Comparative Evaluation of Machine Learning Algorithms for Network Intrusion Detection Using Weka

Nureni Ayofe Azeez, Obinna Justin Asuzu, Sanjay Misra, Adewole Adewumi, Ravin Ahuja and Rytis Maskeliunas

1 Introduction

Even before the advent of mainstream computers, data security has been one of the topics of vital importance. Over the past few years, more computers have been plugged to the Internet, while many are being networked together, thereby increasing the risk of security threats to these systems.

A network intrusion can simply be defined as any activity considered illegal, unauthorized, unapproved, and unethical to the smooth running of computer network

N. A. Azeez · O. J. Asuzu
School of Computer Science and Information Systems, North West University, Potchefstroom, South Africa
e-mail: nazeez@unilag.edu.ng

O. J. Asuzu
e-mail: bnnsz384@gmail.com

N. A. Azeez · O. J. Asuzu
University of Lagos, Lagos, Nigeria

S. Misra (✉) · A. Adewumi
College of Engineering, Covenant University, Ota 1023, Nigeria
e-mail: sanjay.misra@covenatuniversity.edu.ng; ssopam@gmail.com

A. Adewumi
e-mail: Wole.adewumi@covenatuniversity.edu.ng

R. Ahuja
University of Delhi, New Delhi, India
e-mail: ravinahujadce@gmail.com

R. Maskeliunas
Kaunas University of Technology, Kaunas, Lithuania
e-mail: rytis.maskeliunas@ktu.lt

S. Chakraverty et al. (eds.), *Towards Extensible and Adaptable Methods in Computing*, https://doi.org/10.1007/978-981-13-2348-5_15

activities. In an effort to detect intrusion, the defender must have a very clear and precise understanding of the attack process, how it works and penetrates [1].

Usually, whenever there is a network intrusion, some of the available and valuable network resources meant for authorized user are completely absorbed and trampled upon. Therefore, the need to design and deploy an efficient network intrusion system which will prevent the intruders and hackers is essential [2].

An intrusion detection system therefore can simply be defined as an application that controls, monitors, manages, and directs a network of system from malicious and unauthorized activities that might violate various established network [1].

A network intrusion detection system is a type of intrusion detection technique that combines outputs from multiple sources and uses different approaches to detect these activities and distinguish the unwanted activity from authentic ones [3]. A lot of works have gone into developing efficient systems to detect and predict network intrusions [4].

From written literature on the development of computer framework, one noteworthy is the work of [5], who proposed a model of intrusion-detection by monitoring and statistically analyzing the auditing of system's records should in case there is abysmal and abnormal style in system usage and application. This approach is known as misuse-based detection which can detect only the known attack, but new attacks cannot be identified [6].

2 Literature Review

As available in many publications (both old and recent), it is very evident that many works have been carried out in the area of network intrusion [7]. In this section, therefore, efforts are made to review related works carried out by prominent researchers.

In an attempt to finding lasting solution to network intrusion [8, 9], applied a method that made use of k-means clustering to produce numerous training subsets. Neuro-fuzzy models were employed, and finally, an SVM classification model known as the radial SVM model to demonstrate the ability to attain a higher intrusion detection rate.

The result of the demonstration showed a higher performance against back-propagation and decision tree machine learning algorithms. The data used in this procedure was the KDD Cup 99 datasets.

A multiclass chi-square feature selection was proposed by [10] to reduce the number of feature to an optimal set. Also, the proposed method was chosen due to the lack of works using multiclass SVM in intrusion detection [11]. The results of the analysis show a better performance in the reduced number of false alarm when compared to other techniques.

Nidhi et al. [12] proposed the use of a four-layer framework of consecutively preprocessing, encoding, and classifying the data while integrating with a neural network to effectively detect intrusion attacks.

The experiment was conducted using KDD Cup 99 dataset. The result showed a better performance than the most existing system. In the second model, there was an increase in the accuracy of attack while significantly reducing the time.

Mahalanobis distance characteristic ranking was used by Zhao et al. [13] to choose important and vital features as well as improved search algorithm to select a variety and combination of features. The procedure adopted was very helpful to identify and decrease useless and unimportant features with the aim of improving the accuracy and precision of the results. Both the KNN and SVM algorithms were used for evaluating the techniques on the 99 datasets of KDD CUP. The experimental results showed that false rate is low, especially with the reduced feature subsets.

Bayesian network classifier was used by Fengli and Dan [14] to carry out an efficient and effective feature selection with NSL-KDD dataset. The aim of this approach was to carry out a comprehensive comparative analysis with other feature selection techniques. The results obtained show that Bayesian network classifier used low time rate for attack detection and provide an increased precision rate of detection.

Zhao et al. [13] proposed a framework to reduce redundant features, thereby reducing computational cost on the intrusion detection process. This approach uses the information gain algorithm together with chi-square for feature selection after which maximum entropy classifier was used to analyze the data. The dataset used for this procedure is the KDD CUP 99. The results show a 100% accuracy even though some features have been removed.

Fengli and Dan [14] proposed a hybrid feature selection framework based on principal component analysis (PCA) and fuzzy adaptive resonance theory (FART) for improving the detection accuracy of intrusion attacks especially the root2 local attacks. The PCA algorithm was used to reduce the dimensionality of the dataset attributes without losing vital information. The fuzzy adaptive resonance theory was used for classifying the resulting dataset after the feature selection process. The procedure was carried out on the benchmark data from KDD Cup 99 dataset.

The results of the proposed framework as compared to the result of procedures for both [15] and [16] show that the method outperforms the two in terms of detection rate and false alarm rate.

Poojitha et al. [17] proposed an approach based on multiple classifier systems. It uses a pattern recognition approach to extract suitable feature based on the characteristics that distinguish each network activity to produce three main classes. The classification problem was then subdivided into smaller classification tasks of each task related to one of the three classes. The classification result of each task was then fused into a single output based on three fusion techniques, namely voting rule, average rule, and the belief function. The procedure was carried out on UCI KDD dataset by DARPA, and the results show that the error rates of most attacks were kept very low also with low false alarm rate.

A multi-layered scheme was proposed by Adel and Mohsen [18], for intrusion detection. They adopted genetic algorithms, neuro-fuzzy networks, and fuzzy inference approach for the effective analysis of KDD Cup 99 dataset. The scheme has two main layers, with the first layer utilizing a neuro-fuzzy classifier to produce a distinct class labeled result, while the second layer utilizes a fuzzy inference module.

The result of the experimentation showed a high precision and a good performance in DOS attacks analysis [19].

3 Methodology

3.1 Data Exploration

1999 KDD cup dataset was used for the implementation. This dataset was formed by processing portions of the 1998 DARPA IDS evaluation dataset which was also established by MIT Lincoln Lab. A close network was used to generate the artificial data. Hand-injected attacks were used to produce many different types of attack with normal activity in the background.

As the initial goal was to produce a large training set for supervised learning algorithms, there is a large proportion (80.1%) of abnormal data which is unrealistic in real world and inappropriate for unsupervised anomaly detection which aims at detecting "abnormal" data. The KDD 99 dataset generated over the span of 7 weeks is made up of half a billion records, 42 features, and 23 different types of attack. There are three main types of features in this dataset.

There are nine (9) recognized features of individual TCP connections, and content feature has thirteen (13) standard features within a connection, while there are nineteen (19) traffic features when a two-second time window is computed (Table 1).

3.2 Classification Models

Basically, three different models were used to evaluate the KDD dataset and demonstrate performance. So, what are the classification models?

A classification model is a data mining operation that attempts to draw some conclusion by observing a set of tuples. Given one or more tuples, a classification model will try to predict the value of one or more outcomes.

Native Bayes, decision tree, and random forest classification models were used in this analysis.

Decision was reached on the first three algorithms because of their popularity along with observable contradictory results obtained on them from previous researches. What is more, they can provide relatively good performance on the classification task.

Naïve Bayes. It is considered as one of the prominent machine learning algorithms for data classification with belief of independence between a pair of features. This theorem basically tries to use a known outcome to predict a sequence of events that may have led to that outcome. It is basically used for text classification and involves the use of high-dimensional training dataset.

Table 1 Feature description

Name	Description
Basic features	
Duration	Length (number of seconds) of connection
Protocol_type	Type of protocol, e.g., tcp, udp
Service	Network service on the duration, e.g., http, telnet
src_byte	Number of data byte from the source to destination
dst_byte	Number of data bytes from destination to source
Flag	Normal or error status of the connection
Land	1 if connection is from the same host/port, 0 otherwise
Wrong_fragment	Number of "wrong" fragments
Urgent	Number of urgent packets
Content features	
Hot	Number of "hot" indicators
num_failed_logins	Number of failed login attempts
logged_in	1 if login successful else 0
num_compromised	Number of compromised conditions
root_shell	1 if root shell is obtained else 0
su_attempted	1 if "su command" attempted else 0
num_root	Number of "root" accessed
num_file_creation	Number of file creation operation
num_shells	Number of shell prompts
num_access_files	Number of operation on access control files
num_outbound_cmds	Number of outbound commands in an ftp session
is_hot_login	1 if the login belongs to the hot else 0
is_geust_login	1 if login as guest else 0
Traffic features	
Count	Number of connection to the same host as the current connection in the past 2 s
serro_rate	% of connections that have "SYN" error
rerror_rate	% of connection that "REJ" error
same_srv_rate	% of connection to the same service
diff_srv_rate	% of connection to different services
srv_count	Number of connections to the same service as the current connection in the past 2 s
srv_serror_rate	% of connection that have "STN" error
srv_rerror	% of connection that have "REJ" error
srv_diff_host_rate	% of connection to different hosts

There are several applications attributed to Naïve Bayes algorithm. The prominent among them are document categorization, email spam detection, sexually explicit content detection, personal email sorting as well as language and sentiment detection.

This is regarded as a form of simple probabilistic classifiers that depends on Bayes' theorem with independence belief and assumption between the pair of features. With Naive Bayes algorithm, training of dataset can be done efficiently and reliably. With it, one can make accurate and fast predictions on the dataset [20]. It assists to compute the conditional probability distribution of each feature in a given dataset. Naïve Bayes is majorly used in some areas of application such as text retrieval, text categorization, and the problems related to judging documents. It is mathematically represented as:

$$\rho(\boldsymbol{A}|\boldsymbol{B}) = \frac{\rho(\boldsymbol{B}|\boldsymbol{A})\rho(\boldsymbol{A})}{\rho(\boldsymbol{B})} \tag{1}$$

ρ(A): Likelihood that A and B occur independent of each other.
ρ(B|A): Likelihood that B occurs given that A also occurs.
ρ(A|B): Likelihood that A occurs given that B occurs.

Decision Tree Model Algorithm. Decision tree assists in building classification recursively by dividing a given dataset into subsets. It uses a tree-like graph for decision making where the possible consequences include but not limited to resource cost, chance event outcomes, and utility. It is usually used in decision analysis to identify a technique that can be best used to achieve a goal [19].

Algorithm 1: Algorithm for decision tree model

start
- Begin at the root
- Carry out the test
- Follow and trace the edge corresponding to the outcome
- Go to 2 except leaf
- Forecast the outcome associated with the leaf

stop

Decision trees used in data mining are of two main types:

- **Classification tree** which evaluates different combinations of features of an instance to determine the class to which the instance belongs.
- **Regression tree** is used when the outcome of the evaluation is a numeric value (e.g., the price of a good).

The term **classification and regression tree (CART)** was first introduced by LEO BREIMAN in 1984. This is a simple method for building both classifiers and regressors. The input space is partitioned into two dimensions.

Decision tree learning is a machine learning algorithm that uses the tree models to try to predict a set of outcomes base on an input. This is one of the prominent and important techniques for data classification. It has a structure of flowcharting with

each internal node connotes a status test on an attribute. Each of the branches stands for the outcome of a test, while each node embraces a class label. The root node is regarded as the topmost node. There are many types of decision tree algorithms. Among them are 4.5 (this is an extension of the basic ID3 algorithm), chi-squared automatic interaction detector (CHAID), classification and regression tree (CART) and Iterative Dichotomiser 3 (ID3), conditional inference trees, and multivariate adaptive regression splines (MARS) [19].

Random Forests. These are known machine learning algorithms that ensemble decision trees. They are majorly used for regression and classification. One of the main benefits of random forests is in their capability to reduce the risk of overfitting after combining many decision trees. Random forests have similar features like decision trees which include capturing feature interaction and nonlinearity. What is more? They also handle categorical features. Random forests train a set of decision trees in parallel. They do this by injecting randomness into the process of training. Combination of predictions from different trees enhances the performance on test data.

3.3 Data Analysis

The KDD Cup 99 consists of both training (labeled) and test (unlabeled) datasets. Because these datasets contain over a billion records, only 10% subset of each was used for this analysis. The raw data at this point is messy and cannot be used directly for this work. This data must undergo data clean up and feature engineering steps to make it suitable for analysis.

These two steps addressed the quality issue in these datasets which are as follows:

- Inconsistent values
- Duplicate records
- Missing values
- Invalid data
- Outliers
- Bias data
- Low variance data (irrelevant features).

4 Implementation

4.1 Cleaning up the Data

The training dataset was too large and time-consuming for building models, so only 10% of the data was used amounting to about 494,021 records in the dataset. Duplicate rows were removed from the dataset resulting in a total of 145,586 of the training data and 77,291 of the testing data.

The distribution of the training data shows that about 60% of the training data has been labeled as normal which makes this dataset bias. A biased data like this will have a large impact on the analysis, especially where the analysis is focused on classifying intrusion. To resolve this, the data was normalized by reducing the number of records labeled as normal.

4.2 Feature Engineering

This helps to reduce the data amount because there are a lot of features in this dataset. There are 41 features in total, and some of these features are either useless or irrelevant to the intrusion detection problem, so by selecting only useful features, any meaningless calculation can be avoided.

This also helps to improve the accuracy by removing misleading or unrelated features. Some feature correlated with each other can cause overfitting. Also, the variance of the values of each feature is calculated; this is done to remove features in the dataset that have the same set of values and therefore reducing the amount of unnecessary work to be done in training the dataset.

Finally, feature normalization was done on the dataset. Some features have a range of vales which are very high. For example, in the dataset, the features src_byte and src_dest have their values ranging from 0 to more 50,000, while many features have their values ranging from 0 to 1. It is wise to normalize the features so that all the features will have influence on the result.

Also, there are other methods to reduce noise in the output values like early blockage. In achieving this, we use algorithms for identifying noisy training and completely eliminate the likely and suspected noisy training. This is good as early detection is good and not expensive to implement. At the end of the preprocessing stage, there are 39 features and 145,586 records left for classification.

The experiment was set up on Intel Core i7 processors, 8 GB RAM, 1 TB HDD, Windows 10 PC, and Weka machine learning workbench was utilized for the classification task. The classification was performed based on the 22 attack categories. The testing of dataset was processed using the Naive Bayes, decision tree, and random forest classifiers.

In measuring the performance of each of the techniques used, we adopted accuracy, precision, sensitivity, and specificity rate with expressions hereunder:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{3}$$

$$\text{Specificity} = \frac{TN}{FP + TN} \tag{4}$$

$$FScore = \frac{2TP}{2TP + FP + FN} \tag{5}$$

Table 2 Confusion matrix

		Predicted classes		
		a	b	c
Actual classes	a	TP		
	b		TP	
	c			TP

Table 3 Accuracy for training dataset for each algorithm

Accuracy (%)				
	First run	Second run	Third run	Average
Naïve Bayes	77.04	76.77	75.41	76.41
Decision tree	99.86	99.85	99.83	99.85
Random forest	99.93	99.91	99.92	99.92

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

where FN is false negative, TN is true negative, TP is true positive, and FP is false positive. The accuracy is determined by finding the probability of a correct classification which is calculated by dividing the total number of attacks detected by the total number of attacks in the dataset.

The sensitivity is the ability of the system to detect an anomaly, and specificity is the ability of the system to correctly rule out an attack in a normal connection.

A "confusion matrix" in most cases can be used to signify the result, as shown in Tables 2. This table correlates all the actual classes in the row against the predicted classes in the columns. Each class is represented by a short character. For example, the class "back" is represented by the character "a". In confusion matrix, a cell which has the same class for both the row index and column index is the true positive, while other cells are either false positive or false negative.

The total accuracy of the algorithm was calculated from the confusion matrix. The accuracy is the ratio of a number of correctly classified instances or record to the total number of instances or record set.

From Tables 1, 2, 3 and 4, it can be deduced that the diagonal cell shows the numbers of correctly classified records (instances) which are known as the true positives, while the rest of the cell holds are miss-classification count for the corresponding class. The miss-classified instances can be referred to as either false negative or false positive depending on context. The total accuracy is the ratio of sum of TP divided by the total number of records

$$Total\ Accuracy = \frac{\sum TP}{Total} = \frac{TP + TN}{Total} \tag{7}$$

Table 4 Performance evaluation of each model for all classes

	Precision			Sensitivity			Specificity			Accuracy		
	NB	DT	RF	NB	DT	RF	NB	DT	RF	NB	DT	RF
Back	97.32	99.73	100.00	97.40	99.14	99.90	99.98	100.00	100.00	99.96	99.99	100.00
Teardrop	98.54	100.00	100.00	99.94	100.00	100.00	99.99	100.00	100.00	99.99	100.00	100.00
Loadmodule	2.17	22.22	88.89	83.33	19.44	63.89	99.83	100.00	100.00	99.83	99.99	100.00
Neptune	99.99	99.98	99.98	99.50	99.98	100.00	99.99	99.99	99.99	99.76	99.99	99.99
Rootkit	0.34	0.00	–	37.50	0.00	0.00	98.06	100.00	100.00	98.05	99.99	99.99
phf	2.16	83.33	100.00	100.00	100.00	100.00	99.91	100.00	100.00	99.91	100.00	100.00
Satan	46.80	98.18	99.92	93.23	97.16	97.23	99.16	99.99	100.00	99.11	99.97	99.98
Buffer_overflow	11.54	77.41	86.10	25.00	81.67	97.78	99.92	100.00	100.00	99.90	99.99	100.00
ftp_write	0.18	33.33	100.00	50.00	16.67	50.00	99.00	100.00	100.00	98.99	100.00	100.00
Land	48.23	88.80	89.93	96.97	83.11	93.64	99.98	100.00	100.00	99.98	100.00	100.00
Spy	–	0.00	–	–	–	–	100.00	100.00	100.00	100.00	100.00	100.00
ipsweep	8.94	99.40	99.82	83.94	98.47	97.11	95.15	100.00	100.00	95.09	99.99	99.99
Multihop	4.60	0.00	100.00	55.56	0.00	77.78	99.95	100.00	100.00	99.95	100.00	100.00
Smurf	6.47	100.00	100.00	99.42	98.79	99.69	92.65	100.00	100.00	92.69	99.99	100.00
pod	1.86	100.00	100.00	98.82	100.00	99.70	92.36	100.00	100.00	92.37	100.00	100.00
Perl	0.00	100.00	100.00	0.00	66.67	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Warezclient	13.75	97.44	99.25	49.14	97.35	98.15	97.47	99.99	100.00	97.09	99.97	99.99
nmap	6.21	92.32	99.00	21.81	92.83	97.02	99.52	99.99	100.00	99.41	99.99	100.00
imap	23.98	83.34	100.00	81.11	28.89	67.78	99.98	100.00	100.00	99.97	99.99	100.00
Warezmaster	3.71	68.89	95.83	82.87	76.85	82.87	99.67	100.00	100.00	99.67	99.99	100.00
Portsweep	17.12	97.00	98.27	91.44	96.71	98.06	98.20	99.99	100.00	98.18	99.98	99.99
Normal	99.90	99.88	99.91	62.18	99.93	99.99	99.91	99.82	99.86	76.96	99.88	99.94
Guess_passwd	31.16	97.70	100.00	90.49	90.49	92.25	99.91	100.00	100.00	99.91	100.00	100.00

5 Results and Discussion

Table 3 shows the accuracy of each classifier for each run and a computed average. Naïve Bayes classifier performed the worst in detecting most of the attacks with an average accuracy of 76.41%, while random forest algorithm is the best with an average accuracy of 99.92 followed by decision tree with 99.85% accuracy in average.

The test result for each classifier is summarized in Table 4. The table represents the measure in terms of average precision, sensitivity, specificity, and accuracy for the three classifiers. The averages are computed from the three different test runs labeled as NB for Naïve Bayes, DT for decision tree, and RF for random forest.

Figure 1 shows the average precision of each algorithm against all the attack types. Naïve Bayes scores the least in this evaluation. Also, Fig. 1 shows a very low precision loadmodule, rootkit, spy meaning the algorithm falsely flag them as attacks especially Naïve Bayes.

Figure 2 also shows the average sensitivity of each algorithm. Again, there is very low sensitivity loadmodule, rootkit, spy, meaning the algorithm could not correctly flag them as attacks.

Figure 3 shows the average specificity of each algorithm. This figure shows that all three algorithms could to some degree correctly specify which attack it was, and Naïve Bayes still scores the least in this evaluation.

Figure 4 shows a good performance on the accuracy of the algorithm, especially the random forest which had the best accuracy across all attacks, while Naïve Bayes still performed the least.

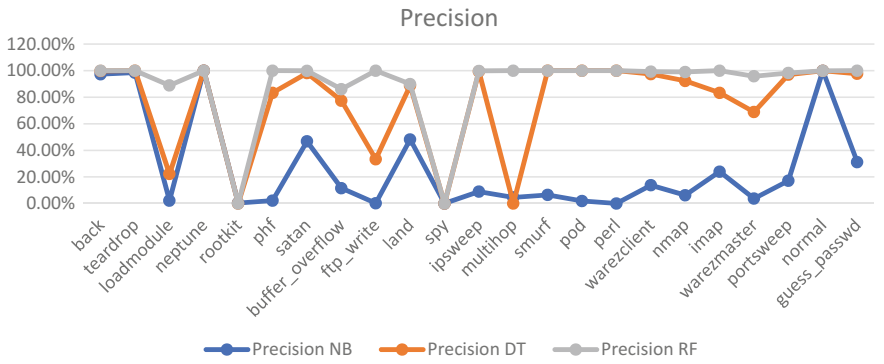

Fig. 1 Precision evaluation of each model for all classes

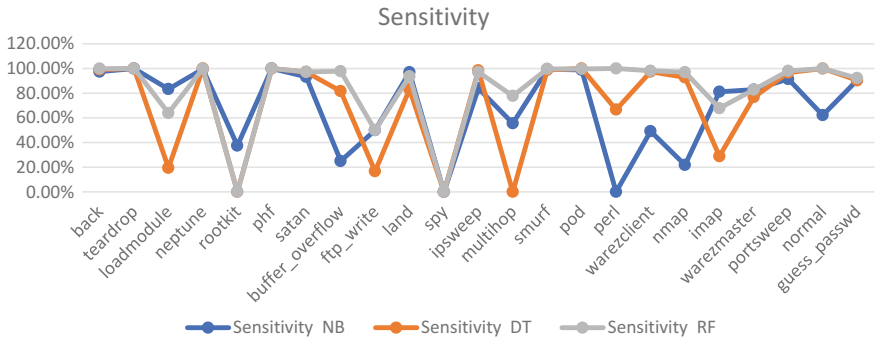

Fig. 2 Sensitivity evaluation of each model for all classes

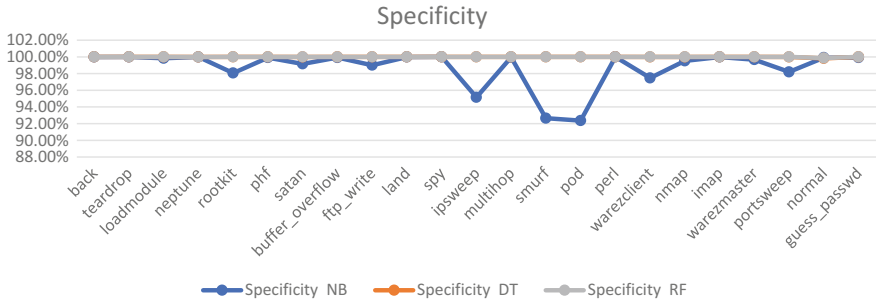

Fig. 3 Specificity evaluation of each model for all classes

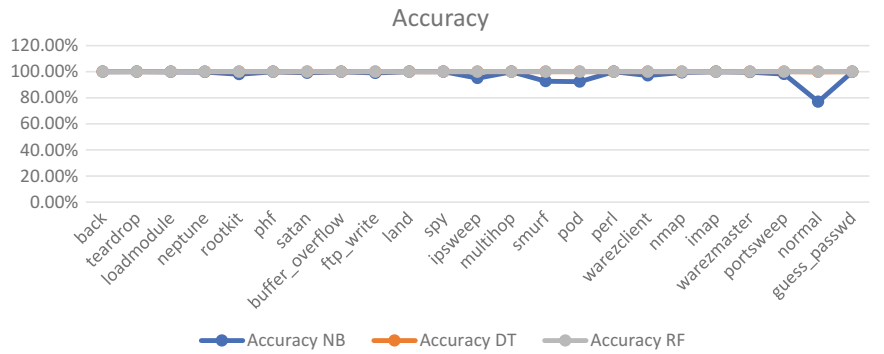

Fig. 4 Accuracy evaluation of each model for all classes

6 Conclusion

This work reviewed and evaluated the performance of three of the commonly used machine learning algorithms for intrusion detection. These algorithms were evaluated using a big data and machine learning data processing tool developed by University of Waikato, New Zealand, called Weka. The authors used a real-time artificial dataset generated by MIT Lincoln Lab by simulating a closed network and hand-injected attacks. There are three runs in the procedure of which the dataset was split into different ratios (60 : 40, 50 : 50, 40 : 60) for both training and test data for each run. To improve the performance of the result, a lot of preprocessing was carried out on the data to remove correlated and useless features, overfitted, duplicate, biased and noisy data. This procedure also improved the time taken to classify the attacks. The accuracy of the all the algorithms was improved as the ratio of training data to test data was increased. This procedure saw a very good performance across the three runs for decision tree and random forest while experiencing a poor performance from Naïve Bayes algorithm.

References

1. Azeez, N. A., & Ademolu, O. (2016). CyberProtector: Identifying compromised URLs in electronic mails with Bayesian classification. In *International Conference Computational Science and Computational Intelligence,* pp. 959–965.
2. Azeez, N. A., Okunoye, O. B., Oladeji, F. A., & Edafeadjeke, E. O. (2015). Towards an adaptive and scalable access control model for a cloud-based environment. In *Nigerian Computer Society (NCS) 12th Annual Conference International Conference on Information Technology for Inclusive Development* (Vol. 26, pp. 214–223).
3. Singh, R., & Singh, D. (2014). A review of network intrusion detection system. *International Journal of Engineering and Technoscience, 5*(1), 10–15.
4. Giorgio, G., & Fabio, R. (2003). Intrusion detection in computer networks by multiple classifier systems. *Pattern Recognition Letters*, 1795–1803.
5. Denning, E. D. (1987). An Intrusion-detection model. *IEEE Transaction on Software Engineering*, 222–232.
6. Azeez, N. A., & Babatope, A. B. AANtID: An alternative approach to network intrusion detection. *Journal of Computer Science and Its Application, 23*(1) (2016).
7. Azeez, N. A., & Venter, I. M. (2013). Towards ensuring scalability, interoperability and efficient access control in a multi-domain grid-based environment. *SAIEE Africa Research, 104*(2), 54–68.
8. Azeez, N. A., & Irwin, B. (2010). Cyber security: Challenges and the way forward. *GESJ: Computer Science and Telecommunications*, 1512–1232.
9. Chandrasekhar, A. M., & Raghuveer, K. (2013). Intrusion detection technique by using k-means, fuzzy neural network and SVM classifiers. In *2013 International Conference Computer Communication and Informatics (ICCCI)*. Coimbatore, India.
10. Sumaiya, T. I., & Aswani, K. C. (2016). Intrusion detection model using fusion of chi-square and multi class SVM. *Journal of King Saud University.*
11. Smaha, R. E., & Haystack. (1988). An intrusion detection system. In *Proceedings of the IEEE Fourth Aerospace*. Orlando, FL.

12. Nidhi, S., Krishna, R., Rama, K. C. (2013). Novel intrusion detection system integrating layered framework with neural network. In *IEEE 3rd International Advance Computing Conference (IACC)*. Ghaziabad.
13. Zhao, Y., Zhang, Y., Tong, W., & Chen, H. (2013). An improved feature selection algorithm based on MAHALANOBIS distance for network intrusion detection. In *Sensor Network Security Technology and Privacy Communication System (SNS & PCS), 2013 International Conference.* Nangang, China.
14. Fengli, Z., & Dan, W. (2013). An effective feature selection approach for network intrusion detection. In *Networking, Architecture and Storage (NAS), 2013 IEEE Eighth International Conference.* Xi'an, China.
15. Yang, L., Bin-xing, F., You, C., & Li, G. A (2006). lightweight intrusion detection model based on feature selection and maximum entropy model. In *Communication Technology, 2006. ICCT '06. International Conference.* Guilin, China.
16. Preecha, S., & Woraphon, L. (2015). Anomaly traffic detection based on PCA. *The International Arab Journal of Information Technology*, 253–260.
17. Poojitha, G., Naveen, K. K., & Jayarami, P. R. (2010). Intrusion detection using artificial neural network. In *2010 International Conference on Computing Communication and Networking Technologies (ICCCNT).* Karur, India.
18. Adel, N. T., & Mohsen, K. (2007). A new approach to intrusion detection based on an evolutionary. *Computer Communications,* 2201–2212.
19. Hui, L., & Jinhua, X. (2009). Three-level hybrid intrusion detection system. In *International Conference on Information Engineering and Computer Science, 2009. ICIECS 2009.* Wuhan, China.
20. Zhang, H. (2004). *The optimality of Naive Bayes*. New Brunswick, Canada: University of New Brunswick.
21. Azeez, N. A., Ademola, P. A., Ademola, O. A., & Kehinde, K. A. (2011). Ancaee: A novel clustering algorithm for energy efficiency in wireless sensor networks. *Wireless Sensor Network*, 307–312.
22. Balogun, A. O., & Jimoh, R. G. (2015). Anomaly intrusion detection using an hybrid of decision tree. *A Multidisciplinary Journal Publication of the Faculty of Science, Adeleke University, Ede, Nigeria*, 67–73.
23. Heady, R., Luger, G., Maccabe, A., & Servilla, M. The architecture of a network. Technical Report, Department of Computer Science, University (1990).
24. Govind, P. G., & Manish, K. (2016). A framework for fast and efficient cyber security network. In *6th International Conference on Advances in Computing & Communications, ICACC 2016.* Cochin, India.
25. Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to Information Retrieval.* Cambridge University Press.