ULUSLARARASI 3B YAZICI TEKNOLOJİLERİ
VE DİJİTAL ENDÜSTRİ DERGİSİ

INTERNATIONAL JOURNAL OF 3D PRINTING TECHNOLOGIES AND DIGITAL INDUSTRY

ISSN:2602-3350 (Online)

URL: https://dergipark.org.tr/ij3dptdi

A COMPARATIVE EVALUATION OF THE BOOSTING ALGORITHMS FOR NETWORK ATTACK CLASSIFICATION

Yazarlar (Authors): Koray Çoşkun, Gürcan Çetin*

Bu makaleye şu şekilde atıfta bulunabilirsiniz (To cite to this article): *Çoşkun K., Çetin G., "A Comparative Evaluation of The Boosting Algorithms For Network Attack Classification" Int. J. of 3D Printing Tech. Dig. Ind., 6(1): 102-112, (2022).*

DOI: 10.46519/ij3dptdi.1030539

Araştırma Makale/ Research Article

Erişim Linki: (To link to this article): https://dergipark.org.tr/en/pub/ij3dptdi/archive

A COMPARATIVE EVALUATION OF THE BOOSTING ALGORITHMS FOR NETWORK ATTACK CLASSIFICATION

Koray Çoşkun[a], Gürcan Çetin[a]*

[a]Muğla Sıtkı Koçman University, Technology Faculty, Information Systems Engineering Department, TURKEY

* *Corresponding Author: gcetin@mu.edu.tr*

(**Received:** 30.11.2021; **Revised:** 28.03.2022; **Accepted:** 25.04.2022)

ABSTRACT

The security of information resources is an extremely critical problem. The network infrastructure that enables internet access, in particular, may be targeted by attackers from a variety of national and international locations, resulting in losses for institutions that utilize it. Anomaly detection systems, sometimes called Intrusion Detection Systems (IDSs), are designed to identify abnormalities in such networks. The success of IDSs, however, is limited by the algorithms and learning capacity used in the background. Because of the complex behavior of malicious entities, it is critical to adopt effective techniques that assure high performance while being time efficient. The success rate of the boosting algorithms in identifying malicious network traffic was studied in this study. The boosting approach, one of the most used Ensemble Learning techniques, is accepted as a way to cope with this challenge. In this work, Google Colab has been used to model well-known boosting algorithms. The AdaBoost, CatBoost, GradientBoost, LightGBM, and XGBoost models have been applied to the CICID2017 dataset. The performance of the classifiers has been evaluated with accuracy, precision, recall, f1-score, kappa value, ROC curve and AUC. As a result of the investigation, it was discovered that the XGBoost algorithm produced the greatest results in terms of f1-score, with 99.89 percent, and the AUC values were extremely near to 1, with 0.9989. LightGBM and GradientBoost models, on the other hand, have been shown to be less effective in detecting attack types with little data.

Keywords: Boosting Algorithms. Ensemble Learning. Intrusion Detection Systems. Network Attacks.

1. INTRODUCTION

Cybersecurity has become one of the fastest growing industries in recent years due to the need to create new tools that detect, prevent, and respond to many types of attacks. When developing these tools, time-related risks to network traffic are analyzed. However, due to the complexity of network traffic, providing accurate and effective solutions is a significant challenge [1]. Moreover, as information about existing assaults, weaknesses, and security measures improves, attacks get more complicated. So, it's vital to use some techniques such as signature or anomaly detection systems and deep learning or machine learning based algorithms to defend critical network infrastructures. In recent years, the number of applications based on machine learning and deep learning has increased. For instance, Kanimozhi and Jacob [2] classified bot attacks with 99.97% accuracy with an artificial intelligence-based IDS. The comprehensive performance analysis of machine learning algorithms on IDSs can be found in the paper [3] released by Saranya et al.

Preliminary investigations using de-facto datasets such as DARPA, KDD-Cup'99, NSL-KDD, Kyoto2006+, CAIDA-2007, and TU-DDoS are conducted in the literature in order to build real-time IDSs. Aside from these, the CICIDS2017 dataset has gotten a lot of interest from academics in recent years since it covers additional concerns based on today's network threats [4]. In this study, CICIDS2017 was chosen as the current dataset for the use of boosting techniques in the development of a novel IDS, which is one of the study's objectives.

The literature on the research carried out to detect and classify network attacks using the

CICIDS2017 dataset can be summarized as follows. Sharafaldin et al. [5] used K-Nearest Neighbors (KNN), Random Forest, Decision Tree (ID3), Naive Bayes (NB), Adaboost, Multilayer Perceptron (MLP), and Quadratic Discriminant Analysis (QDA) methods in their study. When the results are analyzed, it is determined that the ID3 algorithm produces the best results, with an f1-score value of 0.98. In another work, Özekes and Karakoç [6] employed decision tree and random forest algorithms. When the study was simply examined in terms of the categorization of normal and abnormal network traffic, the accuracy scores for the decision tree and random forest techniques were 0.99957 and 0.99966, respectively. Tama et al. [7] discovered outliers in web traffic using the CISC-2010v2 in addition to the CICIDS2017 dataset. The algorithms Random Forest (RF), Gradient Boosting Machine (GBM), and XGBoost were chosen as methods. Then, these algorithms were compared with the proposed stacked ensemble approach. In addition to the 99.98 accuracy value, the proposed approach also addressed the data imbalance problem in the CICIDS-2017 dataset. Abdulrahman and Ibrahem [8] compared the C5.0, NB, Support Vector Machine, and RF performance findings. The RF and C5.0 classifiers outperformed the others, with average accuracy values of 86.80% and 86.45%, respectively. The major goal of Hosseini and Seilani's research [9] is to improve system accuracy while lowering training time. On the CICIDS2017 and NSL-KDD datasets, Logistic Regression, RF, Decision Tree, and KNN machine learning methods were utilized. More than 99 percent accuracy was achieved in a run time interval of 27.36 seconds as a consequence of the experiments.

When the papers are examined, it is discovered that the success rates of various machine learning approaches are compared. However, recent efforts in the field of machine learning have resulted in the creation of new approaches throughout time. And each of these approaches has benefits and drawbacks over the others. There are not enough studies in the literature to compare the performance of these new methods in classifying network attacks. One of these methods is the Ensemble Learning (EL) method. Ensemble learning, in its classic terms, is a strategy for determining unknown data by voting on the outcomes of estimations done by basic classifiers. The two most used Ensemble Learning methods are the boosting algorithm and the bagging algorithm [10]. The boosting algorithms studied in this paper are a sequential technique based on slow learning that tries to learn from failures. According to boosting algorithms, it is much easier to specify a large number of general rules than to specify a single highly accurate prediction rule. In the boosting method, high performance is achieved by giving certain parts of the training samples to the learning algorithm over and over again [11]. Different studies in the fields of bank failure [12], intrusion detection [13], wind speed forecasting [14], health [15], and so on have all confirmed this achievement.

The success rate of the boosting approach in identifying malicious network traffic was studied in this study. The AdaBoost, CatBoost, Gradient Boost, LightGBM, and XGBoost algorithms were modeled on Google Colab and used on the CICID2017 dataset in this context. After normalization, the CICIDS 2017 dataset was subjected to the Recursive Feature Elimination (RFE) technique. Following the RFE process, 25 features were deleted from the data set, reducing it to 53 features. The classifiers' performance was measured using accuracy, precision, recall, f1-score, kappa value, ROC curve, and AUC. As a consequence of the research, while the XGBoost method is the most effective model, the LightGBM model is less successful in properly identifying the attack types, which are very few in the dataset, and remains the lowest in terms of overall performance when compared to other models.

The following sections of the study are organized as follows: Ensemble Learning and boosting methods are explained in Section 2. It also explains the boosting techniques that were utilized in the research. The preprocessing process on the dataset is explained in Section 3 using the CICIDS2017 dataset as an example. Section 4 describes how to classify network intrusions using boosting techniques. This part also includes the models' assessment criteria, the study's experimental results, the findings, and a comparison of the models. Section 5 brings the research to a close.

2. EL AND BOOSTING ALGORITHMS

Ensemble Learning (EL) methods, the distribution of training datasets is altered in a

specific way to produce several training subsets. Multiple base-classifiers are then created in order to make predictions about unknown data by voting on these base-classifiers. The ensemble learning uses voting, stacking, bagging, and boosting methods to combine different classifiers [16]. Voting entails producing a forecast that is the average of several different regression models. Stacking is a data mining technique that combines many machine learning models. The base learners are trained first, followed by a meta-classifier that generates a final prediction based on the base learners' predictions [17]. On the other hand, the bagging method, given in Figure 1(a), combines multiple learners that adapt to diverse samples by altering the training set, enhancing the accuracy rate of the entire sample [18]. Bagging is highly successful for issues where a minor change in the learning set can result in a significant change in the predictions [19]. In the boosting algorithm procedure, which is depicted with its general model in Figure 1(b), random samples are created from the training dataset first. For this sample, a weak classifier is trained, and the complete training dataset is tested [20]. For each predicted value, the error is computed. If the sample is misclassified, the weight for that sample is raised, and a new sample is produced. These procedures are continued until the system achieves high accuracy. When compared to the bagging method, the boosting approach has a reduced error rate. Furthermore, in datasets where decision trees are successful for classification, the boosting technique considerably improves classification rates. Within the scope of boosting algorithms, the AdaBoost, GradientBoost, XGBoost, LightGBM, and CatBoost techniques were utilized in this work.

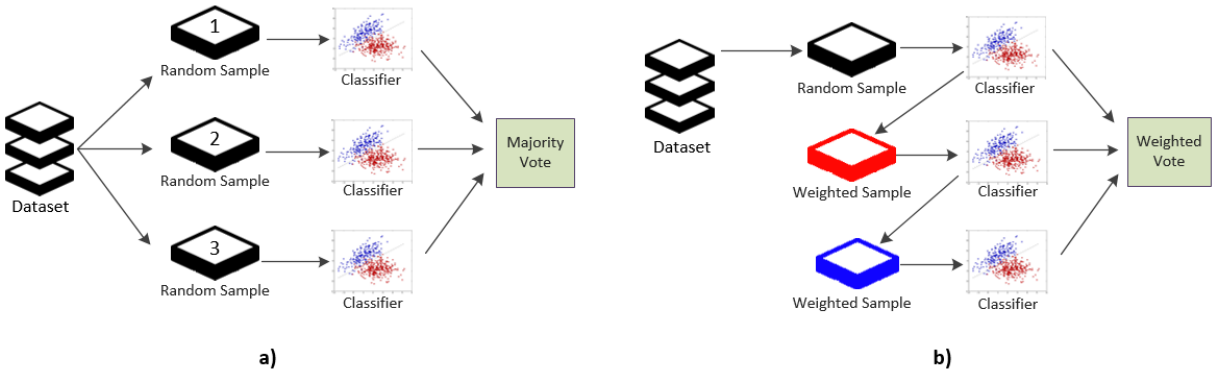


Figure 1. **a)** Bagging **b)** Boosting.

2.1. AdaBoost

The AdaBoost algorithm, developed by Freund and Schapire [21], manipulates training samples to generate multiple hypotheses. It works on the weighting principle, in which each model corrects the error of the previous model [22]. In this method, a probability distribution value is kept on the training data. It produces an m-dimensional training set iteratively by sampling with variation in accordance with this value. This is followed by using the learning algorithm in order to create a classifier. The classifier's error rate is computed using the training data [23]. And then, the weights are given using the error value and more weight is given to the data points that are misclassified. In this way, the error will be corrected in later models.

2.2. CatBoost

Categorical Boosting is an algorithm put forward by Yandex to deal with categorical data more easily. It is based on the gradient boosting algorithm. CatBoost's unique encoding capability of categorical attributes makes it advantageous in data preprocessing. It uses the ordered boosting with ordered Target Statistics technique to solve the problem of prediction shifts [24].

2.3. GradientBoost

Friedman [25] introduced Gradient Boosting in 1999 as an ensemble approach for regression and classification. It is an iterative process of building a strong ensemble classifier. Its goal is to combine weaker models in a greedy manner to get stronger estimators [26]. As the model develops, new trees are generated based on the prediction mistakes of prior trees [27].

Moreover, random subsampling of the training data speeds up and improves the accuracy of gradient boosting for execution. This also helps prevent overfitting [28]. The following tuning parameters are required in gradient boosting regression: ntrees and shrinkage rate, where ntrees is the number of trees to be generated and the shrinkage parameter is commonly referred to as the learning rate applied to each tree.

2.4. Light GBM

Researchers from Microsoft and Peking University created the LightGBM to overcome the efficiency and scalability concerns with GBDT and XGBoost when used to solve problems such as high input features and large data sizes. LightGBM deals with the amount of data and the number of variables with novel techniques such as Gradient Based One Way Sampling (GOSS) and Exclusive Feature Bundling (EFB). These two methods make the boosting algorithm more efficient and scalable [29]. These integrated characteristics allow data scanning, sampling, clustering, and classification to be conducted smoothly and correctly in a short amount of time. When memory consumption, processing time, and arithmetic speed are all taken into account, the LightGBM becomes a good solution for quicker training, appropriate efficiency, optimum memory, acceptable accuracy, parallelism, and large-scale data processing [30].

2.5. XGBoost

XGBoost is a boosting algorithm developed by Tianqi Chen and Carlos Guestrin [31]. It combines Cause Based Decision Tree (CBDT) and Gradient Boosting Machine (GBM) in a single efficient method to handle almost all data kinds fast and reliably. With such distinct properties, it can be used to effectively build forecasting models as well as analyse large datasets with a large number of features and classifications [30]. The capabilities of XGBoost can be listed as follows [32]:

- XGBoost has special regularization methods like Lasso (L1) and Ridge (L2).
- It has a depth priority approach that creates splits based on the highest depth and prunes non-gain splits.
- It can work fast with its cache awareness and out of computing feature, where it can use the cache at the maximum level.
- To find the most accurate split points, XGBoost employs the weighted quartile sketch algorithm.

3. DATASET AND PREPROCESSING

3.1. Dataset

CICIDS2017 dataset was created by the Canadian Cyber Security Institute. In contrast to prior datasets used in the literature, the victim network includes all common and essential infrastructure, such as a router, firewall, switch, and several versions of Windows, Linux, and Macintosh operating systems. The attack network consists of four machines running Kali and Windows 8.1, as well as a router and a switch. Table 1 shows the types of cyberattacks that take place every day in CICID2017 dataset.

Table 1. Daily Label of Dataset [5].

Days	Labels
Monday	BENING
Tuesday	BForce, SFTP and SSH
Wednesday	DoS and Heartbleed Attacks, slowloris, Slowhttptest, Hulk and GoldenEye
Thursday	Web and Infiltration Attacks, Web BForce, XSS and SQL Inject. Infiltration Dropbox Download and Cool disk
Friday	DDoS LOIT, BotNet ARES, PortScans (sS, ST, sF, sX, sN, sP, sV, sU, SO, sA, sW, sR, SL and B)

Figure 2 shows the types of attacks carried out in the CICID2017 dataset, as well as the ratio of these attacks in the overall dataset. In the dataset, normal packet data (BENIGN) accounts for 80% of the total data (2,273,097 records).

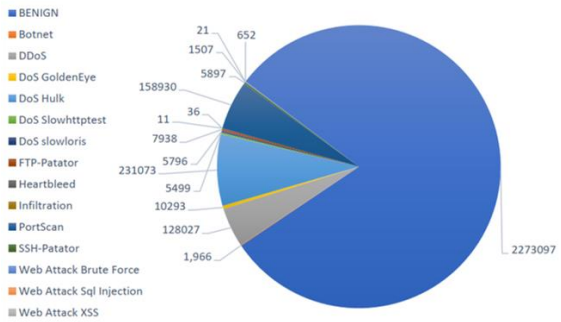


Figure 2. Distribution of BENIGN and attack class in the CICIDS2017 dataset

3.2. Data Preprocessing

In this section, data preprocessing, normalization, and feature selection processes

on the CICIDS2017 dataset are presented before performing classification with boosting algorithms. First of all, the missing, erroneous and corrupt data detected in the data set were investigated and a total of 1358 data were extracted from the dataset. In the next stage, the data indicating the attack class were digitized in the range of 0-14 with the Label Encoding method. Scaling the data set through standardization and normalization methods to improve model accuracy is critical for machine learning algorithms. Because machine learning algorithms provide superior results when data is well-distributed and reorganized. As a result, the cleaned and corrected data was subjected to Z-Score Scaling.

Following the normalization phase, the CICIDS 2017 dataset was subjected to feature selection. The Recursive Feature Elimination (RFE) approach was employed in the study to choose features. This technique is based on removing features that do not help to gradually separate distinct classes from all features [33]. The weights are determined in this case by applying a classification algorithm to the features. The procedure of elimination is repeated until only the characteristics with the greatest discriminating rate remain. By examining the algorithms employed and their performance rates, the threshold value of 0.005 was chosen as the importance value providing the best performance in the study. As a consequence, 25 features with values lower than this threshold were eliminated from the data set, resulting in a data set of 53 features. Figure 3 is a graph depicting the significance values of the attributes in the CICID2017 data set. In Figure 3, for example, characteristics like Packet Length and Destination port are highlighted as critical for identifying the attack, whereas the Idle Std property, which indicates the standard deviation of a stream's idle time, is shown as insignificant.

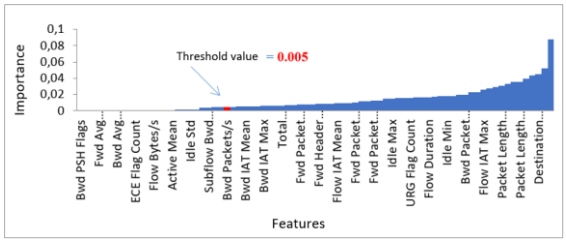


Figure 3. Features importance with RFE method in the CICIDS 2017 dataset.

4. CLASSIFYING WITH BOOSTING ALGORITHMS

4.1. Dataset Modelling of the Boosting Algorithms and Evaluation Metrics

The research was conducted using the Colab platform utilizing the Python programming language and the Jupyter Notebook virtual server. A computer with an NVIDIA Tesla P100-PCIE-16GB GPU processor on the Google Colab platform was utilized as a hardware tool.

In order to apply machine learning techniques, the CICIDS2017 dataset was split into two parts: 70% training and 30% testing. At this point, no data was obtained from random rows of the data set while constructing the training and test data sets, and homogenous acquisitions were carried out according to the attack types. In both the training and test data sets, proportionate additions were made from attack categories and regular traffic data. 70% of the slowloris packages, for example, were put into the training set and 30% into the test set. This holds true for other sorts of attacks as well. In addition, cross validation (kfold = 10) was used to improve the study's accuracy.

Learning rate = 0.1, n estimators = 100, and max depth = 50 were used as the basic parameters for Adaboost, Catboost, GradientBoost, LightGBM, and XGBoost models in the study. Following the training procedure, the boosting algorithms' performances were measured using accuracy, precision, recall, f1-score, Kappa, receiver operating characteristic (ROC) curve, and area under roc curve (AUC).

The accuracy expressed by Eq.(2) is the ratio of true positive (TP) and true negative (TN) results to the total number of observations [34]. The precision, on the other hand, is defined as the ratio of observations with a real value of positive and classified as positive (TP), as provided in Eq.(3), to all observations with a positive value. The recall is the ratio between the number of observations with a real value of positive and categorized as positive (TP), the number of observations with a real value of positive, and the estimated value of positive and negative (False Negative - FN). It is expressed as Eq.(4). The f1-score is obtained by the harmonic average of precision and recall values expressed in Eq.(5). The f1-score value takes a value between 0 and 1, and the value that gets

closer to 1 indicates that the model is better. The Kappa value measures the reliability of compatibility between the results of two or more models and is calculated by Eq.(6). The actual value is represented by po, while the predicted value is represented by pe. The value of Kappa varies from -1 to +1. As the Kappa value approaches +1, it shows that the findings of the model are consistent, as it approaches 0, it suggests that compatibility between models is totally random, and as it approaches -1, it indicates that the outcomes are evaluated inversely. The ROC curve, on the other hand, is employed in the binary classification problem. When the AUC value is near 1, the model is successful [35].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

$$f1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

$$Kappa = \frac{p_o - p_e}{1 - p_e} \quad (6)$$

4.2. Experimental Results and Discussion

Table 2 shows the macro average values of the performance outcomes of the models used to detect and classify network attacks in the CICID2017 data set. The XGBoost model provided the best estimation based on the accuracy, precision, and f1-score criteria, with values of 0.9989, 0.9942, and 0.8937, respectively. The AdaBoost model, on the other hand, became the most successful model after XGBoost and even achieved the greatest recall value with 0.8760. The LightGBM model has the lowest success rate among the models. It has an f1-score of 0.4817. The fact that this value is less than 1 shows that the model is unable to identify all kinds of network attacks.

Table 2. Comparison of evaluation metrics.

Classifier	Acc.	Prec	Recall	f1
AdaBoost	0.9985	0.9039	**0.8760**	0.8865
CatBoost	0.9982	0.9052	0.8745	0.8740
G.Boost	0.9984	0.8351	0.8304	0.8322
LightGBM	0.9773	0.4653	0.5126	0.4817
XGBoost	**0.9989**	**0.9242**	0.8747	**0.8937**

Figure 4 depicts the findings achieved by the models using the Kappa and AUC criteria. The XGBoost model produced the highest results in terms of Kappa and AUC, with values of 0.9969 and 0.9989, respectively. The LightGBM model was the most unsuccessful, with Kappa = 0.9339 and AUC = 0.9445.

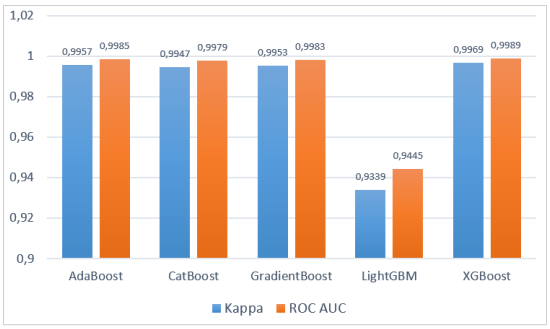


Figure 4. Kappa and AUC values.

BENIGN	0.9997	0.9993	0.9995	682324
Bot	0.8689	0.8112	0.8391	588
DDoS	0.9999	0.9997	0.9998	38238
DoS GoldenEye	0.9981	0.9952	0.9966	3097
DoS Hulk	0.9985	0.9995	0.9990	69212
DoS Slowhttptest	0.9902	0.9926	0.9914	1625
DoS slowloris	0.9949	0.9915	0.9932	1762
FTP-Patator	1.0000	0.9996	0.9998	2317
Heartbleed	1.0000	1.0000	1.0000	2
Infiltration	1.0000	0.6000	0.7500	10
PortScan	0.9942	0.9997	0.9969	47590
SSH-Patator	0.9994	1.0000	0.9997	1757
Web Attack Brute Force	0.7297	0.8178	0.7712	472
Web Attack Sql Injection	0.8000	0.5714	0.6667	7
Web Attack XSS	0.4903	0.3423	0.4032	222
accuracy			0.9989	849223
macro avg	0.9242	0.8747	0.8937	849223
weighted avg	0.9989	0.9989	0.9989	849223

a)

	precision	recall	f1-score	support
BENIGN	0.9886	0.9889	0.9887	682324
Bot	0.0000	0.0000	0.0000	588
DDoS	0.9494	0.9940	0.9712	38238
DoS GoldenEye	0.7498	0.7162	0.7326	3097
DoS Hulk	0.9695	0.9231	0.9457	69212
DoS Slowhttptest	0.3402	0.3748	0.3567	1625
DoS slowloris	0.0436	0.0108	0.0173	1762
FTP-Patator	0.6352	0.8727	0.7353	2317
Heartbleed	0.0000	0.0000	0.0000	2
Infiltration	0.0000	0.0000	0.0000	10
PortScan	0.9724	0.9780	0.9752	47590
SSH-Patator	0.8722	0.9169	0.8940	1757
Web Attack Brute Force	0.2612	0.5572	0.3556	472
Web Attack Sql Injection	0.0000	0.0000	0.0000	7
Web Attack XSS	0.1970	0.3559	0.2536	222
accuracy			0.9773	849223
macro avg	0.4653	0.5126	0.4817	849223
weighted avg	0.9778	0.9773	0.9774	849223

b)

Figure 5. Classification report of the **a)** XGBoost model and, **b)** LightGBM model.

The LightGBM model seems to be the most unsuccessful model in classifying network attacks according to precision, recall and f1-score values. However, accuracy = 0.9773 and Kappa = 0.9339 values were close to 1. At this point, the results of the LightGBM model in predicting and classifying network attacks were

examined in detail and the reasons were investigated. In addition, Figure 5 (a)-(b) shows a comparison with the most successful model, the XGBoost. The XGBoost method, shown in Figure 5 (a), assessed Bening, DDoS, Dos GoldenEye, Dos Hulk, Dos Slowhttptest, DoS Slowloris, FTP-Patator, Hearthbleed, PortScan, and SSH-Patator classes with an accuracy of more than 0.99. The f1-score values, on the other hand, remained low, resulting in inaccurate estimations in the Bot, Infiltration, Web Attack Brute Force, Web Attack SQL Injection, and Web Attack XSS attack classes.

Figure 5 (b) shows that the LightGBM model was unable to make proper predictions for Bot, Heartbleed, Infiltration, and Web Attack SQL Injection attacks, with accuracy, recall, and f1-score values of 0.0. The major reason for this is that there aren't many of these kinds of attacks in the dataset. Only 2 of the 849,223 entries in the test dataset belong to the Hearthbleed attack class, and 7 of them belong to the Web Attack SQL Injection attack class, according to the number of attack classes listed in the "Support" column. A dataset for network attacks, on the other hand, is naturally imbalanced, and precise forecasts for all forms of network attacks are required. It can be said that the LightGBM model will produce better results when the data is balanced. In other words, the performance rate for selecting classes with fewer data in a dataset is low, but the performance rate for high-rate data classes is high. In imbalanced data, the weighted average value is also used to determine the model's overall performance. The weighted average considers each element's categorization weight depending on its ratio within the dataset. In Figures 5 (a) and (b), the weighted average f1-score values for XGBoost and LightGBM are 0.9989 and 0.9774, respectively. As a result, it categorized network traffic successfully in both models in a broad sense. However, the LightGBM model has been less successful in properly identifying smaller-scale attacks.

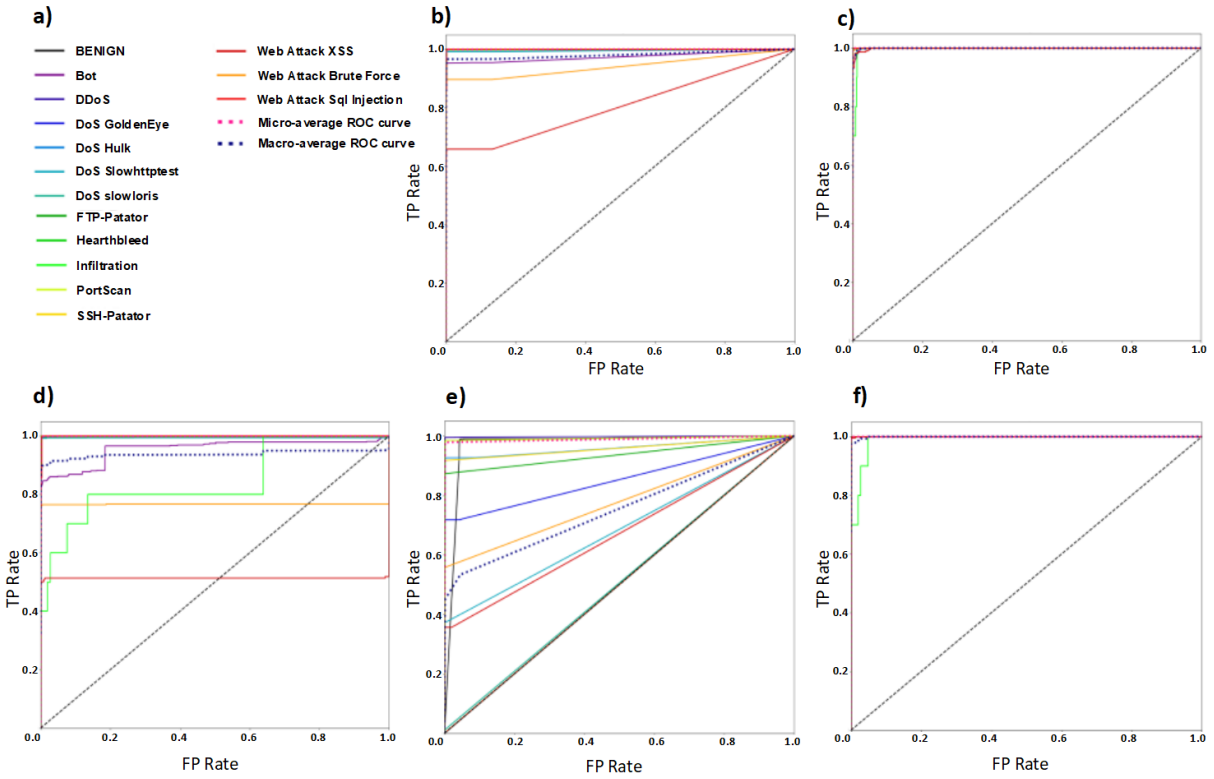
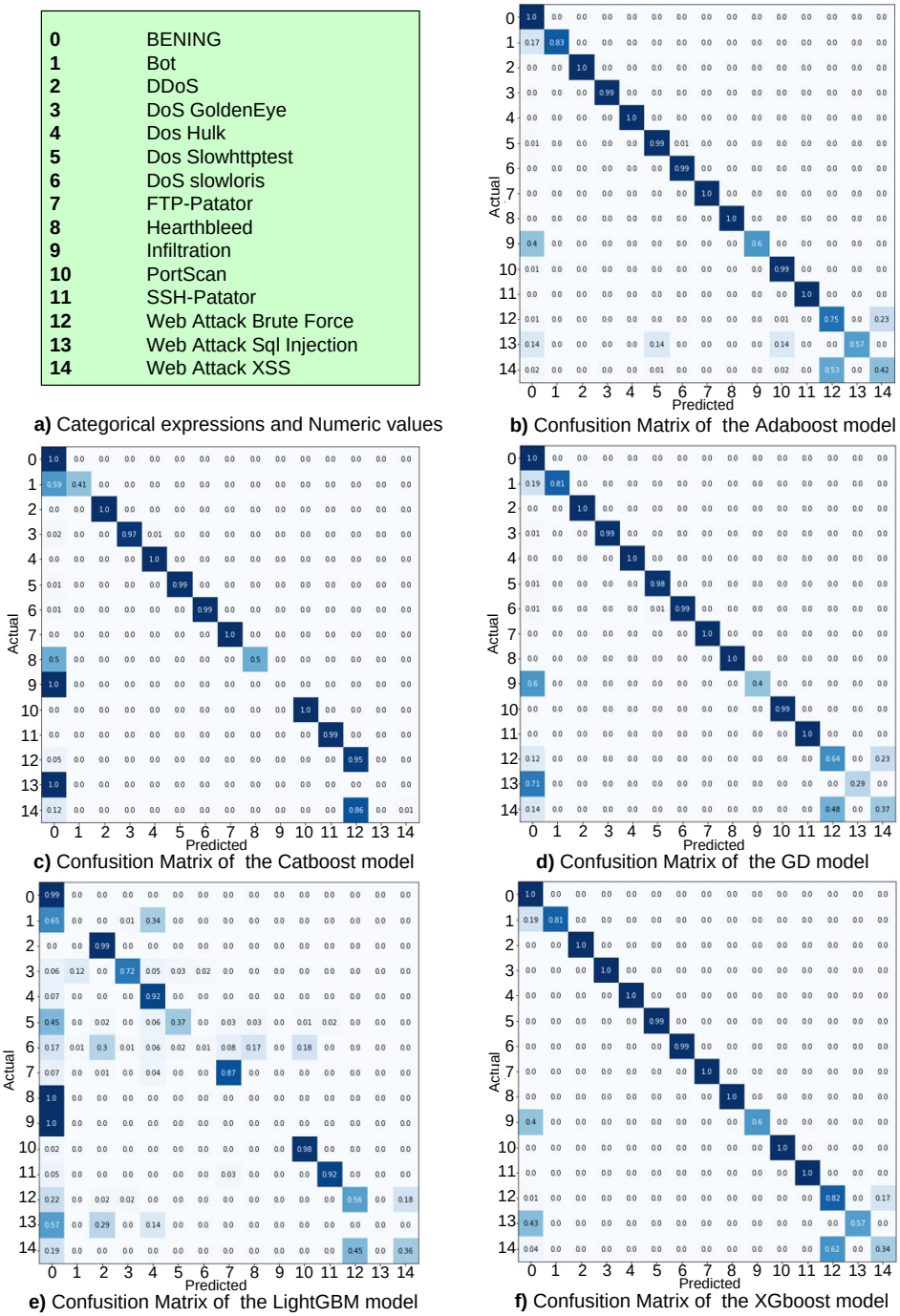


Figure 6. a) ROC curve colours to network attacks, ROC curve of the **b)** AdaBoost **c)** CatBoost **d)** GradientBoost **e)** LightGBM **f)** XGBoost models.

Figure 6(b)–(e) depicts the classification of each attack type and normal network traffic using ROC curves. According to the ROC graph, the greater the area under a curve, the more effective the classification process for that curve's class. As an example, the Adaboost models's findings are presented in Figure 6(b). The AUC was determined to be 0.9985. AUC

value close to 1.00 indicates that the model does not memorize and the performance of the model is high. When the ROC curve is examined, the attacks titled 'Bot' and 'Web Attack' remained below the AUC value. That is, incorrect classifications have been made for these attack classes. One of the models that are graphically clear that misclassification is made is the GradientBoost model shown in Figure 6(c). The model's AUC is 0.9983. The ROC curve showed that the attacks named 'Bot,' 'Infiltration,' and 'Web Attack' were below the AUC value. As a result, the GradientBoost model, like the LightGBM model, failed to classify classes with a small number of data in the data set. The CatBoost and XGBoost models, on the other hand, have a high success rate. When the ROC curves are analyzed, it is clear that the classification process completed fast, with less data. However, there are still some incorrect findings when determining the Infiltration attack.



a) Categorical expressions and Numeric values

b) Confusion Matrix of the Adaboost model

c) Confusion Matrix of the Catboost model

d) Confusion Matrix of the GD model

e) Confusion Matrix of the LightGBM model

f) Confusion Matrix of the XGboost model

Figure 7. a) Categorical expressions and Numerical Values, Confusion Matrix of the **b)** AdaBoost **c)**CatBoost **d)**GradientBoost **e)**LightGBM **f)**XGBoost models.

The confusion matrices of the models shown in Figure 7(b)-(f) indicate how well the classes were predicted, as well as which classes were wrongly predicted. Some data relating to the categories of attacks labeled 'Bot', 'Infiltration,' and 'Web Attack,' for example, are erroneously classified in all models. The AdaBoost model predicted the 'Bot' attack with an accuracy of 83% (Figure 7 (b)) and classified it as 17% BENING. In the LightGBM model (Figure 7 (e)), on the other hand, no proper classification was made, with 65% classified as BENING and 35% classified as 'DosHulk'. In the case of the 'Infiltration' attack, the CatBoost and LightGBM models were unable to accurately classify the data and misclassified it as regular traffic. The AdaBoost and XGBoost models produced the best classification results. 60% of the packages were correctly classified in both models, whereas 40% were labeled as BENING. The major reason for this is that, because the Infiltration attack uses malicious file sending, it might be mistaken for regular file sending in network traffic, and therefore, models can label it as 'BENING.'

5. CONCLUSION

AdaBoost, CatBoost, Gradient Boost, Light GBM, and XGBoost were employed to detect malicious network traffic in this research. CICIDS2017 was chosen as the dataset. The performance of the trained and tested classifiers was examined in terms of evaluation criteria such as accuracy, precision, recall, f1-score, Kappa, ROC curve and AUC.

The XGBoost model provided the best estimate based on accuracy, precision, f1-score, Kappa and ROC AUC criteria. The LightGBM model, on the other hand, has the lowest success rate among the models. It was determined that it could not make accurate predictions, especially for Bot, Heartbleed, Infiltration, and Web Attack SQL Injection attacks in the data set. The biggest reason for this is that these types of network attacks are small in number in the dataset. It can be said that the performance rate of the LightGBM model is low for unbalanced datasets. On the other hand, when evaluated in terms of classification of network attack types, all models of 'Bot', 'Infiltration,' and 'Web Attack' attacks were the most misclassified attack types.

With this proposed study, it is possible to detect network attacks in real-world systems and take necessary precautions. It's also important to figure out what kind of attack you're up against. As technology continues to advance, it is expected that the types of attacks will expand, making it more difficult to identify them in the future. Meanwhile, CPU and GPU capabilities are also increasing at a rapid rate. Using deep learning algorithms and machine learning algorithms on computers with great processing capacity, it is possible to identify malicious network traffic with high accuracy and speed.

REFERENCES

1. Perez, S.I., Moral-Rubio, S., Criado, R., "A new approach to combine multiplex networks and time series attributes: Building intrusion detection systems (IDS) in cybersecurity", Chaos, Solutions and Fractals, Vol. 150, Pages 1-11, 2021.

2. Kanimozhi, V. and Jacob, T.P, "Artificial Intelligence based Network Intrusion Detection with hyper-parameter optimization tuning on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing", ICT Express, Vol. 5, Issue 3, Pages 211-214, 2019.

3. Saranya, T., Sridevi, S., Deisy, C., Chung, T.D., Ahamed, K.M., "Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review", Third IC on Computing and Network Communications (CoCoNet'19), Trivandrum, 2020.

4. Ghurab, M., Gaphari, G., Alshami, F., Alshamy, R., Othman, S., "A Detailed Analysis of Benchmark Datasets for Network Intrusion Detection System" Asian Journal of Research in Computer Science, Vol. 7, Issue 4, Pages 14-33, 2021.

5. Sharafaldin, I., Lashkari, A., Ghorbani, A., "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization", 4th International Conference on Information Systems Security and Privacy, Portugal, 2018.

6. Özekes, S. and Karakoç, E.N., "Makine Öğrenmesi Yöntemleriyle Anormal Ağ Trafiğinin Tespit Edilmesi", Düzce Üniversitesi Bilim ve Teknoloji Dergisi, Vol. 7, Issue 1, Pages 566-576, 2019.

7. Tama, B.A., Nkenyereye, L., Islam, S.R., Kwak, K.S., "An Enhanced Anomaly Detection in Web Traffic Using a Stack of Classifier Ensemble", IEEE Access, Vol. 8, Pages 24120 – 24134, 2020.

8. Abdulrahman, A.A. and Ibrahem, M.K., "Toward Constructing a Balanced Intrusion Detection Dataset Based on CICIDS2017", Samarra Journal of Pure and Applied Science, Vol. 2, Issue 3, Pages 132-142, 2020.

9. Hosseini, S. and Seilani, H., "Anomaly process detection using negative selection algorithm and classification techniques", Evolving Systems, Vol. 12, Pages 769–778, 2021.

10. Hongle, D., Yan, Z., Gang, K., Lin, Z., Chen, Y.C., "Online ensemble learning algorithm for imbalanced data stream", Applied Soft Computing, Vol. 107, Pages 1-12, 2021.

11. Schapire, R.E., "The Boosting Approach to Machine Learning an Overview", In: Denison DD, Hansen MH, Holmes CC et al editors, Nonlinear Estimation and Classification. Lecture Notes in Statistics, Vol. 171, Springer, New York, Pages 1-23, 2003.

12. Pham, X.T. and Ho, T.H., "Using boosting algorithms to predict bank failure: An untold story", International Review of Economics & Finance, Vol. 76, Pages 40-54, 2021.

13. Shahraki, A., Abbasi, M., Haugen, Q., "Boosting algorithms for network intrusion detection: A comparative evaluation of Real AdaBoost, Gentle AdaBoost and Modest AdaBoost", Engineering Applications of Artificial Intelligence, Vol. 94, Pages 1-14, 2020.

14. Li, Y., Shi, H., Duan, Z., Liu, H., "Smart wind speed forecasting approach using various boosting algorithms, big multi-step forecasting strategy", Renewable Energy, Vol. 135, Pages 540-553, 2019.

15. Ma, B., Meng, F., Yan, G., Yan, H., Chai, B., Song, F., "Diagnostic classification of cancers using extreme gradient boosting algorithm and multi-omics data", Computers in Biology and Medicine, Vol. 121, Pages 1-10, 2020.

16. Abro, A.A, Taşcı, E., Uğur, A.A., "Stacking-based Ensemble Learning Method for Outlier Detection", Balkan Journal of Electrical & Computer Engineering, Vol. 8, Issue 2, Pages 191-185, 2020.

17. Wen, L., Hughes, M., "Coastal Wetland Mapping Using Ensemble Learning Algorithms: A Comparative Study of Bagging, Boosting and Stacking Techniques", Remote Sensing, Vol. 12, Issue 10, Pages 1-18, 2020.

18. Xia, T., Zhuo, P., Xiao, L., Du, S., Wang, D., Lifeng, X. "Multi-stage fault diagnosis framework for rolling bearing based on OHF Elman AdaBoost-Bagging algorithm", Neurocomputing, Vol. 433, Pages 237-251, 2021.

19. Andiojaya, A. and Demirhan, H., "A bagging algorithm for the imputation of missing values in time series", Expert Systems with Applications, Vol. 129, Pages 10-26, 2019.

20. Yin, S., Liu, H., Duan, Z., "Hourly PM2.5 concentrations multi-step forecasting method based on extreme learning machine, boosting algorithm and error correction model", Digital Signal Processing, Vol. 118, Pages 1-21, 2021.

21. Freund, Y. and Schapire, R.E., "A decision-theoretic generalization of on- line learning and an application to boosting", Journal of Computer and System Sciences, Vol. 55, Issue 1, Pages 119-139, 1997.

22. Chengsheng, T., Huacheng, L., Xu, B., "AdaBoost typical Algorithm and its application research", MATEC Web of Conferences, Vol. 139, Issue 2, France, 2017.

23. Qi, C., Wang, Y., Tian, W., Wang, Q., "Multiple kernel boosting framework based on information measure for classification", Chaos, Solutions and Fractals, Vol. 89, Pages 175-186, 2016.

24. Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V., Gulin, A., "CatBoost: unbiased boosting with categorical features", NeurIPS - 32nd Conference on Neural Information Processing Systems, Montreal, 2018.

25. Friedman J.H., "Greedy function approximation: a gradient boosting machine", Annals of statistics, Vol. 29, Issue 5, Page s1189-1232, 2001.

26. Kearns, M. and Valiant, L., "Cryptographic limitations on learning Boolean formulae and finite automata", Journal of the ACM, Vol. 41, Issue 1, Pages 67-95, 1994.

27. Friedman, J.H. "Stochastic gradient boosting", Computational Statistics & Data Analysis, Vol. 38, Issue 4, Page 367-378, 2002.

28. Dahiya, N., Saini, B., Chalak, H.D., "Gradient boosting-based regression modelling for estimating the time period of the irregular precast concrete structural system with cross bracing", Journal of King Saud University - Engineering Sciences, Pages 1-8, 2021.

29. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y., "LightGBM: a highly efficient gradient boosting decision tree", NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc. California, 2017.

30. Shehadeh, A., Alshboul, O., Al Mamlook, R.E., Hamedat, O., "Machine learning models for

predicting the residual value of heavy construction equipment: An evaluation of modified decision tree, LightGBM, and XGBoost regression", Automation in Construction, Vol. 129, Pages 1-16, 2021.

31. Chen, T. and Guestrin, C., "XGboost: A scalable tree boosting system", 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 785–794, San Francisco, 2016.

32. Ma, J., Zhongqi, Y., Qu, Y., Xu, J., Cao, Y., "Application of the XGBoost Machine Learning Method in PM2.5 Prediction: A Case Study of Shanghai", Aerosol and Air Quality Research, Vol. 20, Issue 1, Pages 128-138, 2019.

33. Sharma, N.V. and Yadav, N.S., "An optimal intrusion detection system using recursive feature elimination and ensemble of classifiers", Microprocessors and Microsystems, Vol. 85, Pages 1-11, 2021.

34. Aksoy, B., Usta, U., Karadağ, G., Kaya, A.R., Ömür, M., "Classification of Environmental Sounds with Deep Learning", Advances in Artificial Intelligence Research, Vol. 2, Issue 1, Pages 20-28, 2022.

35. Aksoy, B. and Salman, O.K.M., "Detection of COVID-19 Disease in Chest X-Ray Images with capsul networks: application with cloud computing", Journal of Experimental & Theoretical Artificial Intelligence, Vol. 33, Issue 3, Pages 527-541, 2021.