Acta Scientific COMPUTER SCIENCES

Volume 5 Issue 11 November 2023

Research Article

Detecting Cybersecurity Attacks Using Machine Learning Techniques

Saif Rawashdeh*

Department of Computer Science, Jordan University of Science and Technology, Jordan

***Corresponding Author:** Saif Rawashdeh, Department of Computer Science, Jordan University of Science and Technology, Jordan.

Received: October 11, 2023

Published: October 20, 2023

Abstract

The goal of this study is to detect anomaly assaults using a variety of machine learning methods (Decision Tree, Random Forest, Gradient Boosting, XGBoost, AdaBoost, Multilayer Perceptron, and Voting) using the well-known dataset NSL-KDD. Accuracy, precision, recall, and f1-score are the four assessment measures used to evaluate the performance of these algorithms. As a result, we will run two experiments to look for different kinds of assaults on this dataset: 1) Two categories of binary classification (normal and malicious attacks). 2) Multiclass classification (malicious attacks types). These tests check if the algorithms can distinguish between the many types of harmful attacks that can be found in the NSL-KDD dataset. The outcomes demonstrated that in both studies, the XGB classifier had the greatest performance results.

Keywords: NSL-KDD Dataset; Machine Learning; Cybersecurity Attacks; Detection Attacks

Introduction

Information security incidents, such as unauthorized access, denial of service (DoS), malware attacks, zero-day attacks, data breaches, social engineering or phishing, etc., have risen exponentially over the past ten years due to the rapidly growing importance of information technology. In 2010, there were fewer than 50 million unique malware executables that the security industry had a record of. This estimated population doubled to over 100 million in 2012. In 2019, the security sector found over 900 million malicious executables, and this number is growing, according to AV-TEST statistics [1]. Businesses and individuals may suffer large financial losses as a result of cybercrime and network assaults. For instance, studies show that a typical data breach costs USD 8.19 million globally and USD 3.9 million in the United States, and that cybercrime costs the global economy USD 400 billion annually [1]. The number of records breached is predicted to roughly treble over the next five years, according to the security community's projections. So, firms must develop and implement a thorough cybersecurity plan in order to reduce future losses. The nation's security is dependent on governments, people with access to data, applications, and instruments that demand high security clearance, according to the most current socioeconomic research, which demonstrate that this is true [1]. Also, it depends on companies granting access to their staff members, who are capable of and knowledgeable about recognizing such cyber-threats immediately and effectively. So, the main issue that requires immediate attention is the need to intelligently identify various cyber events, whether known or unknown, and appropriately protect key systems from such cyber-attacks.

Cybersecurity is the field of technology and practices that guards against unauthorized access, attacks, and damage to computers, networks, programs, and data [2]. Cybersecurity can be broken down into multiple categories and applies to a variety of contexts, including business and mobile computing. These are I network security, which focuses on preventing cyber-attackers or intruders from gaining access to a computer network; (ii) application security, which considers preventing risks or cyber-threats from affecting devices and software; (iii) information security, which primarily considers the security and privacy of relevant data; and (iv) operational security, which refers to the protocols for handling and protecting data assets. In network and computer security systems, traditional cybersecurity solutions include a firewall, antivirus software, or an intrusion detection system [2,3].

Techniques for artificial intelligence are regarded as some of the most sophisticated and useful in recent years. As a result, these strategies are important in numerous sectors, including information and cyber security [5]. Artificial intelligence is the ability for robots, electronics, software, applications, and gaming consoles to be aware of, recall, and utilize data in a manner that is comparable to how the human brain functions and makes decisions [4]. These methods gather data from experiments and then put them to use. In other words, artificially intelligent machines have electronic brains that can process information and carry out necessary tasks. Due to the widespread usage of Internet networks and their accessibility, especially with the introduction of 5G technology, the word "cybersecurity" has only lately gained popularity [5].

Computers and other electronic equipment are vulnerable to theft and illegal access by people who want to perpetrate a variety of cybercrimes. In this way, businesses aim to develop artificial intelligence-based methods for anticipating cybercrime activities, assaults, and computer intrusions. These methods can more effectively determine if people accessing the network are permitted to access the information therein than can experts. Due to their tremendous capacity for learning, remembering, and completing tasks rapidly, experts also benefit greatly from these strategies in terms of time and effort savings. Repetitive patterns can also be preserved using artificial intelligence techniques [4,5]. This capability in cybersecurity can record the patterns and actions of each user connecting to a network. In other words, using artificial intelligence approaches, it is possible to forecast the presence of harmful software penetration or any anomalous activity by studying user habits and practices [4,5].

In order to determine whether each sample is normal or anomalous and then to differentiate between the anomalous attacks in this dataset, many machine learning methods are applied to a well-known network intrusion dataset in this research. The rest of this essay is structured as follows: Some earlier works that are relevant to this study are presented in Section 2. The research approach is described in Section 3 of this publication. The findings are illustrated in Section 4. The paper's conclusion is provided in Section 5, which also makes some recommendations for additional research.

Related Work

Table 1 shows several authors applications of machine learning algorithms in the cybersecurity field to detect different types of attacks based on real-time datasets or existing datasets from several resources.

Amaizu., *et al*. [6] employed a deep learning model for network intrusion detection called Deep Neural Network (DNN). They used the NSL-KDD, UNSW-NB15, and CSECIC-IDS2018 datasets. The model's performance was then assessed using four assessment metrics: accuracy, precision, recall, and F1-Score. The following results demonstrate how well the DNN model performed in each dataset: 97.89% in NSL-KDD, 89.99% in UNSW-NB15, and 76.47% in CSECIC-IDS2018. Anwer., *et al*. [7] used four machine learning techniques to detect malicious network traffic based on a well-known dataset. They used a popular cybersecurity dataset named NSL-KDD that has 148,517 samples divided into training (125,973) and testing (22,544) datasets. This dataset contains five classes divided into two categories (normal and non-normal attacks), and each of the non-normal attacks has several types. R2L (Xsnoop, Guess_Password, Named, Ftp_write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock, Snmpguess, Httptunnel, Sendmail), DoS (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), U2R (Xterm, Buffer_overflow, Sqlattack, Rootkit, Perl, Loadmodule, Ps), and Probe (Portsweep, Satan, Nmap, Mscan, and Saint) are non-normal attacks. Support Vector Machine (SVM), Gradient Boosted Decision Trees (GBDT), and Random Forest are the machine learning algorithms (RF). They evaluated these algorithms using four metrics: accuracy, specificity, training time, and prediction time. In comparison to the other algorithms, RF's accuracy of 85.34% provided the best performance.

Su., *et al*. [8] showed the BAT model, which is a deep learning method for finding network threats that are trying to get in.They used the NSL-KDD dataset, which is well-known in the field of network attacks. This dataset contains 148,517 samples divided into training (125,973) and testing (22,544) datasets. This dataset contains five classes divided into two categories (normal and non-normal attacks), and each of the non-normal attacks has several types. R2L (Xsnoop, Guess_Password, Named, Ftp_write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock, Snmpguess, Httptunnel, Sendmail), DoS (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), U2R (Xterm, Buffer_overflow, Sqlattack, Rootkit, Perl, Loadmodule, Ps), and Probe (Portsweep, Satan, Nmap, Mscan, and Saint) are non-normal attacks. In the intrusion detection procedure, this model gave an accuracy value of 84.25%. Xu., *et al*. [9] proposed a new 5-layer autoencoder (AE)-based model that is more suited for detecting network anomalies. They used the NSL KDD dataset, which is well-known in the field of network attacks. This dataset contains 148,517 samples divided into training (125,973) and testing (22,544) datasets. This dataset contains five classes divided into two categories (normal and non-normal attacks), and each of the non-normal attacks has several types. R2L (Xsnoop, Guess_Password, Named, Ftp_write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock, Snmpguess, Httptunnel, Sendmail), DoS (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), U2R (Xterm, Buffer_overflow, Sqlattack, Rootkit, Perl, Loadmodule, Ps), and Probe (Portsweep, Satan, Nmap, Ipsweep, Mscan, Saint) are non-normal attacks. They have shown that this model gave 90.61% as an accuracy value in the intrusion detection process.

Kavitha., *et al*. [10] came up with a new algorithm called One Class Support Vector Machine (OCSVM) to find network attacks that break in. They used the NSL-KDD dataset, which is well-known in the field of network attacks and contains 148,517 samples divided into training (125,973) and testing (22,544) datasets. This dataset contains five classes divided into two categories (normal and non-normal attacks), and each of the non-normal attacks has several types. R2L (Xsnoop, Guess_Password, Named, Ftp_write, Phf, Multihop, Imap, Warezmaster, Warezclient, Snmpgetattack, Spy, Xlock, Snmpguess, Httptunnel, Sendmail), DoS (Smurf, Back, Land, Processtable, Neptune, Pod, Apache2, Udpstorm, Worm, Teardrop), U2R (Xterm, Buffer_overflow, Sqlattack, Rootkit, Perl, Loadmodule, Ps), and Probe (Portsweep, Satan, Nmap, Ipsweep, Mscan, Saint) are non-normal attacks. They have shown that this model gave an accuracy value of 81.29% in the intrusion detection procedure.

Table 1: Previous Related Work for Cybersecurity Attacks Detection.

Ref	Year	Attack	Cybersecurity Dataset	Algorithm	Result
[6]	2020	* NSL-KDD: • R2L • DoS • U2R • Probe * UNSW-NB15: • Analysis • Reconnaissance • DoS • Exploits • Fuzzers • Generic • Normal • Worms • Backdoor • Shellcode * CSECIC-IDS2018 • Infiltration • Web Attacks • DoS • DDoS • Botnet • Brute-force • Normal	NSL-KDD, UNSW-NB15, and CSECIC-IDS2018	DNN	Accuracy = 97.89%, 89.99% and 76.47 in three datasets, respectively
[7]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	SVM, GBDT, and RF	RF Accuracy = 85.34%
[8]	2020	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	BAT model	Accuracy = 84.25%
[9]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	5-layer autoencoder (AE)-based model	Accuracy = 90.61%
[10]	2021	• R2L • DoS • U2R • Probe • Normal	NSL-KDD	One Class SVM	Accuracy = 81.29%

Methodology

The proposed methodology utilized in this study to identify different types of assaults in the NSL-KDD cybersecurity dataset is shown in Figure 1. The suggested methodology is demonstrated in the sections that follow.

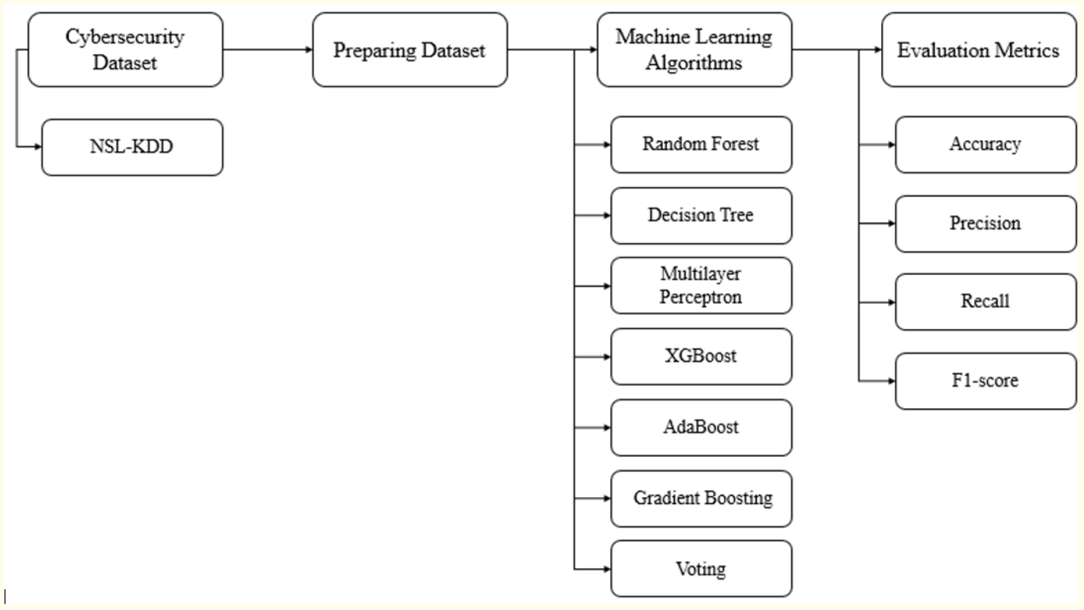

Figure 1: Flow-chart of Proposed Methodology.

Dataset description

There is a well-known dataset named the NSL-KDD dataset, which contains several cybersecurity threats and was employed in our experiment. The names of the dataset, the number of instances, the number of characteristics, and the types of cybersecurity threats are displayed in Table 2.

Table 2: Information of Cybersecurity Datasets.

Dataset Name	No. of Instances	No. of Features	Cybersecurity Attacks
NSL-KDD	148,517	42	R2L, DoS, U2R, Probe, Normal

The NSL-KDD is a popular cybersecurity dataset with 42 features and 148,517 instances that are split into two groups: normal attack (78,588 samples) and malicious attack (69,929 samples) [11]. Table 3 shows the features in this dataset. The malicious attack includes four attack classes (DoS, Probe, R2L, and U2R), and within each class are many types of attacks, as shown in Table 4. Also, Figure 2 presents the counts for each attack class.

Preparing dataset

We use a well-known encoding technique to turn each dataset's non-numerical properties into numerical features in order to apply various machine learning algorithms to them [12]. The "Label Encoder" technique converts non-numerical data into machine-readable forms by substituting a distinct number beginning at 0 for each value [12]. The protocol type, service, and flag are categorical

Table 3: NSL-KDD Features.

No.	Feature	No.	Feature	No.	Feature
1	duration	15	su_attempted	29	same_srv_rate
2	protocol_type	16	num_root	30	diff_srv_rate
3	service	17	num_file_creations	31	srv_diff_host_rate
4	flag	18	num_shells	32	dst_host_count
5	src_bytes	19	num_access_files	33	dst_host_srv_count
6	dst_bytes	20	num_outbound_cmds	34	dst_host_same_srv_rate
7	land	21	is_host_login	35	dst_host_diff_srv_rate
8	wrong_fragment	22	is_guest_login	36	dst_host_same_src_port_rate
9	urgent	23	count	37	dst_host_srv_diff_host_rate
10	hot	24	srv_count	38	dst_host_serror_rate
11	num_failed_logins	25	serror_rate	39	dst_host_srv_serror_rate
12	logged_in	26	srv_serror_rate	40	dst_host_rerror_rate
13	num_compromised	27	rerror_rate	41	dst_host_srv_rerror_rate
14	root_shell	28	srv_rerror_rate	42	labels

Table 4: Attack Classes in NSL-KDD Dataset.

Class	Attack Type
Normal	-
DoS	Back, Land, Neptune, Pod, Smurf, Teardrop, Apache2, Processtable
Probe	Satan, Ipsweep, Nmap, Portsweep, Mscan, Saint
R2L	Guess_Password, Ftp_write, Imap, Phf, Multihop, Warezmaster, Warezclient, Spy, Xlock, Xsnoop, Snmpguess, Snmpgetattack, Httptunnel, Sendmail, Named
U2R	Buffer_overflow, Loadmodule, Rootkit, Perl, Xterm, Ps

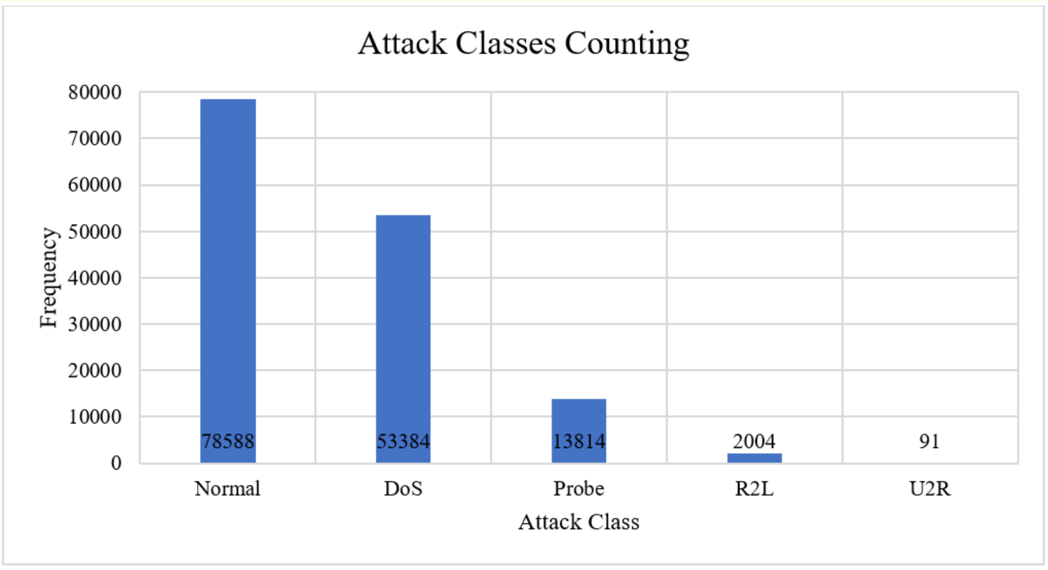

Figure 2: NSL-KDD Attack Classes.

features in the dataset NSL-KDD, and we must convert them into numerical features.

Machine learning algorithms

The dataset is prepared and then fit to seven machine learning algorithms so that the above cyberattacks can be found. We used the hold-out strategy to split the dataset into training and testing datasets with a test size of 0.1. As a result, 0.9 of the total dataset is a training dataset, and 0.1 of the total dataset is a testing dataset. Using the training dataset, many machine learning models are built, and the testing dataset is used to test how well they work.

Random forest algorithm (RF)

A vast number of decision trees are trained in the ensemble supervised learning technique known as random forest, which is used for regression and classification applications. The class selected by the majority of trees serves as the forecast outcome for classification problems. The mean or average forecast of each tree is provided for regression tasks [13]. Random decision forests can address decision trees' propensity to overfit their training set. Each cybersecurity dataset that includes several attacks is therefore classified using the Random Forest classifier [14,15]. Because the label in our experiment is discrete, we employed the random forest with classification type. The settings of the RF that were used were as follows: n estimators (number of decision trees) = 100, max features = sqrt, max depth = None, random state = 42.

Decision tree algorithm (DT)

A supervised machine learning system called Decision Tree makes decisions based on a set of rules, much like people do [16]. One of the approaches to predictive modeling utilized in three disciplines—data mining, statistics, and machine learning—is decision tree learning, also known as induction of decision trees [17]. It moves from making observations about a sample (represented by the branches) to drawing conclusions about the sample's target value (represented by the leaves, which are attack types) using a decision tree [18].

Classification trees are tree models with a discrete target variable; in these tree structures, the leaves stand for the various sorts of assaults and the branches for the characteristics of the dataset that help predict the class labels [19]. Decision trees with a continuous objective variable (usually real numbers) are known as regression trees [20]. Due to their clarity and simplicity, decision trees are one of the well-known machine learning algorithms. Because the label in our experiment is discrete, we utilized the Decision Tree with classification type. The DT settings we used were: criterion = gini and random state = 42.

Multilayer perceptron algorithm (MLP)

A particular kind of feedforward artificial neural network is called a multilayer perceptron (MLP). The term "MLP" can apply to any feedforward ANN in certain circumstances, while in others it refers to networks made up of several layers of perceptrons (with threshold activation). Multilayer perceptrons with a single hidden layer in particular are referred to as "vanilla" neural networks [21].

An MLP has an input layer, a hidden layer, and an output layer as its minimum number of node layers. Except for the input nodes, each node is a neuron with a nonlinear activation function [22]. During training, backpropagation is used by MLP as a supervised learning method. MLP differs from a linear perceptron due to its numerous layers and non-linear activation. It can distinguish between data that cannot be linearly separated [23].

Because the label in our experiment is discrete, we utilized the MLP with classification type, with the following MLP parameters: activation = relu and random state = 42.

eXtreme gradient boosting algorithm (XGBoost)

Extreme Gradient Boosting, often known as XGBoost, is a technique used in regression and classification tasks. The gradient boosting technique has undergone careful parallelization and optimization [24]. Parallelizing the entire boosting process drastically reduces the training time. Instead of building the best model feasible on the data, we train hundreds of models on various subsets of the training dataset [24] and then vote on the model that performs the best (as in traditional approaches). XGBoost frequently performs better than conventional gradient boosting methods [25]. You have access to a large number of inner parameters in the Python implementation, which you can change for increased precision and accuracy [25].

This algorithm's primary function is to transform decision trees, which are weak learners, into strong learners, who then create the final prediction label (the average of each prediction by week classifier) [26]. Several important characteristics of the XGBoost include: 1) Parallelization: The model is built to operate concurrently across several CPU cores. 2) Regularization: To avoid overfitting, XGBoost provides a range of regularization penalties. Regularizations with penalties lead to successful training, which enables the model to successfully generalize. Non-linearity: XGBoost can identify non-linear data patterns and learn from them. 4) Cross-validation is integrated and available right now. 5) Scalability: With the help of distributed servers and clusters like Hadoop and Spark, XGBoost can run distributed, enabling you to manage enormous volumes of data. C++, Java, Python, and Julia are just a few of the many programming languages that are supported [24-26].

In our experiment, we used this algorithm with classification type because the label is discrete and the parameters of XGBoost that used are as the following: colsample_bylevel = 1, learning_rate = 0.1, gamma = 0, n_estimators = 100, and random_state = 42.

AdaBoost algorithm

An example of a statistical classification meta-algorithm is AdaBoost, which stands for Adaptive Boosting. Performance can be improved by combining it with a variety of other learning methods. The output of additional learning algorithms, or "weak learners," is combined to create the final output of the boosted classifier. AdaBoost is adaptive in that it modifies incoming weak learners in favor of situations where previous classifiers misclassified them. In some circumstances, it might be less prone to the overfitting issue than other learning methods. Individual learners might perform poorly, but the final model will converge to a powerful learner as long as their performance is slightly better than random guessing. It has been shown that AdaBoost may be used to integrate strong base learners, such as deep decision trees, in addition to weak base learners (such as decision stumps), producing a more accurate model [27-29].

Every learning algorithm has a variety of parameters and configurations to adjust before it performs at its best on a dataset, and the majority of them are better suited to particular issue types than others. With decision trees acting as weak learners, AdaBoost is commonly referred to as the best out-of-the-box classifier [27-29]. Data gained from each stage of the AdaBoost method on the relative "hardness" of each training sample is input into the tree-growing process when combined with decision tree learning, causing later trees to concentrate on more challenging-to-classify samples.

Due to the discrete nature of the label in our experiment, we employed the AdaBoost with classification type. The settings of GB that were used were as follows: algorithm = SAMME.R, learning rate = 1.0, n estimators = 50, and random state = 42.

Gradient boosting algorithm (GB)

An technique to machine learning called gradient boosting has applications in regression and classification, among other things. It provides a prediction model in the form of a collection of decision trees that are weak prediction models. A method called gradient boosting turns several weak learners (decision trees) into one powerful learner. Individual decision trees are ineffective learners in this situation. Each tree in the succession is connected to the one before it and works to fix the flaw of the one before it. Boosting algorithms are frequently time-consuming to train yet very accurate because of this sequential link. Models that learn slowly perform better in statistical learning. The weak learners are fitted in a way that, as the model gets better, each new student fits into the residuals of the stage before them. Each phase's results are combined in the final model to create a powerful learner. To find residuals, a loss function is employed. For instance, mean squared error (MSE) can be used in regression work, and logarithmic loss (log loss) can be used in classification work. It's important to note that nothing changes when a new tree is included in the model. The additional decision tree matches the residuals of the present model [30-32].

Because the label in our experiment is discrete, we utilized the GB with classification type. The GB parameters we used were as follows: subsample=1.0, learning rate=0.1, criterion=friedman mse, n estimators=100, and random state=42.

Voting algorithm

An ensemble machine learning model known as a "voting classifier" learns from a variety of other models and predicts an output (class) based on the output's likelihood of being the desired class [33]. It merely predicts the output class with the most votes by adding the results of each classifier given into the voting classifier [34]. Instead of creating separate, specialized models and evaluating their performance, we suggest a single model that trains on many models and predicts output based on the cumulative majority of votes for each output class. Voting Classifier supports two different voting methods [35,36]. 1) Hard voting: The projected output class that receives the greatest number of votes, or the one that is most likely to be predicted by each classifier, is the one that has the highest likelihood of being predicted by each classifier. 2) Soft voting: The forecast is based on an average of the likelihood given to that class and is based on the output class.

Because the label in our experiment is discrete, we employed voting with classification type. The specifications of voting that were used were as follows: estimators = DT, RF, and XGB; and voting type = hard.

Results and Discussion

This section presents the experimental results for each cybersecurity dataset in each machine learning algorithm based on four evaluation metrics.

Evaluation metrics

There are several assessment measures to examine the machine learning algorithms that were utilized, including accuracy, precision, recall, and f1-score [37]. These metrics' formulas are as follows: TP = True Positives, TN = True Negative, FP = False Positives, and FN = False Negative:

Accuracy: is the most intuitive performance metric is the ratio of properly predicted samples to total samples, which is simply a ratio of correctly predicted samples to total samples..

$$Accuracy = \frac{\text{TP+TN}}{\text{TP + TN + FP + FN}} \quad \text{------------- (1)}$$

Precision: is the ratio of correctly predicted positive tweets to the total predicted positive samples.

$$Precision = \frac{TP}{TP+FP} \quad \text{-------------- (2)}$$

Recall: is the proportion of accurately anticipated positive samples to the total number of positive samples predicted.

$$Recall = \frac{TP}{TP+FN} \quad \text{---------------- (3)}$$

F1-score: is the weighted average of Precision and Recall.

$$F1\text{-}score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad \text{------------ (4)}$$

We used two tests on this dataset to find numerous different sorts of attacks: 1) Binary gradation (normal and malicious attack classes). Multiclass classification, second (attacks types in each malicious attack class). These tests are performed to determine whether each algorithm can distinguish between the different harmful attack types in the NSL-KDD dataset.

Results

Binary classification experiment

Machine learning methods are used in this experiment to determine if the sample in this dataset represents a legitimate attack or a malicious one. The performance results for the machine learning methods utilized are based on four metrics: precision, accuracy, f1-score, and recall, as shown in Table 5 and Figure 3. When compared to the other classifiers, the XGB classifier outperforms the higher performance outcomes.

Multiclass classification experiment

Machine learning algorithms are used in this experiment to identify the R2L, DoS, U2R, and Probe types of each malicious attack class in the dataset. The performance results for the machine learning methods utilized are based on four metrics: precision, accuracy, f1-score, and recall, as shown in Table 6 and Figure 4. As compared to the others, the XGB classifier outperforms the higher performance results in the detection attack procedure.

Conclusion and Future Work

In this study, we developed a number of machine learning techniques to identify anomaly attacks, including Decision Tree, Random Forest, Gradient Boosting, XGBoost, AdaBoost, Multilayer Perceptron, and Voting. We used the NSL-KDD dataset, a well-known

Models	Accuracy	Precision	Recall	F1-score
DT	0.996027	0.995413	0.996127	0.99577
RF	0.997239	0.997559	0.996557	0.997058
GB	0.985793	0.990424	0.9792	0.98478
XGB	0.998249	0.997849	0.998422	0.998136
AdaBoost	0.970576	0.973341	0.963707	0.9685
MLP	0.992324	0.994804	0.988811	0.991799
Voting	0.997576	0.997561	0.997274	0.997418

Table 5: Performance Results of Binary Classification.

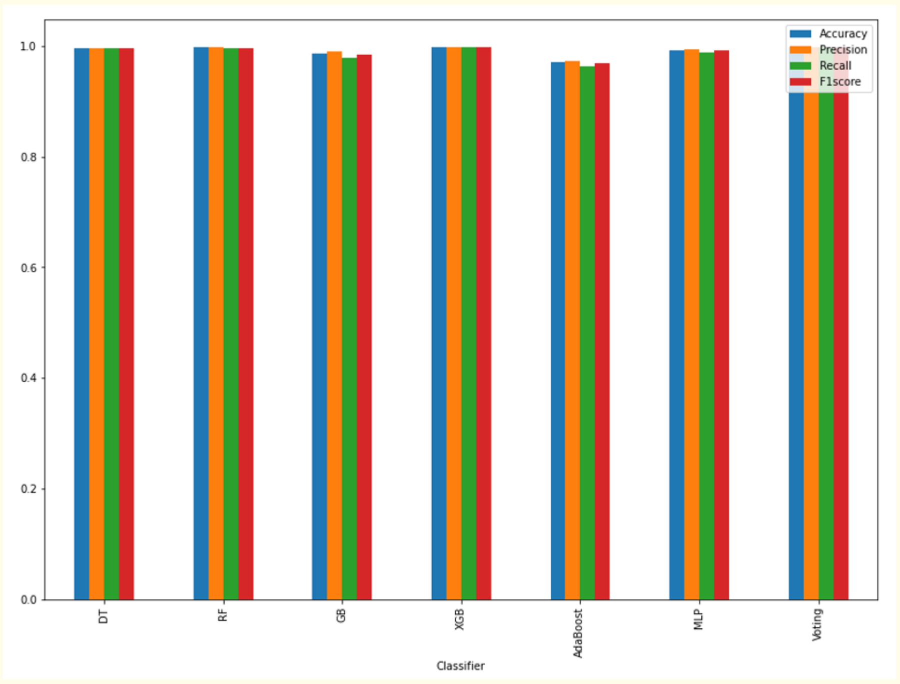

Figure 3: Performance Results of Binary Classification.

Models	Accuracy	Precision	Recall	F1-score
DT	0.993136	0.713237	0.736366	0.72117
RF	0.995424	0.80128	0.792541	0.79146
GB	0.961247	0.63763	0.647018	0.616836
XGB	0.997855	0.856876	0.845016	0.847671
AdaBoost	0.730588	0.04961	0.106949	0.060603
MLP	0.994709	0.788667	0.761176	0.768417
Voting	0.996282	0.796958	0.81584	0.804374

Table 6: Performance Results of Multiclass Classification.

dataset that we downloaded from the Kaggle website, to conduct our tests. Accuracy, precision, recall, and f1-score are the four standard assessment measures used to assess the effectiveness of these algorithms. We conducted two experiments on this dataset to identify various types of attacks. 1) Two categories of binary classification (normal and malicious attacks). 2) Classification of multiple classes into different categories (malicious attack types). The purpose of these tests is to determine how well these algorithms can distinguish between the various types of damaging assaults in this dataset. The aforementioned findings show that in both cases, the XGB algorithm outperformed the best algorithm. We intend to use further machine learning and deep learning methods on this dataset in subsequent research. In order to assess how well the aforementioned machine learning methods, perform, we will also apply them to the other dataset.

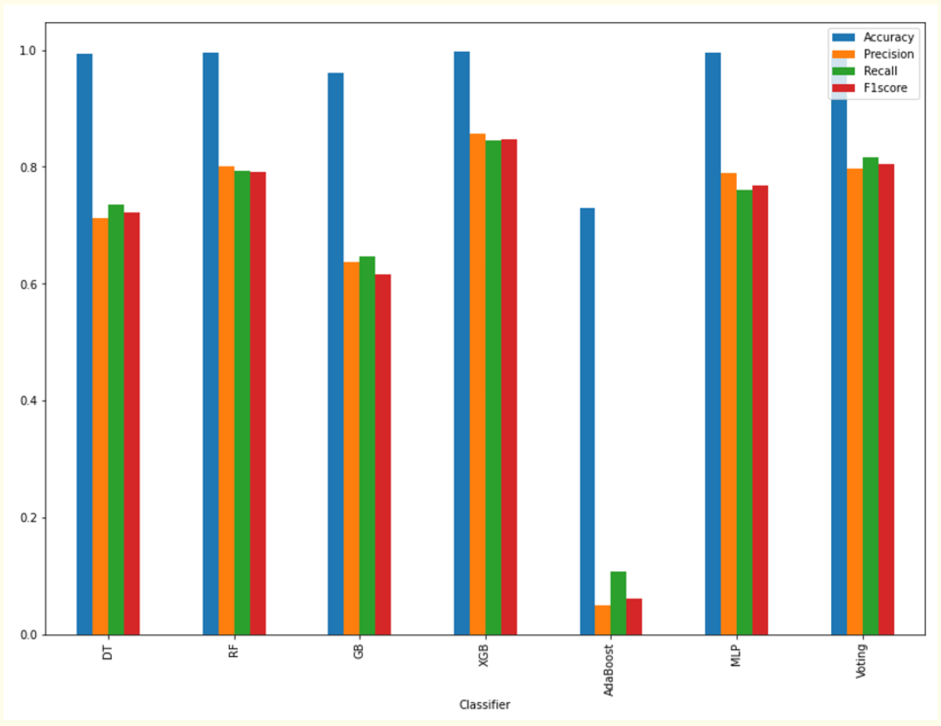

Figure 4: Performance Results of Multiclass Classification.

Bibliography

1. Ahsan Mostofa., *et al*. "Cybersecurity threats and their mitigation approaches using Machine Learning—A Review". *Journal of Cybersecurity and Privacy* 2.3 (2022): 527-555.

2. Corallo A., *et al*. "Cybersecurity awareness in the context of the Industrial Internet of Things: A systematic literature review". *Computers in Industry* 137 (2022): 103614.

3. AlDaajeh S., *et al*. "The role of national cybersecurity strategies on the improvement of cybersecurity education". *Computers and Security* (2022): 102754.

4. Mijwil M., *et al*. "The Significance of Machine Learning and Deep Learning Techniques in Cybersecurity: A Comprehensive Review". *Iraqi Journal for Computer Science and Mathematics* 4.1 (2023): 87-101.

5. Annamalai C. "Application of Factorial and Binomial identities in Information, Cybersecurity and Machine Learning". *International Journal of Advanced Networking and Applications* 14.1 (2022): 5258-5260.

6. Amaizu G C., *et al*. "Investigating Network Intrusion Detection Datasets Using Machine Learning". In 2020 International Conference on Information and Communication Technology Convergence (ICTC) (2020): 1325-1328.

7. Anwer M., *et al*. "Attack Detection in IoT using Machine Learning". *Engineering, Technology and Applied Science Research* 11.3 (2021): 7273-7278.

8. Su T., *et al*. "BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset". *IEEE Access* 8 (2020): 29575-29585.

9. Xu W., *et al*. "Improving Performance of Autoencoder-Based Network Anomaly Detection on NSL-KDD Dataset". *IEEE Access* 9 (2021): 140136-140146.

10. Kavitha S and Uma Maheswari N. "Network Anomaly Detection for NSL-KDD Dataset Using Deep Learning". *Information Technology in Industry* 9.2 (2021): 821-827.

11. Dhanabal L and Shantharajah S P. "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms". *International Journal of Advanced Research in Computer and Communication Engineering* 4.6 (2015): 446-452.

12. Hancock J T and Khoshgoftaar TM. "Survey on categorical data for neural networks". *Journal of Big Data* 7.1 (2020): 1-41.

13. Pal M. "Random forest classifier for remote sensing classification". *International Journal of Remote Sensing* 26.1 (2005): 217-222.

14. Farnaaz N and Jabbar M A. "Random forest modeling for network intrusion detection system". *Procedia Computer Science* 89 (2016): 213-217.

15. Idhammad M., *et al*. "Detection system of HTTP DDoS attacks in a cloud environment based on information theoretic entropy and random forest". *Security and Communication Networks* (2018).

16. Kingsford C and Salzberg S L. "What are decision trees?". *Nature Biotechnology* 26.9 (2008): 1011-1013.

17. Quinlan JR. "Induction of decision trees". *Machine Learning* 1.1 (11986): 81-106.

18. De Ville B. "Decision trees". Wiley Interdisciplinary Reviews: Computational Statistics 5.6 (2013): 448-455.

19. Kotsiantis S B. "Decision trees: a recent overview". *Artificial Intelligence Review* 39.4 (2013): 261-283.

20. Amor N B., *et al*. "Naive bayes vs decision trees in intrusion detection systems". In Proceedings of the 2004 ACM symposium on Applied computing (2004): 420-424.

21. Noriega L. "Multilayer perceptron tutorial". School of Computing. Staffordshire University (2005).

22. Tang J., *et al*. "Extreme learning machine for multilayer perceptron". *IEEE Transactions on Neural Networks and Learning Systems* 27.4 (2015): 809-821.

23. Ramchoun H., *et al*. "Multilayer perceptron: Architecture optimization and training" (2016).

24. Mitchell R and Frank E. "Accelerating the XGBoost algorithm using GPU computing". *PeerJ Computer Science* 3 (2017): e127.

25. Pan B. "Application of XGBoost algorithm in hourly PM2. 5 concentration prediction". In IOP conference series: earth and environmental science 113.1 (2018): 012127.

26. Dong W., *et al*. "XGBoost algorithm-based prediction of concrete electrical resistivity for structural health monitoring". *Automation in Construction* 114 (2020): 103155.

27. Hu W and Hu W. "Network-based intrusion detection using Adaboost algorithm". In The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05) (2005): 712-717.

28. Jabri S., *et al*. "Moving vehicle detection using Haar-like, LBP and a machine learning Adaboost algorithm". In 2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS) (2018): 121-124.

29. Yuan L and Zhang F. "Ear detection based on improved adaboost algorithm". In 2009 International Conference on Machine Learning and Cybernetics 4 (2009): 2414-2417.

30. Son J., *et al*. "Tracking-by-segmentation with online gradient boosting decision tree". In Proceedings of the IEEE international conference on computer vision (2015): 3056-3064.

31. Peter S., *et al*. "Cost efficient gradient boosting". *Advances in Neural Information Processing* Systems 30 (2017).

32. Lusa L. "Gradient boosting for high-dimensional prediction of rare events". *Computational Statistics and Data Analysis* 113 (2017): 19-37.

33. Kumar U K., *et al*. "Prediction of breast cancer using voting classifier technique". In 2017 IEEE international conference on smart technologies and management for computing, communication, controls, energy and materials (ICSTM) (2017): 108-114.

34. El-Kenawy E S M., *et al*. "Novel feature selection and voting classifier algorithms for COVID-19 classification in CT images". *IEEE Access* 8 (2020): 179317-179335.

35. Khan MA., *et al*. "Voting classifier-based intrusion detection for iot networks". In Advances on Smart and Soft Computing (2022): 313-328.

36. Mahabub A. "A robust technique of fake news detection using Ensemble Voting Classifier and comparison with other classifiers". *SN Applied Sciences* 2.4 (2020): 1-9.

37. Dalianis H. "Evaluation metrics and evaluation". In Clinical text mining (2018): 45-53.