COMPARISON OF NEURAL NETWORK, J48, AND RANDOM TREE BASED ALGORITHM FOR ANORMALY INTRUSION DETECTION

[1]VICTOR ONOMZA WAZIRI, PHD, [2]USMAN ABDULQAHAR OZOVEHE, & [3]AUDU ISAH
[1&2]Department of Cyber Security Science,
School of Information and Communication Technology,
Federal University of Technology, Minna, Niger State Nigeria
[3]School of Physical Sciences, Department of Statistics,
Federal University of Technology, Minna, Niger State Nigeria
E-mail: [1]victor.waziri@futminna.edu.ng[2]abdulqaharusuman@gmail.com, & [3]aisah@futminna.edu.ng
Phone No: +234-806-351-8931

Abstract
Currently, the number of computer users is growing exponentially, so are the number of network intrusions and cyber-attacks known variously as Hacking, Hacktivism, Distribution of Denial Attacks (DDoS)m cybercrime just any Internet Network attack for some nefarious reasons. Attacks whose signatures (commonly known as zero-day attack) that have not been identified have posed the biggest threat over the years. This situation has led to lots of research in intrusion detection aimed at curbing the threat. This work compares the strength of Neural Networks algorithms in network traffic classification with the ransom tree and J48 models for efficiency and performance accuracy. By using a portion of the NSL-KDD dataset and splitting it into three parts in the WEKA soft computing software environment, it is possible to determine the strength of the three models in classifying anomaly network intrusion detection systems. The classification rate so obtained (based on the trained numerical values as provided) all gave tests indication of very high correct classification values, high true positive rate and very low false positive rate. When the comparison was performed against Random Tree results to that of Bayesnet and J48 classifiers, Ramdom Tree performed better. However, from the experimental results given, it could be concluded that the use of Random Tree and Neural Network can be more effective in developing real world Intrusion Detection Systems.

Keywords: Neural Network, Naive Bayes, Decision Tree, Intrusion Dictation, Performance Evaluation

Introduction
The exponential growth in the use of computer network world wide has really created an issue of real apprehension and also an area in which further researches should really be carried out. Policies used by some companies are unable to cover some lapses coupled with some information security infrastructures are badly configured and some computer programs used in interfacing to the internet are having vulnerabilities that could be exploited. The process of monitoring events occurring in a computer system or network and analyzing them for sign of intrusion is known as intrusion detection (Abraham *et al.*, 2004). An intrusion detection system can be a piece of installed software or physical device that monitors network traffic in order to detect unwanted activities and events such as illegal and malicious traffic that violates the security policy, and traffic that goes against the acceptable use policy of the network (Wu, 2009). If an intrusion detection system is properly deployed, it would play an important role in determining if a system is under attack or identifying any breach in security.

Generally, intrusion detection can be grouped into two categories: (a) anomaly detection and (b) misuse detection. This classification is based on the method of analysis.

(a) Anomaly Detection: In this method, normal system behavior is monitored and recorded. A proper knowledge base is created for normal behavior; anomaly based IDS identify action that deviates too far from normal activity or behavior, and alarms an administrator. This method is particularly resourceful in the detection of unwanted traffic that is not specifically known (Rohit, 2010)

(b) Misuse Detection: In this method, the activities are documented and then compared with known attack signature database. If these activities come close to certain benchmarks or have values that are equal to those benchmarks, these events are flagged as intrusion. The IDS would then alarm the administrator of the possibility of a break-in; it then falls on the administrator to take action as he deems fit to address the situation (Rohit, 2010).

Classification based on the location of censor (a) Network based IDS (b) Host based IDS:

(a) Network Based IDS: Network packets are the source of data in this method. Analysis and monitoring of the traffic are done in real time on the network adapter; various detection techniques are used to analyze and identify the attack type. Upon detection of the attack by the network administrator, appropriate actions are taken to mitigate the attack. Network based IDS include wireless network monitoring and network analysis (Rohit, 2010).

(b) Host Based IDS: Host based IDS relies primarily on security logs and information gathered through a monitoring system, events and system logs. Intrusions are identified by analyzing these logs and frequent checks for unexpected deviations from what has been specified as normal system activities. An appropriate counteraction is taken, if an attack is identified, by either disconnecting the user login or disabling the account (Rohit, 2010).

Motivation

The use of machine learning algorithms and methods to analyze network traffic has always been a prosperous approach. With a firm believe that there is no information security infrastructure that can be perfectly secured, therefore, comparing algorithms that are employed in network traffic analysis lead to the authors undergoing this paper.

Related Works

A lot of research has been conducted in the field of intrusion detection with the aim at improving detection accuracy and reduce the false alarm rates; and defend against novel attacks.

Mrutyunjaya and Manas (2007) proposed a model using naïve Bayes classifiers over the KDD cup 99 dataset. The experiment was carried out on 10% of the KDD cup 99 dataset which contained approximately 65525 connections; with a full training set and 10 fold cross validation for the testing purpose. In the 10 fold cross validation the available data was randomly divided into 10 disjoint subsets of approximately equal sizes, with one subset being used as the test set and remaining 9 subsets used for building the classifier.

Bharti, Jain, and Shukla (2010) proposed an intrusion detection model that makes use of feature selection, fuzzy K-mean clustering and J48 algorithm. Their model eliminated hard assignment for assigning data-points to corresponding clusters, which is a major problem of k-mean clustering, fuzzy K-mean clustering was used to solve this problem. J48 was used to tackle class to cluster assignment.

Chandolikar and Nandavadekar, (2012) in their paper evaluated the performance of two data mining algorithms to ascertain their effectiveness in intrusion detection. BayesNet a Bayes based algorithm and J48 algorithm which is a decision tree algorithm.

Aziz, Salama, Hassanien, and Hanafi (2012) in their paper used the artificial immune system (AIS) inspired genetic algorithm approach to tackle anomaly network intrusion detection. The algorithm was implemented in the project WEKA. The NSL- KDD dataset was used as test data. The test was carried out using the Minowski and Euclidean distance function.

Tang and Cao (2009) proposed a new approach to detect network attacks; an illustration of the model is shown in figure 2.1 (Tang & Cao, 2009). The aim was to study the efficiency of Neural Networks and support vector machine methods in intrusion detection. The algorithms were applied to the KDD cup 99 dataset. The test was carried out on varying percentage of the test data. They observed that with increasing percentage of the test data, the accuracy was also increased, while the detection rate reduced with increased percentage of the test data. From their results, based on accuracy Neural Networks slightly outperform Support Vector Machines. However, for detection rate, support vector machines carried the day.

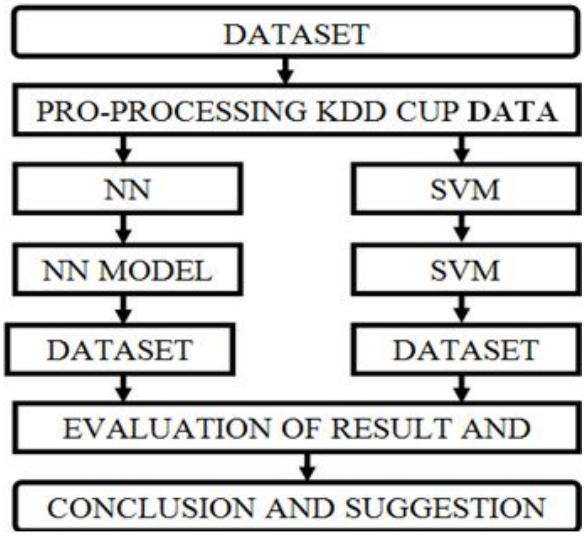

Figure 2.1: Proposed Intrusion Detection Model

Hoque, Mukit, and Bikas (2012) in their paper, proposed an IDS that was based on genetic algorithms (GA), their model was based on Darwin's principle of evolution and survival of the fittest. The system they used was divided into two phases; the pre-calculation phase and detection phase. In the pre-calculation phase, a set of chromosomes is created using the training data, these chromosomes are then used in the detection phase for comparison. In the detection phase, a population is created for the test data and then goes through an evaluation process which involves selection, cross-over and mutation, these determines the type of the test data.

Methodology

This section is on the presentation of the experiment and also discusses the parameters used in evaluating the algorithms used in this research.

Table 2.1: Experimental Design Setup tools

Operating System	Microsoft Windows 7 professional 64-bit.
Software	Weka Version 3.6.10
Neural Network Algorithms	BayesNet, J48, and Random Tree.
Data set	NSS-Kdd
Laptop	Dell D630, intel Duo Core 2,4Gig RAM, T7300 2.00GHz.

Parameters: Here we discuss the various indices used in building and evaluating a neural network.

Attributes: These make up the input nodes for the network:

Hidden Layer: These are intermediate nodes between the input and output layer, in our experiment, the hidden layer is set to 'a' which is, (attribs +classes)/2. This enables greater processing power and system flexibility.

Mean Absolute Error (MAE): This is used to measure the closeness of a prediction to its eventual result.

Root Mean Squared Error (RMSE): This is used to measure how well the system learned the model, that is, it measures how close the predictions came to the actual class.

Relative Absolute Error (RAE): This evaluates the error relative to what might have been if a simpler predictor had been used.

Root Relative Squared Error (RRSE): This measures the error relative to what it would have been, had a simpler predictor been used (Note: for RAE and RRSE the simpler predictor is just an average of the actual value).

Confusion Matrix: This has details of the actual and predicted classes.

True positive (TP): This is when an anomaly is correctly classified as one; true positive rate (TPR) is a ratio of the identified normal data to that of the entire normal data.

False positive (FP): This is the anomaly that is incorrectly labeled as normal; false positive rate (FPR) is the ratio of incorrectly labeled anomaly to the entire malicious data, rate has to be very low.

True Negative: This occurs when there is no attack and no alarm. True negative rate (TNR) is the ratio of correctly classified anomaly to the entire malicious dataset.

False Negative: This is the normal data labeled as anomaly. False negative rate (FNR) is the ratio of incorrectly labeled normal data to that of the entire normal data.

Precision: It is the proportion of predicted positive instances that have been found to be correct.

Accuracy: This is the percentage of the correctly predicted instances

Kappa Statistics: It is a chance–corrected measure of agreement between the actual and predicted classes. It does not take cost into account. Where a value of 1 indicates complete agreement while 0 indicates no agreement at all. A value close to 1 is an indication of good performance for the classifier.

$$\mathbf{K} = \frac{p(A) - P(E)}{1 - p(E)} \qquad (3.1)$$

From equation (3.1) defines the parameters as follow:
P(A) is the proportion of times that the coders, it is the accuracy of the classifier;
P(E) is the proportion of times that we would expect them to agree by chance, which are calculated along the lines of the intuitive argument.

In analyzing the results obtained from the test on various samples of the data, the use of accuracy only would not be generally acceptable, as it is not sensitive to class distribution and therefore not chance corrected.

For this paper, the Bayesnet, J48, and the Random Tree neural network algorithms in Weka were used to analyze the network traffic NSS-kdd downloaded from http://nsl.cs.unb.ca/NSL-KDD/ to carry out our test.

Experimental Layout
The test and training files are saved as arff formats in a folder on the desktop.

Description WEKA software is started:
Retrieve the training and test data from the desktop.
Choose the Classifier.
Set classifier parameters.
Run the experiment and document results.
Repeat step i to vi for the each data set.

Note:
The 42 attributes of the data are the input nodes at the input layer
The number of hidden layers is 22
The Experiment steps
Download the network traffic from http://nsl.cs.unb.ca/NSL-KDD/
Process the dataset; divide it into three parts named NSL-kdd_4000, NSL-kdd_3500 and NSL- kdd_3000, representing 4000, 3500 and 3000 connections respectively.
Start WEKA and load the dataset from the folder it is located on the disk, figure 3.1 shows the screenshot of the loaded dataset.

Figure 3.1: Screenshot Showing Weka Startup On Windows

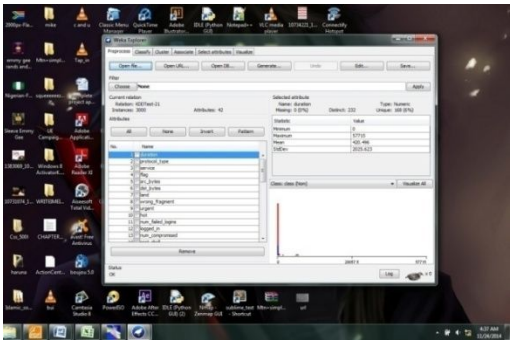

Figure 3.2: Screenshot Showing Loaded Dataset

Start the training and testing process using 10-fold cross validation.
After the learning rate and momentum has been varied, result of the BayesNet for the three set of data would be compared with that of J48, and Random Tree neural network algorithms.
Retrieve and evaluate the results gotten

Experimental Results

This section discusses the experimental results that show some simple graphical analysis as presented in normalized Table 4.1.

The results for the experiments on the 3 datasets and varying Kappa statistics, root mean square error (RMSE), correctly classified instances (CCI), True positive (TP), and the false negative performance accuracy are compared for the analysis by applying the three algorithms (BayesNet, J48, and Random Tree algorithms) as summarized in table 4. The results as shown in the graphical histograms clearly show the difference in terms of performance between the algorithms. The comprehensive results of the tables summarized is obtained in Table 4.1The experiment was carried out on Dataset NSS-kdd_4000

Table 4.1: Result from the iteration of NN1 Model.

Training Parameters	NN1 Train	1st Retrain	2nd Retrain	3rd Retrain	4th retrain	5th retrain	6th retrain	7th retrain
Iterations	13	9	15	8	12	14	15	13
Training MSE	0.00618566	0.00735707	0.00583144	0.00787549	0.00662532	0.00551117	0.00657987	0.00553799
Validation MSE	0.00944252	0.00698658	0.00793371	0.00603267	0.00744463	0.00833006	0.00862000	0.00816243
Testing MSE	0.00788424	0.00681670	0.00954940	0.00520126	0.01047210	0.00934746	0.00962382	0.00838818
Regression	0.48749	0.47795	0.53002	0.50716	0.50484	0.5807	0.49665	0.59163
Duration	1sec	3secs	7secs	3secs	4secs	0sec	4secs	41secs
Mu	0.00100	0.00100	0.00001	0.00100	0.00010	0.00010	0.00100	0.00100

Table 4.2: Result for Dataset NSSkdd4000

Dataset	CLASSIFIER	KAPPA	RMSE	CCI	TPR	FPR
NSSkdd_4000	Bayesnet	0.6898	0.2923	3587 (89.675%)	0.888	0.161
	J48	0.863	0.1993	3840 (96 %)	0.955	0.121
	Random Tree	0.8788	0.1867	3857 (96.42%)	0.958	0.108

Kappa Characteristics: The kappa statistics for the data set NSSkdd_4000 analyzed is 0.6898, 0.863, and 0.8788 representing the BayesNet, J48, and the Random Tree classifier algorithm respectively. The Random tree classifier algorithm show the highest and the better result, followed by the J48 algorithm with a value of 0.863, then the Bayesian algorithm produced the lowest of all with a value of 0.6898.

Root Mean Square Error: In Table 4.1 above, the RMSE field shows that the BayesNet gave the highest value of 0.2923. The Random Tree gave a value of 0.1867 and the J48 show the value of 0.1993. The Random Tree algorithm had the lowest value.

Correctly Classified Instances: Under the correctly classified instance in the table above, the result clearly shows that Random Tree algorithm out scored the other algorithms used with a value of 3857 (96.42%). Followed by the J48 and the Bayes algorithms with values of 3840 (96%) and 3587 (89.675%) respectively.

True Positive Rate: This value has to be high because it is a ratio of the identified normal data to that of the entire normal data. The BayesNet had the highest value of 0.888, followed by the J48 with a value of 0.955. In this case also, the Random Tree algorithm showed a promising value of 0.958 which is the highest value.

False Positive Rate: This value has to be low because it is used measure ratio of incorrectly labeled anomaly to the entire malicious data. The Bayes Network had the highest value of 0.161, followed by the J48 with a value of 0.121. In this case also, the Random Tree algorithm showed a promising value of 0.108 which is the lowest value.

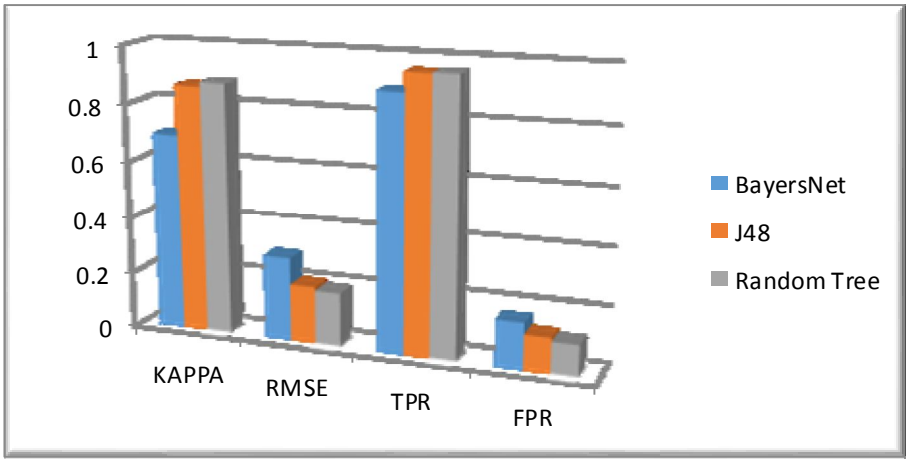

Figure 4.1: Comparison of the three algorithms on NSS-Kdd_4000 dataset

Result Discussion for the Dataset NSS-kdd_3500

In Table 4.1, we present features of our experimental results; Table 4.3 shows further values of the various data attributes experimented upon as stated above.

Table 4.3 Result for Dataset NSS-kdd_3500

SAMPLE	CLASSIFIER	KAPPA	RMSE	CCI	TPR	FPR
NSSkdd3500	BayesNet	0.6538	0.3145	3098 (88.514%)	0.885	0.150
	J48	0.8621	0.1844	3361(96.028%)	0.960	0.119
	Random Tree	0.8723	0.1824	3370 (96.28%)	0.963	0.104

Kappa Characteristics: The Kappa statistics as shown above was 0.6538, 0.8621, and 0.8723 for BayesNet, J48, and Random Tree algorithms classifier respective. The highest score recorded was 0.8723 from the Random tree algorithm classifier. The Kappa characteristics indicates the agreement between the classifications and the true classes, the resulting value for our results most give very high but must not be greater than one (1). With the result obtained, the Random Tree algorithm classifier is having the highest value of 0.8723 while the lowest score recorded was from the BayesNet algorithm classifier.

Root Mean Square Error: In Table 4.2 above, the RMSE field shows that the BayesNet gave the highest value of 0.3145. The Random Tree got a value of 0.1824 and the J48 show the value of 0.1844. The Random Tree algorithm had the lowest value.

Correctly Classified Instances: Under the correctly classified instance in the table above, the result clearly shows that Random Tree algorithm out scored the other algorithms used with a value of 3370 (96.28%). Followed by the J48 and the BayesNet algorithms with values of 3361(96.028%) and 3098 (88.514%) respectively.

True Positive Rate: This value has to be high because it is a ratio of the identified normal data to that of the entire normal data. The BayesNet had the highest value of 0.885, followed by the J48 with a value of 0.960. In this case also, the Random Tree algorithm also showed a promising value of 0.963 which is the lowest value.

False Positive Rate: This value has to be low because it is used to measure the ratio of incorrectly labeled anomaly to the entire malicious data. The BayesNet has the highest value of 0.150, followed by the J48 with a value of 0.119. In this case also, the Random Tree algorithm showed a promising value of 0.104 which is the lowest value. Figure 4.2 depicts the graphical features of the three experimental algorithms

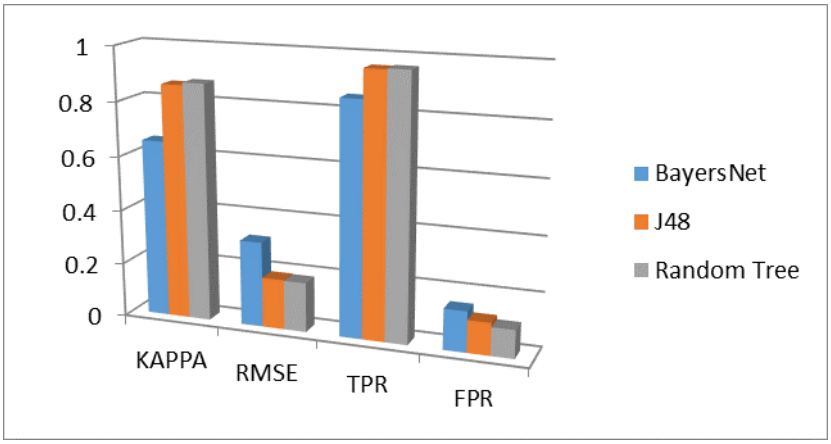

Figure 4.2 comparison of the three algorithms on NSS-kdd_3500 dataset

4.3 Result Discussion for the Dataset NSS-kdd_3000

This subsection depicts details of the experiment carried out on the Dataset NSS_KDD_3000

Table 4.4: Result for Dataset NSS - KDD_3000

Sample	Classifier	KAPPA	RMSE	CCI	TPR	FPR
NSSkdd3000	BayesNet	0.656	0.306	2664 (88.8%)	0.888	0.161
	J48	0.8439	0.1915	2864 (95.46%)	0.960	0.115
	Random Tree	0.8561	0.2047	2874 (95.8 %)	0.964	0.095

Kappa Characteristics: The kappa statistics values for the data set NSSkdd_3000 analyzed are 0.656, 0.8439, and 0.8561 representing the BayesNet, J48, and the Random Tree classifier algorithm respectively. The Random Tree classifier algorithm show the highest and the better result, followed by the J48 algorithm with a value of 0.863, and the Bayesian algorithm produced the lowest of all with a value of 0.6898.

Root Mean Square Error: In Table 4.3, the RMSE field shows that the BayesNet gave the highest value of 0.306. The Random Tree gave a value of 0.2047 and the J48 show a value of 0.1915. The J48 algorithm had the lowest value.

Correctly Classified Instances: Under the correctly classified instance in the Table 4.4, the result clearly shows that Random Tree algorithm out scored the other algorithms used with a value of 2874 (95.8 %). Followed by the J48 and the Bayes algorithms with values of 2664 (88.8%) and 2864 (95.46%).

True Positive Rate: This value has to be high because it is a ratio of the identified normal data to that of the entire normal data. The BayesNet had the lowest value of 0.888, followed by the J48 with a value of 0.960. In this case also, the Random Tree algorithm showed a promising value of 0.964 which is the highest recorded value.

False Positive Rate: This value has to be low because it is used to measure the ratio of incorrectly labeled anomaly to the entire malicious data. The BayesNet had the highest value of 0.161, followed by the J48 with a value of 0.115. In this case also, the Random Tree algorithm showed a promising value of 0.095 which is the lowest value.

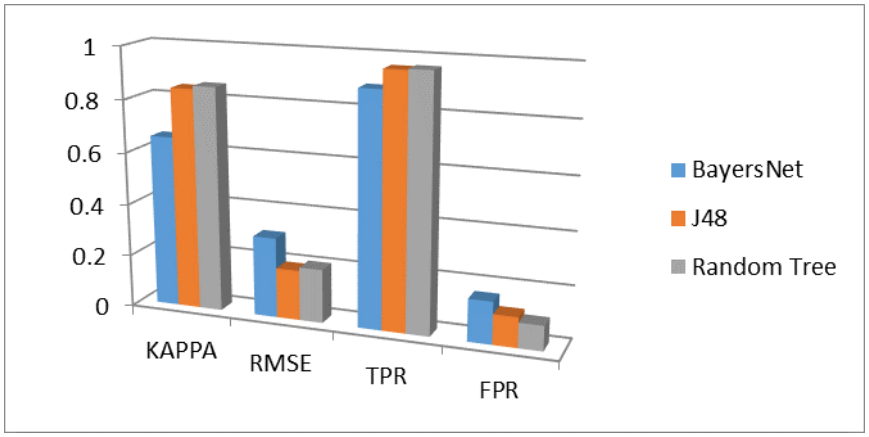

Figure 4.3: Comparison of the three algorithms on NSS-kdd_3000 dataset

Receiver Operating Characteristics ROC

The Receiver Operating Characteristics was developed in the 1950's to help set the operating point of a communication system and has also been used to set the parameters

(before learning) or thresholds (after learning) of many learning algorithms for any two classes. In figure 4.4, we can see that the Random Tree has the smallest area coverage.J48 curves shows the area runs exponentially up while the BayesNet curve has more area coverage. Thus, under this ROC construct the BayesNet is more efficient. ROC is calculated using the existing Dataset processed from NSS-kdd_3000 Dataset

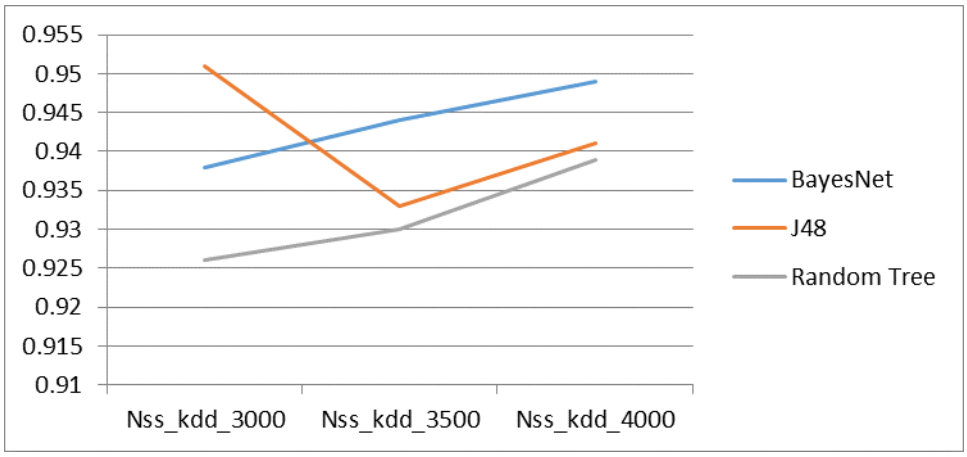

Figure 4.4: Depicts the ROC curve showing performance of the three Algorithms

Conclusion and Future Works

From the above tables and figures, we can say that the Random Tree Neural Network performs relatively well. When the Random Tree Neural network result is compared with that of other classifiers namely, BayesNet and J48; the Random Tree Neural Network outperforms all the other classifiers, except for the Root Mean Square Error in Table 4.3 where the J48 smartly edges past the Random Tree with 0.1915 Root Mean Square Error (RMSE) for the Random Tree Neural Network was 0.2047.

On the basis if the experiment carried out and the result obtained, the Random Tree algorithm did show the potential of being more efficient when compared with other two. The Random Tree Neural Network algorithm has really shown that there is a bright future for having an algorithm for analyzing network traffic with low False Positive Rate (FP), a high True Positive Rate (TP) and a very low Root Mean Square Error (RMSE) which clearly defines how reliable information technology security infrastructures are. Future works could be done with the comparison of the efficiencies of WEKA, Matlab and Java software.

References

Abraham, A., Peddabachigari, S. & Thomas, J. (2004). Intrusion detection system using decision tree and support vector machines. *NA,* 1 - 16.

Aziz, S. A., Salama, A. M., Hassanien, A. & Hanafi, O. E. S. (2012). Artificial immune system inspired intrusion detection system using genetic Algorithm. *Informatica*, 36, 347 - 357.

Bharti, K., Jain, S., & Shukla, S. (2010). Fuzzy K-mean clustering via J48 for intrusion detection system. *International Journal of Computer Science and Information Technologies,* 1(4), 315 - 318.

Chandolikar, N., & Nandavadekar, V. (2012). Comparative analysis of two Algorithms for intrusion attack classification using KDD cup 99 dataset. *International Journal of Computer Science and Engineering,* 1(1), 81 - 88.

Hoque, M. S., Mukit, A. M., & Bikas, A. M. (2012). An implementation of intrusion detection system using genetic Algorithm. *International Journal of Network Security and Its Applications,* 4, 109 -120.

Mulay, A. S., Devale, R. P., & Garje, G. (2010). Intrusion detection system using support vector machine and decision tree. *International Journal of Computer Applications,* 3, 40 - 43.

Rohit, P. (2010). *Instantaneous intrusion detection system.* Master's Thesis, Oklahoma State University.

Sujitha, B., Ramani, R., & Parameswari. A. (2012). Intrusion detection system using fuzzy genetic. *International Journal of Advanced Research in Computer and Communication Engineering,* 1(10), 827 - 831.

Tang, H. & Cao, Z. (2009). Machine learning based intrusion detection Algorithm. *Journal of Computational Information Systems*, 1825 - 1831.

Wu, M. T. (2009). Information assurance tools report. *Intrusion Detection Systems*, VA.