AnalyzingNB, DT and NBTree IntrusionDetection Algorithms

Deeman Yousif Mahmood[*], Dr. Mohammed Abdullah Hussein [**]

* Computer Science dept. University of Sulaimani, Sulaimani, Iraq, e-mail: demo_y86@yahoo.com

** Electrical Engineering dept. University of Sulaimani, Sulaimani, Iraq, e-mail: mohamed.hussain@univsul.net

Received:7 Jan. 2014,Revised: 1 Feb. 2014, Accepted : 24 Feb. 2014
Published online: 26 Mar. 2014

Abstract:

This work implements data mining techniques for analysing the performance of Naive Bayes, C4.5 Decision Tree, and the hybrid of these two algorithms the Naive Bayes Tree (NBTree). The goal is to select the most efficient algorithm to build a network intrusion detection system (NIDS). For our experimental analysis we used the new NSL-KDD dataset, which is a modified dataset of the KDDCup 1999 intrusion detection benchmark dataset, with a split of 66.0% for the training set and the remainder for the testing set. In the testing process Weka has been used, which is a Java based open source framework consisting of a collection of machine learning algorithms for data mining applications. In terms of accuracy the experimental results show that the hybrid NBTree is more precise than the other two approaches and the decision tree is better than the Naive Bayes algorithm. Otherwise, in terms of speed of response the Naive Bayes outperform the other two algorithms followed by Decision Tree and NBTree, respectively.

Keywords: Decision Tree (C4.5); Intrusion detection System (IDS); Naïve Bayes (NB); NBTree; NSL-KDD; Weka

1. Introduction :

Intrusion detection systems (IDS) are becoming a very important tool of today's network security architectures, where it analyses the network traffic and looks for intrusive activities [1]. Intrusion detection systems mainly use two techniques: misuse based detection and anomaly based detection. In a misuse (signature) based intrusion detection system, intrusions are detected by looking for activities that correspond to known signatures of intrusions or vulnerabilities, while an anomaly based intrusion detection system detect intrusions by searching for abnormal network traffic. The abnormal traffic pattern can be defined either as the violation of accepted thresholds for frequency of events in a connection, or as a user's violation of the legitimate profile developed for normal behaviour [2, 3]. There are several types of intrusion detection systems and the choice of which one to use depends on the overall risks to the organization and the resources available. There are two primary types of IDS: host-based (HIDS) and network-based (NIDS). HIDS resides on a particular host and looks for indications of attacks on that host while NIDS resides on a separate system that watches network traffic and looks for indications of attacks that traverse that portion of the network [1].A commonly used intrusion detection analysis raw data is the NSL-KDD dataset. NSL-KDD is a new version of KDD Cup 99 dataset that is considered as a standard

benchmark for intrusion detection evaluation and it consists of approximately 4,900,000 single connection vectors, each of which contains 41 features and is labelled as either normal or attack type [4]. This work is a continuation of our previous work in the field of IDS and it aims to design an enhanced IDS using hybrid algorithm with high detection rate and keeping a low false alarm rate. The paper starts with a literature review of related works then a description of the Decision Tree algorithm. After that the Naive Bayes and the NBTree algorithm is described. In the results section, algorithm-wise a 2 class (Normal or Attack) classifications is presented and an analysis of the three algorithms is shown.

Related works

In [1] we proposed an intrusion detection system model based on K-star and Information gain for feature set reduction. The key idea of this paper is to take advantage of instance-based classifier and dataset features reduction for intrusion detection system, the model has the ability to recognize attacks with high detection rate and low false negative.Shanmugavadiva and Nagarajan[5] have designed a fuzzy logic-based system for effectively identifying the intrusion activities within a network. The proposed fuzzy logic-based system is able to detect intrusion behaviour of the networks due to the availability of a better set of rules. Here, they used automated strategy for the generation of fuzzy rules, which were obtained from the definite rules using frequent items. The experiments and evaluations of the proposed intrusion detection system are performed with the KDD Cup 99 intrusion detection dataset. The experimental results did show that the proposed system achieved higher precision in identifying whether the records are normal or attack one.In [6] Stein and Chen applied the genetic algorithm and the decision tree algorithm for intrusion detection. They used the genetic algorithm technique for the feature reduction.Shishupal and Parvat in [7] proposed layered approach and compared the proposed layered approach with the Decision Tree and the Naive Bayes classification methods. Their system is based upon serial layering of multiple hybrid detectors.In [8] authors proposed adaptive intrusion detection based on combining Naive Bayes and ID3 Decision Tree. KDD 99 bench marks intrusion detection datasets has been used in the work. 19 and 41 attributes have been used for determining detection rate and false positive of normal and four types of attacks records using the three algorithms.

Proposed Algorithms for IDS

In this section we will present the algorithms used in this work.

A. *Decision Tree (C4.5)*

In this section the decision tree algorithm is described. Decision trees are well known machine learning techniques and it is composed of three basic elements [9]:

- A decision node specifying a test attributes.
- An edge or a branch corresponding to one of the possible attributes values.
- A leaf, usually named an answer node, and it contains the class to which the object belongs.

In decision trees, two major phases should be ensured:

Building the tree: Based on a given training set.

Classification: Order to classify a new instance.

At start the root of the tree is determined, and then the node specified property is tested. The test results allow moving down the tree relative to a given instance of the attribute value. This process is repeated until it encounters a leaf.

The instance is then classified in the same class based on leaves characteristics.

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier [10].

The pseudo code for building C4.5 decision trees is written below [11]:

1. Check for a base case
2. For each attribute find the normalized information gain ratio.
3. Let *a_best* be the attribute with the highest normalized information gain
4. Create a decision *node* that splits on *a_best*
5. Recurse on the sublists obtained by splitting on *a_best. Add the obtained nodes* as children of the a_best*node*

Decision Tree algorithms use the strategy of future generations, from root to leaves. To ensure this process, the attribute selection measure is used, taking into account the discriminative power of each attribute over the classes in order to choose the "best" one as the root of the (sub) decision tree [12]. In other words, best attribute should be used as a root node for splitting the tree. Objective criteria for judging the efficiency of the split is needed, and information gain measure is used to select the test attribute at each node in the tree. The attribute with the highest information gain (or greatest entropy reduction) is chosen as the test attribute for the current node [13]. This attribute minimizes the information needed to classify samples in the resulting partitions. In the classification setting, higher entropy (which measures the amount of disorder or uncertainty in a system) corresponds to a sample that has a mixed collection of labels. Lower entropy corresponds to a case where we have mostly pure partitions. In information theory, the entropy of a sample *D* is defined as follows:

$$H(D) = -\sum_{i=1}^{k} P(c_i|D)\log_2 P(c_i|D) \qquad (1)$$

Where $P(ci|D)$ is the probability of a data point in *D* being labeled with class*ci*, and *k* is the number of classes. $P(ci|D)$can be estimated directly from the data as follows:

$$P(c_i|D) = \frac{|\{x_j \in D | x_j \text{ has label } y_j = c_i\}|}{|D|} \qquad (2)$$

We can also define the weighted entropy of a decision/split as follows:

$$H(D_L, D_R) = \frac{|D_L|}{|D|}H(D_L) + \frac{|D_R|}{|D|}H(D_R) \qquad (3)$$

Where *D* has been partitioned into D_L and D_R due to some split decision. Finally, we can define the information gain for a given split as:

$$Gain(D, D_L, D_R) = H(D) - H(D_L, D_R) \qquad (4)$$

Gain is the expected reduction in entropy caused by knowing the value of an attribute [14].

B. Naïve Bayes

Naive Bayes (NB) is a method of supervised classification commonly used to predict the likelihood of group members. It assumes conditional independence of a class, and it's based on Bayes theorem [12]. Bayesian network is one of the most widely used graphical model used for representing and processing of uncertain information. Bayesian network is specified by two elements:

- A graphical component composed of a directed acyclic graph (DAG) where vertices represent events and edges are relations between events.
- A numerical component used in the quantification of different links in the DAG by using conditional probability of distribution for each node in the context of its parents.

Naive Bayes is a simple and easy to build Bayesian network. It consists from a root that originate from the same DAG node (called mother, and that node is usually ignored) and several children nodes [15]. Having no complicated iterative parameter estimation

makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and it's widely used. Sometimes it outperforms more sophisticated classification methods.

Bayes theorem provides a way of calculating the posterior probability, $P(c|x)$, from $P(c)$, $P(x)$, and $P(x|c)$. Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of the other predictors. This assumption is called class conditional independence.

Likelihood — Class Prior Probability

$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)} \quad (5)$$

Posterior Probability — Predictor Prior Probability

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c) \quad (6)$$

- $P(c|x)$ is the posterior probability of the *class*.
- $P(c)$ is the prior probability of the *class*.
- $P(x|c)$ is the likelihood, or the probability of *predictor* for a given *class*.
- $P(x)$ is the prior probability of *predictor*.

C. *NBTree*

NBTree is a hybrid algorithm that represents a cross between Naive Bayes classifier and C4.5 Decision Tree classification and it's best described as a decision tree with nodes and branches [15]. The NBTree algorithm is written below with input of T sets of labeled instances and a decision-tree with Naive Bayes category at the output (leaves):

1. For each attribute Xi, evaluate the utility, u(Xi), of a split on attribute Xi. For continuous attributes, a threshold is also evaluated at this stage.
2. Let J = AttMax(Ui). The attribute with highest utility (Maximum utility).
3. If Uj is not significantly better than the utility of the current node, create a Naive Bayes classifier for the current node and return.
4. Partition T according to the test on Xj. If Xj is continuous, a threshold split is used; if Xj is discrete, a multi-way split is made for all possible values.
5. For each child, call the algorithm recursively on the portion of T that matches the test leading to the child.

Experiments and Results:

D. Evaluation*:*

To evaluate the classifier used in this work, we applied the evaluation indices as follows:

True positive (TP): classifying an intrusion as an intrusion.

False positive (FP): incorrectly classifying normal data as an intrusion, also known as a *false alarm*.

True negative (TN): correctly classifying normal data as normal. The true negative rate is also referred to as *specificity*.

False negative (FN): incorrectly classifying an intrusion as normal.

These metrics are derived from a basic data structure known as the confusion matrix [1,18]. A sample confusion matrix for a two class case can be represented as shown in Table1.

Table.1: Confusion Matrix.

		Predicted Class	
	Activity	Attack	Normal
Actual Class	Attack	TP	FN
	Normal	FP	TN

These metrics are defined as follows:

$$True\ Positive\ Rate(TPR) = \frac{TP}{TP+FN} = \frac{\#Correct\ Intrusions}{\#Intrusions} \quad (7)$$

True Positive Rate is also referred to as Sensitivity or Recall.

$$Precision = \frac{TP}{TP+FP} = \frac{\#Correct\ Intrusions}{\#Instances\ Classified\ as\ Intrusion} \quad (8)$$

Precision is also referred to as Positive predictive value (PPV).

$$False\ Positive\ Rate(FPR) = \frac{FP}{FP+TN} = \frac{\#Normal\ as\ Intrusions}{\#Normal} \quad (9)$$

$$True\ Negative\ Rate(TNR) = \frac{TN}{TN+FP} = \frac{\#Correct\ Normal}{\#Normal} \quad (10)$$

True Negative Rate is also called Specificity.

$$False\ Negative\ Rate(FNR) = \frac{FN}{FN+TP} = \frac{\#Intrusion\ as\ Normal}{\#Intrusions} \quad (11)$$

Three additional performance metrics are also commonly used, referred to as*accuracy, Error rate and F-measure*:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = \frac{\#Correct\ Classification}{\#All\ Instances} \quad (12)$$

Accuracy is the most basic measure of the performance of a learning method. This measure determines the percentage of correctly classified instances and the overall classification rate.

$$Error\ rate = 1 - Accuracy \quad (13)$$

F-measure is a measure of a test's accuracy. It considers both the precision and the recall of the test to the F-measure and it can be interpreted as a weighted average of precision and recall. F-measure reaches its best value at 1 and worst score at 0.The traditional F-measure is the harmonic mean of precision and recall:

$$F - measur = \frac{2*Precision*Recall}{Precision+Recall} \quad (14)$$

E. Results:

The first classification model that we used was the Decision Tree (C4.5). J48 is Weka open source java implementation of C4.5 decision tree algorithm. The results of this algorithm are summarized in table 2.

Table.2: Results of C4.5 Decision Tree Classification Model.

Parameter	Normal Class	Anomaly Class	Average Value %
True Positive	97.6	97.1	97.4
False Positive	2.9	2.4	2.6
Precision	97.1	97.7	97.4
Recall	97.6	97.1	97.4
F-Measure	97.4	97.4	97.4
Time to construct	0.25 Sec.		
Accuracy	97.3761		
Error Rate	2.6239		

The second classification model that we used was the Naive Bayes. The results of this algorithm are summarized in table 3.

Table.3: Results of Naïve Bayes Classification Model.

Parameter	Normal Class	Anomaly Class	Average Value %
True Positive	90.6	93.6	92.1
False Positive	6.4	9.4	7.9
Precision	93.3	91	92.2
Recall	90.6	93.6	92.1

F-Measure	91.9	92.3	92.1
Time to construct model	0.06 Sec.		
Accuracy	92.1283		
Error Rate	7.8717		

The third classification model that we used was the hybrid algorithm with Decision Tree and Naive Bayes. The results of this algorithm are summarized in table 4.

Table.4: Results of NBTree Classification Model.

Parameter	Normal Class	Anomaly Class	Average Value %
True Positive	99.4	98.3	98.8
False Positive	1.7	0.6	1.2
Precision	98.3	99.4	98.8
Recall	99.4	98.3	98.8
F-Measure	98.8	98.8	98.8
Time to construct	3.95 Sec.		
Accuracy	98.8338		
Error Rate	1.1662		

Figure 1, is showing a comparison between the three classifiers in a graphical way.

The figure clearly indicates thatin terms of accuracy the NBTree classifier is better than the other two and the DT comes next.

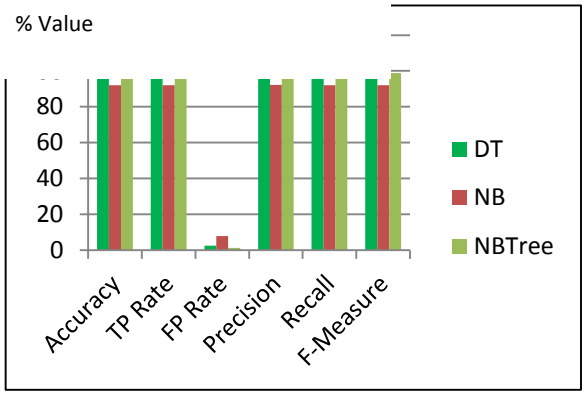

Fig.1 Classifiers comparison chart

Conclusion:

In this work a comparison between three classifiers algorithms has been carried out. The Decision tree, Naive Bayes and the NBTree were used for classifying traffics to either normal or attack by using a standard data set NSL-KDD [16]. Using Weka framework [17], the obtained results show the precedence of NBtree over the others in terms of accuracy of detection. NBTree had the best predictive power with high accuracy and less error rate. It's clear that the hybrid algorithm of Decision Tree and Naïve Bayes is more accurate as a classification model to design an intrusion detection system rather than using each algorithm alone but it needs more construction and processing time. The Naïve Bayes needs the least construction and processing time but its lease accurate than the decision tree alone. Hence, if the speed of response is a major goal the Naive Bayes is recommended but if accuracy is the main goal either the Decision Tree or the Hybrid algorithms are recommended.

References

[1] Deeman Y. Mahmood, Mohammed A. Hussein, "Intrusion Detection System Based on K-Star Classifier and Feature Set Reduction", International Organization of Scientific Research Journal of Computer Engineering (IOSR-JCE) Vol.15, Issue 5, PP. 107-112, Dec. 2013.

[2] D. A. Frincke, D. Tobin, J. C. McConnell, J. Marconi, and D.Polla, "A framework for cooperative intrusion detection", In Proc. 21st NIST-NCSC National Information Systems Security Conference, pages 361-373, 1998.

[3] Denning D, "An Intrusion-Detection Model", IEEE Transactions on Software Engineering, Vol. SE-13, No 2, Feb 1987.

[4] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani "A Detailed Analysis of the KDD CUP 99 Data Set", proceeding of the 2009 IEEE symposium on computational Intelligence in security and defense application.

[5] R. Shanmugavadiva, Dr. N. Nagarajan "Network Intrusion Detection System Using Fuzzy Logic", Indian journal of Computer Science and Engineering(IJCSE) Janeuary 2011.

[6] Gary Stein, Bing Chen, "Decision Tree Classifier for network intrusion detection with GA based feature selection", University of Central Florida. ACM-SE 43, proceedings of 43rd annual Southeast regional Conference. Volume 2, 2005, ACM, New York, USA.

[7] Rupali S. Shishupal , T.J.Parvat, " Layered Framework for Building Intrusion Detection Systems", International Journal of Advances in Computing and Information Researches ISSN:2277-4068, Volume 1– No.2, April 2012.

[8] Dewan Md. Farid, Nouria Harbi, and Mohammad Zahidur Rahman, "COMBINING NAIVE BAYES AND DECISION TREE FOR ADAPTIVE INTRUSION DETECTION",International Journal of Network Security & Its Applications (IJNSA), Volume 2, Number 2, April 2010.

[9] http://en.wikipedia.org/wiki/Decision_tree

[10] Manasi Gyanchandani, R. N. Yadav, J. L. Rana, "Intrusion Detection using C4.5: Performance Enhancement by Classifier Combination", ACEEE Int. J. on Signal & Image Processing, Vol. 01, No. 03, Dec 2010.

[11] S. B. Kotsiantis "Supervised Machine Learning: A Review of Classification Techniques", Informatica 31:249–268 (2007).

[12] Ian H. Witten, Eibe Frank, Mark A. Hall "Data Mining Practical Machine Learning Tools and Techniques", Copyright © 2011 Elsevier Inc.

[13] Gaffney John E., Ulvila, J.W., "Evaluation of intrusion detectors: a decision theory approach", Security and Privacy, 2001. S&P 2001. Proceedings. 2001 IEEE Symposium on.

[14] Yogendra Kumar Jain, Upendra,"An *Efficient Intrusion Detection Based*on*Decision. Tree Classifier Using Feature Reduction*",IJSRP, Volume 2, Issue 1, January 2012 Edition [ISSN 2250-3153].

[15] Pumpuang P., Srivihok A. , Praneetpolgrang P. , "Comparisons of Classifier Algorithms: Bayesian Network, C4.5, Decision Forest and NBTree for Course Registration Planning Model of Undergraduate Students", SMC 2008. IEEE International Conference.

[16] The Knowledge Discovery in Databases, NSL-KDD dataset, http://nsl.cs.unb.ca/NSL-KDD/

[17] University of Waikato, WEKA: Waikato environment for knowledge analysis. Data Mining Software in Java.http://www.cs.waikato.ac.nz/ml/weka/.

Yogendra Kumar Jain, Upendra, "Intrusion Detection using Supervised Learning with Feature Set Reduction", International Journal of Computer Applications (0975 – 8887), Volume 33– No.6, November 2011.