Analysis on Improving the Performance of Machine Learning Models Using Feature Selection Technique

N. Maajid Khan, Nalina Madhav C, Anjali Negi, and I. Sumaiya Thaseen(✉)

School of Information Technology and Engineering, Vellore Institute of Technology, Vellore, India
sumaiyathaseen@gmail.com

Abstract. Many organizations deploying computer networks are susceptible to different kinds of attacks in the current era. These attacks compromise the confidentiality, integrity and availability of network systems. It is a big challenge to build a reliable network as several new attacks are being introduced by the attackers. The aim of this paper is to improve the performance of the various machine learning algorithms such as KNN, Decision Tree, Random Forest, Bagging Meta Estimator and XGBoost by utilizing feature importance technique. These classifiers are chosen as they perform superior to other base and ensemble machine learning techniques after feature selection. Feature Importance technique is utilized to obtain the highest ranked features. Reduced attributes improve the accuracy as well as decrease the computation time and prediction time. The experimental results on UNSW-NB dataset show that there is a drastic decrease in the computation time with reduced attributes compared to evaluating the model using the dataset with the entire set of attributes.

Keywords: Accuracy · Attributes · Feature selection · Machine learning · Computation time

1 Introduction

With the epic development of computers and network, most of the systems experience security vulnerabilities. Also due to increase in the cyber-crime rate in current ecommerce world, security is the most important factor in a high performance networking system. An internet-attack is a planned way to exploit computer systems that challenges the confidentiality, integrity and availability of the computer network and is an international concern. The attackers use malicious ways to enter into the system and use the data for detrimental purposes leading to system infiltration, information modification etc. These attacks can be monitored using an intrusion detection system (IDS). IDS are used to monitor any type of suspicious activity that might happen in the network traffic. Suspicious activities include attacks like denial of service (DOS), fuzzers, backdoor, generic etc. The analysis for anomaly detection can be done using machine learning techniques on different train and test datasets. As researchers start finding out different ways to increase accuracy and improve the model, there is still enough scope of increasing the accuracy and efficiency of the model.

The methods developed earlier were not potential enough to detect the attacks. Systems used to execute days to track and find the attack that has occurred. Nowadays machine learning algorithms can detect these suspicious activities and categorize the attack that causes the most damage to the system. The system learns by itself based on the past experiences and can process and classify the attack category for the unseen data.

2 Related Work

Many intrusion detection systems (IDS) have been developed using machine learning techniques. An IDS [4] was proposed by using Association Mining Rule (AMR) for data pre-processing. They used Naive Bayes and Logistic Regression algorithms for classifying the normal and attack data. The authors [4] suggested that removal of redundant data minimizes the processing time and increases accuracy. The characteristics of features of UNSW-NB15 and KDD99 datasets are analyzed by the authors [5]. AMR algorithm is used to select the best features on which classifiers are utilized to evaluate accuracy. The results showed that KDD99 attributes were not too efficient compared to UNSW-NB15 attributes but the accuracy produced by KDD99 dataset was better. Machine learning is utilized for detecting network anomalies. Belouch [8] evaluated the accuracy, model building time and prediction time of three classification algorithms using Apache Spark and addresses the issue related to low accuracy and prediction time on UNSW-NB15 dataset when all 42 attributes were utilized. Experiment results showed that the accuracy of Random Forest Classifier was high with 97.49% and fastest detection time which was 0.08 s for binary classification. According to [9], the results showed that Random Forest ensemble classifier outperformed other base classifiers. They proposed a model fusing Random Forest classifier and was efficient with high detection rate and low false rate on NSL-KDD dataset. Random Forest classifier eliminates over fitting problem.

Muniyal [3] states that feature selection has to be implemented as a two-step process. Random Forest classifier is used and features with high importance score is selected and sorted based on the importance score which is then used for the classification. Due to the redundant data in the NSL-KDD dataset, the data is pre-processed to avoid biased accuracy towards the repeated data. The results show that Random forest classifier produced better accuracy with reduced attributes with increase in performance rate and reduction in False Positive Rate (FPR).

In this literature work [6], several feature selection methods have been studied and analysed. They [6] used filter, wrapper and hybrid approaches for choosing the best feature subset. Different feature selection techniques were applied to KDD99 dataset and determined the evaluation criteria for each method. Using these methods, they eliminated the repetitive attributes or features to increase the accuracy of the intrusion detection system.

Feature fusion and the decision fusion techniques were used by [7] in NIDSs. They proposed evaluation criteria for evaluating different data fusion techniques. They suggest that hybrid fusion techniques seem efficient compared to SVM and other methods. They [7] also found that ensemble algorithms like Adaboost can combine multiple decisions than any other method. The experiment was carried out on KDD99 dataset. They [7] concluded that for developing an excellent network intrusion detection system, data fusion techniques prove to be better as it help in improving the performance of the detection system.

Three important factors for developing an IDS system are pre-processing, feature selection and algorithm design. This paper [2] proposes a modified Naïve Bayes algorithm that improves accuracy and reduces the prediction time. In the proposed system, the algorithm approximates the attribute interactivity using conditional probabilities. They compared the proposed algorithm with Naïve Bayes, J48 and REPTree using several measurement parameters. Results show that good results in detecting intrusion.

In this paper [1], the authors have focused on false positive and false negative performance metrics to increase the detection rate of the IDS. Various machine learning classifiers such as J48, Random Forest, Random Tree, Decision Table, MLP, Naive Bayes, and Bayes Network were evaluated based on several tests on KDD intrusion detection dataset. Random forest classifier was successful in achieving the highest average rate.

3 Preliminary Examination of Data

3.1 Data Description

The UNSW-NB 15 (“UNSW-NB15 data set,” 2015) data set was created for producing both normal and attack behaviours generated using tcpdump tool to capture raw traffic. This dataset contains 9 attack types. 42 attributes are considered for our research work. The dataset is divided into testing set and training set. The class distribution is shown in Table 1.

Table 1. Dataset class distribution

Category	Training set	Testing set
Normal	37,000	56,000
Analysis	677	2000
DoS	4089	12,264
Backdoor	583	1,746
Exploits	11,132	33,393
Fuzzers	6,062	18,184
Generic	18,871	40,000
Shell code	378	1,133
Worms	44	130
Reconnaissance	3,496	10,491
Total records	82,332	175,341

The UNSW-NB 15 dataset includes complex patterns compared to NSL KDD99 dataset and contains 9 different attack types unlike NSL KDD99 dataset with only 5 different attack types. Hence for the analysis, UNSW-NB 15 dataset is used to evaluate various classification methods.

3.2 Data Preprocessing

Feature Scaling. Standardization of a dataset is very important for many machine learning algorithms which uses Euclidean distance. If they are not standardized, then there is a possibility that attributes which have values in larger range may be given higher importance. Since each feature may not be represented in same unit of measurement, features with varying magnitude will have an adverse effect on the machine learning algorithms. This can be avoided by transforming the data to same range through standardization. Standard Scalar is a standardization technique utilizing centring and scaling. Standard deviation is calculated for each feature by computing the statistics on the samples in the training set. This will distribute in such a way that mean will be 0 and standard deviation will be 1.

Every feature in the dataset is scaled as

$$\text{Mean, } \bar{X} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{1.1}$$

$$\text{Standard Deviation, } \sigma_X = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (X - \bar{X})^2} \tag{1.2}$$

$$\text{Standardization, } X' = \frac{(X - \bar{X})}{\sigma_X} \tag{1.3}$$

Where, X is the attribute, N is the total number of instances, X' is the new rescaled, $\bar{X}$ is the mean and σ_X is the standard deviation.

4 Proposed Methodology

4.1 System Architecture

In this section, we describe the architecture of adapting the important features from the dataset which would improve the accuracy of the model and also drastically improve the speediness of training and computation time of all the models. Figure 1 shows the proposed architecture to improve the performance of the model and minimizing the computation time. The step-by step approach of the proposed model is discussed below:

- Choose an input data set, UNSW-NB15.
- Load the dataset as Training and Testing datasets separately.

- Apply Feature scaling technique. Determine the mean (1.1) and standard deviation (1.2) for the feature and the features are further scaled using the formula (1.3).
- Evaluate the models-KNN, Decision tree, Random forest, bagging Meta estimator and XG Boost, determine the accuracy, training and prediction time of these models.
- Apply Feature selection by using Feature Importance model applied on Random Forest classifier.
- Consider only the reduced attributes which are important for the model obtained from the feature selection technique.
- Execute all the models again with reduced features and evaluate the performance in terms of accuracy, training and prediction time.
- Compare the results.

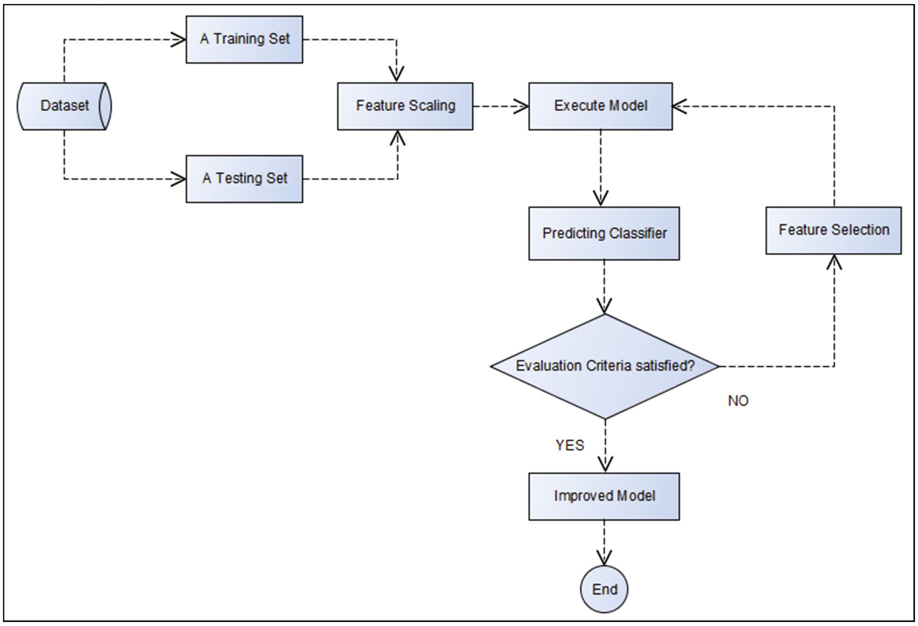

Fig. 1. The proposed architecture for improving computation speed of the models

4.2 Feature Selection

Feature Importance. Feature importance is implemented in our model by utilizing model.feature_importances in Sklearn Random Forest package to analyze about the essential features of the data. The features that are all the more closely related with dependent variable and contribute more for variation of the dependent variable are identified as important features. A large number of features are added to the random forest model and the classifier evaluates and returns back those features which contribute more for improving the performance of the model.

Steps in identifying the important features:

1. Train the random forest model assuming that right hyper-parameters have been considered.
2. Discover prediction score of models and highlight as a benchmark score.
3. Assess prediction scores ‘p’ more times where p is number of features and shuffle randomly across the column of jth feature.
4. Evaluate all the ‘p’ scores by a comparison with the already marked benchmark score. If the result of jth column after shuffling it randomly is troubling the score then the feature is required for the model.
5. Eliminate the features that do not trouble the benchmark score and retrain the model with newly reduced subset of features.

4.3 Evaluation Criteria

Accuracy: It is the most widely used metric for classification models which is determined as the fraction of samples predicted correctly.

$$Accuracy = \frac{(TN + TP)}{(TN + TP + FN + FP)} \tag{1.4}$$

The accuracy score is obtained by considering the input as actual labels and predicted labels.

Precision: It is a measure of result relevancy. It makes sure that the negative sample is never identified as positive.

$$precision = \frac{TP}{TP + FP} \tag{1.5}$$

Recall: It is the fraction of relevant instances that have been retrieved over the total amount of relevant instances.

$$recall = \frac{TP}{TP + FN} \tag{1.6}$$

F-measure: It is a measure that combines both precision and recall measures. It is measured as weighted average of precision and recall.

$$F1score = 2 * (\frac{precision * recall}{precision + recall}) \tag{1.7}$$

Where, *TN* = True Negative, *TP* = True Positive, *FN* = False Negative and *FP* = False Positive.

Computation Time: There are a lot of high-performance computing (HPC) platforms which support heterogeneous hardware resources (CPUs, GPUs, storage, etc.) in the current era. The prediction of application execution times over these devices is a great challenge and is crucial for efficient job scheduling.

5 Results

The proposed model is developed using Anaconda Navigator tool which has a Spyder interface to compute all the models. The different models chosen for our analysis are KNN, Decision tree, Random forest, bagging Meta estimator and XG Boost. These models obtained higher accuracy compared to other existing models in the literature and also in our analysis. Hence, these classifiers are retained for further improving the computation and prediction time using the proposed approach. Table 2 shows the overall accuracy for KNN, Decision tree, Random forest, bagging Meta estimator and XG Boost after feature scaling. Thus it is inferred from Table 2 that Random forest gave the highest accuracy with 74.875% followed by bagging Meta estimator at 74.641%, Decision tree at 74.227%, Boost at 71.437% and finally KNN at 71.103%.

Table 2. Results of the models before feature selection

Classifier	Accuracy	Training time(s)	Predict time(s)	Precision	Recall	F-score
Random forest	74.875	3.656	0.532	0.74	0.75	0.72
XG Boost	71.437	109.946	6.331	0.69	0.71	0.67
Bagging meta estimator	74.641	9.817	1.144	0.73	0.75	0.71
Decision tree	74.227	2.753	0.06	0.74	0.74	0.72
KNN	71.103	25.189	118.457	0.70	0.71	0.68

Table 2 shows the computation time of the models such as training and prediction times. It is inferred that the training time of XG Boost model is very high (109.946 s) while decision tree model showed more efficient in computation time (2.753). This is the reason for not utilizing few ensemble models for classification. Hence feature selection technique is utilized to obtain the important features useful for predicting the models and enhance their performance.

In (Fig. 2), X-axis represents the attribute names and Y-axis represents the level of importance of attributes. The 41 attributes were reduced to 11 attributes based on their importance by using feature importance on Random Forest classifier.

The reduced features of **UNSW-NB15 dataset** are Service, State, Sbytes, Dbytes, Rate, Sttl, Ackdat, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, and ct_src_ltm.

These reduced features would be used for computing the accuracy and performance of the models.

Table 3 shows the accuracy results of all the models with 11 important attributes obtained by feature importance. Random forest results in the highest accuracy with

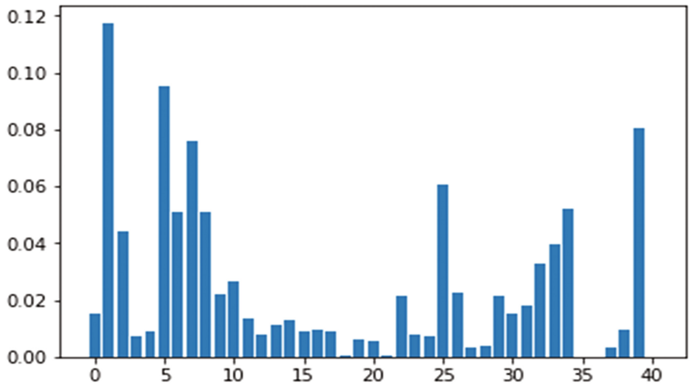

Fig. 2. Feature importance score of 41 attributes

Table 3. Results of the models with reduced attributes

Classifier	Accuracy	Training time(s)	Predict time(s)	Precision	Recall	F-score
Random forest	75.656	0.888	0.447	0.75	0.76	0.73
XG Boost	71.747	34.238	5.876	0.69	0.72	0.67
Bagging meta estimator	75.085	1.901	0.594	0.74	0.75	0.72
Decision tree	74.497	0.35	0.037	0.75	0.74	0.72
KNN	74.395	0.929	8.657	0.74	0.74	0.72

76.656% followed by bagging meta estimator at 75.085%, Decision tree at 74.497%, KNN at 74.395% and XGBoost at 71.747%. It can be observed that there is an improvement in the model accuracy. Table 3 shows the computation time of the models in terms of training and prediction times evaluated on dataset with reduced attributes.

From Table 3, it can be observed that the computation time reduced thereby improving the performance of the model. Thus an analysis of the classifiers that perform better with and without feature importance is studied and results are shown.

6 Conclusion

Accuracy and computation time are critical parameters to assess the performance of the machine learning models. The highest ranked important features are selected utilizing feature importance technique of Random forest classifier. The features with the highest importance score were selected to evaluate other existing machine learning models. The reduction of 41 features to 11 important features improves the XG Boost performance. Performance of Decision tree model also proved to be efficient in comparison to other four models that are used for our analysis. The accuracy of the models increased with reduced attributes and the performance of the model increased by a very big factor. Thus feature importance has been a key factor in decreasing the computation time of the classifier which has not been studied extensively in literature.

7 Future Work

Multiple machine learning models will be integrated to build hybrid architecture of the model. This can further improve the accuracy of the model as well as to speed up the computation time of the models.

References

1. Almseidin, M., Alzubi, M., Kovacs, S., Alkasassbeh, M.: Evaluation of machine learning algorithm for intrusion detection. Department of Information Technology, University of Miskolc, Hungary (2018)
2. Kumar, K., Batth, J.S.: Network intrusion detection with feature selection techniques using machine-learning algorithms. Int. J. Comput. Appl. **150**(12), 1–13 (2016)
3. Belavagi, M.C., Muniyal, B.: Performance evaluation of supervised machine learning algorithms for intrusion detection. Procedia Comput. Sci. **89**, 117–123 (2016)
4. Mogal, D.G., Ghungrad, S.R., Bapusaheb, B.B.: NIDS using machine learning classifiers on UNSW-NB15 and KDDCUP99 datasets. In: IJARCCE (2017)
5. Moustafa, N., Slay, J.: The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems. In: 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), Kyoto, Japan (2015)
6. Kumar, D., Singh, H.: A study on performance analysis of various feature selection techniques in intrusion detection systems, vol. 3, no. 6, pp. 50–54 (2015)
7. Li, G., Yan, Z., Fu, Y., Chen, H.: Data fusion for network intrusion detection: a review. Secur. Commun. Netw. **2018**, 16 (2018)
8. Belouch, M., El Hadaj, S., Idhammad, M.: Performance evaluation of intrusion detection based on machine learning using Apache Spark. Procedia Comput. Sci. **127**, 1–6 (2018)
9. Farnaaz, N., Jabbar, M.A.: Random forest modeling for network intrusion detection system. Procedia Comput. Sci. **89**, 213–217 (2016)