Efficient Dimensionality Reduction using Best First Search and Data Mining Classifiers for Intrusion Detection System

[1]B.Kavitha [2] S.Karthikeyan, [3]P. Selvi

ABSTRACT

The area of intrusion detection is central to the concept of computer security. One major challenge in intrusion detection is that we have to identify the camouflaged intrusions from a huge amount of normal communication activities. It is demands to applying data mining techniques to detect various intrusions. The Data Mining process requires high computational cost when dealing with large data sets. Reducing dimensionality can effectively cut this cost. Hence dimensionality reduction is vital when data mining techniques are applied for intrusion detection. In this paper dimensionality reduction is performed to reduce 41 attributes to 7 attributes based on Best First Search method and the classification of normal and abnormal packets are performed using J48, Id3, Random tree and Naïve Bayes updateable classifiers. These models are tested on KDD Cup 99 dataset. The result shows that J48 classifier outperforms ID3, Random tree and Naïve Bayes Updateable. The experimental result renders remarkable improvement in reducing the false alarms and it also reduces the time complexity.

[1]Lecturer, Department of Computer Applications, Karpagam University / Research Scholar, Research and Development Center, Bharathiar University Email : kavitha_gana2006@yahoo.co.in

[2] Director & Professor, School of Computer Science, Karpagam University, Email: skaarthi@gmail.com

[3]M. Phil Scholar, Department of Computer Science, Karpagam University, Email: pselvi99@yahoo.com

Keywords : Intrusion Detection System, Best first search, J48, ID3, Random Tree, Naïve Bayes Updateable.

1. INTRODUCTION

Intrusion Detection System (IDS) is a type of security management system for computers and networks. An IDS gathers and analyzes information from various areas within a computer or a network to identify possible security breaches, which include both intrusions and misuse organization [1].

The problem of intrusion detection has been studied extensively in computer security ([2], [3] and [4]), also has received a lot of attention in machine learning and data mining. Data mining provides an extra level of intrusion detection by identifying the boundaries for usual network activity so it can distinguish common activities from uncommon activities. Data mining significantly improves intrusion detection using a variety of different methods ([3], [4] and [5]).

A key challenge for many researchers is how to choose the optimal set of features since not all features are relevant to the learning algorithm, and in some cases, irrelevant and redundant features can introduce noisy data that distracts the learning algorithm and therefore severely degrade the accuracy of the detector and causes slow training and testing process. To overcome this problem feature selection has proven to have a significant impact on the performance of the machine learning algorithms. Dimensionality reduction is usually performed either by selecting a subset of the original dimensions and/or by constructing new dimensions. This paper deals with

feature subset selection for dimensionality reduction in machine learning.

The paper is organized as follows, Section 2 describes the related work in the field of Intrusion Detection System. Section 3 explains the overview of the proposed framework. Section 4 presents the description of Best First Search algorithm for feature selection. In section 5 provides details of classification models used in this work are presented. Section 6 shows experiment and results. The final section provides conclusion and summary report.

2. Related Work

The Existing dimensionality reduction methods can roughly be categorized into two classes: feature extraction and feature selection. In feature extraction problems [5], [6], the original features in the measurement space are initially transformed into a new dimension-reduced space via some specified transformation. Significant features are then determined in the new space. Although the significant variables determined in the new space are related to the original variables, the physical interpretation in terms of the original variables may be lost.

In addition, although the dimensionality may be greatly reduced using some feature extraction methods, such as principal component analysis (PCA) [7], the transformed variables usually involve all the original variables. Often, the original variables may be redundant when forming the transformed variables. In many cases, it is desirable to reduce not only the dimensionality in the transformed space, but also the number of variables that need to be considered or measured [8], [9].

In fact, in many cases, the inclusion of insignificant variables will inevitably complicate data inspection and modeling without providing any extra information, because the insignificant variables are, in a sense, irrelative or redundant and, thus, can be ignored [12]. Detailed discussions on various feature selection algorithms can be found in [5], [10], [13]. Reducing dimensionality (the number of attributed or the number of records) can effectively cut this cost of large datasets. This article focuses a pre-processing step which removes dimension from a given data set before it is fed to a data mining algorithm. This work explains how it is often possible to reduce dimensionality with minimum loss of information. Clear dimension reduction taxonomy is described and techniques for dimension reduction are presented theoretically

3 Overview of the Frame Work

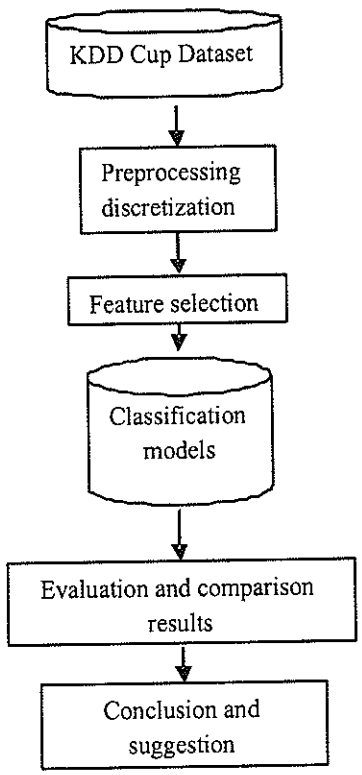

Fig. 1. Frame work of the Proposed Model

The data mining process of building intrusion detection models is depicted in Fig. 1. In the first phase the dataset is collected from KDD Cup 99[20] dataset. The second phase raw dataset is then preprocessed using the discretization method to convert the numeric values to

the nominal values. In the third phase dimensionality reduction of the attributes is performed using Best First Search which reduces the attributes from 41 to 7.In the fourth phase four different classification algorithms are used to classify the dataset into normal or abnormal. In the fifth phase the performance of each classification algorithms are discussed based on several evaluation models [21].

4. Dataset Description

4.1. KDDcup'99 Dataset

In this paper, Kddcup'99 data set is used which is based on the 1998 DARPA [20]. Normal connections are created to profile that those expected in a military network. The various types of attack in our experimental dataset are classified into four categories are shown in the Table 1.

TABLE 1 Various Attack Types

Categories	Attack Types
DOS	Apache2, Back, Land, Mail bomb, Neptune, Pod, process Table, Smurf, Tear dr op, Udpstrom
PROBE	IPsweep, Mscan, nMap, Portsweep, Saint, Satan
U2R	Buffer Overflow, http tunnel, load module, perl, root kit, ps, sqlattack, xterm
R2L	Ftpwrite, guesspasswd, imap, multihop, named, phf, send mail, snmp getattack,
	snmpguess, warezmaster , worm, xlock, xsnoop

The KDDCup'99 Intrusion Detection benchmark is comprised of 3 components. In this work corrected KDD set is used because a dataset with different statistical distributions than either "10% KDD" or "Whole KDD" is provided by the "Corrected KDD" and is comprised of 14 additional attacks. Hence, the "Corrected KDD" dataset is being used for our experiment. The value of each connection is being predicted by this task.

4.2. Exclusion of Dataset

As in our previous work [6] 65000 records have been selected as sample dataset out of 3, 11,029 Corrected KDD dataset connections for the work done by us. However, because the sample number of Probe, U2R, and R2L is being less, the number of records of above attack types will be constant in any sample rate. The remaining records out of 65,000 are 44,417 which are the outcome of excluding the Probe, U2R and R2L types of records. Out of 44417, 20% of Normal connection is selected, and remaining 80% of the dataset is accounted by the Dos. The data sampling number and ratio are shown in Table 2

Table 2 : Amount and ratio of data sampling

Category	Corrected Dataset		Randomly Selected Sampled Records	
Normal	60593	19.48%	8883	13.67%
Probe	4166	1.34%	4166	6.4%
DOS	229853	73.9%	35534	54.67%
U2R	70	.02%	70	.11%
R2L	16347	5.26%	16347	25.15%
Total	3,11,029	100%	65000	100%

5. Feature Selection

The feature selection technique is applied for dimensionality reduction to removes irrelevant, weakly relevant and redundant attribute. In this work attributes selection is done based on Best First Search. Best First Search uses classifier evaluation model namely cfsSubsetEvaluation to estimate the merits of attributes. The attributes with high merit value is considered as potential attributes and used for classification.

5.1 Best First Search Algorithm:

- ✓ If the Successor's merit value is better than its parent, the Successor is set at the front of the queue and the loop restarts.
- ✓ Else the Successor is appended into the queue in place (determined by merit value). The procedure evaluates the successor of the parent.

6. Constructing Classification Models

In this section the details of J48, ID3, Random tree, Naïve Bayes Updateable are discussed.

6.1 J48

J48 is an open source Java implementation of the C4.5 algorithm in the weka data mining tool [14]. It builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. The training data is a set S = s1,s2,... of already classified samples. Each sample si = x1,x2,... is a vector where x1,x2,... represent attributes or features of the sample. The training data is augmented with a vector

C = c1,c2,... where c1,c2,... represent the class to which each sample belongs.

At each node of the tree, J48 chooses one attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. We normalized information gain for choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision.

For Example:

Information gain and Entropy is calculated as follows:

Suppose S is a set of 14 examples in which one of the attributes is wind speed. The values of Wind can be weak or strong. The classification of these 14 examples are 9 YES and 5 NO. For attribute Wind, suppose there are 8 occurrences of Wind = Weak and 6 occurrences of Wind = Strong. For Wind = Weak, 6 of the examples are YES and 2 are NO. For Wind = Strong, 3 are YES and 3 are NO. Therefore

$$\text{Gain}(S, \text{Wind}) = \text{Entropy}(S) - (8/14)*\text{Entropy}(S_{weak}) - (6/14)*\text{Entropy}(S_{strong}) \quad (1)$$

$$= 0.940 - (8/14)*0.811 - (6/14)*1.00$$

$$= 0.048$$

$$\text{Entropy}(S_{weak}) = -(6/8)*\log_2(6/8) - (2/8)*\log_2(2/8)$$

$$= 0.811$$

$$\text{Entropy}(S_{strong}) = -(3/6)*\log_2(3/6) - (3/6)*\log_2(3/6)$$

$$= 1.00$$

For each attribute, the gain is calculated and the highest gain is used in the decision node.

In pseudocode, the general algorithm for building decision trees is[15]:

- Check for base cases
- For each attribute a
  - Find the normalized information gain from splitting on a
- Let a_best be the attribute with the highest normalized information gain
- Create a decision node that splits on a_best .
- Recur on the sublists obtained by splitting on a_best, and add those nodes as children of node

6.2 ID3

ID3[16] is based on the Concept Learning System (CLS) algorithm. The basic CLS algorithm over a set of training instances C is:

Step 1:

- ✓ If all instances in C are positive, then create YES node and halt.
- ✓ If all instances in C are negative, create a NO node and halt.
- ✓ Otherwise select a feature, F with values v1,.., vn and create a decision node.

Step 2:

Partition the training instances in C into subsets C1, C2, ..., Cn according to the values of V.

Step 3:

Apply the algorithm recursively to each of the sets Ci.

Note, the trainer (the expert) decides which feature to select. ID3 improves on CLS by adding a feature selection heuristic. ID3 searches through the attributes of the training instances and extracts the attribute that best separates the dataset. If the attribute perfectly classifies the training sets then ID3 stops; otherwise it recursively operates on the n (where n = number of possible values of an attribute) partitioned subsets to get their "best" attribute.

6.3 Random Tree

Random trees have been introduced by Leo Breiman and Adele Cutler [17]. The algorithm can deal with both classification and regression problems. The classification works as follows: the random trees classifier takes the input feature vector, classifies it with every tree in the forest, and outputs the class label that received the majority of "votes.

All the trees are trained with the same parameters, but on the different training sets, which are generated from the original training set using the bootstrap procedure. For each training set we randomly select the same number of vectors as in the original set (N). The vectors are chosen with replacement. That is, some vectors will occur more than once and some will be absent. For every node of each tree trained not all the variables are used to find the best split, rather than a random subset of them. With each node a new subset is generated, however its size is fixed for all the nodes and all the trees. It is a training parameter, set to $\sqrt{number - of - variables}$ by default. None of the trees that are built are pruned.

In random trees there is no need for any accuracy estimation procedures, such as cross-validation or bootstrap, or a separate test set to get an estimate of the training error. The error is estimated internally during the training. When the training set for the current tree is drawn by sampling with replacement, some vectors are left out so-called *oob (out-of-bag) data*. The size of oob data is about N/3. The classification error is estimated by using this oob-data as following:

- Get a prediction for each vector, which is oob relatively to the i-th tree, using the every i-th tree.
- After all the trees have been trained, for each vector that has ever been oob, find the class-"winner" for it and compare it to the ground-truth response.
- Then the classification error estimate is computed as ratio of number of misclassified oob vectors to all the vectors in the original data. In the case of regression the oob-error is computed as the squared error for oob vectors difference divided by the total number of vectors.

6.4 Naive Bayes Updateable

The naive Bayes is a classifier that maps a number of input features to a set of labels (classes). It has two ports, one (top) that takes the input signal and the other (bottom) that takes the desired classification. The signal on the desired classification port has to have nominal enumeration (one feature per class). Naïve Bayes Updateable [18] is a Standard Naive Bayes that implements the Updateable interface which can be trained incrementally. The Naive update calculates the prior probabilities of a class P_C, and the conditional prior probabilities for a set of features given a class $P_{F|C}$.

Probability of class membership:

$$P_C = P(C) \qquad (2)$$

Probability of feature F given a class C:

$$P_{F|C} = P(F|C) \quad (3)$$

There are two tables that are updated, one containing the class frequencies (probability of a class) and the other containing the probability of a feature given a class. The update reaches its final value in just one step.

7. Estimation of Model Performance

Weka 3.6[19] data mining tool kit is used for analyzing the results. The correctly and incorrectly classified instances show the percentage of test instances that were correctly and incorrectly classified. The percentage of correctly classified instances is often called accuracy. The classification models can be evaluated using 10 fold cross validation method, ROC curve and other Statistical Methods.

The true positive rate (TPR) or sensitivity is defined as the fraction of positive examples predicted correctly by the model,

$$TPR = TP / (TP + FN) \quad (4)$$

False positive rate (FPR) is defined as the fraction of negative examples predicted as a positive class the model, ie,

$$FPR = FP / (TN + FP) \quad (5)$$

7.1 10 Fold Cross Validation:

- First step: data is split into 10 subsets of equal size (usually by random sampling).
- Second step: each subset in turn is used for testing and the remainder for training.
- The error estimates are averaged to yield an overall error estimate

7.2 The Receiver Operating Characteristic Curve (ROC)

ROC[22] curve, is a graphical plot of the sensitivity vs. (1 " specificity) for a binary classifier system as its discrimination threshold is varied.

- ✓ *X* axis shows percentage of false positives (FP) in the sample.
- ✓ *Y* axis shows percentage of true positives (TP) in the sample

7.3 Other statistical Methods

A. Kappa Statistics[21]: The kappa measure of agreement is the ratio

$$K = P(A) - P(E) / (1 - P(E)) \quad (6)$$

Where, P (A) is the proportion of times the k raters agree, and P (E) is the proportion of times the k raters are expected to agree by chance alone.

B. Mean Absolute Error[21]: In statistics, the mean absolute error is a quantity used to measure how close forecasts or predictions are to the eventual outcomes. The mean absolute error (MAE) is given by

$$\frac{1}{n}\sum_{i=1}^{n} | fi - yi | = \frac{1}{n}\sum_{i=1}^{n} | e_i | \quad (7)$$

The mean absolute error is an average of the absolute errors ei = fi " yi, where fi is the prediction and yi the true

C. Root Mean Squared Error (RMSE)[21]: It is a frequently-used measure of the differences between values predicted by a model or an estimator and the values actually observed from the thing being modeled or estimated.

D. Relative Absolute Error[21]: The relative absolute error E_i of an individual program i is evaluated by the equation):

$$E_i = \frac{\sum_{j=1}^{n} \left| P_{(ij)} - T_j \right|}{\sum_{j=1}^{n} \left| T_j - \overline{T} \right|} \qquad (8)$$

where $P_{(ij)}$ is the value predicted by the individual program i for sample case j (out of n sample cases); T_j is the target value for sample case j; and $\overline{T}$ is given by the formula:

$$\overline{T} = \frac{1}{n} \sum_{j=1}^{n} T_j \qquad (9)$$

For a perfect fit, the numerator is equal to 0 and $E_i = 0$. So, the E_i index ranges from 0 to infinity, with 0 corresponding to the ideal.

E. Root Relative Squared error[21]: The root relative squared error E_i of an individual program *i* is evaluated by the equation:

$$E_i = \sqrt{\frac{\sum_{j=1}^{n} \left(P_{(ij)} - T_j \right)^2}{\sum_{j=1}^{n} \left(T_j - \overline{T} \right)^2}} \qquad (10)$$

F. Recall - precision (information retrieval) [21]:

- ✓ Precision (retrieved relevant / total retrieved)

$$TP / (TP+FP) \qquad (11)$$

- ✓ Recall (retrieved relevant / total relevant)

$$TP / (TP + FN) \qquad (12)$$

8. Performance Evaluation and Result

The work is begun with original dataset which is comprised of 41 attributes and one class label. Applying the Best First Search we obtained 7 potential attributes as the result of dimensionality reduction. From the reduced dimensionality the experimental result shows that the performance of the reduced feature also predicts the classification in efficient manner.

8.1 Dimensionality Reduction

The original dataset consist of 41 attributes and one class label. The following names shows the list of 41 Attributes: duration, protocol type, service, Flag,, src_bytes, dst_bytes, land, wrong _ fragment, urgent, Hot, num_field_logins, logged_in, num_compromised, root_shell, su_attempted, num_root, num_file_creation, num_shells, num_access_files, num_outbounds_cmds, is_hist_login, is_guest_login, count, srv_count, serror_rate, srv_serror_rate, rerror _ rate, srv _ rerror_rate, same _ srv _ rate , diff _ srv _ rate , srv_diff_host_rate, dst_host_count, dst_host_srv_count, dst_hosdst_same_srv_rate,dst_host_diff_srv_rate, dst_host_same _ src _ port _ rate, dst _ host _ srv _ diff _ host _ rate, dst _ host _serror_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst _ host_srv_rerror_rate.,

After the dimensionality reduction the 7 potential attributes obtained are: Protocol Type, Service, Src_bytes, Dst_bytes, count, diff_srv_rate, dest_host_srv_count.

8.1 Result

The analysis and interpretation of classification is a time consuming process that requires a deep understanding of statistics. The result of table 3 shows that J48 classifier outperforms with the remaining algorithms. Next to the J48 algorithm, ID3 and Random Tree perform better. NaiveBayes Updateable has the worst classification but the Time Taken is very less. The figures 2, 3, 4 and 5 depict the performance of the 7 and 41 attributes of j48, ID3, Random tree and Naïvebayes Updateable. It is observed from the table 3 that the classification accuracy increases better after dimensionality reduction. The summary report shows various statistical evaluations which was discussed in the section 7 & result of the j48, ID3, Random Tree and Naivebayes updatable is shown in the tables 4, 5, 6 and 7 respectively.

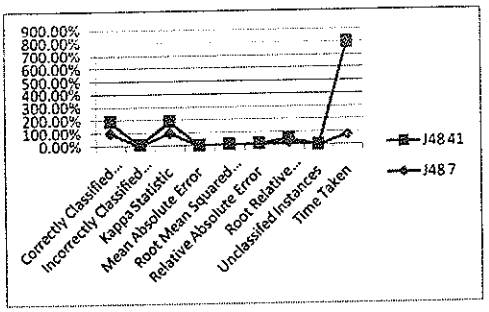

Fig . 2. Performance of J48 with 7 & 41 attributes

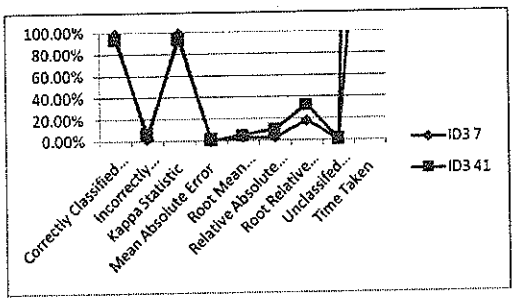

Fig . 3. Performance of ID3 with 7 & 41 attributes

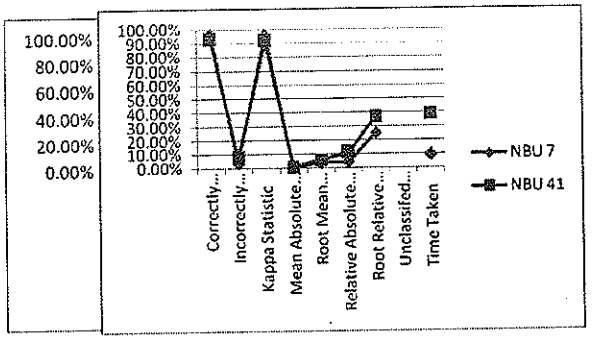

Fig. 4. Performance of Random Tree with 7 & 41 attributes

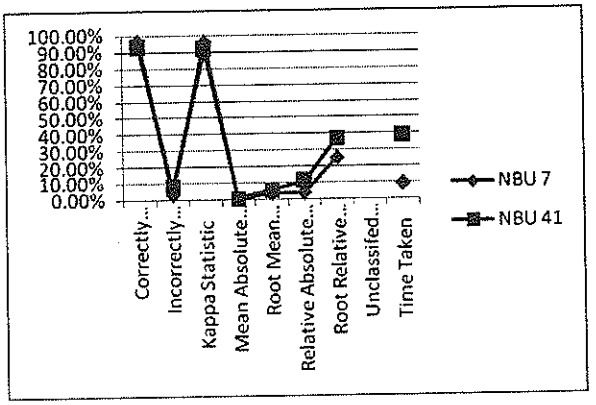

Fig. 5. Performance of Naïve Bayes Updateable with 7 & 41 attributes

Table 3 Comparison of four models

	J48		ID3		Random Tree		NaiveBayes Updateable	
	7	41	7	41	7	41	7	41
Correctly Classified Instances	98.0123%	93.98%	97.91%	93.83%	97.74%	93.76%	96.76%	92.67%
Incorrectly Classified Instances	1.9877%	6.08%	1.77%	5.85%	2.26%	6.24%	3.24%	7.33%
Kappa Statistic	0.9771	0.9303	0.9796	0.932	0.974	0.9278	0.9627	0.9153
Mean Absolute Error	0.0016	0.0048	0.0013	0.0046	0.0016	0.0048	0.0019	0.005
Root Mean Squared Error	0.0292	0.0496	0.0269	0.0491	0.0303	0.0505	0.0379	0.0558
Relative Absolute Error	3.5141%	10.60%	2.96%	10.1562	3.16%	10.55%	4.10%	10.98%
Root Relative Squared error	19.3057%	32.86%	17.81%	32.53%	20.03%	33.42%	25.11%	36.96%
Unclassified Instances	—	—	0.32%	0.31%	—	—	—	—
Time Taken	0.77 sec	7.25 sec	3.14 sec	9.2 sec	4.16 sec	46.59 sec	0.09 sec	0.39 sec

Table 4 J48 with 7 attributes Summary Report

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.987	0	1	0.987	0.993	0.998	ipsweep
0.97	0	0.98	0.97	0.975	0.998	mscan
0	0	0	0	0	0.968	nmap
0.937	0.001	0.782	0.937	0.852	0.98	portsweep
0.549	0.001	0.867	0.549	0.672	0.976	saint
0.964	0.005	0.872	0.964	0.915	0.996	satan
0.724	0	0.538	0.724	0.618	0.965	buffer_overflow
0.949	0	0.915	0.949	0.932	0.987	httptunnel
0.5	0	0.25	0.5	0.333	1	loadmodule
1	0	0.6	1	0.75	1	perl
0	0	0	0	0	0.763	rootkit
0.313	0	0.294	0.313	0.303	0.811	ps
1	0	1	1	1	1	saqlattack
0	0	0	0	0	0.807	xterm
0	0	0	0	0	5	ftp_write
1	0	0.999	1	0.999	1	guess_passwd
0	0	0	0	0	0.499	imap
0.273	0	0.429	0.273	0.333	0.772	multihop
0.471	0	0.615	0.471	0.533	0.911	named
0	0	0	0	0	0.75	phf
0.75	0	0.833	0.75	0.789	0.975	sendmail
0.999	0.01	0.937	0.999	0.967	0.997	snmpgetattack
0.999	0	1	0.999	0.999	1	snmpguess
0.996	0	0.988	0.996	0.992	1	warezmaster
1	0	1	1	1	1	worm
0.333	0	0.429	0.333	0.375	0.72	xlock

0.5	0	6	0.5	0.545	0.917	xsnoop
0.934	0.001	0.994	0.934	0.963	0.996	normal
0.999	0	0.997	0.999	0.998	1	apache2
0.999	0	0.999	0.999	0.999	1	back
0.889	0	1	0.889	0.941	0.944	land
1	0	0.978	1	0.989	1	pod
1	0	1	1	1	1	mailbomb
1	0	1	1	1	1	teardrop
0	0	0	0	0	1	udpstorm
0.978	0.001	0.954	0.978	0.966	0.999	processtable
0.999	0.001	0.996	0.999	0.998	1	neptune
1	0	1	1	1	1	Smurf

Table 5 ID3 with 7 attributes Summary Report

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.99	0	1	0.99	0.995	0.999	ipsweep
0.975	0	0.979	0.975	0.977	0.994	mscan
0	0	0	0	0	0.968	nmap
0.98	0.001	0.787	0.98	0.873	0.968	portsweep
0.566	0.001	0.868	0.566	0.686	0.978	saint
0.972	0.005	0.874	0.972	0.92	0.995	satan
0.778	0	0.724	0.778	0.75	0.862	buffer_overflow
0.961	0	0.993	0.961	0.977	0.972	httptunnel
0	0	0	0	0	0.5	loadmodule
1	0	1	1	1	1	perl
0	0	0	0	0	0.5	rootkit
0.4	0	0.5	0.4	0.44	0.625	ps
1	0	1	1	1	1	sqlattack
0	0	0	0	0	0.654	xterm

0	0	0	0	0	0.5	ftp_write
1	0	1	1	1	1	guess_passwd
0	0	0	0	0	0.5	imap
0.647	0	0.733	0.647	0.688	0.75	multihop
0.588	0	0.714	0.588	0.645	0.853	named
0	0	0	0	0	0.5	phf
0.944	0	0.81	0.944	0.872	0.925	sendmail
1	0.01	0.937	1	0.968	0.977	snmpgetattack
1	0	0.999	1	0.999	1	snmpguess
0.998	0	0.995	0.998	0.997	0.998	warezmaster
1	0	1	1	1	1	worm
0.429	0	0.75	0.429	0.545	0.667	xlock
1	0	1	1	1	0.917	xsnoop
0.936	0.001	0.997	0.936	0.965	0.994	normal
0.996	0	0.996	.996	0.996	0.992	apache2
0.999	0	0.999	0.999	0.999	1	back
0.889	0	0.889	0.889	0.889	0.944	land
1	0	1	1	1	1	pod
1	0	1	1	1	1	mailbomb
1	0	1	1	1	1	teardrop
1	0	1	1	1	1	udpstorm
0.976	0	0.966	0.976	0.971	0.998	processtable
1	0	1	1	1	1	neptune
1	0	1	1	1	0.999	smurf

Table 6 Random Tree with 7 attributes Summary Report

TP Rate	FP Rate	Precision	Recall	F- Measure	ROC Area	Class
0.984	0	0.984	0.984	0.984	0.997	Ipsweep
0.953	0.001	0.95	0.953	0.951	0.995	mscan
0	0	0	0	0	0.974	Nmap

0.945	0.001	0.783	0.945	0.856	0.982	Portsweep
0.545	0.001	0.842	0.545	0.662	0.975	Saint
0.966	0.005	0.867	0.966	0.914	0.996	Satan
0.69	0	0.488	0.69	0.571	0.845	buffer_overflow
0.905	0	0.941	0.905	0.923	0.971	Httptunnel
0	0	0	0	0	0.75	Loadmodule
1	0	0.6	1	0.75	1	Perl
0	0	0	0	0	0.615	Rootkit
0.375	0	0.5	0.375	0.429	0.718	Ps
0	0	0	0	0	0.5	Sqlattack
0	0	0	0	0	0.73	Xterm
0	0	0	0	0	0.5	ftp_write
0.997	0	0.994	0.997	0.995	0.999	guess_password
0	0	0	0	0	0.5	Imap
0.545	0	0.667	0.545	0.6	0.841	multihop
0.588	0	0.667	0.588	0.625	0.882	named
0	0	0	0	0	0.75	phf
0.65	0	0.722	0.65	0.684	0.925	sendmail
0.999	0.001	0.935	0.999	0.966	0.998	snmpgetattack
0.998	0	0.996	0.998	0.997	1	snmpguess
0.995	0.001	0.966	0.995	0981	0.999	warezmaster
0	0	0	0	0	0.5	worm
0.222	0	0.4	0.222	0.286	0.833	xlock
0.833	0	0.714	0.833	0.769	0.917	xsnoop
0.922	0.001	0.992	0.922	0.956	0.996	normal
0.989	0	0.995	0.989	0.992	0.996	apache2
0.993	0	0.99	0.993	0.991	0.996	back
0.778	0	0.875	0.778	0.824	0.889	land
0.966	0	0.977	0.966	0.971	0.983	pod
0.999	0	0.997	0.999	0.998	1	mailbomb
0.667	0	1	0.667	0.8	0.833	teardrop
1	0	1	1	1	1	udpstorm
0.978	0.001	0.945	0.978	0.961	0.999	processtable
0.999	0	0.998	0.999	0.999	1	neptune
0.999	0	1	0.999	0.999	1	smurf

Table 7 NaiveBayesUpdateable with 7 attributes Summary Report

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC curve	Class
0.987	0.003	0.614	0.987	0.757	0.998	ip sweep
0.922	0.001	0.956	0.922	0.939	1	mscan
0	0	0	0	0	0.997	nmap
0.942	0.002	0.758	0.942	0.84	0.999	portsweep
0.367	0.002	0.73	0.367	0.488	0.993	saint
0.976	0.006	0.856	0.967	0.912	0.999	satan
0.793	0	0.5	0.793	0.613	0.999	buffer_overflow
0.956	0.001	0.786	0.956	0.863	1	httptunnel
0	0	0	0	0	0.99	loadmodule
0	0	0	0	0	0.997	perl
0	0	0	0	0	0.988	rootkit
0.063	0	0.111	0.063	0.08	0.997	ps
0	0	0	0	0	0.994	sqlattack
0.077	0	0.333	0.077	0.125	0.996	xterm
0	0	0	0	0	0.992	ftp_write
1	0	0.999	0	1	1	guess_passwd
0	0	0	0	0	0.993	imap
0.364	0.001	0.178	0.364	0.239	0.997	multihop
0.174	0	0.6	0.176	0.273	0.993	named
0	0	0	0	0	0.967	phf
0.8	0	0.444	0.8	0.571	1	sendmail
0.999	0.011	0.931	0.999	0.964	0.996	snmpgetattack
0.999	0.001	0.984	0.999	0.991	1	snmpguess
0.999	0.003	0.911	0.999	0.953	1	warezmaster
0	0	0	0	0	0.996	worm
0.222	0	1	0.222	0.364	0.998	xlock
0	0	0	0	0	0.998	xsnoop

0.87	0	0.996	0.87	0.929	0.993	normal
0.997	0.001	0.938	0.997	0.967	1	apache2
0.997	0	1	0.997	0.999	1	back
0	0	0	0	0	0.995	land
1	0.002	0.42	1	0.592	1	pod
1	0	1	1	1	1	mailbomb
0.667	0	0.364	0.667	0.471	1	teardrop
0	0	0	0	0	0.966	udpstorm
0.991	0.001	0.895	0.991	0.941	1	processtable
0.994	0	1	0.994	0.997	1	neptune
1	0	1	1	1	1	smurf

9. Conclusion

In this contribution Best First Search a model learning algorithm is used to rank the features extracted for detecting intrusions and generate Intrusion Detection models. From the result, it is observed that after reducing the dimensionality from 41 attributes to 7 attributes the accuracy of classification increases. The performance of J48 outperforms the remaining three algorithms. Next ID3 works well but it takes more time while comparing with J48 classifier. NaiveBayesUpdateable has the worst performance with 7 & 41 attributes but the Time Taken by this algorithm is very less.

References

1. R. Heady, G. Luger, A. Maccabe, and M. Sevilla. The Architecture of a Network-level Intrusion Detection System, Technical report, CS90-20. Dept. of Computer Science, University of New Mexico, Albuquerque, NM 87131

2. Edward Amoroso, "Intrusion detection", Intrusion.net Books, January 1999.

3. Julia Allen at all, "State of the practice of intrusion detection technologies", Technical Report CMU/ SEI99 - TR-028, ESC-99-028, Carnegie Mellon, Software Engineering Institute, Pittsburgh, Pennsylvania, 1999.

4. Stefan Axelsson, "Intrusion detection systems: A survey and taxonomy", Technical Report No 99-15, Dept. of Computer Engineering, Chalmers University of Technology, Sweden, March 2000

5. T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes, and T. Garvey. A real-time intrusion detection expert system (IDES) - final technical report.Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, February 1992.

6. B. Kavitha, S. Karthikeyan, and B. Chitra. Efficient Intrusion Detection With reduced

Dimension Using Data Mining Classification Methods and Their Performance Comparison

7. Dorosz P., Kazienko P.: Omijanie intrusion detection systems. Software 2.0 no 9 (93), September 2002, pages 48-54. (In Polish only).

8. Dorosz P., Kazienko P. Systems wykrywania intruzów. VI Krajowa Konferencja Zastosowan Kryptografii ENIGMA 2002, Warsaw 14-17 May 2002,pg.no47-78.

9. Elson D: Intrusion Detection, Theory and Practice. March 27, 2000,

10. Fan W., Miller M., Stolfo S., Lee W., Chan P.: Using Artificial Anomalies to Detect Unknown and Known Network Intrusions. In Proceedings of the First IEEE International Conference on Data Mining, San Jose, CA, November 2001.

11. Frederick K. K.: Network Intrusion Detection Signatures. December 19, 2001,

12. Ajit Abraham, Ravi Jain, Johnson Thomas, Sang Yang Han "D-SCIDS: Distributed softcomputing intrusion detection system" Journal of Network and Computer Applications, Elsevier, 2005.

13. Jones A.K., Sielken R.S.: Computer system intrusion detection: a survey. 09.02.2000

14. J48classifier, http: // www.d.umn.edu / ~padhy 005 /Chapter5.html

15. S.B. Kotsiantis, Supervised Machine Learning: A Review of Classification Techniques, Informatica 31(2007) 249-268, 2007

16. Mitchell, Tom M. Machine Learning. McGraw-Hill, 1997

17. Wald I, Machine Learning,July 2002. http://stat-www.berkeley.edu/users/breiman/wald2002-1.pdf

18. Harry Zhang "The Optimality of Naive Bayes". FLAIRS 2004 conference, http:// www.resample.com/xlminer/help/NaiveBC/ classiNB_intro.htm

19. WEKA: Data Mining Software in Java (2008), http://www.cs.waikata.ac.nz/ml/weka

20. MIT Lincoln Lab., Information Systems Technology Group, The 1998 Intrusion detection off-line evaluation plan (March 25, 1998)

21. Cortj . Willmott, Steveng. Ackleso r, oberte. Davis, David r. Legate1s james O'donnell, and Clinton M. Rowe, Statistics for the evaluation and comparison of models, Journal of Geophysical Research, vol. 90, no. C5, pages 8995-9005, september 20, 1985

22. ROC Signal detection theory and ROC analysis in psychology and diagnostics : collected papers; Swets, (1996).

Author's Biography

Kavitha B. completed her M.Phil in Computer Science from Bharathiar University in 2007. She is working as a Lecturer in Department of Computer Applictions, Karpagam University, Coimbatore. Her teaching experience is 6 yrs. Currently She is pursuing Ph.D in Bharathiar University. She has published 5 papers in International Journals and presented paper in 1 International Conference. Her research interests are Data mining, Network Security and Cryptography.

Karthikeyan S. received the Ph.D. Degree in Computer Science and Engineering from Alagappa University, Karaikudi in 2008. He is working as a Professor and

Director in School of Computer Science and Applications, Karpagam University, Coimbatore. At present he is in deputation and working as Assistant Professor in Information Technology, College of Applied Sciences, Sohar, Sulatanate of Oman. He has published more than 14 papers in Natrional/International Journals. His research interests include Cryptography and Network Security

Selvi. P completed her M.Sc Degree in Computer Science from Sri Rama Krishna College of Arts and Science for Women. She is currently pursuing M.Phil in Computer Science from Karpagam University.