PERFORMANCE EVALUATION OF J48 AND BAYES ALGORITHMS FOR INTRUSION DETECTION SYSTEM

[1]Uzair Bashir & [2]Manzoor Chachoo

[1]Mewar University,Chittorgarh, Rajasthan,India

[2]University of Kashmir,Srinagar, India

ABSTRACT

Recent advances in the field of Artificial Intelligence has urged many security experts to follow this novel area. Extensive work has already been done, and much more efforts are being put to use the techniques of AI to improve the real time security. With lot of algorithms and techniques already in the market, it has become quite difficult to choose a particular algorithm,judging the fact that none proves to be accurate and efficient than the others. During the detailed study of major machine learning algorithms, we found certain algorithms more accurate than the other. In this paper, we have chosen two different techniques of differentiating the normal traffic from the intrusions. The results show J48 more efficient and accurate than the Naïve Bayes algorithm.

INDEX TERMS

Intrusion, IPV4, techniques, Bayes, Tree, Classification.

1. INTRODUCTION

Without the Internet, an organization or an individual is like someone living in remote area with no or minimal connection to rest of the world. To be an active participant, connection to the Internet has become an obligation. If you are a business and you are not on the Internet, you might be losing your most important deals or communications with your customers. Using the services of the Internet like websites and emails for your operations might be beneficial on one hand but then at the same time you get exposed to the attacks and intrusions, which is a dark reality. In the past, there have been various attacks carried out on the Internet and fortunately we have been answering these attacks with major research in the area of network security[1]. These attacks are possible because of loopholes in the programming code while designing softwares or in the design of hardware itself. Although, we have come up quite far in this area but the routine of attacks keeps on changing and improving. Tools like intrusion detection systems aid us to defend against these intrusions or attacks. These systems have been particularly designed for this purpose with capabilities of dealing with newer attacks also. The method of learning about the new intrusions/attacks, however, remains an area of research and many techniques have been developed recently to improve the learning capabilities of IDS [2].

Intrusion detection systems act as a shield protecting the networks or computer systems against the external intruders as well as internal ones. It constantly monitors system and network activities and looks for any suspicious behaviors. This is unlike firewall systems, which drops or allows a packet based on a certain predefined rule set. There are typically two techniques that are used by IDS to handle intrusions: anomaly and misuse (signature) detection. Anomaly based detection techniques report an intrusion based on an activity or event that occurs outside the predefined boundaries. On the other hand, signature based detection techniques have a database of already known attacks which it uses to trap an intrusion [2] [3].

Both techniques of IDS are quite capable in their own ways but also have certain disadvantages. Neither of the techniques has proven to be better than the other. One technique may have an upper hand in one domain but lacks something in other domain and vice versa. Anomaly detection, for example, proves to be more offensive in catching new attacks but may generate false alarms most of the times. Signature detection reduces the false alarm rate; however trapping novel attacks becomes more difficult. From the concise explanation of these two techniques, it is obvious that if, somehow, it was possible to feed the system with a fresh set of rules or heuristics, attack types and likewise, it may be possible for an IDS to deal with intrusions(old or new) without producing false notifications(or at least reducing them)[4][6]. Things could be better if this input was not manual and actually generated by the system itself by constantly learning like a human does. Expert system was a major breakthrough for artificial intelligence, which later added to substitution of a human in analyzing and reporting intrusions. We can understand the working of these systems by looking at deductive databases[23]. Real time IDS have found more interest of researchers in the recent past [24].

As we know that input to an IDS is data (which can be both- a normal activity or an intrusion), use of machine learning can be of great help. Machine learning in fact is associated with construction and improvement of systems over time by learning from new data constantly (an obvious definition of artificial intelligence). This gives us two main advantages- A manual system (which is otherwise slow) is replacedby a fast and a more effective system, and we can track down new intrusions, while constantly improving our knowledge about related intrusions. The good news, however, is that we do not have to worry about feeding the system; it does all this on its own.

Machine learning is defined as "*a field of study that gives computers the ability to learn without being explicitly programmed*" [13].Witten[17] describes machine learning as the acquisition of structural descriptions from examples, which could be used for prediction, explanation and understanding. It involves algorithms and architectures that automatically ‘learn’ from datasets, i.e. change their behavior in order to improve performance at a given task. More precisely, a machine learns with respect to a particular task T, performance metric P, and type of experience E, if the system reliably improves its performance P at task T, following experience [18]. Machine Learning techniques can be categorized into supervised or unsupervised learning techniques[22].

Supervised learning (classification) techniques require a set of input instances to be first labeled with the correct outputs. The algorithm then uses these pre-labelled instances as training examples in order to build a model. This is the first phase of classification, known as the training phase. Once the algorithm has been adequately trained, the model can be used for classification of future inputs. This second phase of classification is called the testing phase. Well known supervised machine learning techniques include decision tree based algorithms like C4.5 & J48, rule based algorithms like RIPPER, perceptron based techniques like Single-Layer Perceptron and Multilayer Perceptron, statistical techniques like Naïve Bayes and Bayesian Networks, instance-based techniques like k-Nearest Neighbor and Support Vector Machines (SVM)[19] [25].

Unsupervised learning techniques (clustering techniques) do not require input instances to be pre-labeled. They group together instances into clusters according to similarity in features. The number and nature of the clusters is not pre-defined but is automatically determined by the algorithm. Unlike supervised techniques, these techniques are able to discover new classes of inputs without any retraining. Well known unsupervised learning techniques include partitioning methods like k-Means, hierarchical methods like BIRCH, density based methods like DBSCAN, and grid based methods like CLIQUE. [20]

2. ALGORITHMS CHOSEN FOR INTRUSION DETECTION SYSTEM

The algorithm which have been chosen to implement IDS are as under:

J48 ALGORITHM:

C4.5 is a successor of ID3 developed by Ross Quinlan and is implemented in WEKA as J48 using Java. All of them adopt a greedy and a top-down approach to decision tree making. It is used for classification in which new data is labelled according to already existing observations (training data set). Decision tree induction begins with a dataset (training set) which is partitioned at every node resulting in smaller partitions, thus following a recursive divide and conquer strategy. In addition to a data set, which is a collections of objects, a set of attributes is also passed. Objects can be an event, an activity and the attributes are the information related to that object. To every tuple in the data set is associated a class label which identifies whether an object belongs to a particular class or not. Splitting can further be performed only if the tuples fall in different classes.

The partition of dataset uses heuristics that chooses an attribute that best partitions a data set. This is referred to a as attribute selection measure. These attribute selection measures are responsible for the type of branching that occurs on a node. Gini index, information gain are some examples which partition a node into a binary or multiway, respectively. Certain other ways to convert multi-branch trees into strict binary can be used if need be.

C4.5 uses gain ratio as the attribute selection measure which has an advantage over information gain used in its predecessor ID3. Since ID3 can produce n-ary branch trees if the attribute on which partitioning is data has unique values, and therefore cannot be used for classification.The attribute which we have to keep in mind before selecting a node are as:

1. Each attribute has information gain associated with it and can be defined as the reduction in entropy of the attribute caused by splitting the instances based on the values obtained by that attribute. The information gain for a particular attribute say A at a node is calculated as under:

$$Information\ Gain(\ N, A) = Entropy(N) - \sum_{values(A)} \frac{|N_i|}{|N|} Entropy(N) \qquad \ldots.(\ 1\ )$$

$$(\mathbf{6.4.1.1})$$

Where N is set of instances at that particular node and |N| is its cardinality, Niis the subset of N for which attribute A has value i, and entropy of the set N is calculated as:

$$entropy(N) = \sum_{i=1}^{No.of Classes} -P_i log2P_i \ldots.(2)$$

Where Piis proportion of instances in N that have their ith class value as output attributes.

2. At all the nodes information gain is calculated and the highest information gain is selected for further process.

When the node acquisition is done using above mentioned steps, new branches are added for each value taken by test attribute below their respective nodes. At each node training instances are passed down the branch along with their associated test attribute value, and furthermore this subset of training instances are used recursively to create new nodes. If there is no change in the output value then a leaf is generated to end the recursion of nodes in the branch and output attribute is assigned the same class value. In those cases when no instances are passed down then a leaf node is created having common class value for the output attribute [20]. The above stated process of generating nodes recursively continues until all the instances are exhausted i.e, they are correctly classified or all their attributes have been used or at the time when it's not possible to divide the dataset.

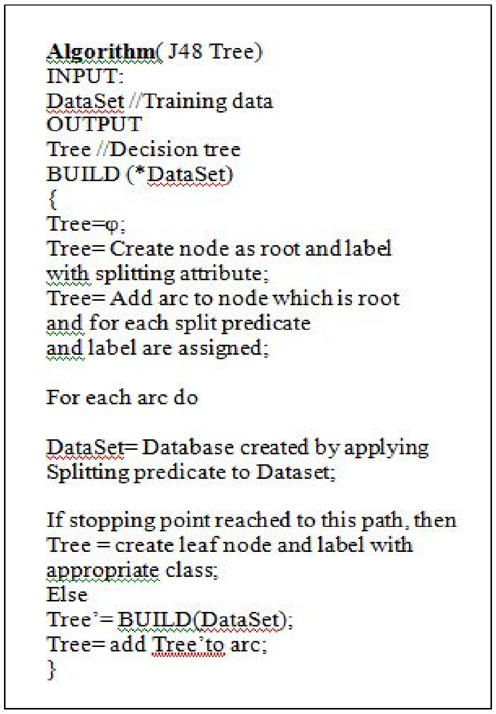
Extensions were made to the basic ID3 algorithm to

i. Algorithm deals with continuous attributes.

ii. Algorithm deal with missed values.

iii. Algorithm to prevent over fitting the data.

Instead of continuous values, when any discrete valued attribute is selected at a particular node, the number of branches formed at that node is equal to the number of values taken by that attribute. But in case of continuous valued attributes, two branches are formed based on the threshold value that best splits them into two. The value of threshold is selected in such a way, so that it may lead to maximum information gain for the given instance. Fayyad &Irani[21] further extended the concept and split a continuous- valued attribute into two or more than two intervals, also extended later in [27]. In this case, there may arise a situation where instance has no value for a missing attribute or may has some unknown value. In this case, the missing value may be replaced by the value that occurs most common in training instances for which it is being tested successfully. In the said algorithm, each and every possible value taken by the attribute having missing value can be calculated on the number of times that an instance can be seen in the training instances at a node. The algorithm of the j48 is as:



```
Algorithm( J48 Tree)
INPUT:
DataSet //Training data
OUTPUT
Tree //Decision tree
BUILD (*DataSet)
{
Tree=φ;
Tree= Create node as root and label
with splitting attribute;
Tree= Add arc to node which is root
and for each split predicate
and label are assigned;

For each arc do

DataSet= Database created by applying
Splitting predicate to Dataset;

If stopping point reached to this path, then
Tree = create leaf node and label with
appropriate class;
Else
Tree'= BUILD(DataSet);
Tree= add Tree' to arc;
}
```

BAYESIAN'S CLASSIFICATION

Bayesian's classification uses probability theory and statistical inferences to identify the class for a given tuple. A single tuple can be associated with different probabilities for different classes. This means instead of finding a particular class for a tuple, a range of classes is selected with probability distributed over these classes. Usually the class with highest probability is selected at the end. This property makes it suitable for datasets that are large.

Bayesian classification is associated with finding two kinds of probabilities- Prior probability and posterior probability. Prior probabilities are independent of any information and is associated only with identifying that a tuple belongs to a certain class irrespective of any other information. Posterior probability or a posteriori probability is finding the probability of a tuple belonging to a certain class based on some information. It is a conditional probability. The Baye's theorem is given as:

$$P(H|T) = \frac{P(T|H)P(H)}{P(T)} \text{........(3)}$$

Where 'T' is a tuple from a dataset D and H denotes the probability that a given tuple T falls under a particular class. P(H|T) and P(T|H) are a posteriori probabilities that denote conditional evaluation of H on T and T on H, respectively. P(H) and P(T) denote prior probabilities and are unconditional meaning independent of others.

We calculate the posterior probability of a tuple 'T' against each class 'C' i.e. the classifier will depict that the tuple belongs to a class with highest posterior probability. We then represent the above equation in a more specific way as:

$$P(C_i|T) = \frac{P(T|C_i)P(C_i)}{P(T)} \text{........ (4)}$$

If classes C (C1, C2,Cm) then the classifier finds:

$$P(C_i|T) > P(C_j|T) \qquad \text{For } 1 \leq j \leq m, j \neq I \text{ ....(5)}$$

$P(C_i|T)$Is called maximum posterior hypothesis. Since P(T) is constant and we assume prior probabilities P(Ci) to be equal i.e P(C1) = P(C2) = … = P(Cm). Therefore, we maximize$P(T|C_i)$.

III. EXPERIMENTAL SETUP AND RESULTS

The experimental setup has been done in data mining tool WEKA 3.4. We have used J48 classifier of weka with KDD-NSL Data set. The NSL-KDD is the improved data set of KDD-99. The number of records present in training data set is 125973and for testing data set is 6808. The improvements which have been made on NSL-KDD data set for intrusion detection system are as under:

1. The Data Set does not contain the redundancy of records in training dataset, which may make classifier not be biased for more frequent records.
2. In the testing data set duplicate records are not contained, therefore, the performance of the learners will be better and not biased by the methods that have better detection rates on these frequent records.
3. It is necessary to state that there is inverse proportionality between the number of selected records at each difficulty level group and the percentage of records in the KDD data set. This inverse proportionality results in varying classification rates for each distinct machine learning methods. This makes it more efficient having accurate evaluation of each learning technique.

IV. RESULTS AND DISCUSSION

As two algorithms are being taken for the implementation of intrusion detection system. The two Algorithms are as:

1. J48 Tree Algorithm
2. Bayes Algorithm

J48 algorithm

== Evaluation on training set ====== Summary ===

Correctly Classified Instances	125861	99.9111 %
Incorrectly Classified Instances	112	0.0889 %
Kappa statistic	0.9982	
Mean absolute error	0.0017	
Root mean squared error	0.029	
Total Number of Instances	125973	

=== Detailed Accuracy by Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	1.000	0.001	0.999	0.999	0.998	normal
	0.999	0.000	0.999	0.999	0.998	anomaly
Weighted Avg	0.999	0.001	0.999	0.999	0.998	

=== Confusion Matrix ===

a	b	<-- classified as
67310	33	a = normal
79	58551	b = anomaly

//

=== Evaluation on test set ====== Summary ===

Correctly Classified Instances	6802	99.9119 %
Incorrectly Classified Instances	6	0.0881 %
Kappa statistic	0.9982	
Mean absolute error	0.0017	
Root mean squared error	0.0296	
Total Number of Instances	6808	

=== Detailed Accuracy by Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	1.000	0.002	0.998	0.999	0.998	normal
	0.998	0.000	1.000	0.999	0.998	anomaly
Weighted Avg	0.999	0.001	0.999	0.999	0.998	

=== Confusion Matrix ===

a	b	<-- classified as
3615	0	a = normal
6	3187	b = anomaly

=== Run information ===

```
Instances:              125973
Test mode:              3-fold cross-validation
Number of Leaves:       605
Size of the tree:       719
```

=== Stratified cross-validation ====== Summary ===

```
Correctly Classified Instances      125672     99.7611 %
Incorrectly Classified Instances    301         0.2389 %
Kappa statistic                                 0.9952
Mean absolute error                             0.0035
Root mean squared error                         0.0476
Total Number of Instances                       125973
```

=== Detailed Accuracy by Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	0.998	0.003	0.998	0.998	0.995	normal
	0.997	0.002	0.997	0.997	0.995	anomaly
Weighted Avg	0.998	0.002	0.998	0.998	0.995	

=== Confusion Matrix ===

a	b	<-- classified as
67189	154	a = normal
147	58483	b = anomaly

Bayes algorithm

=== Evaluation on training set ====== Summary ===

```
Correctly Classified Instances      113902      90.4178 %
Incorrectly Classified Instances    12071        9.5822 %
Kappa statistic                                  0.8067
Mean absolute error                              0.0962
Root mean squared error                          0.3054
Total Number of Instances            125973
```

=== Detailed Accuracy by Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	0.937	0.134	0.890	0.913	0.808	normal
	0.866	0.063	0.923	0.894	0.808	anomaly
Weighted Avg	0.904	0.101	0.905	0.904	0.808	

=== Confusion Matrix ===

a	b	<-- classified as
63106	4237	a = normal
7834	50796	b = anomaly

//

=== Evaluation on test set ====== Summary ===

```
Correctly Classified Instances         6130     90.0411 %
Incorrectly Classified Instances        678      9.9589 %
Kappa statistic                                  0.7994
Mean absolute error                              0.0998
Root mean squared error                          0.3114
Total Number of Instances                        6808
```

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	0.933	0.136	0.886	0.909	0.801	normal
	0.864	0.067	0.919	0.891	0.801	anomaly
Weighted Avg	0.900	0.104	0.901	0.900	0.801	

=== Confusion Matrix ===

a	b	<-- classified as
3371	244	a = normal
434	2759	b = anomaly

//

=== Stratified cross-validation ====== Summary ===

Correctly Classified Instances	113667 90.2312 %
Incorrectly Classified Instances	123069.7688 %
Kappa statistic	0.803
Mean absolute error	0.0979
Root mean squared error	0.3078
Total Number of Instances	125973

=== Detailed Accuracy by Class ===

	TP Rate	FP Rate	Precision	F-Measure	MCC	Class
	0.932	0.132	0.890	0.911	0.804	normal
	0.868	0.068	0.918	0.892	0.804	anomaly
Weighted Avg	0.804	0.102	0.903	0.902	0.804	

=== Confusion Matrix ===

a	b	<-- classified as
62794	4549	a = normal
7757	50873	b = anomaly

As per the results obtained from the experimentation of algorithm J48 (Formally C4.5) for the intrusion detection, the algorithm works very well and has shown increase in the detection rate and True positive rate and decrease in False positive rate. The Kappa statistics and F-Measure of the algorithm is very good. The True positive rate of the algorithm is 99.8%, which means that the algorithm detects 99.8% of the attacks truly. On the other hand, the false positive rate is as low as 0.2%. Thus in overall the algorithm works very well for the purpose of Intrusion detection system. On the other side the results obtained from the experimentation of algorithm Naïve Bayes for the intrusion detection, the algorithm works very well and has not shown increase in the detection rate and True positive rate and not so much improvement in decrease in false positive rate. The Kappa statistics and F-Measure of the algorithm is not so good. The True positive rate of the algorithm is 90.2%, which means that the algorithm detects 90.2% of the attacks trulybut is not as good as J48 which yields the detection rate of 99.8. On the other hand the false positive rate is not so low which is 10.2%. Thus in overall the algorithm does not work well for the purpose of Intrusion detection system because of missing attributes, if the instances and default value is taken into consideration.

.

V. CONCLUSIONS

The current research is focused on designing and implementation of intrusion detection system using data mining and machine learning algorithms. Our thorough theoretical study about the area of machine learning and its implementation in detecting attacks has narrowed our scope of research to some powerful machine learning algorithms. In this paper, we have chosen decision tree and Naïve Bayes algorithm to find which of the two algorithms is more accurate and efficient. Here in this paper, the efforts have been madeto implement the intrusion detection system using famous J48 and Naïve Bayes algorithm for machine learning. The results obtained by the experiments revealed that J48 performs well for designing and implementation of intrusion detection system than Naive Bayes algorithm. The detection rate of J48 algorithm nearly tends to 100 % mark while as naive Bayes provides only 90 % of detection rate. The scenario for false positive rate is same. Therefore, as per statistics J48 performs well for intrusion detection system.

REFERENCES

[1] CONNOLLY, P. J., 2001. Security protects bottom line. InfoWorld, Vol. 23, No. 15, p. 47

[2] SAKURAI, K., & Kim, T. H. (2008). A Trend in IDS researches. 보안공학연구논문지제권제호년월 (Journal of Security Engineering), 5(4), 8.

[3] Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., & Srivastava, J. (2003). A comparative study of anomaly detection schemes in network intrusion detection. Proc. SIAM.

[4] Mathew, D. (2002). Choosing an intrusion detection system that best suits your organization. GSEC Practical v1. 4b, available at: www. Sans. org/reading_room/whitepapers/detection

[5] Brown, D. J., Suckow, B., & Wang, T. (2002). A survey of intrusion detection systems. Department of Computer Science, University of California, San Diego.

[6] Grandison, T., & Terzi, E. (2009). Intrusion Detection Technology.

[7] Beigh, B. M., & Peer, M. A. (2011). Intrusion Detection and Prevention System: Classification and Quick.

[8] Kovacich, G. L. (2003). The Information Systems Security Officer's Guide: Establishing and managing an information protection program. Butterworth-Heinemann.

[9] Huang, Y. A., & Lee, W. (2003, October). A cooperative intrusion detection system for ad hoc networks. In Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks (pp. 135-147). ACM.

[10] Cavusoglu, H., Mishra, B., &Raghunathan, S. (2004). A model for evaluating IT security investments. Communications of the ACM, 47(7), 87-92.

[11] Banerjee, U., & Arya, K. V. (2013). Optimizing Operating Cost of an Intrusion Detection System. International Journal of Communications, Network and System Sciences, 6(1).

[12] Cohen, G., Meiseles, M., &Reshef, E. (2012). U.S. Patent No. 8,099,760. Washington, DC: U.S. Patent and Trademark Office

[13] Amoroso, E., &Kwapniewski, R. (1998, December). A selection criteria for intrusion detection systems. In Computer Security Applications Conference, 1998. Proceedings. 14th Annual (pp. 280-288). IEEE.

[14] Chaudhary, A., V. N. Tiwari, and A. Kumar. "Analysis of fuzzy logic based intrusion detection systems in mobile ad hoc networks." BharatiVidyapeeth's Institute of Computer Applications and Management (BVICAM) 6.1 (2014): 690-696.

[15] Beigh, Bilal Maqbool. "One-stop: A novel hybrid model for intrusion detection system." Computing for Sustainable Global Development (INDIACom), 2014 International Conference on. IEEE, 2014.

[16] Mitra, Sulata, and ArkadeepGoswami. "Load Balancing in Integrated MANET, WLAN and Cellular Network." BharatiVidyapeeth's Institute of Computer Applications and Management (2011): 304.

[17] Witten, Ian H., and Eibe Frank. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2005.

[18] Moskovitch, Robert, et al. "Improving the detection of unknown computer worms activity using active learning." *KI 2007: Advances in Artificial Intelligence*. Springer Berlin Heidelberg, 2007. 489-493.

[19] Kotsiantis, Sotiris B., I. Zaharakis, and P. Pintelas. "Supervised machine learning: A review of classification techniques." (2007): 3-24.

[20] Han, Hui, et al. "Two supervised learning approaches for name disambiguation in author citations." *Digital Libraries, 2004. Proceedings of the 2004 Joint ACM/IEEE Conference on*. IEEE, 2004.

[21] Irani, Keki B. "Multi-interval discretization of continuous-valued attributes for classification learning." (1993).

[22] Govindarajan, M. "Hybrid Intrusion Detection Using Ensemble of Classification Methods." International Journal of Computer Network & Information Security 6.2 (2014).

[23] Beigh, Bilal Maqbool. "A New Classification Scheme for Intrusion Detection Systems." *International Journal of Computer Network and Information Security (IJCNIS)* 6.8 (2014): 56.

[24] Kenkre, Poonam Sinai, AnushaPai, and Louella Colaco. "Real time intrusion detection and prevention system." *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*. Springer International Publishing, 2015.

[25] Singh, Jayveer, and Manisha J. Nene. "A survey on machine learning techniques for intrusion detection systems." *International Journal of Advanced Research in Computer and Communication Engineering* 2.11 (2013): 4349-4355.

[26] Wagh, Sharmila, et al. "Effective Framework of J48 Algorithm using Semi-Supervised Approach for Intrusion Detection." *International Journal of Computer Applications* 94.12 (2014).