2021 5th International Conference on Electrical Information and Communication Technology (EICT) | 978-1-6654-0906-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/EICT54103.2021.9733679

Evaluation of Machine Learning Algorithms using Feature Selection Methods for Network Intrusion Detection Systems

Md Nafiur Reza
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh
nafiur63@gmail.com

Syed Faysal Kabir
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh
kabirfaysal304@gmail.com

Nushrat Jahan
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh
nushratjahan2511@gmail.com

Maheen Islam
Associate Professor
Department of Computer Science and Engineering
East West University
Dhaka, Bangladesh
maheen@ewubd.edu

***Abstract*—In this modern internet era dominated by technology and automation, network security is a concern for the whole world. Despite many security methods such as virus scanners, firewalls, encryption machines, the present day networks are vulnerable to cyber attacks and data theft. As a result, the Intrusion Detection System (IDS), with their preset blocking and filtering rules, is gaining global attention to detect and prevent the attacks in order to safeguard the network system. This paper evaluates the execution time and accuracy of different feature selection methods using different machine learning algorithms for application in IDSs. Four features selection methods (Pearson correlation, Chi-square, recursive feature elimination, tree-based: select from model) are used with three machine learning algorithms (random forest, logistic regression, k-nearest neighbor) on NSL-KDD datasets using two different machines with high and low configurations. This paper observes that, for both the machines, the best result is obtained by the random forest method in terms of execution time and accuracy.**

***Index Terms*—machine learning, feature selection, intrusion detection, accuracy, execution time**

I. Introduction

Machine learning (ML) is not a separate branch of study anymore, as it has penetrated almost all facets of modern life. The use of ML algorithms can be extended to network security, to prevent the invasion into sensitive systems and processes. Network security systems are primarily classified as intrusion detection system (IDS) [1] and intrusion prevention system (IPS). An IDS works using some blocking and filtering rules preset by the system administrator. An IDS, that will eventually be developed into an IPS, can detect threats by continuously scanning the network traffic. If it finds anything suspicious, it blocks the site and informs the system administrator to take necessary steps against the attack. The focus of this work is the application of ML in network IDS [2]. Before initiating this work, several published papers have been thoroughly studied to understand the state of art of the application of ML in network IDS at present. A brief literature review is presented in Table I, where some noteworthy works have been mentioned with their highlights.

The main contribution of this work is to find the best ML models with the best feature selection method, which results in the fastest execution time and the highest accuracy for use in network IDS. Many other works have worked in a similar approach, as seen from the literature review, but this work differs because no new ML model is built here; rather the existing models are used for the evaluation in two different configuration machines. The rest of the paper is organized as follows: Section II describes the methodology of work adopted in this study, Section III presents and analyses the findings of this work, Section IV highlights the outcome of this study, Section V discusses some future work directions related to this paper, and finally Section VI concludes the work.

II. Methodology of Work

This study has been divided into five working components: dataset selection for training and testing, dataset preprocessing, ML classification methods, system training, and system testing with evaluating. The steps of the methodology of work are shown in Fig. 1.

A. Data-selection for Training and Testing

For evaluating the efficiency of any IDS model, a standard dataset is needed. For this study, the "NSL-KDD" dataset [11] is used, which is the modern version of the "KDDCup99" dataset [12] that is considered as a standard for evaluating ID rates. The NSL-KDD dataset consists of about 4,900,000

TABLE I
LITERATURE REVIEW

Ref.	Dataset used	Algorithms used	Accuracy
[3]	NSL-KDD	Misuse detection and anomaly detection using J48 and Simple Cart	Hybrid Anomaly & Misuse Detection – 99.10% for known attacks and 30.05% for unknown attacks; Light Weight Network - 72.70%
[4]	KDDCUP'99	K-means clustering using Normal, DOS, PROBE, U2R, R2L and RBF Kernel using Gaussian Kernel function	KMSVM (renamed by authors) – 92.86%, KM – 86.67%, SVM – 40% for DOS attack
[5]	NSL-KDD	J48 classifier	J48 gives the highest accuracy
[6]	KDD'99	Pearson Linear Correlation and Mutual Information using Neural Network, CART, ID3Decision tree, Random forest	Neural Network – 99.98%
[7]	NSL-KDD	Neural Network, Nearest neighbor, Decision tree, J48, Meta Pagging, Random tree, REPTREE, DecisionStump, Naive Bayes	Proposed model – 99.81%, Random tree – 99.747%, J48 – 99.74%, Naive Bayes -90.2876%
[8]	UNSW-NB-15	Artificial Neural Network using Chi-square and SVM using Principal Component Analysis and Kernel function	SVM – 91%, ANN – 92%
[9]	NSL-KDD	Correlation-based – wrapper, 41, 17; Chi-square based – filter, 41, 35	SVM-CR- 81%, SVM-CS – 82%, ANN-CR – 94%, ANN-CS – 83%
[10]	Kyoto 2006+	SVM with Gaussian Kernel (SVM-RBF), Random Forest, KNN, and Quadratic Discriminant Analysis	KNN with 3 neighbors has the best testing accuracy between non-clustering and clustering cases; RF – 58%

connection vectors, each of which has 41 unique features. All those features are respectively labeled as normal or attack [13]. The training dataset consists of 21 different attacks from the 37 presents in the test set. The type of attacks in NSL-KDD dataset are basically of four main classes [14], which are Denial of Service attack (DOS attack), Remote to Local (R2L) attack (accessing from an unauthorized remote machine), User to Root (U2R) attack (unauthorized access to the privilege of the root user), and probing attack.

B. Dataset Pre-Processing

Dataset pre-processing is important to increase the efficiency of the models and also to gain more accuracy. NSL-KDD dataset consists of a lot of features and attacks. In real-time attack detection, there can be some irrelevant features and attacks that are of no use for that particular detection. So the data of the NSL-KDD dataset needs to be processed in a proper way. For this work, the data pre-processing has been completed in the following four steps.

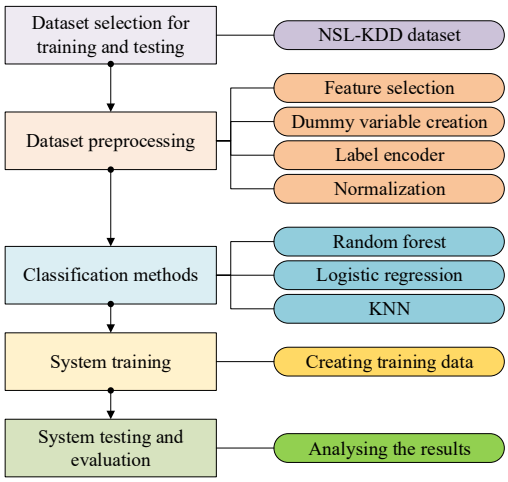

Fig. 1. Steps of the work.

1) Data Transformation: The NSL-KDD dataset has a large number of connection vectors, with 41 features in each with a target class of either "Attack" or "Normal". As the ML models need numerical values to work efficiently and properly, the nominal and symbolic features need to be transformed into numeric data. The "$protocol\_type$" column, "$services$" column and the "$flag$" column of the dataset have been transformed into unique numerical values for every instance based on their characteristics. The label encoder technique has been used for the data transformation.

2) Priority Maintenance: After the transformation of the nominal data into numeric values, the dataset is now formed into usable formation for testing and training. But after transformation, the "$protocol\_type$" column still has some issues with the numerical values as it contains 3 different protocols - TCP, UDP and $ICMP$. When the model acquires the dataset it will be confused about the priority of those protocol types. So, the protocol types should be maintained according to their priority. That is why, dummy variables have been created to keep the protocol types individually in them and set the priority of the types.

3) Dataset Normalization: After the transformation and priority maintenance, the dataset needs to be normalized as it is the rudimentary step of dataset pre-processing to enhance the performance of the IDS, especially when the training and testing dataset is too large. First, the continuous attributes have been normalized using the min-max method of normalization (Eq.1) to make them fit within the range of 0 to 1.

$$X_i = \frac{v_i - min(v_i)}{max(v_i) - min(v_i)}. \tag{1}$$

Here, X_i is the normalized value, and v_i is the factual value of the attribute. In case $max(v_i) = min(v_i)$, then X_i is mapped to 0.

4) Feature Selection: The network IDS often deals with a large amount of data including irrelevant and redundant features. So, to get higher accuracy in shortest time, using feature selection methods is very important [15]. Four feature selection methods are applied in this study, which are as follows:

1) Pearson correlation: Works by selecting the best features based on univariate statistical test.
2) Chi-square: Works with the statistical hypothesis.
3) Recursive feature elimination: Works by choosing smaller features starting from an initial number of features.
4) Tree Based: SelectFromModel: Works by making decision trees.

C. ML Models

After collection of dataset and pre-process of dataset we have selected three ML Algorithms to create models and perform IDS. The algorithms are:

*1) **Logistic Regression (LR)**:* Some notes needed used to build the model using LR algorithm are:

- LR classifier comes with default parameter values, which need to be tested first.
- For better results, some parameter tuning is used. All the parameters will be tuned until a satisfactory result is obtained.
- C penalty, maximum iteration and tol are the important parameters; so, their change will be noted.
- After parameter tuning, the algorithm is used with every feature selection method to get best result.

*2) **Random Forest**:* Some notes needed used to build the model using random forest algorithm are [16]:

- Scikit-learn's [17] built-in classifier model is used from the Ensemble Library.
- Random Forest comes with a lot of default parameter. First checking will be on what result the default parameter setting gives. $n\_estimators$, $criterion$ and $max\_samples$ are the most important parameters.
- After parameter tuning, the algorithm is used with every feature selection method to get best result.

*3) **K Nearest Neighbor (KNN)**:* Some notes needed used to build the model using KNN algorithm are:

- Default parameter values for this classifier needs to change and examine well.
- Weights and $leaf\_size$ parameters need to extract as the dataset is huge.
- Value of n as n-neighbor parameter needs to change in 3,5,7, etc. to get better results.
- After parameter tuning, the algorithm is used with every feature selection method to get best result.

D. System Training

After selecting the ML algorithms, to train and test the models, the dataset was separated into two parts consisting 80% data and 20% data for the training and testing, respectively. To get some unique combinations of data every time for the training of the models, a built-in function of Scikit-learn [17], $train\_test\_split$ is used, which shuffles the data every time in order to train the model.

E. System Testing and Evaluating

The testing phase involves the execution of the created models with 20% data as testing dataset and collecting the accuracy and execution time of the created models with the best number of feature selection.

F. Environmental Limitations

The models of ML algorithms with feature selection methods perform better and faster in high configuration computers. After creating the models, they are tested in two different computers to know the difference of execution time. The computers used in this work are average and comparatively slower than the other high-end computers existing in the present market. The two computers used in this work are mentioned in Table II. Computer 1 is the high configuration machine and Computer 2 is the low configuration machine.

TABLE II
SPECIFICATIONS OF THE TWO COMPUTERS USED IN THIS WORK.

	Processor	RAM	GPU
Computer 1	AMD Ryzen7 2700x	16 GB	Nvidia RTX 2070
Computer 2	Intel Core i5 8250U	8 GB	Intel HD 620

III. Results and Analysis

After preparing the feature selection methods and creating the ML models with parameter tuning of the algorithms, the models have been implemented. The findings are presented in the following sections.

A. Best feature finding

When the feature selection methods were implemented, all the 41 features were tuned. It was found that all the models give their best result when the number of features was 20 in the two machines. So, the accuracy was measured based on those 20 features. For each feature selection method, 20 different features were chosen, which included a combination of features common to all four methods, common to only three methods, common to only two methods, and unique to each method. It has been found that the features common to all the four methods gave the best accuracy. Table III depicts the different features for each feature selection method. In this table, the Rows 1-8 show the common features that were found using by all the feature selection methods, the Rows 9-15 show the features that are common in at least three methods, and the Rows 16-24 show features common in two methods. The rest in Rows 25-30 are the unique features of the methods. From this table, we have concluded that the common features in Rows 1-8 are the most important features, which play an important role in the model accuracy.

TABLE III
FEATURES OF THE NSL-KDD DATASET.

Row	Pearson corelation	Chi-square	Recursive feature elimination	Tree based: Select from model
1	*dst_host_same_src_port_rate*	*dst_host_same_src_port_rate*	*dst_host_same_src_port_rate*	*dst_host_same_src_port_rate*
2	*dst_host_rerror_rate*	*dst_host_rerror_rate*	*dst_host_rerror_rate*	*dst_host_rerror_rate*
3	*srv_serror_rate*	*srv_serror_rate*	*srv_serror_rate*	*srv_serror_rate*
4	*diff_srv_rate*	*diff_srv_rate*	*diff_srv_rate*	*diff_srv_rate*
5	*dst_host_serror_rate*	*dst_host_serror_rate*	*dst_host_serror_rate*	*dst_host_serror_rate*
6	*count*	*count*	*count*	*count*
7	*same_srv_rate*	*same_srv_rate*	*same_srv_rate*	*same_srv_rate*
8	*dst_host_same_srv_rate*	*dst_host_same_srv_rate*	*dst_host_same_srv_rate*	*dst_host_same_srv_rate*
9	*new_protocol_type*	-	*new_protocol_type*	*new_protocol_type*
10	*serror_rate*	*serror_rate*	-	*serror_rate*
11	*dst_host_srv_serror_rate*	*dst_host_srv_serror_rate*	-	*dst_host_srv_serror_rate*
12	*dst_host_diff_srv_rate*	*dst_host_diff_srv_rate*	-	*dst_host_diff_srv_rate*
13	*dst_host_srv_count*	*dst_host_srv_count*	-	*dst_host_srv_count*
14	*difficulty_level*	-	*difficulty_level*	*difficulty_level*
15	-	*dst_host_srv_diff_host_rate*	*dst_host_srv_diff_host_rate*	*dst_host_srv_diff_host_rate*
16	*dst_host_srv_rerror_rate*	*dst_host_srv_rerror_rate*	-	-
17	*new_service*	-	-	*new_service*
18	*srv_rerror_rate*	*srv_rerror_rate*	-	-
19	*rerror_rate*	*rerror_rate*	-	-
20	*new_flag*	-	-	*new_flag*
21	*logged_in*	*logged_in*	-	-
22	-	*root_shell*	*root_shell*	-
23	-	*is_guest_login*	*is_guest_login*	-
24	-	-	*srv_count*	*srv_count*
25	-	*srv_diff_host_rate*	*dst_host_count*	*src_bytes*
26	-	-	*num_failed_logins*	*dst_bytes*
27	-	-	*duration*	-
28	-	-	*land*	-
29	-	-	*wrong_fragment*	-
30	-	-	*hot*	-

B. Accuracy measurement

The accuracy of the models was the same for both of the machines in Table II. Fig. 2 shows the difference in accuracy of the four feature selection methods based on the three ML methods. The random forest algorithm outperforms the other two algorithms in terms of accuracy, and the logistic regression algorithm shows the least accuracy. It is noteworthy that for the KNN algorithm, $n = 3$ yields the highest accuracy.

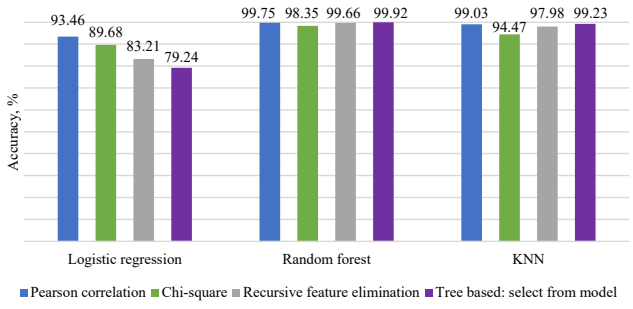

Fig. 2. Accuracy in %, for three ML algorithms with four feature selection methods

C. Execution time of algorithms with feature selection methods

The execution time of the models differed in the two computers in Table II. The time taken by the high configuration machine for every model is nearly half time of that of the low configuration machine. The execution time of the three ML algorithms with the four feature selection methods is depicted in graphical forms in Figs. 3, 4, 5, and 6.

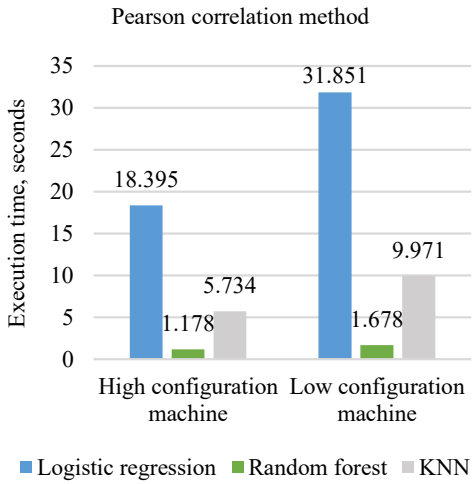

Fig. 3. Execution time with Pearson correlation feature selection.

From the graphs, it can be observed that for the Pearson correlation method, the random forest algorithm takes the least time, followed by KNN and logistic regression. For the Chi-

square, recursive feature elimination, and the tree-based: select from model methods, the ranking of the three algorithms is the same, i.e., random forest outperforms the other two algorithms.

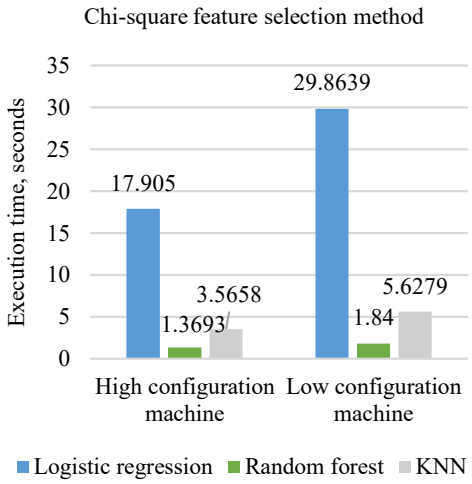

Fig. 4. Execution time with Chi square feature selection.

However, for the random forest algorithm, the execution time is faster in Pearson correlation compared to Chi-square, although the reverse is true for the other two models.

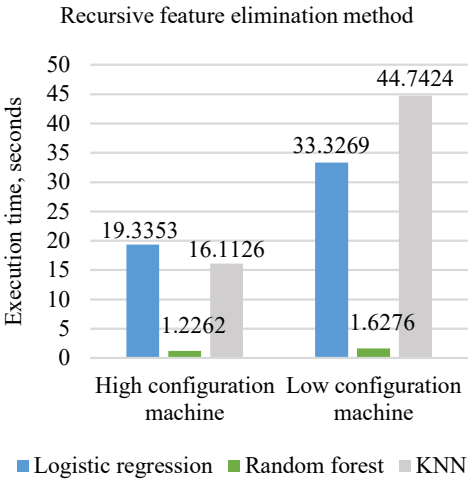

Fig. 5. Execution time with recursive feature elimination method.

Comparing the recursive feature elimination and treebased: select from model method, it is found that the KNN algorithm is faster in the latter than the former, although the other two algorithms are faster in the former. In fact, both logistic regression and KNN algorithm work fastest with the Chi-square method. The random forest algorithm works fastest in the recursive feature elimination method for the low configuration machine, but in the Pearson correlation method for the high configuration machine.

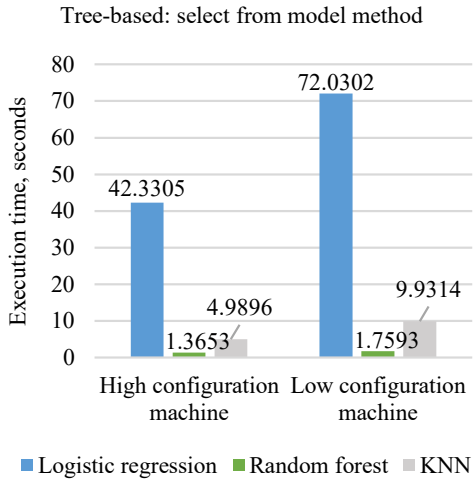

Fig. 6. Execution time with tree-based: select from model feature selection.

IV. Outcome of the Study

The main finding from this study is that no matter the method or algorithm used, the timing of the results will depend on the machine settings. The outcome of this work can be delineated as follows:

- The best accuracy is from the random forest algorithm for every feature selection method which was more than 98%.
- For KNN algorithm, we have got 94% accuracy using the Chi-square method and 97% accuracy from the other three feature selection methods.
- Logistic regression has more than 90% accuracy without using the feature selection method but gave less than 90% accuracy when features selection was used.
- For the Pearson correlation method, the accuracy was more than 93%.
- For execution time the high configured machine had the result executed in half the time required by the comparatively low configured machine.
- The random forest algorithm is the best and fastest algorithm among the three chosen algorithms with the best accuracy rate.

V. Future Works

There are other datasets other than the NSL-KDD dataset that contain other information of connection instances such as UNSW-NB15, AWID, etc. In the future, models can be built from information containing traffic information from all these datasets for better performance. In such cases, the number of total features may change. As a result, other feature selection methods and ML algorithms can be applied for these hybrid models. Real-time detection and collection of datasets can also be implemented. Moreover, IPS can also be built from those hybrid models, and performances of those systems can be judged. Deep learning methods such as artificial neural net-

works can be implemented to create more intelligent systems from these, so that higher accuracy can be achieved.

VI. Conclusion

The security of network traffic data is a growing concern, for which many researches are ongoing to detect and prevent attacks on the security system. In this paper, four feature selection methods and three machine learning algorithms have been applied to detect intrusion in the system. Pearson correlation, Chi-square, recursive feature elimination, tree-based selection from the model are the four feature selection models used; and logistic regression, random forest, K-nearest neighbor are the three ML algorithms used. The goal of this work was to compare the execution time and accuracy of the methods that are used. The NSL-KDD dataset has been used for the purpose of the study. Feature selection was used to tune 41 features, and 20 features got the best result among them. The most common and important features collected after the separation have been enlisted. Two different machines with different configurations are used to test the accuracy and execution times. Random forest had a faster execution followed by the K-nearest neighbor and logistic regression algorithms. For Chi-square, the execution time was more than the Pearson correlation method for all the algorithms. As an extension of this study, the reason of the different execution times for different algorithms and methods can be investigated.

References

[1] Hung-Jen Liao et al. "Intrusion detection system: A comprehensive review". In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 16–24.

[2] Mohammad Almseidin et al. "Evaluation of machine learning algorithms for intrusion detection system". In: *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*. IEEE. 2017, pp. 000277–000282.

[3] Harvinder Pal Singh Sasan and Meenakshi Sharma. "Intrusion detection using feature selection and machine learning algorithm with misuse detection". In: *International Journal of Computer Science and Information Technology* 8.1 (2016), pp. 17–25.

[4] Ujwala Ravale, Nilesh Marathe, and Puja Padiya. "Feature selection based hybrid anomaly intrusion detection system using K means and RBF kernel function". In: *Procedia Computer Science* 45 (2015), pp. 428–435.

[5] Mahsa Bataghva Shahbaz et al. "On efficiency enhancement of the correlation-based feature selection for intrusion detection systems". In: *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE. 2016, pp. 1–7.

[6] Amir Javadpour, Sanaz Kazemi Abharian, and Guojun Wang. "Feature selection and intrusion detection in cloud environment based on machine learning algorithms". In: *2017 IEEE international symposium on parallel and distributed processing with applications and 2017 IEEE international conference on ubiquitous computing and communications (ISPA/IUCC)*. IEEE. 2017, pp. 1417–1421.

[7] Shadi Aljawarneh, Monther Aldwairi, and Muneer Bani Yassein. "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model". In: *Journal of Computational Science* 25 (2018), pp. 152–160.

[8] Nada Aboueata et al. "Supervised machine learning techniques for efficient network intrusion detection". In: *2019 28th International Conference on Computer Communication and Networks (ICCCN)*. IEEE. 2019, pp. 1–8.

[9] Kazi Abu Taher, Billal Mohammed Yasin Jisan, and Md Mahbubur Rahman. "Network intrusion detection using supervised machine learning technique with feature selection". In: *2019 International conference on robotics, electrical and signal processing techniques (ICREST)*. IEEE. 2019, pp. 643–646.

[10] Fadi Salo et al. "Clustering enabled classification using ensemble feature selection for intrusion detection". In: *2019 International Conference on Computing, Networking and Communications (ICNC)*. IEEE. 2019, pp. 276–281.

[11] L Dhanabal and SP Shantharajah. "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms". In: *International journal of advanced research in computer and communication engineering* 4.6 (2015), pp. 446–452.

[12] Mahbod Tavallaee et al. "A detailed analysis of the KDD CUP 99 data set". In: *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE. 2009, pp. 1–6.

[13] Deeman Yousif Mahmood. "Classification Trees with Logistic Regression Functions for Network Based Intrusion Detection System". In: *IOSR Journal of Computer Engineering* 19 (2017), pp. 48–52.

[14] S Revathi and A Malathi. "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection". In: *International Journal of Engineering Research & Technology (IJERT)* 2.12 (2013), pp. 1848–1853.

[15] K Anusha and E Sathiyamoorthy. "Comparative study for feature selection algorithms in intrusion detection system". In: *Automatic Control and Computer Sciences* 50.1 (2016), pp. 1–9.

[16] Gérard Biau and Erwan Scornet. "A random forest guided tour". In: *Test* 25.2 (2016), pp. 197–227.

[17] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.