rate [1-2]. Intrusion detection is the first step to prevent security attack. Hence the security solutions such as Firewall, Intrusion Detection System (IDS), Unified Threat Modeling (UTM) and Intrusion Prevention System (IPS) are getting much attention in studies. IDS detects attacks from a variety of systems and network sources by collecting information and then analyzes the information for possible security breaches [3]. The network based IDS analyzes the data packets that travel over a network and this analysis are carried out in two ways. Till today anomaly based detection is far behind than the detection that works based on signature and hence anomaly based detection still remains a major area for research [4-5]. The challenges with anomaly based intrusion detection are that it needs to deal with novel attack for which there is no prior knowledge to identify the anomaly. Hence the system somehow needs to have the intelligence to segregate which traffic is harmless and which one is malicious or anomalous and for that machine learning techniques are being explored by the researchers over the last few years [6]. IDS however is not an answer to all security related problems. For example, IDS cannot compensate weak identification and authentication mechanisms or if there is a weakness in the network protocols.

* 1. **History of IDS**

Software to detect network intrusions protects a computer network from

unauthorized users, including perhaps insiders.  The intrusion detector learning task is to build a predictive model (i.e. a classifier) capable of distinguishing between ``bad'' connections, called intrusions or attacks, and ``good'' normal connections.

The 1998 DARPA Intrusion Detection Evaluation Program was prepared and managed by MIT Lincoln Labs. The objective was to survey and evaluate research in intrusion detection.  A standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment, was provided.  The 1999 KDD intrusion detection contest uses a version of this dataset.

Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a local-area network (LAN) simulating a typical U.S. Air Force LAN.  They operated the LAN as if it were a true Air Force environment, but peppered it with multiple attacks.The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic.  This was processed into about five million connection records.  Similarly, the two weeks of test data yielded around two million connection records.A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol.  Each connection is labeled as either normal, or as an attack, with exactly one specific attack type.  Each connection record consists of about 100 bytes.

Studying the field of intrusion detection first started in 1980 and the first such model was published in 1987 [7]. For the last few decades, though huge commercial investments and substantial research were done, intrusion detection technology is still immature and hence not effective [7]. While network IDS that works based on signature have seen commercial success and widespread adoption by the technology based organization throughout the globe, anomaly based network IDS have not gained success in the same scale. Due to that reason in the field ofIDS, currently anomaly based detection is a major focus area of research and development [8]. And before going to any wide scale deployment of anomaly based intrusion detection system, key issues remain to be solved [8]. But the literature today is limited when it comes to compare on how intrusion detection performs when using supervised machine learning techniques [9]. To protect target systems and networks against malicious activities anomaly-based network IDS is a valuable technology. Despite the variety of anomaly-based network intrusion detection techniques described in the literature in recent years [8], anomaly detection functionalities enabled security tools are just beginning to appear, and some important problems remain to be solved.

* 1. **Scope of the research**

The network based IDS analyzes the data packets that travel over a network and this analysis are carried out in two ways. Till today anomaly based detection is far behind than the detection that works based on signature and hence anomaly based detection still remains a major area for research [4-5]. The challenges with anomaly based intrusion detection are that it needs to deal with novel attack for which there is no prior knowledge to identify the anomaly. Hence the system somehow needs to have the intelligence to segregate which traffic is harmless and which one is malicious or anomalous and for that machine learning techniques are being explored by the researchers over the last few years [6]. IDS however is not an answer to all security related problems. For example, IDS cannot compensate weak identification and authentication mechanisms or if there is a weakness in the network.

* 1. **Importance of the research**

Several anomaly based techniques have been proposed including Linear Regression, Support Vector Machines (SVM), Genetic Algorithm, Gaussian mixture model, k-nearest neighbor algorithm, Naive Bayes classifier, Decision Tree [3,5]. Among them the most widely used learning algorithm is SVM as it has already established itself on different types of problem [10]. One major issue on anomaly based detection is though all these proposed techniques can detect novel attacks but they all suffer a high false alarm rate in general. The cause behind is the complexity of generating profiles of practical normal behavior by learning from the training data sets [11]. Today Artificial Neural Network (ANN) are often trained by the back propagation algorithm, which had been around since 1970 as the reverse mode of automatic differentiation [12].

The major challenges in evaluating performance of network IDS is the unavailability of a comprehensive network based data set [13]. Most of the proposed anomaly based techniques found in the literature were evaluated using KDD CUP 99 dataset [14]. In this paper we used SVM and KNN –two machine learning techniques, on KDDCup99 [15] which is a popular benchmark dataset for network intrusion.

The promise and the contribution machine learning did till today are fascinating. There are many real life applications we are using today offered by machine learning. It seems that machine learning will rule the world in coming days. Hence we came out into a hypothesis that the challenge of identifying new attacks or zero day attacks facing by the technology enabled organizations today can be overcome using machine learning techniques. Here we developed a supervised machine learning model that can classify unseen network traffic based on what is learnt from the seen traffic. We used both SVM,ANN and DNN learning algorithm to find the best classifier with higher accuracy and success rate. This paper is organized as follows: overview of the system model and design is explained in section III, the system experimental analysis is given in section IV. Section V contains the implementation discussion. Finally, section VI concludes the paper.

**Chapter-2: Types of IDS Presence in Network**

In this section some of the fame intrusion categorization traces [26,27] discussed for the birth of zero-day exploits during a decade. They are known to be,

**2.1 Types of Attacks**

1. **Denial-Of-Service attacks**

Denial-Of-Service attacks make the compromised systems or Host service temporarily suspended over connected to network resources. In other terms, make specific system service temporarily interrupt its network resources connected to the internet. Based on these intrusion categorization ping-of-death attacks, a teardrop attack exists.

1. **Probing attacks**

Probing attacks is an attempt that resembles working nature of Denial-Of-Service attacks. Moreover it act as initial phase of detecting the information about supporting infrastructure of database services and web applications. Based on these port scanning attacks, path revealing, directory traversal, application scanning of entire site such as remote execution, transactions flow etc.

1. **User-To-Root attacks**

User-To-Root attack is type of exploit in which attacker starts access the user's account intentionally by sniffing passwords with intent to gain root access to the specific or compromised machine. Buffer overflow attack is typical example for this categorization in intrusion traces.

1. **Root-To-Local attacks**

Root-To-Local attacks are an attempt to make unauthorized access to local machine by guessing of user name and passwords. Dictionary attack is most relevant example for these intrusion categories. Moreover, some of the fame intrusion traces are ftp, telnet, pop, r10gin are prominent examples rooted from Root-To-Local intrusion traces.

1. **Botnet attacks**

Botnet is an attempt to make single machine or group of machines make them to act as remote control under the compromised networks with intent to spread enormous of malware functions to cause collateral damage either to neither compromised single machine nor group of machines. Some of malicious activities are handled as botnet known as virus, denial-of-service attacks, spam, zero day exploits etc.

**2.2 IDS Approaches**

**A. Data mining based detection approaches**   
  
 Designing detection system over intrusion problem seems to be research challenge [28,29] to cyber community. The objective of Intrusion Detection Systems is to provide efficient wall of defense act over new type of intrusion detecteds connected to network. And also detection model should be built to monitor the network traffic and capable to analyze the suspicious intrusion detected in case of presence. Based on this, intrusion detection has categorized into Signature based Intrusion Detection Systems and Anomaly based Intrusion Detection Systems. Major credit of Signature based detection systems is able to trap malicious content presence over network traffic in faster time. But major critic is able to identify only known suspicious content which is already trained by the system. Anomaly detection approaches has designed to overcome drawback of Signature based detection systems. Merit of Anomaly detection approaches is to trap network anomalies avail over network traffic. Based on this data mining based detection approaches have designed with four variant classifier [30-34] algorithms and they are known to be flow-correlation algorithm, classification algorithm, clustering and association rule. Inflow-correlation algorithm, flow objects will be compared to characterize to create metric other than packet content. Merit of this approach is very effective in utilize the characteristic value as input into one or more functions objective to create metric to decide whether flows are correlated. In classification based detection approach, incoming packet will be compared with one of previous patterns. But the draw back behind these, are not efficient to detect new attacks. In clustering based approaches, split up the entire dataset into subcategories of specified groups or clusters by determining identical features but the limitations, it does not require labeled dataset for training. In association rule based detection approaches, derive the implied relationship between data items related to the set of project types and measures and finally ends with analyzing the fact. The common draw back behind these approaches is no real-time detection. Because they are structure and protocol dependent i.e., rely on same pattern of work and finally fails to analyze the intrusion detects presence over network. Apart from these, categorizing the abnormal behavior from normal network traffic patterns is analyzed using classification approach to detect the malicious traces. The methodology works efficient in capturing anomalous activities based on audit data but it fails to identify known attacks.

**B. Machine Learning based detection approaches**

Anomaly based Intrusion Detection Systems concentrate to build the system using machine learning based approaches [35,36] with intent to design detection technique harder over presence of new intrusion traces. The common classifier approaches used to enhance detection techniques tougher are Neural Networks, Bayesian Networks, Fuzzy Logic, Decision Tree, Genetic Algorithm and Support Vector Machine. In Neural Networks based Intrusion Detection Systems designed as highly interconnected and performs similarly to operation of human braini.e., consists of voluminous number of elements which transforms set of input nodes to the desired set of output nodes. The most prominent example of Neural Network is MultiLayer Perceptron built up with arbitrary accuracy to form classification decision boundary i.e., to represent non-linear discriminate function i.e., rely as indirectly proportional destination as curve. But the major draw back behind these approach, it takes long duration of training time and also neural networks turn to be unstable when handling complexity task i.e., operational problem encountered when tried to simulate the parallelism of neural networks.

Fuzzy Logic based detection approaches represent as Boolean-value approach to differentiate the values between absolutely true and absolutely false. It is explicitly expressive inspecifying human thinking and interpretation of things like 'often', 'minimal level' and 'maximum level'. Based on this intrusion detection systems designed to slip up anomalous behavior from normal behavior presence over network traffic. Fuzzy set theory mathematically designed to represent vagueness and uncertainty with formalized logic tools for dealing with imprecision inherit. Fuzzy based Intrusion Recognition Engine developed using fuzzy rules and fuzzy sets. Some of the major characteristics utilizing Fuzzy logic in designing the intrusion detection systems are known to be, fuzzy rules allow us to construct the model using if-then rules that describing security attacks. Moreover, degree of alert indicates the presence of intrusion traces innet work traffic. Most of fuzzy logic approach combined with evolutionary computation model i.e., Genetic Algorithm, Rule based approach to optimize fuzzy membership function. Their work is to extract an omalous behavior from normal behavior to provide crisp value which represent as output variable required. Data processing is major bottle neck in existing intrusion detection approaches. The pitfall behind these knowledge based techniques in designing anomaly based detection techniques is that providing high-quality document is difficult and time consuming. This pitfall is also applicable to other common Anomaly based detection approaches exclusively obtained when analyzing the training data. Another pitfall observed in developing proposed variants using fuzzy are fails to provide descriptive analysis of reasoning the detection technique has need to chosen for specified task.

Genetic Algorithm based detection [37] approaches placed a platform for designing the intrusion detection technique by deriving the biological concept of selection, mutation, inheritance and finally recombination. Major draw back of this detection approaches are required high resource consumption.

Bayesian Networks based detection [38] approaches is associated with machine learning technique which represent as graphical model in form of inference i.e., an description about whole picture can be drawn through evidence and reasoning. In other terms, make ethical judgments on the basis of proven solution and prediction rather than on the basis of direct observation. As per designing the intrusion detection system over Bayesian networks is concerned with statistical combination of representing relationships among variables of interest. But the major pitfall against these are highly rely on these hold based systems, due to these hypotheses derivation leads to detection errors, and makes the system to be in efficient active.

**Chapter-3: Description Machine Learning Techniques**

**3.1 System Model**

The system proposed is composed of feature selection and learning algorithm show in Fig.2. Feature selection component are responsible to extract most relevant features or attributes to identify the instance to a particular group or class. The learning algorithm component builds the necessary intelligence or knowledge using the result found from the feature selection component. Using the training dataset, the model gets trained and builds its intelligence. Then the learned intelligences are applied to the testing dataset to measure the accuracy of home much the model correctly classified on unseen data.

Fig.3.1 – Performance evaluation steps

**3.2 Logistic Regression Theory**

The Logistic Regression is a regression model in which the response variable (dependent variable) has categorical values such as True/False or 0/1. It actually measures the probability of a binary response as the value of response variable based on the mathematical equation relating it with the predictor variables.

The general mathematical equation for logistic regression is −

y = 1/(1+e^-(a+b1x1+b2x2+b3x3+...))

Following is the description of the parameters used −

* **y** is the response variable.
* **x** is the predictor variable.
* **a** and **b** are the coefficients which are numeric constants.

The function used to create the regression model is the **glm()** function.

### Syntax

The basic syntax for **glm()** function in logistic regression is −

glm(formula,data,family)

Following is the description of the parameters used −

* **formula** is the symbol presenting the relationship between the variables.
* **data** is the data set giving the values of these variables.
* **family** is R object to specify the details of the model. It's value is binomial for logistic regression.

### Example

The in-built data set "mtcars" describes different models of a car with their various engine specifications. In "mtcars" data set, the transmission mode (automatic or manual) is described by the column am which is a binary value (0 or 1). We can create a logistic regression model between the columns "am" and 3 other columns - hp, wt and cyl.

# Select some columns form mtcars.

input <- mtcars[,c("am","cyl","hp","wt")]

print(head(input)

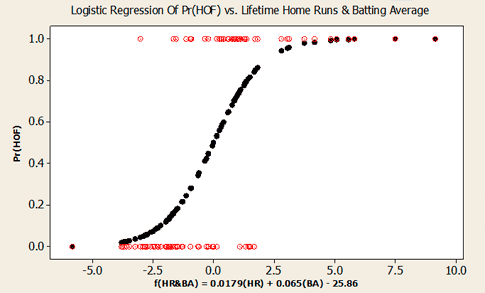


Fig 3.2 Logistic Regression

**3.3.**  **Naïve Bayes Classifier**:

The naïve Bayes model is a heavily simplified Bayesian probability model [12]. The naïve Bayes classifier operates on a strong independence assumption [12]. This means that the probability of one attribute does not affect the probability of the other. Given a series of n attributes,the naïve Bayes classifier makes 2n! independent assumptions. Nevertheless, the results of the naïve Bayes classifier are often correct. The work reported in[13] examines the circumstances under which the naïve bayes classifier performs well and why. It states that the error is a result of three factors: training data noise, bias, and variance. Training data noise can only be minimised by choosing good training data. The training data must be divided into various groups by the machine learning algorithm. Bias is the error due to groupings in the training data being very large. Variance is the error due to those groupings being too small.

The fundamental Naive Bayes assumption is that each feature makes an:

* independent
* equal

contribution to the outcome.

With relation to our dataset, this concept can be understood as:

* We assume that no pair of features are dependent. For example, the temperature being ‘Hot’ has nothing to do with the humidity or the outlook being ‘Rainy’ has no effect on the winds. Hence, the features are assumed to be independent.
* Secondly, each feature is given the same weight(or importance). For example, knowing only temperature and humidity alone can’t predict the outcome accuratey. None of the attributes is irrelevant and assumed to be contributing equally to the outcome.

**3.3.1 Bayes’ Theorem**

Bayes’ Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes’ theorem is stated mathematically as the following equation:

P(A/B) = {P(B/A) P(A)}/{P(B)}

where A and B are events and P(B) ? 0.

* Basically, we are trying to find probability of event A, given the event B is true. Event B is also termed as evidence.
* P(A) is the priori of A (the prior probability, i.e. Probability of event before evidence is seen). The evidence is an attribute value of an unknown instance(here, it is event B).
* P(A|B) is a posteriori probability of B, i.e. probability of event after evidence is seen.

Now, with regards to our dataset, we can apply Bayes’ theorem in following way:

P(y/X) = {P(X/y) P(y)}/{P(X)}

where, y is class variable and X is a dependent feature vector (of size n) where:

Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

X = (Rainy, Hot, High, False)

y = No

So basically, P(X|y) here means, the probability of “Not playing golf” given that the weather conditions are “Rainy outlook”, “Temperature is hot”, “high humidity” and “no wind”.

**3.3.2 Naive assumption**

Now, its time to put a naive assumption to the Bayes’ theorem, which is, **independence** among the features. So now, we split **evidence** into the independent parts.

Now, if any two events A and B are independent, then,

Hence, we reach to the result:

P(y|x\_1,...,x\_n) = { P(x\_1|y)P(x\_2|y)...P(x\_n|y)P(y)}/{P(x\_1)P(x\_2)...P(x\_n)}

**3.4 K-Nearest Neighbors Algorithm**

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. Pause! Let us unpack that. A supervised machine learning  algorithm (as opposed to an unsupervised machine learning algorithm) is one that relies on labeled input data to learn a function that produces an appropriate output when given new unlabeled data.

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

“Birds of a feather flock together.”



Fig 3.3 K-Nearest neighbors algorithm

## Notice in the image above that most of the time, similar data points are close to each other. The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph. There are other ways of calculating distance, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice.

**The KNN Algorithm:**  
1.Load the data  
2.Initialize K to your chosen number of neighbors  
 3. For each example in the data  
 3.1 . Calculate the distance between the query example and the current example from the data.  
 3.2 Add the distance and the index of the example to an ordered collection  
4. Sort the ordered collection of distances and indices from smallest to largest (in ascending order) by the distances  
5. Pick the first K entries from the sorted collection  
6. Get the labels of the selected K entries  
7. If regression, return the mean of the K labels  
8. If classification, return the mode of the K labels.

**3.5 Random forests**

The random forests is an ensemble of un-pruned classification or regression trees. Random forest generates many classification trees. Each tree is constructed by a different bootstrap sample from the original data using a tree classification algorithm. After Patterns Network Traffic Pre- Processors Detector Training Dataset Pattern Builder On-line Off-line Alerts

the forest is formed, a new object that needs to be classified is put down each of the tree in the forest for classification. Each tree gives a vote that indicates the tree’s decision about the class of the object. The forest chooses the class with the most votes for the object. The main features of the random forests algorithm are listed as follows:

•It is unsurpassable in accuracy among the current data mining algorithms.  
 •It runs efficiently on large data sets with many features.   
•It can give the estimates of what features are important.  
 •It has no nominal data problem and does not over-fit.   
•It can handle unbalanced data sets.

In random forests, there is no need for cross-validation or a test set to get an unbiased estimate of the test error. Since each tree is constructed using the bootstrap sample, approximately one-third of the cases are left out of the bootstrap samples and not used in training. These cases are called out of bag (o ob) cases. These o ob cases are used to get a run-time unbiased estimate of the classification error as trees are added to the forest.

The error rate of a forest depends on the correlation between any two trees and the strength of each tree in the forest. Increasing the correlation increases the error rate of the forest. The strength of a tree relies on the error rate of the tree. Increasing the strength decreases the error rate of the forest. When the forest is growing, random features are selected at random out of the all features in the training data. The best split on these random features is used to split the node. The number of random features (Mtry ) is held constant. Reducing (Increasing) Mtry reduces (increases) both the correlation and the strength. The number of features employed in splitting each node for each tree is the primary tuning parameter (Mtry). To improve the performance of random forests, this parameter should be optimized.

We use training data to find the optimal value of the parameter Mtry. The minimum error rate corresponds to the optimal value. Therefore, we use the different value of Mtry to build the forest, and evaluate the error rate of the forest. Then, we select the value corresponding to the minimum error rate to build the pattern. There are two ways to evaluate the error rate. One is to split the dataset into training part and test part. We can employ the training part to build the forest, and then use the test part to calculate the error rate. Another way is to use the oob error estimate. Because random forests algorithm calculates the oob error during the training phase, we do not need to split the training data. We choose the oob error estimate, since it is more effective by learning from the whole training dataset

**3.6 Decision Trees**

Decision Trees are a class of very powerful Machine Learning model cable of achieving high accuracy in many tasks while being highly interpretable. What makes decision trees special in the realm of ML models is really their clarity of information representation. The “knowledge” learned by a decision tree through training is directly formulated into a hierarchical structure. This structure holds and displays the knowledge in such a way that it can easily be understood, even by non-experts.

Induction is where we actually build the tree i.e set all of the hierarchical decision boundaries based on our data. Because of the nature of training decision trees they can be prone to major overfitting. Pruning is the process of removing the unnecessary structure from a decision tree, effectively reducing the complexity to combat over fitting with the added bonus of making it even easier to interpret.

From a high level, decision tree induction goes through 4 main steps to build the tree:

1. Begin with your training dataset, which should have some feature variables and classification or regression output.

2. Determine the “best feature” in the dataset to split the data on; more on how we define “best feature” later

3. Split the data into subsets that contain the possible values for this best feature. This splitting basically defines a node on the tree i.e each node is a splitting point based on a certain feature from our data.

4. Recursively generate new tree nodes by using the subset of data created from step 3. We keep splitting until we reach a point where we have optimized , by some measure, maximum accuracy while minimising the number of splits / nodes.

**Mathematical Overview**

Under a probabilistic adaptive basis function specification the model f(x)f(x) is given by[1]:

Where wm is the mean response in a particular region, Rm, and vm represents how each variable is split at a particular threshold value. These splits define how the feature space in Rp into MM separate "hyper block" regions.

**3.7 Support Vector Machine (SVM)**   
  
 In SVM a separating hyper plane defines the classifier depending on the type of problem and available datasets. In case where dataset is one dimensional, the hyper plane is a point, for two dimensional data it is a separating line as shown in Fig 2, for three dimensional dataset, it is a plane and if the data dimension is higher it is a hyper plane. For a linearly separable dataset, the classifier or the decision function will have the form –

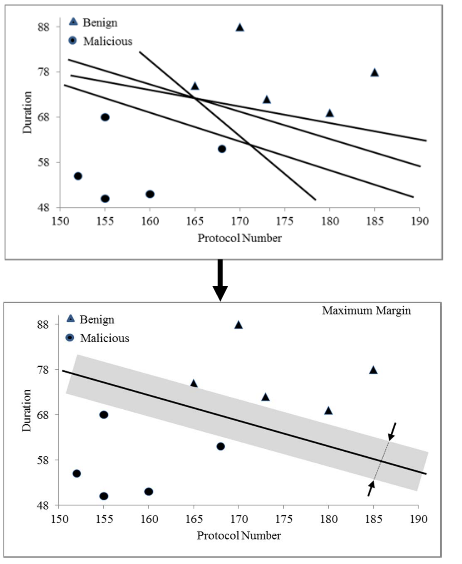


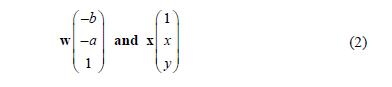
Fig.3.4: SVM classifier in two dimensional problem spaces

ax+by+c=0 (1)

For a given data points (x,y), the above decision function will classify the point in one class if ax + by ≥ c or it will categorize if ax + by < c. The equation of a line y=ax+b can be rewritten as y−ax−b=0 that can be represent using two vectors as below-

For a given data points (x,y), the above decision function will classify the point in one class if ax + by ≥ c or it will categorize if ax + by < c. The equation of a line y=ax+b can be rewritten as y−ax−b=0 that can be represent using two

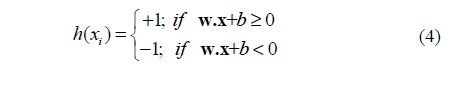
vectors as below-

****

which says we can write the linear equation of a line using two vectors as below-



The reason of using the hyper plane equation **w**T**x** instead of y=ax+b is because it is easier to work in more than two dimensions with this notation and the vector **w** will always be normal to the hyper plane. Once the hyper plan with maximum margin has been found, this hyper plane can be used to make predictions [11]. The hypothesis function h will be-



**3.8 Adaboost**

Adaboost, “Adaptive Boosting,” is a boosting method that builds a highly accurate method by combining multiple simple ones. For our proposed method, we use three well-known Adaboost approaches, namely, discrete Adaboost approach, real Adaboost algorithm, and gentle Adaboost method.  
The output of the weak method ℎ(𝑥)h(x) in the discrete Adaboost approach, as defined by Yoav Freund and Robert Schapire,[24](https://journals.sagepub.com/doi/full/10.1177/1550147719846052) is binary, that is, ℎ(𝑥)∈{+1,−1}h(x)∈{+1,−1}. The real Adaboost algorithm[25](https://journals.sagepub.com/doi/full/10.1177/1550147719846052) uses weak methods, which return a class probability estimate ℎ(𝑥)∈[0,1]h(x)∈[0,1] to update the additive logistic model, rather than the classifications themselves. Based on the real Adaboost approach, gentle Adaboost algorithm was proposed by Viola and Jones.[26](https://journals.sagepub.com/doi/full/10.1177/1550147719846052) This algorithm uses the difference of the conditional class probabilities for the given value of the features to construct the updated values. In the gentle Adaboost approach, the weak method functions are updated by ℎ(𝑥)=𝑃(𝑦=1|𝑥)−𝑃(𝑦=−1|𝑥)h(x)=P(y=1|x)−P(y=−1|x), while real Adaboost algorithm is defined by half the log ratio, ℎ(𝑥)=12log𝑃(𝑦=1|𝑥)𝑃(𝑦=−1|𝑥).h(x)=12logP(y=1|x)P(y=−1|x).

The main difference between gentle Adaboost method and real Adaboost algorithm is the way they use the estimates of the weighted class probabilities.

Adaboost algorithm: AdaBoost is an ensemble based machine learning algorithm, which can be combined with many other classification machine learning algorithms in order to improve its classification and attack detection performance. It calls a base learner for a specified amount of iterations in a loop. For each iteration, distribution of weights Dt is calculated and updated that indicates the importance of examples in the data set for the classification. On each iteration of the loop, the weights of each incorrectly classified samples are modified which is based on the distribution of the sample in the data set so that the new classifier will concentrate more on those samples classified as incorrect (Zan et al., 2007; Sabhnani and Serpen, 2003). The pseudo code of Adaboost algorithm is given in Fig. 1.

Input: Sequence of m training examples Let the set of training sample data be {( y1), . . . , ( , . . . , (, )} with labels {Normal, Dos, Probe, R2L, U2R}, where xi denotes i th feature vector and m is the size of the dataset.

Let T be the number of iterations. Initialize the weights (i) = 1/m for all i.

Repeat for t = 1, 2, . . ., T the following steps

(1) Call the weak classifier, and provide it with the instances of distribution

(2) Calculate the error rate for each category of attacks on each round of the hypothesis i : X → Y

=

If > 0.5, then set T=t-1 and abort loop. Here is the error rate for each category.

(3) Calculate the reweight value for each category of attack instances by using the equation, = /1 −

(4) Update distributin for each category of attacks: +1(i) = { (i) (i) (else) (() = ) (3) Where is a normalization constant.

(5) Repeat the steps from (2) to (4) for all category of attacks with multiple combination of weak classifiers

**Chapter – 4: Dataset descriptions, Methodology, Step diagram**

**4.1 KDD CUP 99 DATASET** With the widespread use of computer networks, the number of attacks has grown extensively, and many new hacking tools and intrusive methods have appeared. Using an intrusion detection system (IDS) is one way of dealing with suspicious activities within a network. An intrusion detection system (IDS) monitors networked devices and looks for anomalous or malicious behavior in the patterns of activity in the audit stream. There are two general types of Intrusion Detection systems, they are Host based Intrusion Detection System and Network based Intrusion Detection System. The Host based Intrusion Detection system has host based sensors and the network based Intrusion detection system has network-based sensors. Data mining-based intrusion detection systems can be classified according to their detection strategy. There are two main strategies: misuse detection, which uses patterns of well known attacks or weak spots of the system to identify intrusions and anomaly detection, which tries to determine whether deviation from the established normal usage patterns can be flagged as intrusions. (a)Misuse Detection: Misuse detection attempts to model abnormal behavior based on signatures of the known attacks and known system vulnerabilities. (b) Anomaly Detection: Normal behavior patterns are useful in predicting both user and system behavior. Anomaly detectors construct profiles that represent normal usage and then use current behavior data to detect a possible mismatch between profiles and recognize possible attack attempts. Signature-based schemes provide very good detection results for specified, well-known attacks. However, they are not capable of detecting new, unfamiliar intrusions, even if they are built as minimum variants of already known attacks. On the contrary, the main benefit of anomaly-based detection techniques is their potential to detect previously unseen intrusion events.

The main intention of our research is to develop an anomaly-based network intrusion detection system utilizing data mining, soft computing and evolutionary techniques. The research work is done by intrusion detection using fuzzy rules, obtained from the association rules Here, we have decided to use KDD Cup 1999 Dataset that contains 41 features labeled as either normal or attack. The Thesis was tested by using the KDD CUP 99 DATA SET. In 1998, DARPA in concert with Lincoln Laboratory at MIT launched the DARPA 1998 dataset for evaluating IDS .The DARPA 1998 dataset contains seven weeks of training and also two weeks of testing data. In total, there are 38 attacks in training data as well as in testing data. The refined version of DARPA dataset which contains only network data (i.e. Tcp dump data) is termed as KDD dataset. The Third International Knowledge Discovery and Data Mining Tools Competition were held in colligation with KDD-99, the Fifth International Conference on Knowledge Discovery and Data Mining. KDD dataset is a dataset employed for this Third International Knowledge Discovery and Data Mining Tools Competition. The input KDD Cup 1999 dataset is divided into two subsets such as, training dataset and testing dataset. At first, the training dataset is classified into five subsets using classification technique so that, four types of attacks (DoS (Denial of Service), R2L (Remote to Local), U2R (User to Root), Probe) and normal data are separated. Then, we mine the association rules from the classified five subsets of data separately. This mined association rule is used to find the consequent part of randomly generated fuzzy rules, which contain only the antecedent part i.e. it will not represent whether the rule is for normal data or attack data. The consequent part of fuzzy if-then rule is formed by matching the randomly generated fuzzy rules with each and every obtained association rule. Thus, we obtain a set of fuzzy if-then rules with consequent parts that represent

whether it is a normal data or an abnormal data. In the testing phase, the test data is matched with fuzzy rules to detect whether the test data is an abnormal data (with attack name) or a normal data. KDD training dataset consists of relatively 4,900,000 single connection vectors where each single connection vectors consists of 41 features and is marked as either normal or an attack, with exactly one particular attack type. These features had all forms of continuous and symbolic with extensively varying ranges falling in four categories: • In a connection, the first category consists of the intrinsic features which comprises of the fundamental features of each individual TCP connections. Some of the features for each individual TCP connections are duration of the connection, the type of the protocol (TCP, UDP, etc.) and network service (http, telnet, etc.). • The content features suggested by domain knowledge are used to assess the payload of the original TCP packets, such as the number of failed login attempts. • Within a connection, the same host features observe the recognized connections that have the same destination host as present connection in past two seconds and the statistics related to the protocol behaviour, service, etc are estimated. • The similar same service features scrutinize the connections that have the same service as the current connection in past two seconds. A variety of attacks incorporated in the dataset fall into following four major categories: Denial of Service Attacks: A denial of service attack is an attack where the attacker constructs some computing or memory resource fully occupied or unavailable to manage legitimate requirements, or reject legitimate users right to use a machine. User to Root Attacks: User to Root exploits are a category of exploits where the attacker initiate by accessing a normal user account on the system (possibly achieved by tracking down the passwords, a dictionary attack, or social engineering) and take advantage of some susceptibility to achieve root access to the system. Remote to User Attacks: A Remote to User attack takes place when an attacker who has the capability to send packets to a machine over a network but does not have an account on that machine, makes use of some vulnerability to achieve local access as a user of that machine. Probes: Probing is a category of attacks where an attacker examines a network to collect information or discover well-known vulnerabilities. These network investigations are reasonably valuable for an attacker who is staging an attack in future. An attacker who has a record, of which machines and services are accessible on a given network, can make use of this information to look for fragile points.

**4.2 Methodology**

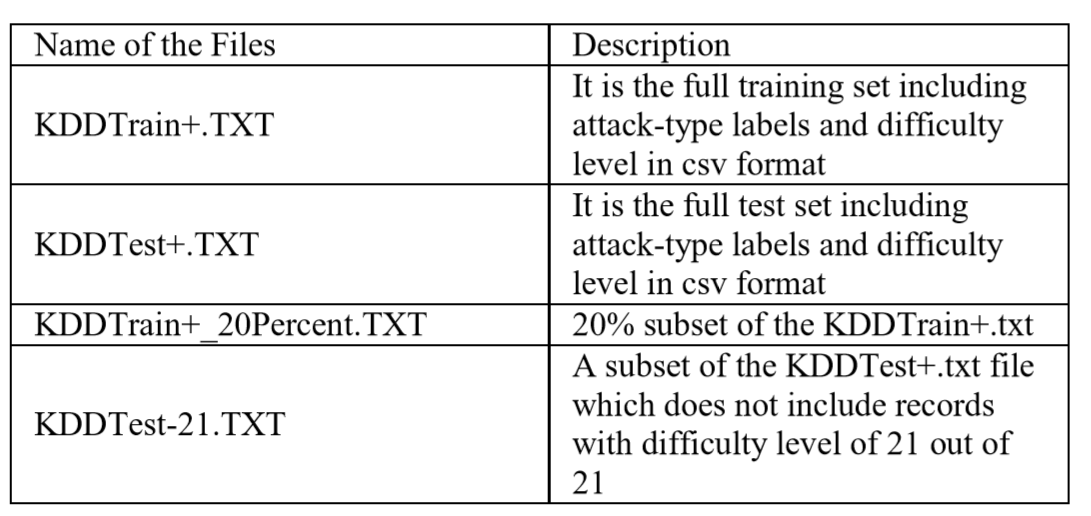
In [19], the classification model consists of two-stages: i) P-rules stage to predict the presence of the class, and ii) N-rules stage to predict the absence of the class. This performed well in comparison with the aforementioned KDDCup 99 results except for the user-to-root (’U2R’) category. In [20], the significance of feature relevance analysis was investigated for IDS with the most widely used dataset, KDDCup 99. For each feature they were able to express the feature relevance in terms of information gain. In addition, they presented the most relevant features for each class label. Reference [21] discussed random forest techniques in misuse detection by learning patterns of intrusions, anomaly detection with outlier detection mechanism, and hybrid detection by combining both the misuse and anomaly detection. They reported that the misuse approach worked better than winning entries of KDDCup 99 challenge results, and in addition anomaly detection worked better compared to other published unsupervised anomaly detection methods. Overall, it was concluded that the hybrid system enhances the performance with the advantage of combining both the misuse and anomaly detection approaches [22], [23], [72]. In [24], an ID algorithm using AdaBoost technique was proposed that used decision stumps as weak classifiers. Their system performed better than other published results with a lower false alarm rate, a higher detection rate, and a computationally faster algorithm. However, the drawback is that it failed to adopt the incremental learning approach. In [25], the performance of the shared nearest neighbor (SNN) based model in ID was studied and reported as the best algorithm with a high detection rate. With the reduced dataset they were able to conclude that SNN performed well in comparison to the K-means for ’U2R’ attack category. However, their work failed to show the results on the entire testing dataset.

In [26], Bayesian networks for ID was explored using Naive Bayesian networks with a root node to represent a class of a connection and leaf nodes to represent features of a connection. Later, [27] investigated the application of Naive Bayes network to ID and through detailed experimental analysis, they showed that Bayesian networks performed equally well and sometimes even better in ’U2R’ and ’Probe’ categories in comparison with the winning entries of KDDCup 99 challenge. In [28], a non-parametric density estimation method based on Parzen-window estimators was studied with Gaussian kernels and Normal distribution. Without the intrusion data, their system was comparatively favorable to the existing winning entries that was based on ensemble of decision trees. In [29], a genetic algorithm based NIDS was proposed that facilitates to model both temporal and spatial information to identify complex anomalous behavior. An overview of ensemble learning techniques for ID was given in [30], and swarm intelligence techniques for ID using ant colony optimization, ant colony clustering and particle swarm optimization of systems were studied in [31]. A comparative study in such research works show that the descriptive statistics was predominantly used.

Overall, a comprehensive literature review shows very few studies use modern deep learning approaches for NIDS and the commonly used benchmark datasets for experimental analysis are KDDCup 99 and NSL-KDD [3], [32]–[33][34]. The IDS based on recurrent neural network (RNN) outperformed other classical machine learning classifiers in identifying intrusion and intrusion type on the NSL-KDD dataset [32]. Two level approach proposed for IDS in which the first level extracts the optimal features using sparse autoencoder in an unsupervised way and classified using softmax regression [33]. The application of stacked autoencoder was proposed for optimal feature extraction in an unsupervised way where the proposed method is completely non-symmetric and classification was done using Random forest. Novel long short-term memory (LSTM) architecture was proposed and by modeling the network traffic information in time series obtained better performance. The proposed method performed well compared to all the existing methods and as well as KDDCup 98 and 99 challenge entries [3]. The performance of various RNN types were evaluated by [34]. Various deep learning architectures and classical machine learning algorithms were evaluated for anomaly based ID on NSL-KDD dataset [34]. The configuration of SVM was formulated as bi-objective optimization problem and solved using hyper-heuristic framework. The performance was evaluated for malware and anomaly ID. The proposed framework is very suitable for big data cyber security problems [35]. To enhance the anomaly based ID rate, the spatial and temporal features were extracted using convolutional neural network and long short-term memory architecture. The performance was shown on both KDDCup 99 and ISCX 2012 datasets [36]. Two step attack detection method was proposed along with a secure communication protocol for big data systems to identify insider attack. In the first step, process profiling was done independently at each node and in second step using hash matching and consensus, process matching was done [37]. An online detection and estimation method was proposed for smart grid system [38]. The method specifically designed for identifying false data injection and jamming attacks in real-time and additionally provides online estimates of the unknown and time-varying attack parameters and recovered state estimates [38]. A scalable framework for ID over vehicular ad hoc network was proposed. The framework uses distributed machine learning i.e. alternating direction method of multipliers (ADMM) to train a machine learning model in a distributed way to learn whether an activity normal or attack [39].

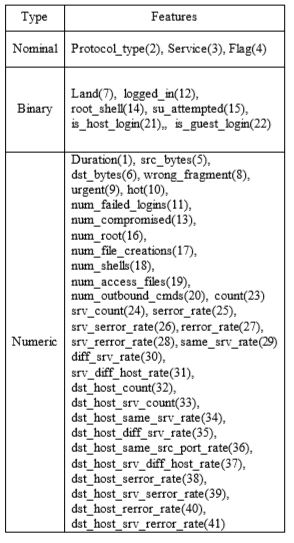
**4.3 Datasets Analysis**

Our proposed methodology applied on NSL-KDD dataset which having 41 attribute, 22 attack type and one class attribute.



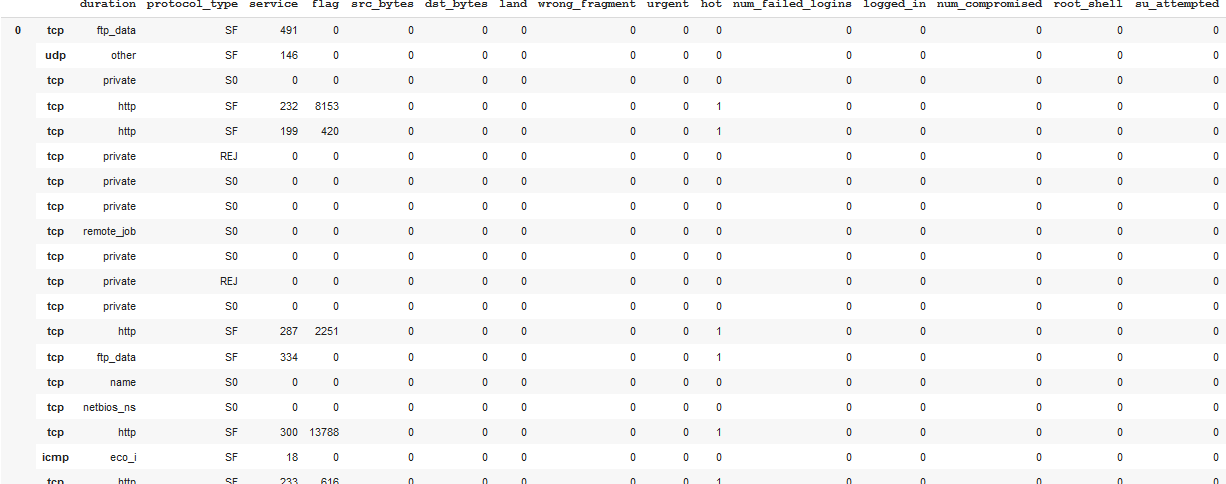
**Table-4.1:** Datasets Analysis

**4.3 Features type in KDD**



**Table-4.2:** Features type in KDD

**4.4 A portion of KDD Datasets**

****

**Fig-4.1:** A portion of KDD Datasets

**Chapter -5**

**Experimental Result Analysis and Performance**

**5.1 Preprocessing**

In this task we have preprocessed the KDD99 dataset using the *Scikit-Learn* preprocessing normalization. **Normalization** process scaled the **KDD99 Datasets individual labels to have unit norm.**

**5.2 Performance Analysis**

For analyzing the performance of different classical supervised machine learning methods to find or accurately detect network intrusion detection the following classical supervised machine learning techniques has been applied.

|  |  |
| --- | --- |
| **Sl** | **Machine Learning Technique** |
| 01 | Logistic Regression |
| 02 | Gaussian Naïve Bayes |
| 03 | K Nearest Neighbors (KNN) |
| 04 | Decision Tree (DT) |
| 05 | AdaBoost |
| 06 | Random Forest |
| 07 | Support Vector Machine (SVM) |

**Table – 5-1:** List of machine classical machine learning technique

For analyzing each technique performance the following classification results and values has been evaluated. The measured parameters are confusion matrix, accuracy, precision value, recall value, harmonic mean of precision (f1-score), false positive rate (fpr), true positive rate (tpr). Each regression or classification results are presented in the next sections.

**5.2.1 Logistic Regression Results:**

After normalization and transformation, we have applied Logistic Regression technique to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 58206 | 2387 |
| 44867 | 205569 |

**Table – 5-2:** Confusion Matrix for Logistic Regression Analysis

|  |  |
| --- | --- |
| **Logistic Regression Results (%)** | |
| **Accuracy** | 84.8 |
| **Precision** | 98.9 |
| **Recall** | 82.1 |
| **f-score** | 89.7 |
| **fpr** | 82.1 |
| **tpr** | 96.1 |

**Table – 5-3:** Logistic Regression Analysis results

**5.2.2 Gaussian Naïve Bayes Results:**

After normalization and transformation, we have applied Gaussain Naïve Bayes technique to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 57879 | 2714 |
| 19223 | 231213 |

**Table – 5-4:** Confusion Matrix for Gaussain Naïve Bayes

|  |  |
| --- | --- |
| **Gaussian Naïve Bayes Results (%)** | |
| **Accuracy** | 92.9 |
| **Precision** | 98.8 |
| **Recall** | 92.3 |
| **f-score** | 95.5 |
| **fpr** | 92.3 |
| **tpr** | 95.5 |

**Table – 5-5:** Gaussain Naïve Bayes results

**5.2.3 K Nearest Neighbors (KNN) Results:**

After normalization and transformation, we have applied K Nearest Neighbors technique to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 60216 | 377 |
| 21791 | 228645 |

**Table – 5-6:** Confusion Matrix for K Nearest Neighbors

|  |  |
| --- | --- |
| **K Nearest Neighbors Results (%)** | |
| **Accuracy** | 92.9 |
| **Precision** | 99.8 |
| **Recall** | 91.3 |
| **f-score** | 95.4 |
| **fpr** | 91.3 |
| **tpr** | 99.4 |

**Table – 5-7:** K Nearest Neighbors results

**5.2.4 Decision Tree Classifier Results:**

After normalization and transformation, we have applied Decision Tree Classifier to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 60268 | 325 |
| 21376 | 229060 |

**Table – 5-8:** Confusion Matrix for Decision Tree Classifier

|  |  |
| --- | --- |
| **Decision Tree Classifier Results (%)** | |
| **Accuracy** | 93.0 |
| **Precision** | 99.9 |
| **Recall** | 91.5 |
| **f-score** | 95.5 |
| **fpr** | 91.5 |
| **tpr** | 99.5 |

**Table – 5-9:** Decision Tree Classifier results

**5.2.5 AdaBoost Classifier Results:**

After normalization and transformation, we have applied AdaBoost technique to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 59508 | 1085 |
| 22295 | 228141 |

**Table – 5-10:** Confusion Matrix for AdaBoost

|  |  |
| --- | --- |
| **AdaBoost Results (%)** | |
| **Accuracy** | 92.5 |
| **Precision** | 99.5 |
| **Recall** | 91.1 |
| **f-score** | 95.1 |
| **fpr** | 91.1 |
| **tpr** | 98.2 |

**Table – 5-11:** AdaBoost results

**5.2.6 Random Forest Classifier Results:**

After normalization and transformation, we have applied random forest technique to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 60308 | 285 |
| 22459 | 227977 |

**Table – 5-12:** Confusion Matrix for Random Forest

|  |  |
| --- | --- |
| **Random Forest Results (%)** | |
| **Accuracy** | 92.7 |
| **Precision** | 99.9 |
| **Recall** | 91.0 |
| **f-score** | 95.2 |
| **fpr** | 91.0 |
| **tpr** | 99.5 |

**Table – 5-13:** Random Forest results

**5.2.7 Support Vector Machine (SVM) Classifier Results:**

After normalization and transformation, we have applied Support Vector Machine (SVM) classifier to the KDD99 datasets and got the following results.

|  |  |
| --- | --- |
| **Confusion Matrix** | |
| 59058 | 1535 |
| 57215 | 193221 |

**Table – 5-14:** Confusion Matrix for Support Vector Machine (SVM) classifier

|  |  |
| --- | --- |
| **Support Vector Machine (SVM) classifier Results (%)** | |
| **Accuracy** | 81.1 |
| **Precision** | 99.2 |
| **Recall** | 77.2 |
| **f-score** | 86.8 |
| **fpr** | 77.2 |
| **tpr** | 97.5 |

**Table – 5-15:** Support Vector Machine (SVM) classifier results

**5.3 Result Analysis**

**Results of Classification**

|  |  |  |
| --- | --- | --- |
| **Sl** | **Machine Learning Technique** | **Accuracy (%)** |
| 01 | Logistic Regression | 84.8 |
| 02 | Gaussian Naïve Bayes | 92.9 |
| 03 | K Nearest Neighbors (KNN) | 92.9 |
| 04 | Decision Tree (DT) | 93.0 |
| 05 | AdaBoost | 92.5 |
| 06 | Random Forest | 92.7 |
| 07 | Support Vector Machine (SVM) | 81.1 |

**Table – 5-16:** Intrusion Detection accuracy for different technique

Fig 5.1: Performance analysis of different Machine Learning Classifiers

Figure 5.2: Intrusion Detection accuracy for different technique

Figure 5.3: Line representation Intrusion Detection accuracy for different technique

In Table 5-16, we have listed the results. While comparing the performance of these classification model Decision Tree learning has the maximum accuracy 93.0% and Support Vector Machine (SVM) has comparatively lower accuracy 81.1%, so among these classical technique Decision Tree leaning classification is much better.

**VII. CONCLUSION**

Intrusion Detection Systems provide the fundamental detection techniques to secure the systems present in the networks that are directly or indirectly connected to the Internet and effectively analysis the problems available in the existing intrusion detection techniques. In this paper we are providing solution on the existing intrusion detection techniques through speedup and accurate anomaly network intrusion detection system. In this work, the proposed method of machine learning for intrusion detection system is presented the proposed method is evaluated on KDD Cup 99 dataset .The performance of the all method is compared with other standard machine learning techniques. The experimental results show that the proposed machine learning technique provides highest classification accuracy of 93.00 %.

**References**

[1] R. Lippmann, J. Haines, D. Fried, J. Korba and K. Das. ”The 1999DARPA off-line intrusion detection evaluation”. Computer networks,vol. 34, no. 4, pp. 579 595, 2000. DOI http://dx.doi.org/10.1016/S1389-1286(00)00139-0.  
[2] W. Lee and S. Stolfo. ”A framework for constructing features and modelsfor intrusion detectionsystems”. ACM transactions on information andsystem security, vol. 3, no. 4, pp. 227261, 2000. DOI http://dx.doi.Org/10.1145/382912.382914.[3] B. Pfahringer. ”Winning the KDD99 classification cup: Bagged boost-ing”. SIGKDD explorations newsletter, vol. 1, pp. 6566, 2000. DOIhttp://dx.doi.org/10. 1145/846183.846200.  
[4] M. Vladimir, V. Alexei and S. Ivan. ”The MP13 approach to the KDD’99classifier learning contest”. SIGKDD explorations newsletter, vol. 1, pp.76 77, 2000. DOI http://dx.doi.org/10.1145/846183. 846202.  
[5] R. Agarwal and M. Joshi. ”PNrule: A new framework for learningclassier models in data mining”. Tech. Rep. 00-015, Department ofComputer Science, University of Minnesota, 2000.  
[6] C.Elkan.”ResultsoftheKDD’99classifierlearning”.SIGKDD explorations newsletter, vol. 1, pp. 63 64, 2000. DOIhttp://dx.doi.org/10.1145/846183. 846199.  
[7] S. Sung, A.H. Mukkamala. ”Identifying important features for intru-sion detection using support vector machines and neural networks”.In Proceedings of the symposium on applications and the Inter-net (SAINT), pp. 209216. IEEE Computer Society, 2003. DOI http://dx.doi.org/10.1109/saint.2003.1183050.  
[8] H. Kayacik, A. Zincir-Heywood and M. Heywood. ”Selecting featuresfor intrusion detection: A feature relevance analysison KDD 99 intrusiondetection datasets”. In Proceedings of the third annual conference onprivacy, security and trust (PST). 2005.  
[9] C. Lee, S. Shin and J. Chung. ”Network intrusion detection throughgenetic feature selection”. In Seventh ACIS international conferenceon software engineering, artificial intelligence, networking, and paral-lel/distributed computing (SNPD), pp. 109114. IEEE Computer Society,2006  
[10] S. Chavan, K. Shah, N. Dave, S. Mukherjee, A. Abraham and S. Sanyal.”Adaptive neuro-fuzzy intrusion detection systems”. In Proceedings ofthe international conference on information technology: Coding andcomputing (ITCC), vol. 1, pp. 7074. IEEE Computer Society, 2004.DOI http://dx.doi.org/ 10.1109/itcc.2004.1286428.  
[11] S. Chebrolu, A. Abraham and J. Thomas. ”Feature deduction andensemble design of intrusion detection systems”. Computers & security,vol. 24, no. 4, pp. 295307, 2005. DOI http://dx.doi.org/10.1016/j.Cose.2004.09.008.

[12] Y. Chen, A. Abraham and J. Yang. ”Feature selection and intrusiondetection using hybrid flexible neural tree”. In Advances in neu-ral networks (ISNN), vol. 3498 of Lecture notes in computer sci-ence, pp. 439 444. Springer Berlin / Heidelberg, 2005. DOI http://dx.doi.org/10.1007/1142746971.  
  
[13] C. Sinclair, L. Pierce and S. Matzner. ”An application of ma-chine learning to network intrusion detection”. In Proceedings ofthe 15th annual computer security applications conference (ACSAC),pp. 371377. IEEE Computer Society, 1999. DOI http://dx.doi.org/10.1109/csac.1999.816048.  
  
[14] H. Debar, M. Becker and D. Siboni. ”A neural network compo-nent for an intrusion detection system”. In Proceedings of the IEEEcomputer society symposium on research in security and privacy,pp. 240250. IEEE Computer Society, 1992. DOI http://dx.doi.org/10.1109/risp.1992.213257.  
  
[15] J. Cannady. ”Artificial neural networks for misuse detection”. In Pro-ceedings of the 1998 national information systems security conference(NISSC), pp. 443456. Citeseer, 1998.  
[16] H. Debar and B. Dorizzi. ”An application of a recurrent network to anintrusion detection system”. In International joint conference on neuralnetworks, 1992. IJCNN., vol. 2, pp. 478 483 vol.2. jun 1992. DOIhttp://dx.doi.org/10.1109/ijcnn. 1992.226942.  
[17] Z. Zhang, J. Lee, C. Manikopoulos, J. Jorgenson and J. Ucles. ”Neuralnetworks in statistical anomaly intrusion detection”. Neural networkworld, vol. 11, no. 3, pp. 305316, 2001.  
[18] J. McHugh. ”Testing intrusion detection systems: A critique of the1998 and 1999 DARPA intrusion detection system evaluations asperformed by Lincoln Laboratory”. ACM transactions on informa-tion and system security, vol. 3, no. 4, pp. 262294, 2000. DOIhttp://dx.doi.org/10.1145/382912.382923.  
[19] M. Tavallaee, E. Bagheri, W. Lu and A. A. Ghorbani. ”A detailed analy-sis of the KDD CUP 99 data set”. In IEEE symposium on computationalintelligence for security and defense applications, Cisda, pp. 16. IEEE,Jul. 2009. DOI http://dx.doi.org/10.1109/cisda. 2009.5356528.  
[20] X. Glorot, A. Bordes, and Y. Bengio, ”Deep sparse rectifier neuralnetworks,” in Proceedings of the Fourteenth International Conferenceon Artificial Intelligence and Statistics, 2011, pp. 315323.

[21] Bengio, Y., Simard, P. and Frasconi, P., 1994. Learning long-termdependencies with gradient descent is difficult. IEEE transactions onneural networks, 5(2), pp.157-166.

[22] Maas, A.L., Hannun, A.Y. and Ng, A.Y., 2013, June. Rectifier nonlin-earities improve neural network acoustic models. In Proc. icml (Vol. 30,No. 1, p. 3).

[23] F. Chollet, ”Keras (2015),” URL http://keras. Io, 2017.

[24] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin,S. Ghemawat, G. Irving, M. Isard et al., ”Tensorflow: A system forlarge-scale machine learning.” in OSDI, vol. 16, 2016, pp. 265283.

[25] Stolfo, S., Fan, W. and Lee, W., KDD-CUP-99 Task Description. 1999-10-28)[2009-05-08]. http://KDD. ics. uci. edu/databases/kddcup99/task,html.

**[**26**]** S.S.Kaushik, Dr.Prof. D.R.Deshmukh,"Detection of Attacks in a Intrusion Detection Systems",International Journal of Computer Science and Information Technologies, Vol.2, no.3,2011, pp.982-986.  
**[**27**]** S.Paliwal, R.Gupta,"Denial-of-Service, Probing & Remote to User(R2L) attack detection using Genetic Algorithm", International Journal of Computer Applications (0975-8887), vol.60, no.19, Dec2012, pp.57-62.  
**[**28**]** G.V.Nadiammai, M.Hemalatha, "Effective approach towards Intrusion Detection System using data mining techniques", Egyptian Informatics Journal, voU5, issueI, Mar2014, pp.37-50.  
[29] Manoj, J.Singh, "Applications of Data Mining for Intrusion Detection", International Journal of Educational Planning & Administration, vo!.l, no.l, 2011, ppJ7-42.  
**[**30**]** D.Upadhyaya, S.Jain, "Hybrid Approach for Network Intrusion Detection System using K-Medoid clusters and Naive Bayes Classification", International Journal of Computer Science Issues, vol.10, issue3,no.l, 2013, pp.231-236.  
**[**31**]** G.P.Spathoulas, S.K.Katsikas, "Methods for post-processing of alerts in intrusion detection: A Survey", International Journal of Information Security Science, vol.2, no.2, 2013, pp.64-80.   
**[**32**]** Z.Tan, A.Jamdagni, X.He, P.Nanda, R.P.L iu, "A System for denial-of-service attack detection based on Multivariate Correlation Analysis", IEEE Transaction on Parallel & Distributed Systems, vol.25, issue no.2, 2014, pp.447-456.  
**[**33**]** M.D.Silva, D.Yora, "Comparative Study of Data mining of Engineering Research and Applications", vol.3, issue I, Jan-Feb2013, pp.1267-1275.  
**[**34**]** G.Gu, PPorras, V.Yegneswaran, M.Fong, W.Lee,"BotHunter: Detecting malware Infection through IDS0Driven Dialouge Correlation", In:Proceedings of 16th USENIX Security Symposium, USENIX Association, Berkeley, 2007, pp.12:1-12:7.  
**[**35**]** V.Das,V.Pathak, S.Sharma, Sreevathsan, MVVNS. Srikanth, T.Gireesh Kumar, "Network Intrusion Detection System based on Machine LEarning Algorithms", International Journal of Computer Science and Information Technology, vol.2, no.6, Dec2010, 138-151.  
**[**36**]** V.Jyothsna, V.V.RamaPrasad, "A Review of Anomaly based Intrusion Detection Systems", International Journal of Computer Applications (0975-8887), vol.2, no.7, Sep2011, pp.26-35.  
**[**37**]** A.Choudhary, V.N.Tiwari, A.Kumar, "Analysis of Fuzzy Logic based Intrusion Detection Systems in Mobile AdHoc Networks", BIJIT-BVICAM's International Journal of Information Technology, vol.6, no.l, 2014, 690-696.  
**[**38**]** M.S.Hoque, Md.A.Mukit, Md.A.N.Bikas, "An Implementation of intrusion detection system using genetic algorithm", International Journal of Network Security and its applications, volA, no.2, Mar2012, pp.109-120.  
  
**[**39**]** S.Taruna, Mrs.Saroj Hiranwal, "Enhnaced Naive Bayes Algorithm for Intrusion Detection in Data mining", International Journal of Computer Science and Information Technologies, volA, no.6, 2013, pp.960-962.