IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 March 2024, accepted 24 March 2024, date of publication 27 March 2024, date of current version 3 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3382539

RESEARCH ARTICLE

A Comprehensive Analysis of the Machine Learning Algorithms in IoT IDS Systems

ERDAL OZDOGAN
IT Department, Gazi University, 06560 Ankara, Turkey
e-mail: erdal.ozdogan@gazi.edu.tr

ABSTRACT In this study, machine learning algorithms in IoT IDS (Internet of Things Intrusion Detection System) systems are comprehensively compared from various aspects. Accuracy, precision, and training time are evaluated. The effects of data preprocessing techniques including normalization, outlier removal, standardization, and regularization on the datasets are examined. Furthermore, the impact of dataset balancing, considering both balanced and imbalanced scenarios, on machine learning performance is investigated. The contribution of feature selection on the four different datasets is also analyzed. Based on findings, it is observed that certain preprocessing operations provide significant advantages in various ML algorithms, whereas others have very low impact, and their performance varies depending on the dataset and feature selection. The aim of this study is to facilitate the complexity and lengthiness of machine learning processes and algorithm selection, providing insights for future academic research. By addressing this objective, an effort is made to shed light on simplifying the utilization of machine learning algorithms. The challenges arising from the complexity of machine learning processes in IoT IDS systems are addressed by this study. This contribution can greatly benefit researchers in their academic endeavors. This multifaceted approach proves beneficial when comparing the methods under consideration, fostering a scientific discourse on their efficacy within contexts.

INDEX TERMS IoT attack detection, IoT IDS, ML algorithm comparison, machine learning preprocessing, machine learning in IoT IDS.

I. INTRODUCTION

In today's world, IoT (Internet of Things) is considered a technology of great importance. IoT enables different devices connected to the internet to communicate with each other and exchange data. This technology provides numerous benefits, such as simplifying our lives, optimizing business processes, and increasing environmental sustainability. There are various fields where IoT can be used, including Smart Home and City Systems, Industrial Applications, Healthcare, Logistics, Agriculture, and Transportation [1], [2]. From a personal usage perspective, security is highly important in IoT. IoT devices have the ability to access personal and sensitive data and can pose security risks as they are connected to networks [3], [4]. Many IoT devices may have low-level security measures and can be vulnerable to malicious attacks [5]. Therefore, special attention must be given to the security of IoT devices. Similarly, when it comes to industrial purposes, it is evident that the security of IoT is of utmost importance. The confidentiality of the data transmitted in industrial IoT applications can be crucial as it may include sensitive information obtained from sensors [6], [7]. This data may include critical information related to product design in the production process or other critical organizational data [8]. In cases where data security is not ensured, information can fall into the hands of unauthorized threat actors, leading to the exposure of trade secrets and unfair competition situations. In addition to data security, destructive attacks by threat actors can disrupt business continuity, damage the brand value of an organization, and leave customers in distress [9], [10], [11]. The development and widespread use of IoT has provided attackers with several advantages that

The associate editor coordinating the review of this manuscript and approving it for publication was Rahim Rahmani.

were not available before [12], [13]. IoT presents a wide attack surface due to the multitude of devices connected to the internet. Each IoT device can be a potential vulnerability or weak point. Attackers can target many devices simultaneously or turn them into a botnet for attacks [14]. This enables attackers to launch larger-scale and more effective attacks. The limitations in the capacity of IoT devices create increased security vulnerabilities, making it easier for attackers to target and carry out attacks. Similarly, IoT devices can collect large amounts of data about users [15]. This data may include personal, financial, or health-related information. Attackers can access this data and engage in activities such as identity theft, fraud, or other malicious activities. The ability of IoT devices to communicate with each other and other networks enables new attack vectors [16]. These risks highlight the importance of IoT application security, which has the potential to affect various segments of society. As seen, IoT security encompasses many dimensions that continue to expand every day, and traditional methods may be insufficient to ensure security [17]. Machine Learning (ML) can detect anomalies by learning normal operating patterns. It can analyze normal behaviors in IoT systems and detect abnormal activities that do not conform to the established patterns [17], [18]. This can also be effective against new and unknown attacks [20]. Machine learning models can learn over time, which means that even if attackers change their attack methods or develop new tactics, security systems can update themselves [21]. This feature is important for detecting current attacks and enhancing defense capabilities. In conclusion, machine learning can provide a more effective approach to IoT security compared to traditional methods. The capabilities of anomaly detection, real-time monitoring, scalability, learning, and analysis of complex relationships offer significant advantages for enhancing the security level of IoT networks.

IoT systems generally generate a large amount of complex data. This data can be obtained from various sensors, devices, and other sources. The complexity of the data structure means that the algorithm needs to effectively analyze this data and produce results with high accuracy. Therefore, it is important to choose an algorithm that can process IoT data in the best possible way and handle unstructured data. IoT applications are typically large-scale and distributed systems. Therefore, the right machine learning algorithm should be able to process large amounts of data and provide scalability. Otherwise, the algorithm may face performance problems as the system expands. Thus, memory consumption, processor power consumption, and energy consumption should be considered in model selection. In machine learning, training time is also an important factor. Training time refers to the time spent on training the model [22]. Faster training time allows the model to be trained more quickly and results to be produced faster. Particularly when working with large datasets or complex models, fast training time is important for performance and efficiency. Training time can affect the amount and quality of data used to train a machine learning model. If the training time is too long, it may be difficult to use more data or access higher-quality data. This can impact the overall performance of the model. Optimizing training time is important for achieving better results within a limited time frame. It helps to make more efficient use of resources and reduce costs. In conclusion, the selection of the right machine learning algorithm in the context of IoT should consider factors such as the complexity of data structure, real-time processing requirements, data security, scalability, and fault tolerance.

In IoT networks, just like in traditional networks, Intrusion Detection Systems (IDS) are effectively used for detecting attacks. IDS is a security measure used to detect and respond to attacks on a network or system. By monitoring network traffic or system logs, IDS attempts to identify known attack signatures or abnormal activities. Machine learning-supported IDS systems offer various advantages over traditional IDSs [23], [24]. The detection of abnormal behaviors, extraction of meaningful relationships from large and complex data, and their ability to learn all indicate the advantages of machine learning-supported IDS systems in enhancing IoT security.

One other crucial aspect of IoT security is the accurate classification of attacks. Attacks on each IoT system can vary significantly. Therefore, organizations may require different approaches, customized or hybrid machine learning models, to prevent the attacks they are most vulnerable to. However, determining which IoT attacks can be more accurately detected using specific machine learning algorithms or combinations of algorithms is a significant challenge [25], [26]. Knowing which machine learning approach is more successful in detecting specific IoT attacks will enhance the effectiveness of hybrid solutions employing multiple models.

In this study, comprehensive comparisons were made to select the most suitable machine learning model for IoT IDS, considering the mentioned factors. The contribution of the article can be summarized as follows:

- A comparison has been made among the most used machine learning classifiers based on various performance metrics in IoT datasets.
- The binary classification performance of ML algorithms has been examined.
- The performance of ML classifiers has been evaluated in terms of training time, accuracy, precision, recall, and f1-score.
- The impact of scaling methods on accuracy has been investigated.
- The impact of feature selection on performance has been studied.
- The performance of algorithms in balanced and imbalanced datasets has been investigated, considering accuracy and f1-score metrics.
- Investigated using four different IoT datasets to determine which attacks could be detected with higher accuracy and f1-score by which classifier.

- Identifying which classifiers are more successful in detecting specific attack types, and consequently, determining which combination of classifiers will yield higher accuracy in preventing a particular IoT attack.

The remaining parts of this article are organized as follows. The second section includes works related to machine learning supported by IDSs. The third section examines machine learning classifiers, performance metrics, and IoT datasets. The fourth section presents the performance comparisons, which are the essence of the study. In the last part of the study, the obtained data is evaluated, and attention is drawn to the future research areas in this field.

II. RELATED WORKS

In recent years, technological advancements have not only made human life easier but have also brought forth numerous challenges in terms of information security. The expanding computer networks, increasing number of malicious users, and evolving security vulnerabilities have made them a target for attacks. The working algorithms of systems developed for attack detection and prevention have become an important field of study to minimize vulnerabilities in computer networks. The long response times of traditional intrusion detection systems and the failure of signature-based systems to detect current attacks have increased the significance of machine learning and deep learning approaches. The comparison of these algorithms, developed for identifying, classifying, and preventing security risks, using different metrics has led to an increasing number of recommendations for intrusion detection systems.

In the study [27], researchers aimed to develop an IDS using ensemble learning methods. They made recommendations regarding Denial of Service (DoS) attacks based on analyses conducted using CIDDS-001, UNSW-NB15, and NSL-KDD datasets. In another study [28], researchers performed an evaluation using the MQTT-IoT-IDS2020 dataset, specifically targeting the deficiencies found in older datasets and their associated attack classes. The study involved a comparison of several classification algorithms, including k-Nearest Neighbor (k-NN), Support Vector Machine (SVM), Naive Bayes (NB), Random Forest (RF), Decision Tree (DT), and Stochastic Gradient Descent (SGD). Reference [29] assessed existing Network Intrusion Detection System (NIDS) tools and datasets, contributing to identifying IoT challenges and proposing solutions. The widely used datasets include KDDCUP99 and NSL-KDD. Reference [30] presented a comparative analysis of selected Intelligent IDS using Microsoft Azure ML Studio (AML-S) platform and datasets containing malicious and benign IoT network traffic. Reference [31] focusing on the analysis of recently published UNSW-NB15, Bot-IoT, and CSE-CIC-IDS2018 datasets, this study emphasized the contribution of data using machine learning algorithms such as Random Forest (RF), Support Vector Machines, Keras Deep Learning models, and XGBoost. Reference [32] established a comparison point for various classification models in different datasets. It showed that some fast-converging algorithms had lower performance, whereas algorithms requiring longer convergence demonstrated more successful outcomes.

In [33], researchers conducted a comparative analysis of various ML models for intrusion detection using the UNSW-NB15 dataset. Integrated ML models achieved 99% accuracy in both binary and multi-class classifiers. In [34], researchers compared k-Means-based, decision tree-based, and hybrid IDS models. Despite achieving a lower detection rate, the Hybrid IDS proved to be more accurate compared to the other two approaches.

Hidayat et al. proposed a hybrid feature selection technique using a random forest model and Pearson correlation coefficient based on the TON_IoT dataset [35]. The dataset underwent training utilizing Adaboost, decision tree, multilayer perceptron, and Long Short-Term Memory (LSTM) networks for multi-layer detection. The research findings indicated that machine learning techniques exhibited superior efficacy in attack detection, as evidenced by enhancements in accuracy, precision, and recall metrics.

Ahmad et al. found that intrusion detection systems need to be trained beyond limited datasets and application scopes by using different datasets [20]. They proposed a hybrid model by sequentially using autoencoders. They expanded TCN architectures to use causal points in a layered manner. Their analysis using MLP, CNN, and LSTM showed that CNN had a higher accuracy rate compared to the others.

Zhang et al. conducted a study [36] where they analyzed research from the past decade. They worked with NSL-KDD and KDD CUP99 datasets, exploring community learning, machine learning, and deep learning techniques. Comparison results, based on metrics like Area Under the Curve (AUC), F1 score, and accuracy, favored community learning algorithms. Additionally, the study delved into the impact of training iteration numbers on deep learning algorithms.

Gamage and Samarabandu evaluated advanced deep learning models such as feed-forward neural networks, autoencoders, deep belief networks, and long short-term memory networks using NSL-KDD, CIC-IDS2017, and CIC-IDS2018 up-to-date datasets [37]. The study unveiled that deep feed-forward neural networks demonstrated high accuracy values across all four datasets, considering accuracy, F1-score, training, and inference time. Additionally, the researchers noted that two widely used semi-supervised learning models, autoencoders, and deep belief networks, failed to surpass supervised feed-forward neural networks.

In study [38],SVM, Naïve Bayes, decision trees, and random forests were employed to detect anomalies in network traffic. The researchers evaluated the UNSW-NB15 dataset in terms of detection accuracy, construction time, and prediction time. Their findings concluded that the RF algorithm demonstrated high accuracy in classifying attacks.

In [39], researchers focused on utilizing deep learning algorithms to develop an intrusion detection system.They concluded that deep learning surpassed machine learning due

to its capability to effectively handle large datasets, detect zero-day attacks, and incorporate self-learning capabilities.

III. BACKGROUND

Machine learning classification is a specific field that focuses on the classification or tagging of data based on its attributes [40]. The objective is to develop algorithms or models capable of automatically assigning new, unseen data points to predefined categories. Essentially, classification algorithms learn from labeled training data to make predictions or decisions about the class labels of unknown instances. Datasets play a crucial role in machine learning and data analysis. They are vital resources that enable the training, evaluation, and validation of machine learning models, facilitate feature extraction and analysis, support fairness assessment, and promote research reproducibility. Datasets form the backbone of data-driven approaches and contribute to the advancement of machine learning and data science.

In this section, classifier algorithms used in machine learning, performance metrics, and IoT datasets used in the study will be briefly explained.

A. MACHINE LEARNING CLASSIFIERS

There are numerous machine learning classifiers employed in Machine Learning-supported IDS design. In this section, commonly used classifiers, which are also evaluated in this study, are briefly discussed.

1) NAÏVE BAYES GAUSSIAN (NBG)

Naïve Bayes Gaussian (NBG) is a simple probability-based classification algorithm [40], [41], [42]. This algorithm relies on the assumption of independence among features in the dataset. Therefore, the training and prediction of the model can be performed quickly. Speed can be a crucial factor in detecting network attacks, and NBG can fulfill this requirement. Network attacks often exhibit diverse features, and the independence assumption can be beneficial in detecting such attacks. This allows the algorithm to capture specific features of attacks and reflect patterns that define an attack. Since network attacks often need to be classified against normal traffic, NBG can be an effective choice for binary classification tasks. However, it has disadvantages such as not considering the true dependencies among features and being tightly bound to the independence assumption.

2) K-NEAREST NEIGHBOR (KNN)

K-Nearest Neighbors is a simple and popular classification algorithm that offers a proximity-based classification approach [43], [44]. When confronted with a new network traffic sample, the KNN algorithm identifies its nearest neighbors (with a specific value of K) and classifies it based on the majority class. This way, it can detect new attacks that exhibit similarity to known attack patterns. KNN can be used with various types of data structures in intrusion detection. For example, network traffic data can be numerical, categorical, or have complex structures. KNN can work with these different data structures and assess similarity by calculating the distance measure for each data point. KNN is suitable for real-time intrusion detection [45], [46]. During the training phase, the algorithm retains the learning data and quickly utilizes the necessary data points for classification. This makes it ideal for swiftly detecting instantaneous attacks. However, it can be computationally expensive depending on the size of the dataset [47]. Distance calculations must be performed for all training data points for each new sample. Hence, the computational time can significantly increase for large datasets or high-dimensional data. Additionally, it may produce biased results in the presence of class imbalances. For instance, if the majority class vastly outweighs the others, the KNN model may favor the majority class and produce incorrect classifications [48]. To address class imbalances, data sampling techniques or class weighting methods can be employed to achieve a proper balance between classes. Moreover, if a particular feature contains much larger values compared to others, it can disproportionately influence the KNN algorithm and diminish the contributions of other features. Therefore, it is important to normalize the features using techniques like Min-Max scaling or Z-Score normalization to bring the features to the same scale.

3) ADAPTIVE BOOSTING (ADABOOST)

Adaptive Boosting is an ensemble method that combines weak classifiers to create a strong classifier [49]. It achieves high accuracy and classification performance [50]. In IDS systems, accurate classification is crucial, and Adaboost can be effective in achieving this goal. Adaboost can be used to evaluate feature importance. During the training of each weak classifier, the importance levels of features can be determined [51]. This algorithm helps identify which features are most effective in detecting attacks and eliminating unnecessary ones. Adaboost learns weak classifiers gradually by updating their weights, boosting the importance of misclassified examples while reducing that of correctly classified ones. This adaptive nature enables Adaboost to excel with challenging examples. However, it may face performance issues with imbalanced datasets. In instances of significant class imbalance (e.g., when the attack class is much smaller), Adaboost may misclassify the majority class. In such cases, careful data sampling or weight adjustment may be necessary. Additionally, if the training data contains noise or misleading information, the Adaboost model can learn these patterns, potentially impairing its generalization performance [52]. To mitigate this risk, appropriate hyperparameter tuning and limiting the complexity of weak classifiers [53], such as decision trees, are necessary to control overfitting.

4) EXTREME GRADIENT BOOST (XGBOOST)

XGBoost, an algorithm based on gradient boosting, is specifically designed to achieve high performance and prediction accuracy in classification and regression problems [54]. It constructs a model by combining multiple decision trees

and updates them step by step to minimize predefined errors. The incremental updates and error reduction of the trees enable XGBoost is used to obtain more precise and reliable results in attack detection due to its fast and scalable algorithm. It performs well on large datasets due to its parallel computing capabilities. In IoT environments, where a vast amount of data flows from numerous devices, XGBoost can swiftly process this data, expediting the detection of attacks. XGBoost offers flexibility through the adjustment of various hyperparameters [55] allowing the algorithm to be fine-tuned for attack detection and tailored to different scenarios. By incorporating regularization techniques and controlling tree size, XGBoost mitigates the risk of overfitting, thereby producing more dependable results in IoT attack detection.

5) SUPPORT VECTOR CLASSIFIER (SVC)

Support Vector Classifier is a classification algorithm commonly used in problems like intrusion detection. SVC creates a hyperplane to effectively classify data points and performs classification using the support vectors on this hyperplane [28]. Support Vector Machine is proficient in handling non-linear datasets, which is crucial in intrusion detection [56] where data frequently exhibits intricate structures and non-linear relationships [57]. It can adapt to such data and classify them by utilizing kernel functions. SVC can be resistant to outliers during the classification process. Outliers can lead to misleading results in intrusion detection. However, the support vectors and the classification process of SVC can minimize the impact of these outliers. SVC has a regularization parameter to improve the generalization performance of the classification model. This parameter reduces the risk of overfitting and allows the model to better fit the overall dataset. In intrusion detection, the ability of the model to correctly detect new attacks is important, and good generalization performance provides an advantage in this aspect. SVC can be computationally expensive for high-dimensional features, particularly when dealing with non-linear data and complex kernel functions [58]. This can increase the computation time, leading to performance problems in large-scale intrusion detection systems.

6) LOGISTIC REGRESSION (LR)

Logistic regression is a commonly used machine learning algorithm for binary classification problems [59], [60]. Logistic regression offers a simple and interpretable model. Understanding the key features describing attacks in an IoT environment is crucial for effective detection. Logistic regression provides coefficients, aiding in the interpretation of feature impact and contribution to the model. This facilitates attack analysis and offers interpretable results to designers. Additionally, logistic regression is known for its speed in both training and prediction times. Given the necessity for real-time attack detection in IoT environments, a fast model is essential. Logistic regression meets this requirement by being trained quickly and capable of swiftly detecting real-time attacks. Moreover, logistic regression is versatile, capable of handling both numerical and categorical features. Since IoT datasets often comprise various feature types, logistic regression's ability to work with diverse data types makes it suitable for such datasets. As a linear classification model, logistic regression has limitations in capturing nonlinear relationships. Consequently, it may not fully capture complex attack patterns in an IoT environment.

B. PERFORMANCE METRICS

Performance metrics in machine learning are used to evaluate how well a model or algorithm performs and how effective it is. Performance metrics quantitatively assess various performance measures of a model, such as accuracy, precision, recall, f1-score and training time [61], [62], [63]. The selection of these metrics reflects our goal to evaluate algorithms' performance and optimize decision-making processes in security sensitive IoT environments, considering both accuracy and speed. These metrics are utilized for comparing different models or algorithms. To make the appropriate model selection among them, evaluating which model better aligns with the desired performance metrics is crucial. For instance, in classification problems, metrics such as accuracy rate or precision are pivotal for assessing the model's ability to classify correctly. Machine learning models are typically configured with hyperparameters, which can impact their performance. Performance metrics assist in evaluating the results of different hyperparameter adjustments, aiding in the selection of the correct parameters to achieve optimal performance. These metrics are also instrumental in refining a developed model. Likewise, they are utilized to comprehend the nature and characteristics of a dataset. Various performance metrics can assess features of the dataset, such as imbalance, error distribution, or class separability. This information is crucial for correctly utilizing the dataset and can significantly impact the success of the model. Confusion matrix is a tool used to evaluate the classification performance of a model in machine learning. It demonstrates the relationship between the true classes and the predicted classes by the model. The confusion matrix consists of four key concepts as shown in Fig.1.

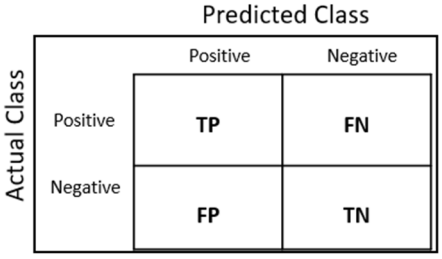


FIGURE 1. Confusion matrix is a tool used to evaluate classification performance.

True Positive (TP), represents the number of examples that the model correctly predicts as positive. These are the

examples that belong to the positive class and are correctly classified as such by the model. True Negative (TN), represents the number of examples that the model correctly predicts as negative. False Positive (FP), represents the number of examples that the model incorrectly predicts as positive. False Negative (FN), represents the number of examples that the model incorrectly predicts as negative. Various evaluation metrics are used based on these four concepts.

Accuracy: Shows the ratio of correctly classified examples by the model to the total number of examples Eq. (1).

$$accurcay = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Precision: Indicates the ratio of correctly predicted positive examples by the model to the total number of positive predictions made by the model Eq.(2).

$$precision = \frac{TP}{TP + FP} \tag{2}$$

Recall or Sensitivity: Indicates the rate of correctly detected positive examples Eq. (3).

$$recall = \frac{TP}{TP + FN} \tag{3}$$

F1-Score: Represents the harmonic mean of precision and recall, considering both precision and recall values Eq. (4).

$$F1 - Score = \frac{2 * precision * recall}{precision + recall} \tag{4}$$

Receiver Operating Characteristic: The Receiver Operating Characteristic (ROC) curve is a graphical representation of the model's performance in binary classification problems. It plots the true positive rate (sensitivity) against the false positive rate (1 - specificity) at various classification thresholds [64]. The curve illustrates the trade-off between sensitivity and specificity for different threshold values. A model with a higher ROC curve that is closer to the top-left corner indicates better performance. The Area Under Curve (AUC) is the area under the ROC curve and provides a single scalar value that represents the overall performance of the model [65]. It ranges between 0 and 1, where a value of 1 represents a perfect classifier, whereas a value of 0.5 represents a random classifier. A higher AUC indicates better discrimination power and overall model performance.

Calibration Plot: Calibration in the context of machine learning refers to the alignment between the predicted probabilities from a classification model and the true probabilities of the predicted outcomes, ensuring that the model's confidence estimates are accurate [66], [67], [68] Calibration plot also known as reliability curve.

In a perfectly calibrated model, the reliability curve would resemble a 45-degree diagonal line from the bottom left to the top right, as illustrated in Fig. 2. This indicates that the predicted probabilities align precisely with the true probabilities. When a reliability curve closely follows this line, it suggests that the model is well-calibrated. However, if the reliability curve deviates above the perfect calibration line, it signifies that the model's predicted probabilities are too high. Conversely, if the curve falls below the line, it indicates that the predicted probabilities are too low. The choice of evaluation metrics depends on the specific problem and the desired performance characteristics. ROC curve and AUC are suitable for binary classification tasks, but precision, recall, F1 score, or specific domain-related metrics may be more appropriate in different contexts.

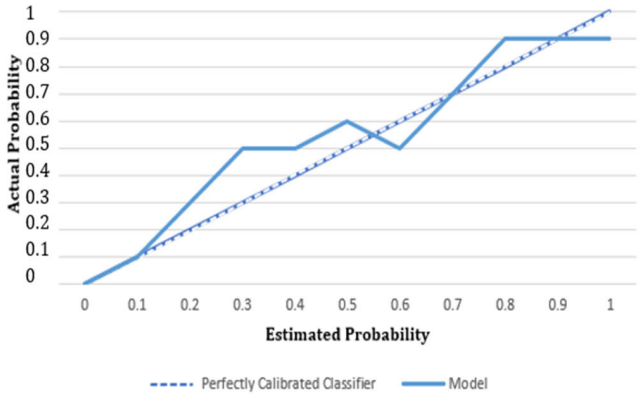


FIGURE 2. Calibration plot ensures the model's confidence.

C. DATASETS

In this study, four datasets containing IoT data have been used for the evaluation of machine learning classifiers. These inherently imbalanced datasets comprise records, feature counts, and the percentage distribution of attacks, as depicted in Table 1.

TABLE 1. Datasets used in the study.

ID	DataSet Name	# of Record	# of features	Attack%
DS1	UNSW_NB15	82332	49	45332
DS2	IoT-ID20	625783	86	585710
DS3	CICIDS2017	283076	80	55766
DS4	CICIoT2023	466868	47	455886

UNSW_NB15 [79], [80], [81], [82], [83]; The UNSW-NB 15 dataset, which has been extensively studied academically, was created by the IXIA PerfectStorm tool at the Australian Cyber Security Center's (ACCS) Cyber Range Lab. The dataset comprises normal traffic and nine attack types, including Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The dataset contains 2,540,044 records and 49 features. In the study, a Testing Dataset consisting of 82332 records was used.

IoT-ID20 [74]: IoT Network Intrusion Dataset consists of 625783 records and 86 features. The dataset includes four attack classes consisting of Mirai, Scan, DoS, and MITM ARP Spoofing, as well as one Normal class.

The distribution of the dataset is provided in Table 3. The IoTID-20 dataset is also not balanced; Mirai attacks comprise more than 66% of the dataset. CICIDS2017 [75];

TABLE 2. Distribution of classes in the DS1 dataset.

Type	Frequency
Analysis	2000
Backdoor	1746
DoS	12264
Exploits	33393
Fuzzers	18184
Generic	40000
Normal	56000
Reconn.	10491
Shellcode	1133
Worms	130

TABLE 3. Distribution of classes in the DS2 dataset.

Type	Frequency
Normal	40073
DoS	59391
Mirai	415677
MITM Arp Spoofing	35377
Scan	75265

The CIC2017 dataset provided by the Canadian Institute for Cybersecurity contains 80 features. The dataset consists of 2,830,743 records with 14 attack classes and one Benign class [76]. The distribution of these traffic types is provided in Table 4.

TABLE 4. Distribution of classes in the DS3 dataset.

Type	Frequency
BENIGN	2273097
DoS Hulk	231073
PortScan	158930
DDoS	128027
DoS GoldenEye	10293
FTP-Patator	7938
SSH-Patator	5897
DoS slowloris	5796
DoS Slowhttptest	5499
Bot	1966
Web Attack Brute Force	1507
Web Attack XSS	652
Infiltration	36
Web Attack Sql Injection	21
Heartbleed	11

Infiltration, Web Attack - SQL Injection, and Heartbleed have been removed from the dataset due to their very low frequencies. Additionally, the database has been reduced based on the frequency percentages of the traffic. In this study, a total of 283,076 records have been selected from the dataset based on the percentage distribution of traffic. CICIoT2023 [77]; The dataset comprises 46,686,579 records and 47 features. Within the dataset, 33 distinct attack traffic types have been consolidated into 7 categories, alongside a Benign traffic class. By scaling according to frequencies, 466,868 records have been selected from this dataset, creating a training dataset. The distribution of traffic in the dataset is presented in Table 5.

TABLE 5. Distribution of classes in the DS4 dataset.

Type	Frequency
DDoS-ICMP_Flood	7200504
DDoS-UDP_Flood	5412287
DDoS-TCP_Flood	4497667
DDoS-PSHACK_Flood	4094755
DDoS-SYN_Flood	4059190
DDoS-RSTFINFlood	4045285
DDoS-SynonymousIP_Flood	3598138
DoS-UDP_Flood	3318595
DoS-TCP_Flood	2671445
DoS-SYN_Flood	2028834
BenignTraffic	1098195
Mirai-greeth_flood	991866
Mirai-udpplain	890576
Mirai-greip_flood	751682
DDoS-ICMP_Fragmentation	452489
MITM-ArpSpoofing	307593
DDoS-UDP_Fragmentation	286925
DDoS-ACK_Fragmentation	285104
DNS_Spoofing	178911
Recon-HostDiscovery	134378
Recon-OSScan	98259
Recon-PortScan	82284
DoS-HTTP_Flood	71864
VulnerabilityScan	37382
DDoS-HTTP_Flood	28790
DDoS-SlowLoris	23426
DictionaryBruteForce	13064
BrowserHijacking	5859
CommandInjection	5409
SqlInjection	5245
XSS	3846
Backdoor_Malware	3218
Recon-PingSweep	2262
Uploading_Attack	1252

In the study, alongside imbalanced datasets, balanced training datasets were created where attack and benign traffic were equally distributed. Evaluations on these balanced datasets have also been taken into consideration.

IV. PERFORMANCE COMPARISONS

During the creation of a machine learning model, various pre-processing steps are undertaken. These processes significantly impact the performance of a well-constructed learning model. However, comprehending the extent of their influence on accuracy, despite being time-consuming, is crucial.

A. METHODOLOGY

In this study, the effects of Scaling, Normalization, Outlier Removal, Balancing, Feature Selection, and Regularization processes have been analyzed across various classifiers and different datasets. The impacts of machine learning classifiers on different datasets have been examined, and accuracy, precision, recall, and time values have been presented in tables. The performance metrics of each classifier on four different datasets were averaged, and the contribution of each process to the classifier's performance was investigated. Furthermore, the contribution of each process to the classifiers was calculated on an average basis, and its effectiveness on the four different datasets was evaluated. Both binary

classification and multi-class classification have been considered separately in the evaluations.

In the study, the performance of ML classifiers was evaluated by conducting assessments on both balanced and imbalanced datasets. This allowed for the measurement of the classifiers' effectiveness under different conditions.

B. HARDWARE AND ENVIRONMENT

In this study, hardware with 128 GB RAM, Intel(R) Xeon CPU 3.00 GHz, NVIDIA GeForce RTX 3090 features were used. The model was trained and evaluated with Python 3.8 and common libraries used for machine learning.

C. ATTACK DETECTION PERFORMANCE

In Machine Learning, binary classification can be employed to assess whether a traffic in IoT datasets constitutes an attack or not. In this section, the impact of pre-processing steps on the accuracy of identifying whether traffic in various datasets constitutes an attack has been examined. The results obtained without any preprocessing on balanced datasets are shown in Table 6. The Accuracy, Precision, Recall, and Learning Time for each classifier are provided in the table. The last row of the table contains the average values.

TABLE 6. Performance metrics of attack detection before pre-processing on balanced datasets.

Classifier	Accuracy	Precision	Recall	Training Time
		DS1		
SVC	68.04	67.78	66.08	13.43
DT	95.33	94.71	95.87	0.14
RF	97.37	97.88	96.75	2.11
GNB	68.18	62.47	88.42	0.04
AdaBoost	94.77	94.71	94.65	1.38
XGBoost	97.63	97.95	97.22	0.72
LR	76.48	82.74	65.92	0.13
Mean	*85.40*	*85.47*	*86.42*	*2.57*
		DS2		
SVC	69.07	64.13	82.20	16.35
DT	95.27	95.44	95.19	0.21
RF	95.50	94.70	96.51	1.6237
GNB	63.90	100	28.61	0.0449
AdaBoost	95.37	94.62	96.31	1.5284
XGBoost	96.20	95.70	96.84	0.6662
LR	83.30	84.65	81.81	0.3320
Mean	*85.51*	*89.89*	*82.50*	*2.97*
		DS3		
SVC	72.01	94.77	46.24	13.2266
DT	99.50	99.39	99.59	0.2575
RF	99.77	99.93	99.59	1.8670
GNB	71.67	80.70	55.98	0.0569
AdaBoost	99.20	99.19	99.19	2.4490
XGBoost	99.90	99.93	99.86	0.6702
LR	76.14	90.47	57.74	0.3421
Mean	*88.31*	*94.91*	*79.74*	*2.70*
		DS4		
SVC	99.20	100	98.38	0.3828
DT	99.33	99.22	99.48	0.1816
RF	99.57	99.93	99.22	2.3138
GNB	97.40	99.39	95.49	0.0459
AdaBoost	99.67	99.74	99.61	1.6459
XGBoost	99.63	99.80	99.48	0.6941
LR	89.70	86.23	94.97	0.1497
Mean	*97.79*	*97.76*	*98.09*	*0.77*

1) REMOVING OUTLIERS ON IMBALANCED DATASET

Outlier removing in ML refers to the process of identifying and eliminating data points that deviate significantly from the majority of the dataset, aiming to improve model performance and robustness [78]. The performance evaluations after removing outliers with a deviation of %1 is presented in Table 7.

TABLE 7. ML classifier performances after outlier removal.

Classifier	Accuracy	Precision	Recall	Training Time
		DS1		
SVC	69.61	71.03	68.53	13.2246
DT	95.37	94.80	96.22	0.1386
RF	96.63	96.92	96.48	1.9877
GNB	65.68	60.34	96.03	0.0389
AdaBoost	94.64	94.49	95.05	1.3175
XGBoost	97.23	97.64	96.94	0.6662
LR	76.71	83.76	67.56	0.1280
Mean	*85.12*	*85.57*	*88.12*	*2.50*
		DS2		
SVC	84.57	85.11	83.08	17.7965
DT	95.23	95.23	95.04	0.1845
RF	95.60	95.45	95.58	1.4500
GNB	65.40	100	29.48	0.0569
AdaBoost	95.47	95.07	95.72	1.4691
XGBoost	96.47	96.09	96.74	0.6766
LR	84.57	85.11	83.08	0.4126
Mean	*88.19*	*93.15*	*82.67*	*3.15*
		DS3		
SVC	72.87	96.21	47.49	13.4331
DT	99.50	99.12	99.86	0.2463
RF	99.83	100	99.66	1.7354
GNB	71.21	91.74	45.34	0.0469
AdaBoost	99.10	98.78	99.39	2.3757
XGBoost	99.83	99.86	99.80	0.6479
LR	78.55	91.47	62.01	0.3580
Mean	*88.7*	*96.74*	*79.08*	*2.69*
		DS4		
SVC	98.97	100	97.94	0.3431
DT	99.03	99.04	98.98	0.1755
RF	99.30	100	98.57	2.0804
GNB	96.83	98.38	95.08	0.0309
AdaBoost	99.23	99.52	98.91	1.7272
XGBoost	99.50	99.93	99.04	0.7141
LR	89.14	84.78	94.74	0.1606
Mean	*97.43*	*97.38*	*97.61*	*0.75*

An average basis analysis reveals an increase of 0.6% in Accuracy, 1.21% in Precision, and 0.2% in Recall. Nevertheless, in this scenario, the time has risen by 0.02 seconds. When evaluated on a classifier-specific basis, the outlier removal process has the most significant impact on the SVC with a contribution of 4.43%. When considering its effect on training time, it has resulted in an additional 0.35 seconds for the SVC.

By examining the values in Table 6 and Table 7 for other classifiers, various insights can be derived, and the contribution levels can be measured.

2) SCALING ON IMBALANCED DATASET

Scaling is one of the preprocessing steps applied in machine learning, and it affects the overall model performance [79].

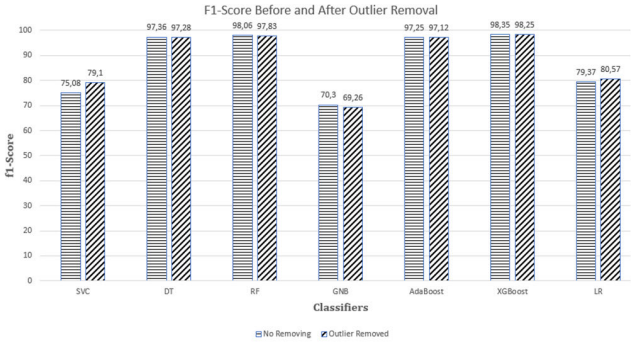


FIGURE 3. The change in accuracy after the outlier removal.

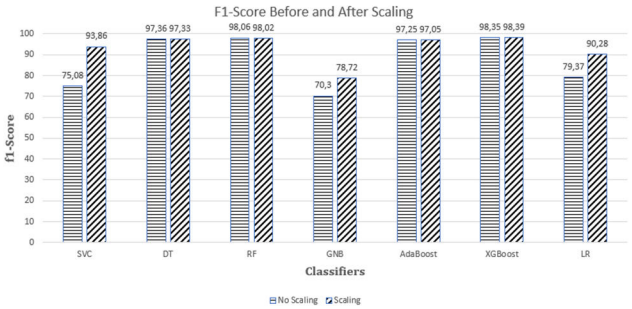


FIGURE 4. The change in accuracy after the scaling.

Table 8 displays the effects of min-max scaling on the performance of ML classifiers.

TABLE 8. Performance of ML classifiers after scaling.

Classifier	Accuracy	Precision	Recall	Training Time
		DS1		
SVC	88.37	87.41	89.79	4.92
DT	94.84	93.84	96.02	0.12
RF	96.60	96.87	96.35	1.98
GNB	77.91	83.35	70.03	0.03
AdaBoost	94.04	94.04	94.10	1.30
XGBoost	97.07	97.65	96.49	0.65
LR	84.34	84.74	83.95	0.25
Mean	*90.45*	*91.13*	*89.53*	*1.32*
		DS2		
SVC	90.37	91.05	90.04	6.65
DT	95.47	95.81	95.31	0.18
RF	95.97	95.97	96.16	1.48
GNB	74.70	99.12	51.04	0.05
AdaBoost	95.13	95.66	94.79	1.45
XGBoost	96.73	96.99	96.61	0.62
LR	87.43	91.30	83.40	0.36
Mean	*90.83*	*95.13*	*86.76*	*1.54*
		DS3		
SVC	97.10	97.13	97.06	2.74
DT	99.53	99.67	99.40	0.23
RF	99.77	99.93	99.60	1.76
GNB	82.15	74.62	97.46	0.05
AdaBoost	99.17	99.13	99.20	2.19
XGBoost	99.90	100	99.80	0.61
LR	91.32	89.79	93.26	0.34
Mean	*95.56*	*94.32*	*97.97*	*1.13*
		DS4		
SVC	99.23	99.93	98.52	0.77
DT	99.30	99.06	99.53	0.17
RF	99.67	99.93	99.39	1.88
GNB	88.50	99.91	76.85	0.04
AdaBoost	99.73	99.87	99.60	1.43
XGBoost	99.80	100	99.60	0.66
LR	98.17	99.65	96.64	0.33
Mean	*97.77*	*99.76*	*95.73*	*0.75*

When analyzing the accuracy, the impact of the scaling process is most noticeable in the SVC classifier. For instance, the accuracy for DS1, which was previously 68%, increased to 88.37% after scaling.

On average, across the four datasets, a 22% increase in accuracy is observed. However, the effect of scaling on SVC is not only an increase in accuracy but also a significant reduction in training time. Looking at the average values for the four datasets, the initial time of 10.85 seconds has decreased to 3.77 seconds. Considering the overall average of all classifiers across all datasets, the overall impact of scaling on accuracy has risen from 89.25% to 93.65%. In terms of training time, it has decreased from 2.25 seconds to 1.75 seconds. This implies an approximate reduction of 22%. In terms of training time, in the assessment conducted, the fastest classifier on average is the GNB with 0.04 seconds, followed by the DT classifier with 0.18 seconds. However, despite this, GNB's accuracy value is 80.82%, whereas DT's accuracy is 97.29%. In cases where training time is considered insignificant, the highest accuracy of 98.38% belongs to the XGBoost classifier. To assess the impact of scaling on the other classifier, Table-6 and Table-8 values can be compared.

3) FEATURE SELECTION

Feature selection is the process of choosing a subset of relevant and significant features from the original set of variables to enhance model performance and reduce complexity [80]. In this section, the impact of the Analysis of Variance (ANOVA) feature selection technique on accuracy has been evaluated. ANOVA is a statistical technique used to identify the most relevant features by measuring the variation in the target variable across different categories of each feature [81], [82]. Additionally, the best parameter selection, which best reflects the classifiers' performance, has also been conducted. The classifiers' accuracies are presented in Table 9 using the best parameters of the classifiers along with the top 10 features on balanced datasets.

TABLE 9. Performance of classifiers after best parameter and top-10 feature in balanced dataset.

Classifier	Accuracy				
	DS1	DS2	DS3	DS4	Mean
SVC	92.69	95.35	89.06	98.9	*94*
DT	93.46	97.08	97.1	98.92	*96.64*
RF	93.97	96.82	97.17	99.06	*96.76*
GNB	73.38	68.94	83.57	95.95	*80.46*
AdaBoost	91.08	96.28	95.75	98.87	*95.5*
XGBoost	94.4	97.31	97.23	99.13	*97.02*
LR	92.69	94.72	88.54	98.4	*93.59*
Mean	*90.24*	*92.36*	*92.63*	*98.46*	*93.42*

When comparing accuracy based on the best parameters, an increase is observed in almost all classifiers compared to the initial values (Table.6). The highest increase of 21.95% is achieved by the SVC, followed by a 14.97% increase in the Logistic Regression classifier. On average, considering the usage of best parameters and the top 10 features, accuracy has been enhanced by 4.7%, resulting in an accuracy rate of 93.42%. In the AdaBoost classifier, a decrease of 1.80% in accuracy has been observed. Overall, when evaluated, the XGBoost classifier has demonstrated the highest successful accuracy of 97.02%. The effects of the feature selection process on accuracy on average are presented in the graph shown in Fig. 5.

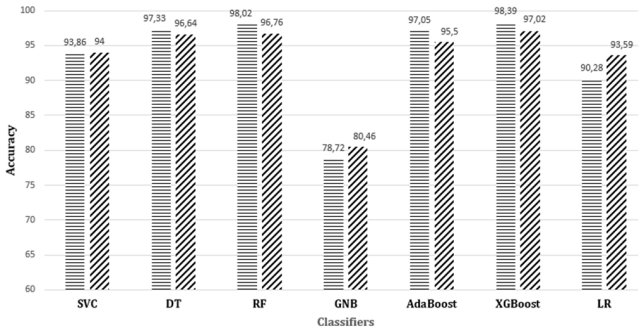


FIGURE 5. The change in accuracy after feature selection.

In the examination conducted on a percentage basis, the highest increase is observed in the LR classifier with 3.67%. In the Adaboost classifier, there is a decrease of 1.6% in accuracy.

TABLE 10. Performance of classifiers after best parameter and top-10 feature in imbalanced datasets.

Classifier	Accuracy				
	DS1	DS2	DS3	DS4	Mean
SVC	87.50	95.35	89.06	98.90	92.71
DT	90	97.08	97.10	98.92	95.78
RF	89.94	96.82	97.17	99.06	95.75
GNB	79.17	68.94	83.57	95.95	81.91
AdaBoost	87.97	96.28	95.75	98.87	94.72
XGBoost	91.18	97.31	97.23	99.13	96.21
LR	76.28	94.72	88.54	98.40	89.49
Mean	*86.01*	*92.36*	*92.63*	*98.46*	*92.37*

In imbalanced datasets, there is an average decrease of approximately 1% in accuracy performance. In other words, if the dataset is balanced, a 1% increase in accuracy is observed. When the dataset is balanced, the highest performance improvement of about 4% is achieved with LR. The number of features in a dataset is an important consideration and can significantly impact the performance, efficiency, and interpretability of a model. The effectiveness of an intrusion detection classifier is contingent on the quantity of features in the detection and classification [83]. For DS1, the accuracy obtained by different classifiers in this context are given in the graph in Fig.6.

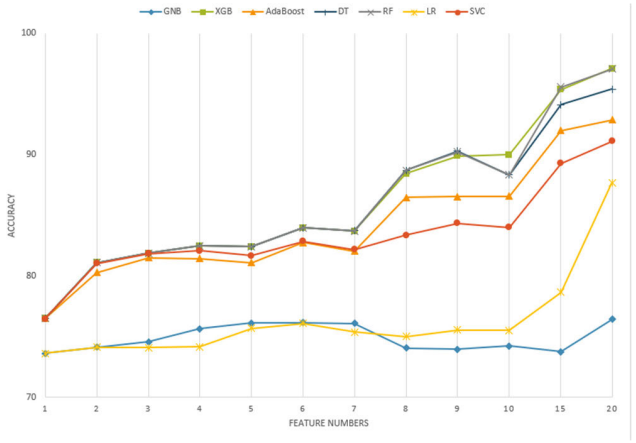


FIGURE 6. Accuracy based on the number of features for DS1.

While the number of features in the GNB classifier has increased, no significant difference in accuracy has been observed. However, both XGB and RF have demonstrated a substantial increase in accuracy. To ascertain if these findings differ across datasets, the same test was conducted on four different datasets, and the average F1-score values are illustrated in Fig.7.

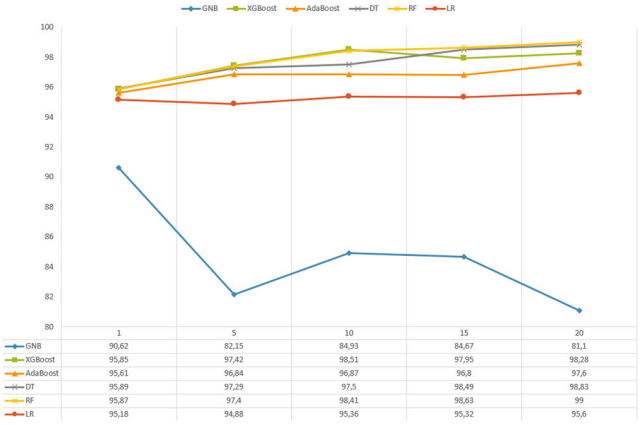


FIGURE 7. F1-score performance of classifiers according to the average values of four different data sets.

It has been observed that as the number of features increases in all classifiers except GNB, the accuracy value increases. The highest accuracy values were observed in the DT and RF classifiers.

4) TRAINING TIME GRAPH

In the study, the training time values of various classifiers depending on the number of features are presented in the graph shown in Fig.8.

As depicted in the graph, GNB and DT classifiers demonstrate no significant increase in training times as the number of features increases. However, in contrast, RF and AdaBoost classifiers show a substantial increase. This increase in training times may lead to delays in the model's performance.

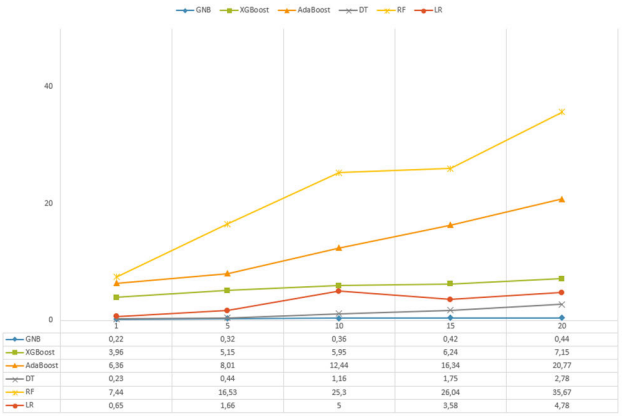


FIGURE 8. Variation of training time based on feature count.

5) ROC CURVE ON IMBALANCED DATASET

AUC indicates better discrimination power and overall model performance. In this section, AUC values have been calculated for the top 10 features and best parameters. The AUC values of the classifiers in imbalanced datasets are presented in Table 11.

TABLE 11. AUC values of classifiers in imbalanced datasets.

Classifier	Area Under Curve (AUC)				
	DS1	DS2	DS3	DS4	Mean
SVC	0.9449	0.9165	0.9478	0.9935	0.9507
DT	0.8976	0.9783	0.9873	0.8731	0.9341
RF	0.9747	0.9818	0.9902	0.9969	0.9859
GNB	0.8612	0.8603	0.781	0.9862	0.8722
AdaBoost	0.9588	0.9686	0.9797	0.9952	0.9756
XGBoost	0.9748	0.9845	0.9918	0.9971	0.9871
LR	0.8839	0.8996	0.914	0.9921	0.9224

From an AUC perspective, on average across the four different datasets evaluated, the XGBoost algorithm achieved the highest performance, followed by the RF classifier. GNB, on the other hand, exhibited the poorest performance with 87.22%.

6) AUC FOR BALANCED DATASETS

According to the ANOVA analysis, the performance achieved in the DS4 dataset using the top 10 selected attributes and the best parameters is depicted in the AUC curves graph in Fig.9.

AUC values for the DS1, DS2, DS3 and DS4 datasets are provided in Table 12. The obtained values in the study have been detailed in the tables located in the Appendix.

In the evaluation conducted on an average basis, the classifier with the highest AUC value is XGBoost at 0.9842. followed closely by the Random Forest classifier at 0.9823. The lowest AUC value is observed with the GNB classifier at 0.8874.

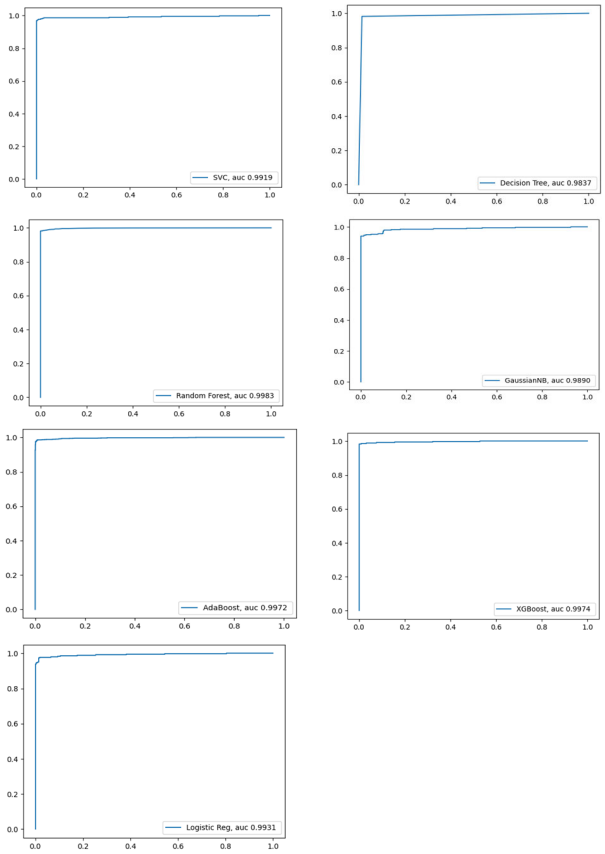


FIGURE 9. The learning curves and AUC values of classifiers in Balanced DS4.

TABLE 12. The AUC values of classifiers in the four datasets.

Classifier	Area Under Curve (AUC)				
	DS1	DS2	DS3	DS4	Mean
SVC	0.9681	0.9367	0.9412	0.9919	0.9675
DT	0.9707	0.9464	0.9784	0.9837	0.9698
RF	0.9856	0.9637	0.9814	0.9983	0.9823
GNB	0.8673	0.9105	0.7827	0.989	0.8874
AdaBoost	0.9737	0.9639	0.9613	0.9972	0.9741
XGBoost	0.9889	0.9694	0.981	0.9974	0.9842
LR	0.9049	0.9488	0.8795	0.9931	0.9316

7) RELIABILITY CURVE

In this section, the reliability curves of machine learning classification algorithms have been evaluated. These curves for all four datasets are shown in the graphs between 2a and 2d included in the Appendix. Fig.10 displays the reliability curve graph for the DS1 dataset.

The reliability curve of classifiers might take an S-shape. This suggests that the model is either overconfident or underconfident. Overconfidence means that the model's predicted probabilities for the positive class are too extreme, whereas under confidence means the model's predicted probabilities are too conservative. It has been observed that the most reliable classifiers for the DS1 dataset are XGBoost and Logistic

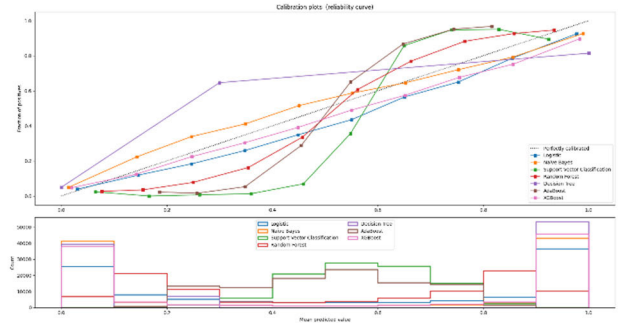


FIGURE 10. Reliability curves of classifiers.

Regression classifiers. The worst performance is exhibited by classifiers showing an S-shape, namely SVC, Random Forest, and Adaboost classifiers.

8) MULTI CLASSIFICATION ACCURACY

In the evaluation of multi-classification performance, four separate datasets have been considered. In this section, the evaluation results for DS3 have been detailed in tabular form. Interpretation for the other datasets has been made based on averages. Additionally, for the DS2 dataset, the top 10 features selected according to ANOVA were determined, and the multi-classification performance of the classifiers was evaluated. Accuracy was calculated here using the one-to-many approach. The confusion matrices (CM) for the evaluation are provided in Table 13 - Table 20. Underneath the classifier names in the tables, the overall accuracy values are displayed.

TABLE 13. Confusion matrix for KNN.

Acc.= 0.7917		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14741	0	18	1	0
	MITM	0	2318	5310	4	1221
	Mirai	0	1748	96643	518	5148
	Benign	7	46	3233	6341	224
	Scan	0	859	14221	6	3747

KNN achieved an overall accuracy of 79.17%. The highest correct classification rate was 99.98% for DoS attacks, followed by 94.36% for MITM ARP attacks. The lowest success rate, on the other hand, was achieved at 80.69% for classifying Mirai traffic.

TABLE 14. Confusion matrix for GNB.

Acc.= 0.6253		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14712	0	45	3	0
	MITM	0	4	1805	490	6554
	Mirai	69	36	61212	6914	35826
	Benign	5	0	1035	4557	4254
	Scan	4	2	696	850	17281

For GNB, the highest correct classification rate was 99.92% for DoS attacks. It also achieved a 94.62% correct detection rate for MITM ARP attacks. The lowest performance, with only a 69.48% correct classification rate, was observed in Scan attacks. The overall accuracy value was measured at 62.53%.

TABLE 15. Confusion matrix for XGB.

Acc.= 0.8088		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14751	0	9	0	0
	MITM	0	1445	7269	2	137
	Mirai	0	38	102600	627	792
	Benign	0	1	3354	6427	69
	Scan	0	4	17592	0	1237

For the XGB classifier, the highest classification success rate was 99.99% for DoS attacks. It was followed by a 95.45% detection rate for MITM ARP attacks. The least successful classification, with only 81.02% accuracy rate, was observed for Mirai traffic. The overall accuracy value was measured at 80.88%.

TABLE 16. Confusion matrix for AdaBoost.

Acc.= 0.7430		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14743	0	16	1	0
	MITM	0	1	8076	0	776
	Mirai	2	1	94900	3749	5405
	Benign	1	0	4846	4717	287
	Scan	0	0	17018	0	1815

The Adaboost classifier achieved an overall accuracy of 74.30%. The highest correct classification rate was 99.99% for DoS attacks, whereas the lowest performance, with only a 74.98% correct classification rate, was observed in Mirai attacks.

TABLE 17. Confusion matrix for LR.

Acc.= 0.7664		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14636	0	56	51	17
	MITM	0	1	7388	601	863
	Mirai	6	3	10356	197	495
	Benign	1829	0	5938	1826	258
	Scan	2	0	18827	0	4

When examining the classification performance of Linear Regression, it achieved an overall accuracy of 76.64%. Although it showed a high accuracy of 96.91% in detecting DoS attacks, it performed poorly in detecting Mirai and Scan attacks. The accuracy values for both attacks were measured at 78.95% and 75.10%, respectively.

The RF classifier achieved an overall accuracy of 81.23%. It obtained accuracy rates of 99.99% for DoS attacks and

TABLE 18. Confusion matrix for RF.

Acc.= 0.8123		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14751	1	8	0	0
	MITM	0	1452	7154	5	242
	Mirai	0	54	102572	268	1163
	Benign	1	3	3111	6637	99
	Scan	0	16	17221	5	1591

95.44% for MITM ARP attacks. The least successful classification was for Mirai traffic, with an accuracy rate of 81.47%.

TABLE 19. Confusion matrix for DT.

Acc. = 0.8123		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14749	1	9	1	0
	MITM	0	1463	7158	5	227
	Mirai	0	73	102624	233	1127
	Benign	1	4	3126	6622	98
	Scan	0	27	17246	10	1550

When examining the performance of the Decision Tree classifier, it achieved an overall accuracy of 76.40% as shown in Table 20. It demonstrated high accuracy in detecting DoS attacks with 99.99%. However, the detection accuracy for Mirai traffic was measured at a lower 81.47%.

TABLE 20. Confusion matrix for SVC.

Acc.= 0.7640		Predicted Class				
		DoS	MITM	Mirai	Benign	Scan
Actual Class	DoS	14695	19	117	0	0
	MITM	2	1441	7459	0	0
	Mirai	46	209	103317	0	0
	Benign	3	6	10120	0	0
	Scan	4	16	18900	0	0

The SVC classifier achieved its highest performance in the detection of DoS attacks with an accuracy of 99.88%, followed by a 95.29% accuracy in detecting MITM ARP attacks. However, the classifier did not detect any traffic as Benign or Scan, resulting in a 0 True Positive for these classes. The overall accuracy value was calculated as 76.40%. A table specifying which types of traffic were most accurately detected by which classifier for this dataset is provided (Table 21).

TABLE 21. Classifier that detects the type of traffic with the best accuracy.

Class	The Best Classifier	Accuracy %
DoS	XGB & RF &DT	99.99
MITM	XGB	95.45
Mirai	RF, DT	81.47
Benign	DT	97.82
Scan	XGB	89.31

The most successful classification achieved by classifiers has been in DoS attacks. XGB, RF, and DT, which demonstrate ensemble learning methods, have detected DoS attacks

TABLE 22. Classifiers achieving the best f1-Score for attacks in the DS1.

Class	The Best Classifier	F1-Score %
Analysis	KNN	12.54
Backdoor	KNN	7.76
DoS	XGB	43.83
Exploits	RF	59.87
Fuzzers	XGB	65.06
Generic	RF	96.32
Benign	RF	94.26
Recon.	XGB	40.07
Shellcode	GNB	3.93
Worms	None	-

TABLE 23. Classifiers achieving the best f1-Score for attacks in the DS2.

Class	The Best Classifier	F1-Score %
DoS	XGB	99.97
MITM ARP	KNN	33.54
Mirai	DT	87.63
Benign	DT	79.20
Scan	GNB	41.77

TABLE 24. Classifiers achieving the best f1-Score for attacks in the DS3.

Class	The Best Classifier	Accuracy
Benign	XGB	99.69%
Bot	XGB	48.72%
DDoS	XGB	99.50%
DDoS- GoldenEye	XGB	96.86 %
Hulk	XGB	72.73%
Slowhttptest	RF	96.95%
slowloris	XGB. RF.DT	98.60%
FTP-Patator	KNN. RF.GNB.DT	100%
PortScan	DT	99.65%
SSH-Patator	GNB	44.36%
Brute Force	GNB	73.33%
XSS	DT.RF	10.00%

TABLE 25. Best accuracy and classifiers for the attacks in the DS4.

Traffic	Classifier	Best Accuracy (%)
Benign	SVM	92.69
Brute Force	Adaboost	3.23
DDoS	LR	99.77
DoS	RF	99.3
Mirai	XGBoost	99.36
Recon	GNB	70.23
Spoof	XGBoost	57.12
Web	DT	0.08

with a high accuracy rate of 99.99%. The lowest accuracy was experienced in detecting Mirai attacks. RF and DT achieved an accuracy rate of 81.47%. It can be concluded that ensemble learning algorithms show success for this dataset. The confusion matrices for all datasets are provided in detail in the Appendix. In the evaluation for DS1, KNN achieved a successful result with an accuracy of 79.45%. However, the KNN classifier relatively underperformed in classifying various attacks. The highest correct classification was 94.53% for "Generic" attacks, followed by 94.35% for Benign traffic. The least successful classification was with 0% accuracy for ShellCode and Worms. The GNB classifier achieved

TABLE 26. ROC curves according to Top 10 features in balanced datasets.

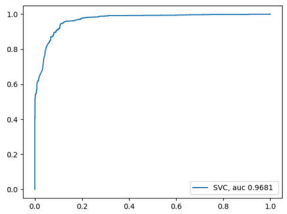
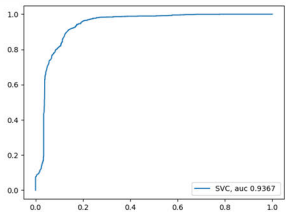
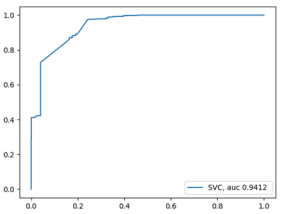
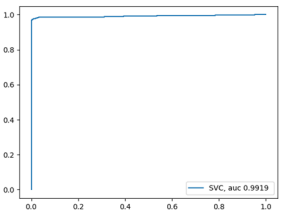
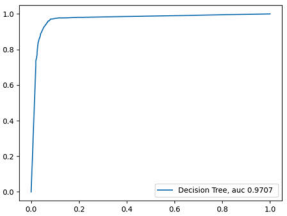
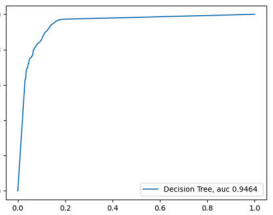
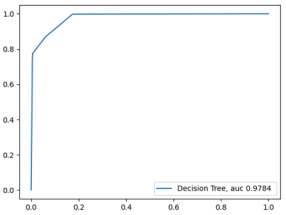
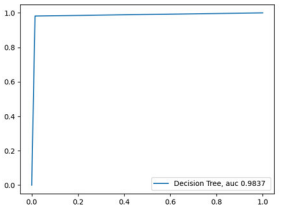
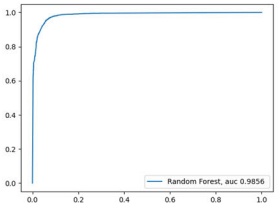
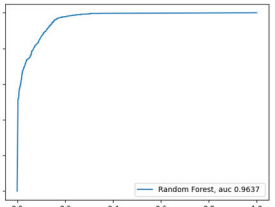
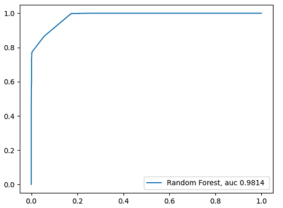
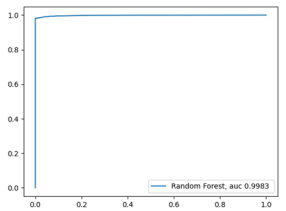
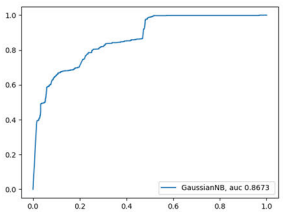
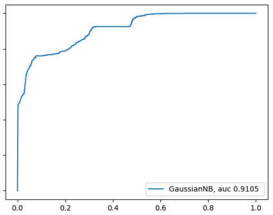
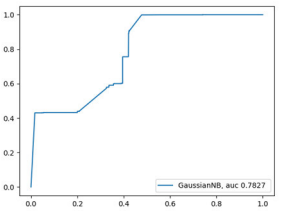
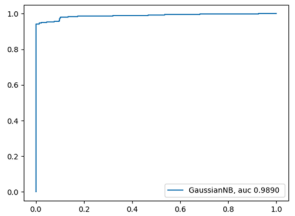
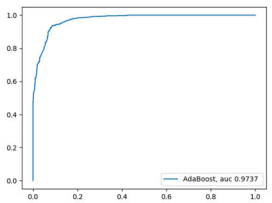
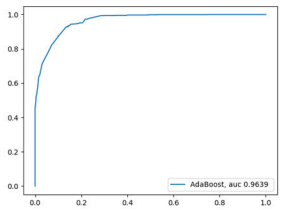
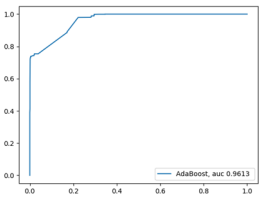
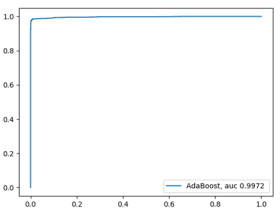
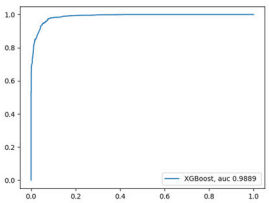
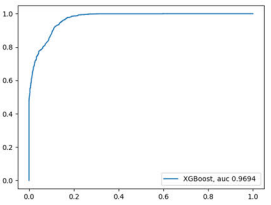
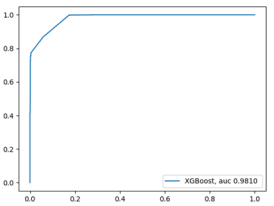
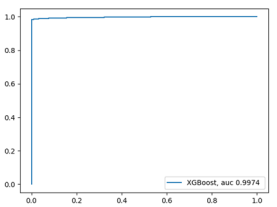
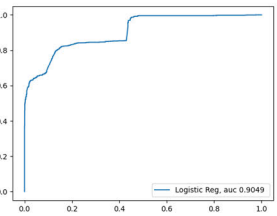
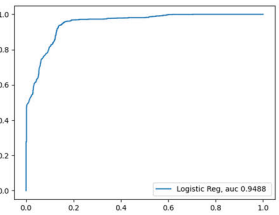
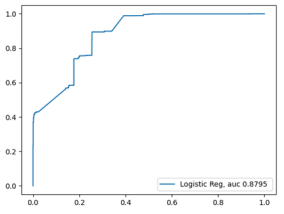
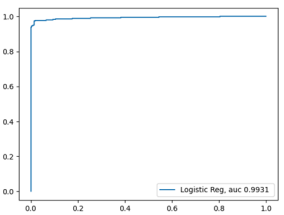
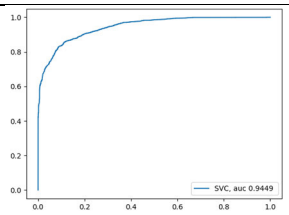
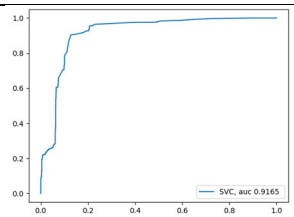
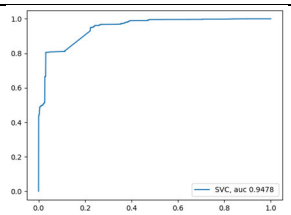
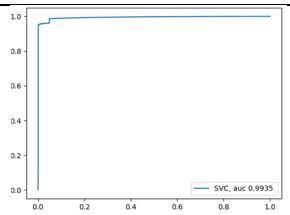
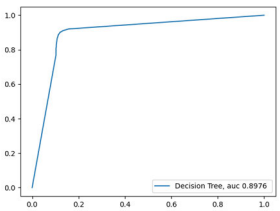
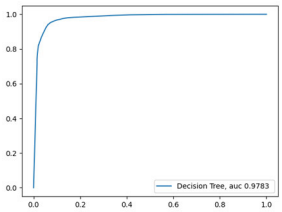
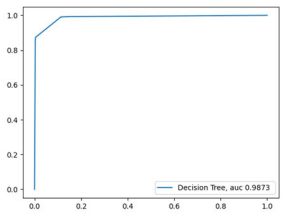
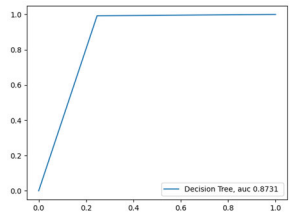
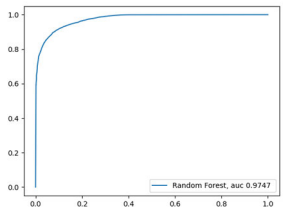
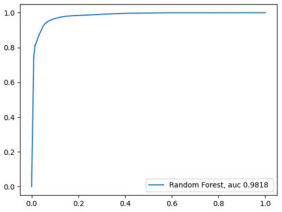
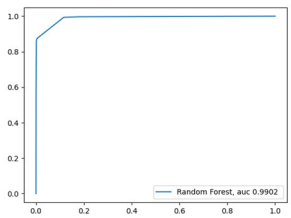
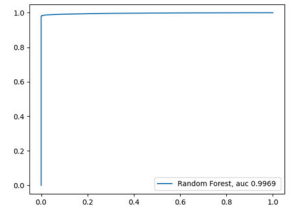
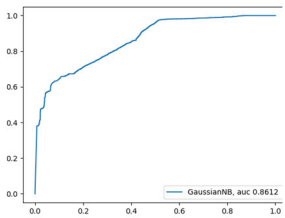
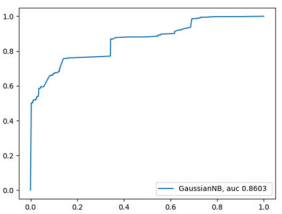
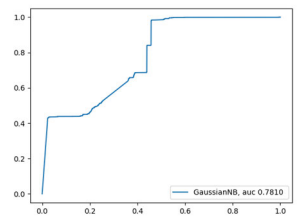
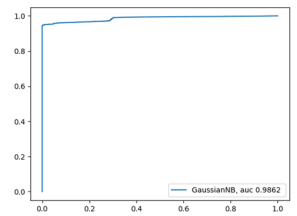
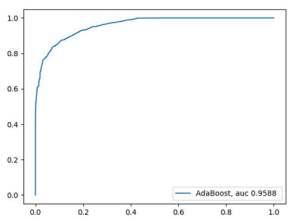
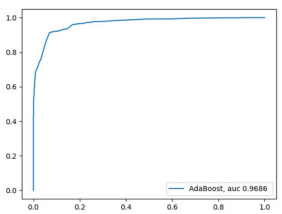
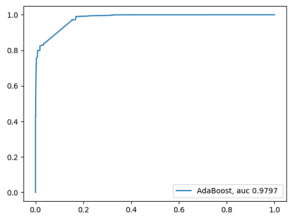
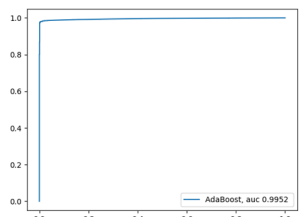
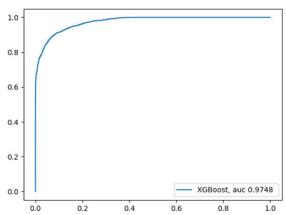
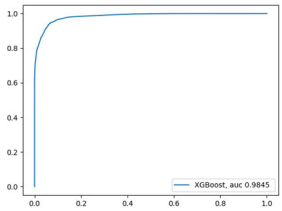
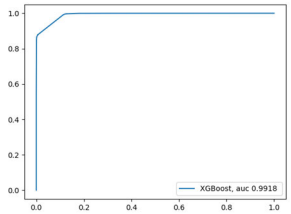
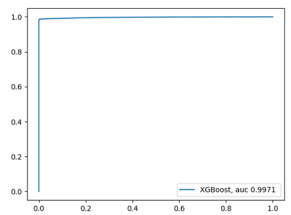
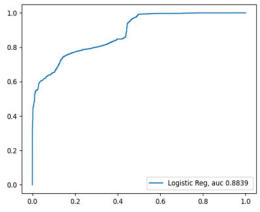
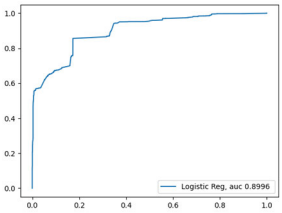
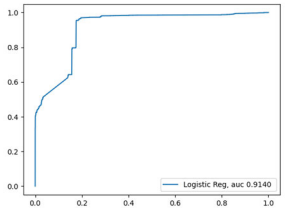
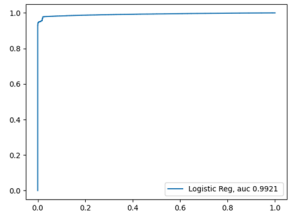
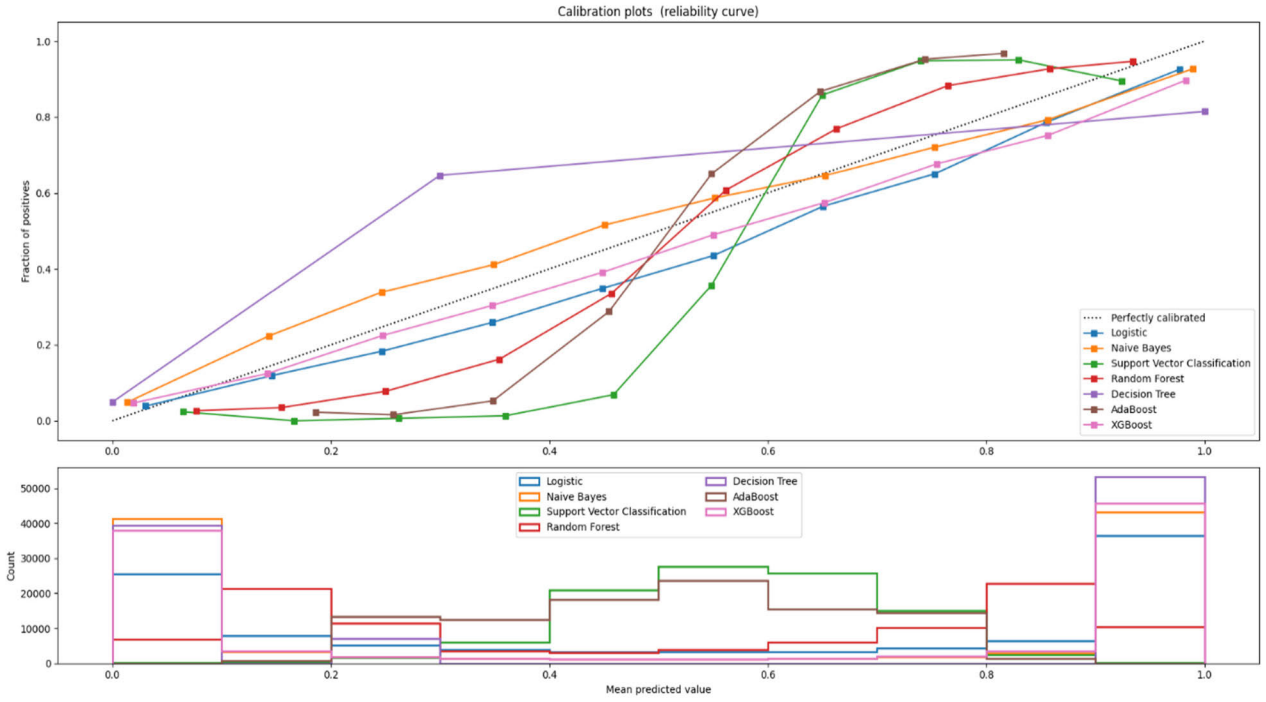
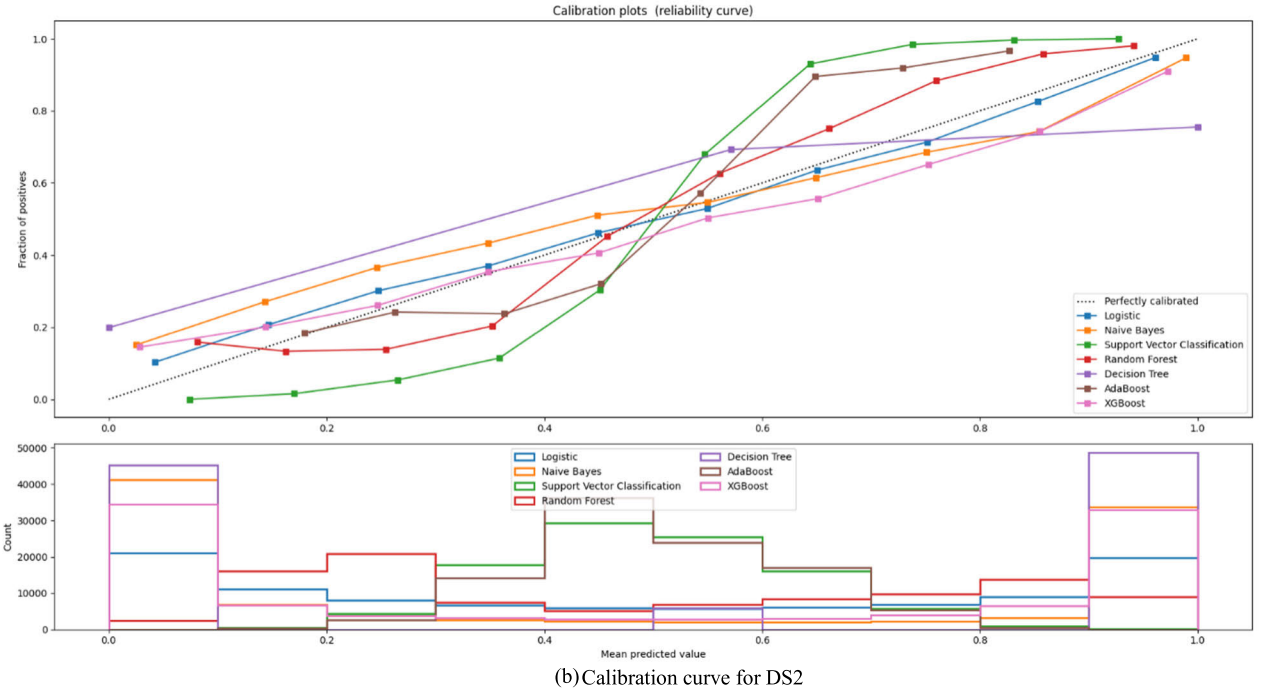
CLS	DS1	DS2	DS3	DS4
SVC				
DT				
RF				
GNB				
AB				
XG				
LR				

TABLE 27. ROC curves according to Top 10 features in unbalanced datasets.

CLS	DS1	DS2	DS3	DS4
SVC				
DT				
RF				
GNB				
AB				
XG				
LR				



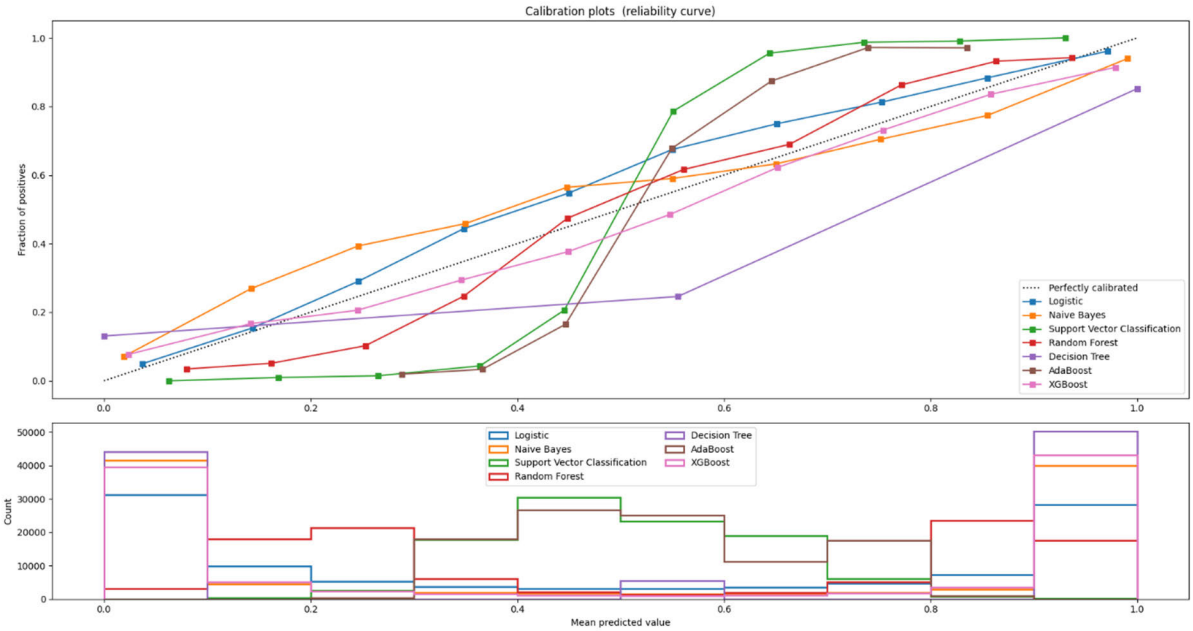
(a) Calibration curve for DS1



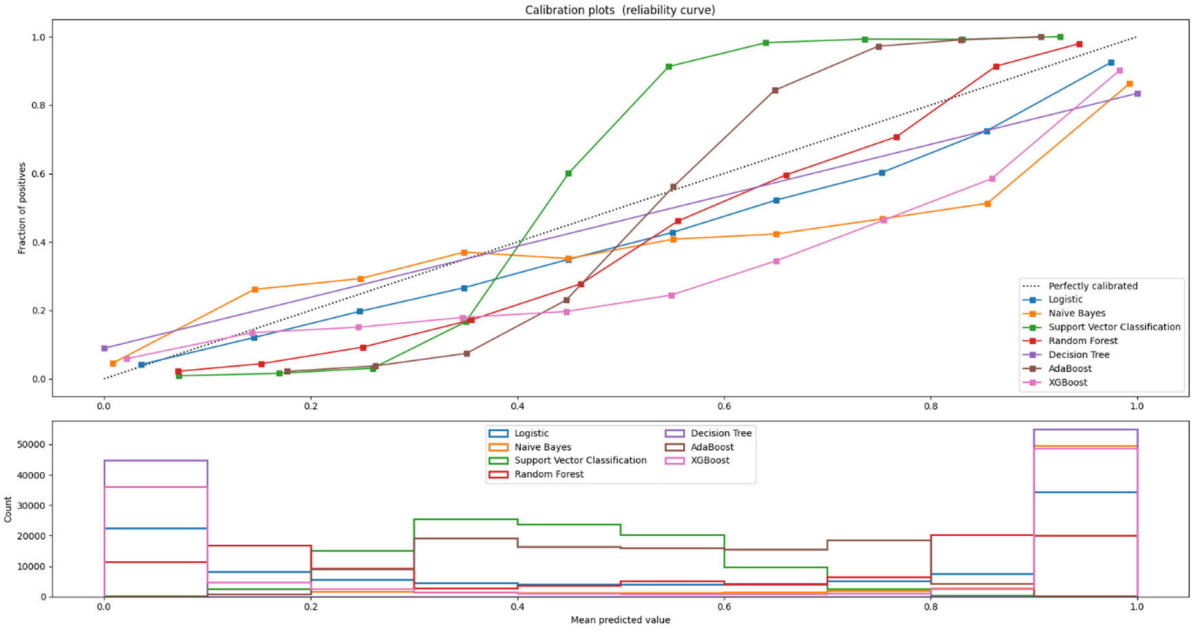
(b) Calibration curve for DS2

FIGURE 11. Calibration Curve charts for four unbalanced datasets.

its highest classification accuracy of 100% for ShellCode attacks. The second-highest classification accuracy was 58% for Generic attacks. The accuracy for Benign traffic was 55%. The least successful classification was 0% for the Worms category. The overall accuracy was measured at 41%. XGB achieved a 95% accuracy in classifying Generic attacks and 94% accuracy in classifying Benign traffic. However, it had 0% accuracy for Backdoor, ShellCode, and Worms. XGB's overall accuracy was measured at 82%. For Adaboost, the highest accuracy value was 85% for Generic traffic. Similar to XGB, it had 0% accuracy for Backdoor, ShellCode, and Worms. Adaboost achieved an overall accuracy of 63%.



(c) Calibration curve for DS3



(d) Calibration curve for DS4

FIGURE 11. ***(Continued.)*** **Calibration Curve charts for four unbalanced datasets.**

When looking at the CM matrix for the LR classifier, detection accuracy for Analysis, Backdoor, ShellCode, and Worms was 0%. The highest accuracy was 91% for Generic traffic. LR's overall accuracy was measured at 63%. The Random Forest classifier also achieved 0% accuracy in classifying BackDoor, ShellCode, and Worms attacks. However, it correctly classified Generic and Benign traffic with 95% and 94% accuracy, respectively. RF's overall accuracy was 81%. According to the DT classifier, Worm and Backdoor detection could not be performed. However, it correctly classified Generic and Benign traffic with 95% and 93% accuracy, respectively. Overall, the accuracy value was measured at 81%. Accordingly, correct classification for Analysis, Backdoor, Recon, ShellCode, and Worms is 0%. However, Benign traffic was correctly classified with 99% accuracy. In total, the accuracy value was measured at 67%. Accordingly, it has been observed that the SVC classifier can accurately detect normal traffic with high accuracy, but it struggles to detect certain types of traffic. Therefore, if a model is to be developed for detecting traffic with 0% accuracy, it may be more

TABLE 28. Confusion matrix (CM) tables of classifiers for DS1 dataset when all features and default parameters are used.

(a)

SUPPORT VECTOR CLASSIFIER (SVC)

SVC 0.7516		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	0	0	0	73	43	0	47	0	0	0
	Backdoor	0	0	0	63	56	0	32	2	0	0
	DoS	0	0	17	436	57	15	444	16	0	0
	Exploits	0	0	29	1671	119	22	973	14	0	0
	Fuzzers	0	0	3	197	230	2	1081	2	0	0
	Generic	0	0	0	84	1	4541	67	2	0	0
	Benign	0	0	0	223	16	0	9010	2	0	0
	Recon.	0	0	3	161	2	4	716	2	0	0
	Shellcode	0	0	0	0	0	0	95	0	0	0
	Worms	0	0	0	4	0	2	4	0	0	0

(b)

GAUSSIAN NAIVE BAYES (GNB) CLASSIFIER

GNB 0.3614		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	141	2	0	13	0	0	2	0	5	0
	Backdoor	132	5	0	3	2	0	0	0	9	2
	DoS	599	43	18	125	28	0	4	19	103	46
	Exploits	644	76	32	1341	66	0	52	56	270	291
	Fuzzers	287	62	1	27	204	38	3	284	541	68
	Generic	8	1829	7	47	6	2696	10	11	41	40
	Benign	43	470	490	2157	450	209	2899	581	1899	53
	Recon.	80	11	1	4	10	0	0	35	647	100
	Shellcode	1	0	0	0	0	0	0	0	94	0
	Worms	0	1	0	1	0	0	0	0	3	5

(c)

XGBOOST CLASSIFIER

XGB 0.8870		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	13	0	43	48	59	0	0	0	0	0
	Backdoor	0	3	8	61	77	1	2	0	1	0
	DoS	0	0	592	271	82	4	23	4	9	0
	Exploits	0	1	471	2045	167	23	69	35	15	2
	Fuzzers	0	1	67	138	1098	1	207	2	1	0
	Generic	0	0	23	66	9	4588	8	0	1	0
	Benign	0	0	5	19	66	1	9145	4	11	0
	Recon.	0	1	75	87	7	0	5	710	3	0
	Shellcode	0	0	1	9	2	1	21	0	61	0
	Worms	0	0	0	6	0	1	0	0	0	3

(d)

ADABOOST CLASSIFIER

AdaBoost 0.4512		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	148	0	0	12	1	0	2	0	0	0
	Backdoor	133	0	0	3	1	0	1	15	0	0
	DoS	652	0	0	160	1	0	48	124	0	0
	Exploits	860	0	0	1266	4	6	312	380	0	0
	Fuzzers	381	0	0	81	99	0	402	552	0	0
	Generic	3438	0	0	118	380	668	47	44	0	0
	Benign	2146	0	0	128	15	1	6479	482	0	0
	Recon.	85	0	0	114	3	0	59	627	0	0
	Shellcode	0	0	0	0	1	0	9	85	0	0
	Worms	1	0	0	5	1	0	0	3	0	0

TABLE 28. *(Continued.)* **Confusion matrix (CM) tables of classifiers for DS1 dataset when all features and default parameters are used.**

(e)
LOGISTIC REGRESSION (LR) CLASSIFIER

LR 0.7693		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	0	0	11	41	44	36	29	2	0	0
	Backdoor	0	0	4	36	51	44	14	4	0	0
	DoS	0	0	191	314	57	104	284	35	0	0
	Exploits	0	0	155	1581	121	148	788	35	0	0
	Fuzzers	0	0	23	129	376	95	854	38	0	0
	Generic	0	0	4	82	4	4546	56	3	0	0
	Benign	0	0	6	247	19	11	8861	107	0	0
	Recon.	0	0	26	106	10	12	454	280	0	0
	Shellcode	0	0	0	0	0	0	74	21	0	0
	Worms	0	0	0	2	0	3	5	0	0	0

(f)
RANDOM FOREST (RF) CLASSIFIER

RF 0.8786		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	13	0	31	65	54	0	0	0	0	0
	Backdoor	0	1	8	80	59	0	4	0	1	0
	DoS	0	0	445	415	74	2	34	3	12	0
	Exploits	0	3	420	2084	193	12	79	26	11	0
	Fuzzers	0	3	51	193	1058	0	208	1	1	0
	Generic	0	0	20	77	8	4578	10	0	1	1
	Benign	0	0	2	23	68	0	9149	1	8	0
	Recon.	0	1	58	108	5	0	12	701	3	0
	Shellcode	0	0	0	6	3	1	29	2	54	0
	Worms	0	0	0	8	0	0	1	0	0	1

(g)
DECISION TREE (DT) CLASSIFIER

DT 0.8549		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	13	4	34	75	35	0	2	0	0	0
	Backdoor	2	4	9	94	39	1	2	0	2	0
	DoS	2	3	486	390	59	14	15	7	9	0
	Exploits	19	12	581	1854	166	48	71	58	13	6
	Fuzzers	19	11	63	199	999	5	209	4	5	1
	Generic	0	2	27	43	10	4600	10	2	1	0
	Benign	0	0	31	75	215	12	8884	18	14	2
	Recon.	0	4	73	78	9	3	11	706	4	0
	Shellcode	0	2	4	16	3	2	20	1	46	1
	Worms	0	0	1	3	1	1	0	0	0	4

(h)
K-NEIGHBOR NEAREST (KNN) CLASSIFIER

KNN 0.7670		Predicted Class									
		Analysis	Backdoor	DoS	Exploits	Fuzzers	Generic	Benign	Recon.	Shellcode	Worms
Actual Class	Analysis	41	21	42	27	17	8	7	0	0	0
	Backdoor	48	16	27	20	28	11	2	1	0	0
	DoS	114	23	376	339	41	16	54	22	0	0
	Exploits	139	51	514	1670	96	39	240	79	0	0
	Fuzzers	86	29	103	144	732	38	355	27	1	0
	Generic	17	8	54	78	39	4447	36	16	0	0
	Benign	7	11	157	413	193	27	8352	88	3	0
	Recon.	11	8	236	225	70	33	151	151	3	0
	Shellcode	0	1	21	19	10	4	18	20	2	0
	Worms	0	0	0	1	1	0	6	2	0	0

TABLE 29. Confusion matrix (CM) tables of classifiers for DS2 dataset when all features and default parameters are used.

(a)
SUPPORT VECTOR CLASSIFIER (SVC)

SVC : 0.7820		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14709	18	91	13	0
	MITM ARP Sp.	2	1441	6297	1	1161
	Mirai	9	209	100470	241	2643
	Benign	2	6	7670	2195	256
	Scan	4	16	15308	141	3451

(b)
GAUSSIAN NAIVE BAYES (GNB) CLASSIFIER

GNB 0.6195		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14790	7	29	5	0
	MITM ARP Sp.	1	7300	271	183	1147
	Mirai	9	27951	62394	9147	4071
	Benign	6	1112	541	8161	309
	Scan	2	12797	1598	315	4208

(c)
XGBOOST CLASSIFIER

XGB 0.8741		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14817	4	7	3	0
	MITM ARP Sp.	1	3621	2918	33	2329
	Mirai	0	828	93224	180	9340
	Benign	2	26	396	9078	627
	Scan	0	250	2542	197	15931

(d)
ADABOOST CLASSIFIER

AdaBoost 0.6871		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	734	3	10	14084	0
	MITM ARP Sp.	1	973	5605	24	2299
	Mirai	16	3320	94355	1185	4696
	Benign	9	26	2532	7148	414
	Scan	2	264	13830	604	4220

(e)
LOGISTIC REGRESSION (LR) CLASSIFIER

LR 0.7804		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14713	18	85	11	4
	MITM ARP Sp.	2	1437	6107	11	1345
	Mirai	45	144	99621	368	3394
	Benign	3	6	7098	2597	425
	Scan	4	12	15107	142	3655

TABLE 29. *(Continued.)* **Confusion matrix (CM) tables of classifiers for DS2 dataset when all features and default parameters are used.**

(f)
RANDOM FOREST (RF) CLASSIFIER

RF 0.8690		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14814	6	7	3	1
	MITM ARP Sp.	0	4131	2439	49	2283
	Mirai	1	1593	92977	224	8777
	Benign	2	42	404	9047	634
	Scan	0	470	3316	235	14899

(g)
DECISION TREE (DT) CLASSIFIER

DT 0.8698		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14818	7	5	1	0
	MITM ARP Sp.	3	4401	2226	51	2221
	Mirai	2	1817	92978	248	8527
	Benign	2	60	412	9061	594
	Scan	0	575	3379	225	14741

(h)
K-NEIGHBOR NEAREST (KNN) CLASSIFIER

KNN : 0.8466		Predicted Class				
		DoS	MITM ARP Sp.	Mirai	Benign	Scan
Actual Class	DoS	14792	7	22	6	4
	MITM ARP Sp.	4	4780	2770	48	1300
	Mirai	2	3515	93968	296	5791
	Benign	4	120	633	8924	448
	Scan	0	2789	5935	288	9908

beneficial to use SVC for detecting normal traffic, followed by the hybrid use of other classifiers.

XGBoost algorithm achieved the highest performance in the imbalanced dataset based on the top 10 features. However, it can be said that there is generally very low success in classifying ShellCode, Worms, and Backdoor traffic, especially. No classifier could detect Worms traffic at all. Therefore, the performance in Worms detection was measured at 0% for all classifiers. For DS1, which types of attacks were most accurately classified by which algorithm are shown in Table 22. In the table, the f1-Score metric was used instead of accuracy.

For DS2 dataset, the classifiers that achieved the highest accuracy in detecting traffic types and their accuracy values are provided in Table 23.

The highest F1-score value for this dataset is 99.97% in detecting DoS traffic, achieved by the XGBoost classifier. However, the F1-score performance in detecting Scan traffic is 33.54%, achieved by the KNN classifier. The best classification and accuracy values for the DS3 database are presented in Table 24.

FTP-patator traffic has been detected with 100% accuracy by KNN, RF, GNB, and DT classifiers. Similarly, PortScan, DDoS, and Benign traffic have been detected with over 99% accuracy by various classifiers. However, classifiers have achieved a maximum of 10% accuracy in detecting XSS traffic. It can be observed that XGB is more successful than other classifiers for this dataset, and this is confirmed by its overall accuracy value of 97.25%. The best-performing classifiers in traffic classification for another dataset, DS4, are provided in Table 25.

According to the table, the classifier that can most successfully detect DDoS attacks is Logistic Regression with 99.77% accuracy. For DoS attacks, Random Forest achieved the highest performance at 99.30%, and for Mirai attacks, the highest accuracy was obtained with XGBoost at 99.36%. The lowest detection success is observed for Web attacks, with a mere 0.8% achieved by the Decision Tree algorithm.

V. CONCLUSION

This study aimed to evaluate the performance of different machine learning classifiers on four distinct datasets for the

TABLE 30. Confusion matrix (CM) tables of classifiers for DS3 dataset when all features and default parameters are used.

(a)

SUPPORT VECTOR CLASSIFIER (SVC)

SVC : 0.9714		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	55628	0	210	1	684	0	1	0	221	0	0	0
	Bot	17	0	0	0	0	0	0	0	22	0	0	0
	DDoS	1	0	3203	0	0	0	0	0	1	0	0	0
	GoldenEye	64	0	0	181	9	0	1	0	0	0	0	0
	Hulk	220	0	0	0	5588	0	0	0	0	0	0	0
	Slowhttptest	98	0	0	0	3	26	4	0	0	0	0	0
	slowloris	79	0	0	0	0	4	60	0	0	0	0	0
	FTP-Patator	202	0	0	0	0	0	0	0	0	0	0	0
	PortScan	12	0	0	0	0	0	0	0	3988	0	0	0
	SSH-Patator	133	0	0	0	0	0	0	0	0	0	0	0
	Brute Force	30	0	0	0	0	0	0	0	0	0	0	0
	XSS	10	0	0	0	0	0	0	0	0	0	0	0

(b)

GAUSSIAN NAIVE BAYES (GNB) CLASSIFIER

GNB 0.9029		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	50326	4384	236	825	110	5	116	12	38	246	428	19
	Bot	0	39	0	0	0	0	0	0	0	0	0	0
	DDoS	3	17	3185	0	0	0	0	0	0	0	0	0
	GoldenEye	2	0	0	253	0	0	0	0	0	0	0	0
	Hulk	12	0	0	157	5472	0	167	0	0	0	0	0
	Slowhttptest	8	0	0	0	1	87	35	0	0	0	0	0
	slowloris	3	0	0	1	0	1	138	0	0	0	0	0
	FTP-Patator	0	0	0	0	0	0	0	202	0	0	0	0
	PortScan	10	25	2	0	0	0	0	0	3963	0	0	0
	SSH-Patator	0	0	0	0	0	0	0	0	0	133	0	0
	Brute Force	0	0	0	0	0	0	0	0	0	0	28	2
	XSS	0	0	0	0	0	0	0	0	0	0	0	10

purpose of detecting IoT attacks. Performance assessment involved the examination of critical metrics such as accuracy, training time, f1-score, and AUC, enabling a comparison of the capabilities of various classifiers. Additionally, the impact of various preprocessing techniques on performance was explored, revealing that these processes had a relatively minor effect on some classifiers but significantly enhanced the performance of others. The results obtained contributed to the identification of the most suitable classifiers for combating IoT attacks. It was observed that certain classifiers exhibited superior ability in detecting specific attack types and could be further optimized through specific preprocessing steps.

This study assessed the multi-class performance of the classifiers, which is crucial for IoT security as different classifiers may excel in detecting different attack types, offering valuable insights for the enhancement of defense strategies.

TABLE 30. *(Continued.)* **Confusion matrix (CM) tables of classifiers for DS3 dataset when all features and default parameters are used.**

(c)

XGBOOST CLASSIFIER

XGB 0.9998		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56740	1	1	0	2	0	0	0	0	0	1	0
	Bot	2	37	0	0	0	0	0	0	0	0	0	0
	DDoS	0	0	3205	0	0	0	0	0	0	0	0	0
	GoldenEye	1	0	0	254	0	0	0	0	0	0	0	0
	Hulk	0	0	0	0	5808	0	0	0	0	0	0	0
	Slowhttptest	0	0	0	0	0	131	0	0	0	0	0	0
	slowloris	1	0	0	0	0	0	142	0	0	0	0	0
	FTP-Patator	0	0	0	0	0	0	0	202	0	0	0	0
	PortScan	3	0	0	0	0	0	0	0	3997	0	0	0
	SSH-Patator	0	0	0	0	0	0	0	0	0	133	0	0
	Brute Force	2	0	0	0	0	0	0	0	0	0	28	0
	XSS	0	0	0	0	0	0	0	0	0	0	0	10

(d)

ADABOOST CLASSIFIER

AdaBoost 0.8565		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56671	0	0	0	66	0	0	8	0	0	0	0
	Bot	39	0	0	0	0	0	0	0	0	0	0	0
	DDoS	3205	0	0	0	0	0	0	0	0	0	0	0
	GoldenEye	90	0	0	0	165	0	0	0	0	0	0	0
	Hulk	1922	0	0	0	3886	0	0	0	0	0	0	0
	Slowhttptest	131	0	0	0	0	0	0	0	0	0	0	0
	slowloris	143	0	0	0	0	0	0	0	0	0	0	0
	FTP-Patator	202	0	0	0	0	0	0	0	0	0	0	0
	PortScan	3998	0	0	0	2	0	0	0	0	0	0	0
	SSH-Patator	133	0	0	0	0	0	0	0	0	0	0	0
	Brute Force	30	0	0	0	0	0	0	0	0	0	0	0
	XSS	10	0	0	0	0	0	0	0	0	0	0	0

We demonstrated which classifiers achieved successful results in detecting specific types of attacks. Since we evaluated the performance of different algorithms in various IoT datasets in our study, selecting the most suitable algorithms for hybrid security solutions could be easier. Hybrid systems can achieve more precise attack detection by combining different algorithms. Understanding which algorithm is better at detecting specific types of attacks enables us to enhance these solutions more effectively. Consequently, it can be observed which combination of classifiers might potentially yield more accurate results if a hybrid model is to be developed for certain attack types.

By examining the effects of scaling and outlier removal processes on accuracy, we demonstrated the importance of data preprocessing techniques. This can help us understand the challenges faced when dealing with real-world IoT data. This information shows us how to adjust enhance algorithm performance while processing real-time data.

TABLE 30. *(Continued.)* **Confusion matrix (CM) tables of classifiers for DS3 dataset when all features and default parameters are used.**

(e)
LOGISTIC REGRESSION (LR) CLASSIFIER

LR 0.9600		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56024	2	184	0	362	4	0	0	169	0	0	0
	Bot	31	0	0	0	0	0	0	0	8	0	0	0
	DDoS	1	0	3203	0	0	0	0	0	1	0	0	0
	GoldenEye	79	0	0	156	20	0	0	0	0	0	0	0
	Hulk	1346	0	0	1	4461	0	0	0	0	0	0	0
	Slowhttptest	104	0	0	0	1	26	0	0	0	0	0	0
	slowloris	120	0	0	0	0	5	18	0	0	0	0	0
	FTP-Patator	202	0	0	0	0	0	0	0	0	0	0	0
	PortScan	12	0	0	0	0	0	0	0	3988	0	0	0
	SSH-Patator	133	0	0	0	0	0	0	0	0	0	0	0
	Brute Force	30	0	0	0	0	0	0	0	0	0	0	0
	XSS	10	0	0	0	0	0	0	0	0	0	0	0

(f)
RANDOM FOREST (RF) CLASSIFIER

RF 0.9996		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56743	0	0	0	2	0	0	0	0	0	0	0
	Bot	6	33	0	0	0	0	0	0	0	0	0	0
	DDoS	0	0	3205	0	0	0	0	0	0	0	0	0
	GoldenEye	1	0	0	253	0	1	0	0	0	0	0	0
	Hulk	0	0	0	0	5808	0	0	0	0	0	0	0
	Slowhttptest	1	0	0	0	0	130	0	0	0	0	0	0
	slowloris	2	0	0	0	0	0	141	0	0	0	0	0
	FTP-Patator	0	0	0	0	0	0	0	202	0	0	0	0
	PortScan	3	0	0	0	1	0	0	0	3996	0	0	0
	SSH-Patator	0	0	0	0	0	0	0	0	0	133	0	0
	Brute Force	2	0	0	0	0	0	0	0	0	0	25	3
	XSS	0	0	0	0	0	0	0	0	0	0	3	7

The wide range of applications in IoT has led to the development of solutions in many sectors, consequently increasing the demand for security. However, the expectations from each IoT solution can vary; for some, training time might be crucial, while for others, it could be of secondary importance. Similarly, the reliability of the algorithm might hold more value than accuracy. Therefore, to ensure diversity and inclusivity in metric selection, multiple metrics have been utilized in the study's results.

In conclusion, this research made a significant contribution to the field of IoT attack detection, providing essential insights into which classifiers or combinations of classifiers can be optimized through specific preprocessing techniques. These findings can serve as a valuable resource

TABLE 30. *(Continued.)* **Confusion matrix (CM) tables of classifiers for DS3 dataset when all features and default parameters are used.**

(g)

DECISION TREE (DT) CLASSIFIER

DT 0.9995		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56720	1	3	6	4	3	1	0	4	0	2	1
	Bot	1	38	0	0	0	0	0	0	0	0	0	0
	DDoS	0	0	3205	0	0	0	0	0	0	0	0	0
	GoldenEye	1	0	0	253	0	0	1	0	0	0	0	0
	Hulk	0	0	0	0	5808	0	0	0	0	0	0	0
	Slowhttptest	1	0	0	0	0	130	0	0	0	0	0	0
	slowloris	1	0	0	0	0	0	142	0	0	0	0	0
	FTP-Patator	0	0	0	0	0	0	0	202	0	0	0	0
	PortScan	3	0	0	0	0	0	0	0	3997	0	0	0
	SSH-Patator	0	0	0	0	0	0	0	0	0	133	0	0
	Brute Force	2	0	0	0	0	0	0	0	0	0	28	0
	XSS	0	0	0	0	0	0	0	0	0	0	0	10

(h)

K-NEIGHBOR NEAREST (KNN) CLASSIFIER

KNN : 0.9991		Predicted Class											
		Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS slowloris	FTP-Patator	PortScan	SSH-Patator	Brute Force	XSS
Actual Class	Benign	56704	9	4	1	5	1	1	0	1	8	11	0
	Bot	5	34	0	0	0	0	0	0	0	0	0	0
	DDoS	1	0	3204	0	0	0	0	0	0	0	0	0
	GoldenEye	3	0	0	250	1	1	0	0	0	0	0	0
	Hulk	1	0	0	0	5807	0	0	0	0	0	0	0
	Slowhttptest	1	0	0	0	0	130	0	0	0	0	0	0
	slowloris	3	0	0	0	0	1	139	0	0	0	0	0
	FTP-Patator	0	0	0	0	0	0	0	202	0	0	0	0
	PortScan	3	0	0	0	0	0	0	0	3997	0	0	0
	SSH-Patator	3	0	0	0	0	0	0	0	0	130	0	0
	Brute Force	3	0	0	0	0	0	0	0	0	0	27	0
	XSS	0	0	0	0	0	0	0	0	0	0	0	10

for researchers and industry professionals aiming to enhance IoT security. The outcomes of this study may also guide organizations seeking to improve their strategies for safeguarding IoT devices.

APPENDIX

TABLES AND FIGURES

See Tables 26–30 and Fig. 11.

REFERENCES

[1] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of Things is a revolutionary approach for future technology enhancement: A review," *J. Big Data*, vol. 6, p. 111, Dec. 2019.

[2] A. Shamsuzzoha, J. Nieminen, S. Piya, and K. Rutledge, "Smart city for sustainable environment: A comparison of participatory strategies from Helsinki, Singapore and London," *Cities*, vol. 114, Jul. 2021, Art. no. 103194.

[3] J. Shahid, R. Ahmad, A. K. Kiani, T. Ahmad, S. Saeed, and A. M. Almuhaideb, "Data protection and privacy of the Internet of Healthcare Things (IoHTs)," *Appl. Sci.*, vol. 12, no. 4, p. 1927, Feb. 2022.

[4] A. Aljeraisy, M. Barati, O. Rana, and C. Perera, "Privacy laws and privacy by design schemes for the Internet of Things: A developer's perspective," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–38, May 2021.

[5] R. Das and E. Ozdogan, "Layered management approach to cyber security of IoT solutions," *Int. J. Grid Utility Comput.*, vol. 14, no. 5, pp. 493–504, 2023.

[6] R. Kumar and N. Agrawal, "Analysis of multi-dimensional industrial IoT (IIoT) data in edge–fog–cloud based architectural frameworks: A survey on current state and research challenges," *J. Ind. Inf. Integr.*, vol. 35, Oct. 2023, Art. no. 100504.

[7] L. L. Dhirani, E. Armstrong, and T. Newe, "Industrial IoT, cyber threats, and standards landscape: Evaluation and roadmap," *Sensors*, vol. 21, no. 11, p. 3901, Jun. 2021.

[8] M. Soori, B. Arezoo, and R. Dastres, "Internet of Things for smart factories in Industry 4.0, a review," *Internet Things Cyber-Phys. Syst.*, vol. 3, pp. 192–204, Jan. 2023.

[9] Q. Wang, X. Zhu, Y. Ni, L. Gu, and H. Zhu, "Blockchain for the IoT and industrial IoT: A review," *Internet Things*, vol. 10, Jun. 2020, Art. no. 100081.

[10] M. Serror, S. Hack, M. Henze, M. Schuba, and K. Wehrle, "Challenges and opportunities in securing the industrial Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 2985–2996, May 2021.

[11] P. Jayalaxmi, R. Saha, G. Kumar, N. Kumar, and T.-H. Kim, "A taxonomy of security issues in industrial Internet-of-Things: Scoping review for existing solutions, future implications, and research challenges," *IEEE Access*, vol. 9, pp. 25344–25359, 2021.

[12] A. Barnawi, S. Gaba, A. Alphy, A. Jabbari, I. Budhiraja, V. Kumar, and N. Kumar, "A systematic analysis of deep learning methods and potential attacks in Internet-of-Things surfaces," *Neural Comput. Appl.*, vol. 35, no. 25, pp. 18293–18308, Sep. 2023.

[13] S. Bharati and P. Podder, "Machine and deep learning for IoT security and privacy: Applications, challenges, and future directions," *Secur. Commun. Netw.*, vol. 2022, pp. 1–41, Aug. 2022.

[14] K. Shaukat, T. M. Alam, I. A. Hameed, W. A. Khan, N. Abbas, and S. Luo, "A review on security challenges in Internet of Things (IoT)," in *Proc. 26th Int. Conf. Autom. Comput. (ICAC)*, Sep. 2021, pp. 1–6.

[15] A. D. Kounoudes and G. M. Kapitsaki, "A mapping of IoT user-centric privacy preserving approaches to the GDPR," *Internet Things*, vol. 11, Sep. 2020, Art. no. 100179.

[16] W. Yao, H. Shi, and H. Zhao, "Scalable anomaly-based intrusion detection for secure Internet of Things using generative adversarial networks in fog environment," *J. Netw. Comput. Appl.*, vol. 214, May 2023, Art. no. 103622.

[17] H. C. Altunay and Z. Albayrak, "A hybrid CNN+LSTM-based intrusion detection system for industrial IoT networks," *Eng. Sci. Technol., Int. J.*, vol. 38, Feb. 2023, Art. no. 101322.

[18] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for Internet of Things data analysis: A survey," *Digit. Commun. Netw.*, vol. 4, no. 3, pp. 161–175, Aug. 2018.

[19] R. Shahbazian, G. Macrina, E. Scalzo, and F. Guerriero, "Machine learning assists IoT localization: A review of current challenges and future trends," *Sensors*, vol. 23, no. 7, p. 3551, Mar. 2023.

[20] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "Zero-day attack detection: A systematic literature review," *Artif. Intell. Rev.*, vol. 56, no. 10, pp. 10733–10811, Oct. 2023.

[21] S. M. Tahsien, H. Karimipour, and P. Spachos, "Machine learning based solutions for security of Internet of Things (IoT): A survey," *J. Netw. Comput. Appl.*, vol. 161, Jul. 2020, Art. no. 102630.

[22] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021.

[23] A. Alhowaide, I. Alsmadi, and J. Tang, "Ensemble detection model for IoT IDS," *Internet Things*, vol. 16, Dec. 2021, Art. no. 100435.

[24] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," *J. Netw. Comput. Appl.*, vol. 28, no. 2, pp. 167–182, Apr. 2005.

[25] A. Churcher, R. Ullah, J. Ahmad, S. Ur Rehman, F. Masood, M. Gogate, F. Alqahtani, B. Nour, and W. J. Buchanan, "An experimental analysis of attack classification using machine learning in IoT networks," *Sensors*, vol. 21, no. 2, p. 446, Jan. 2021.

[26] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, "Modeling intrusion detection system using hybrid intelligent systems," *J. Netw. Comput. Appl.*, vol. 30, no. 1, pp. 114–132, Jan. 2007.

[27] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for IoT applications," *Wireless Pers. Commun.*, vol. 111, no. 4, pp. 2287–2310, Apr. 2020.

[28] N. Saran and N. Kesswani, "A comparative study of supervised machine learning classifiers for intrusion detection in Internet of Things," *Proc. Comput. Sci.*, vol. 218, pp. 2049–2057, Jan. 2023.

[29] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2671–2701, 3rd Quart., 2019.

[30] G. Abdelmoumin and D. B. Rawat, "SmartIDS: A comparative study of intelligent intrusion detection systems for Internet of Things," in *Proc. Future Technol. Conf. (FTC)*, K. Arai, Ed. Cham, Switzerland: Springer, 2022, pp. 420–438.

[31] S. Dwibedi, M. Pujari, and W. Sun, "A comparative study on contemporary intrusion detection datasets for machine learning research," in *Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI)*, Nov. 2020, pp. 1–6.

[32] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "A comprehensive deep learning benchmark for IoT IDS," *Comput. Secur.*, vol. 114, Mar. 2022, Art. no. 102588.

[33] A. Shaver, Z. Liu, N. Thapa, K. Roy, B. Gokaraju, and X. Yuan, "Anomaly based intrusion detection for IoT with machine learning," in *Proc. IEEE Appl. Imag. Pattern Recognit. Workshop (AIPR)*, Oct. 2020, pp. 1–6.

[34] P. Shukla, "ML-IDS: A machine learning approach to detect wormhole attacks in Internet of Things," in *Proc. Intell. Syst. Conf. (IntelliSys)*, Sep. 2017, pp. 234–240.

[35] I. Hidayat, M. Z. Ali, and A. Arshad, "Machine learning-based intrusion detection system: An experimental comparison," *J. Comput. Cognit. Eng.*, vol. 2, pp. 88–97, Jul. 2022.

[36] C. Zhang, D. Jia, L. Wang, W. Wang, F. Liu, and A. Yang, "Comparative research on network intrusion detection methods based on machine learning," *Comput. Secur.*, vol. 121, Oct. 2022, Art. no. 102861.

[37] S. Gamage and J. Samarabandu, "Deep learning methods in network intrusion detection: A survey and an objective comparison," *J. Netw. Comput. Appl.*, vol. 169, Nov. 2020, Art. no. 102767.

[38] D. S. Jodas, L. A. Passos, D. Rodrigues, T. J. Lucas, K. A. P. Da Costa, and J. P. Papa, "OPFsemble: An ensemble pruning approach via optimum-path forest," in *Proc. 30th Int. Conf. Syst., Signals Image Process. (IWSSIP)*, 2023, pp. 1–5.

[39] b. I. Farhan and A. D. Jasim, "A survey of intrusion detection using deep learning in Internet of Things," *Iraqi J. Comput. Sci. Math.*, vol. 3, pp. 83–93, Jan. 2021.

[40] A. Fadil, I. Riadi, and S. Aji, "A novel ddos attack detection based on Gaussian naive Bayes," *Bull. Electr. Eng. Informat.*, vol. 6, no. 2, pp. 140–148, 2017.

[41] B. S. Sharmila and R. Nagapadma, "Intrusion detection system using naive Bayes algorithm," in *Proc. IEEE Int. WIE Conf. Electr. Comput. Eng. (WIECON-ECE)*, Nov. 2019, pp. 1–4.

[42] R. Islam, M. K. Devnath, M. D. Samad, and S. M. Jaffrey Al Kadry, "GGNB: Graph-based Gaussian naive Bayes intrusion detection system for CAN bus," *Veh. Commun.*, vol. 33, Jan. 2022, Art. no. 100442.

[43] S. Uddin, I. Haque, H. Lu, M. A. Moni, and E. Gide, "Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction," *Sci. Rep.*, vol. 12, no. 1, p. 6256, Apr. 2022.

[44] Y. Alharbi, A. Alferaidi, K. Yadav, G. Dhiman, and S. Kautish, "Denial-of-Service attack detection over IPv6 network based on KNN algorithm," *Wireless Commun. Mobile Comput.*, vol. 2021, pp. 1–6, Dec. 2021.

[45] M.-Y. Su, K.-C. Chang, H.-F. Wei, and C.-Y. Lin, "Feature weighting and selection for a real-time network intrusion detection system based on GA with KNN," in *Intelligence and Security Informatics*, C. C. Yang, H. Chen, M. Chau, K. Chang, S.-D. Lang, P. S. Chen, R. Hsieh, D. Zeng, F.-Y. Wang, K. Carley, W. Mao, and J. Zhan, Eds. Berlin, Germany: Springer, 2008, pp. 195–204.

[46] M.-Y. Su, "Real-time anomaly detection systems for denial-of-service attacks by weighted k-nearest-neighbor classifiers," *Exp. Syst. Appl.*, vol. 38, no. 4, pp. 3492–3498, Apr. 2011.

[47] Y. Liao and V. R. Vemuri, "Use of K-nearest neighbor classifier for intrusion detection," *Comput. Secur.*, vol. 21, no. 5, pp. 439–448, Oct. 2002.

[48] S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning K for KNN classification," *ACM Trans. Intell. Syst. Technol.*, vol. 8, no. 3, pp. 1–19, 2017.

[49] A. J. Ferreira and M. A. T. Figueiredo, "Boosting algorithms: A review of methods, theory, and applications," in *Ensemble Machine Learning*, C. Zhang and Y. Ma, Eds. New York, NY, USA: Springer, 2012, pp. 35–85.

[50] A. B. M. S. Ali and Y. Xiang, "Spam classification using adaptive boosting algorithm," in *Proc. 6th IEEE/ACIS Int. Conf. Comput. Inf. Sci. (ICIS)*, 2007, pp. 972–976.

[51] D.-C. Feng, Z.-T. Liu, X.-D. Wang, Y. Chen, J.-Q. Chang, D.-F. Wei, and Z.-M. Jiang, "Machine learning-based compressive strength prediction for concrete: An adaptive boosting approach," *Construct. Building Mater.*, vol. 230, Jan. 2020, Art. no. 117000.

[52] R. Gao and Z. Liu, "An improved AdaBoost algorithm for hyperparameter optimization," *J. Physics: Conf. Ser.*, vol. 1631, no. 1, Sep. 2020, Art. no. 012048.

[53] R. Krithiga and E. Ilavarasan, "Hyperparameter tuning of AdaBoost algorithm for social spammer identification," *Int. J. Pervasive Comput. Commun.*, vol. 17, no. 5, pp. 462–482, Dec. 2021.

[54] S. Ramraj, N. Uzir, R. Sunil, and S. Banerjee, "Experimenting XGBoost algorithm for prediction and classification of different datasets," *Int. J. Control Theory Appl.*, vol. 9, no. 40, pp. 651–662, 2016.

[55] S. Dhaliwal, A.-A. Nahid, and R. Abbas, "Effective intrusion detection system using XGBoost," *Information*, vol. 9, no. 7, p. 149, Jun. 2018.

[56] S. A. Mulay, P. R. Devale, and G. V. Garje, "Decision tree based support vector machine for intrusion detection," in *Proc. Int. Conf. Netw. Inf. Technol.*, Jun. 2010, pp. 59–63.

[57] M. Mohammadi, T. A. Rashid, S. H. T. Karim, A. H. M. Aldalwie, Q. T. Tho, M. Bidaki, A. M. Rahmani, and M. Hosseinzadeh, "A comprehensive survey and taxonomy of the SVM-based intrusion detection systems," *J. Netw. Comput. Appl.*, vol. 178, Mar. 2021, Art. no. 102983.

[58] B. S. Bhati and C. S. Rai, "Analysis of support vector machine-based intrusion detection techniques," *Arabian J. Sci. Eng.*, vol. 45, no. 4, pp. 2371–2383, Apr. 2020.

[59] O. Almomani, M. A. Almaiah, A. Alsaaidah, S. Smadi, A. H. Mohammad, and A. Althunibat, "Machine learning classifiers for network intrusion detection system: Comparative study," in *Proc. Int. Conf. Inf. Technol. (ICIT)*, Jul. 2021, pp. 440–445.

[60] C. Ioannou, V. Vassiliou, and C. Sergiou, "An intrusion detection system for wireless sensor networks," in *Proc. 24th Int. Conf. Telecommun. (ICT)*, May 2017, pp. 1–5.

[61] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. K. A. A. Khan, "Performance analysis of machine learning algorithms in intrusion detection system: A review," *Proc. Comput. Sci.*, vol. 171, pp. 1251–1260, Jan. 2020.

[62] A. H. Azizan, S. A. Mostafa, A. Mustapha, C. F. M. Foozy, M. H. A. Wahab, M. A. Mohammed, and B. A. Khalaf, "A machine learning approach for improving the performance of network intrusion detection systems," *Ann. Emerg. Technol. Comput.*, vol. 5, pp. 201–208, Mar. 2021.

[63] A. Belenguer, J. A. Pascual, and J. Navaridas, "GöwFed: A novel federated network intrusion detection system," *J. Netw. Comput. Appl.*, vol. 217, Aug. 2023, Art. no. 103653.

[64] J. N. Mandrekar, "Receiver operating characteristic curve in diagnostic test assessment," *J. Thoracic Oncol.*, vol. 5, no. 9, pp. 1315–1316, Sep. 2010.

[65] F. S. Nahm, "Receiver operating characteristic curve: Overview and practical use for clinicians," *Korean J. Anesthesiol.*, vol. 75, no. 1, pp. 25–36, Feb. 2022.

[66] P. C. Austin, H. Putter, D. Giardiello, and D. van Klaveren, "Graphical calibration curves and the integrated calibration index (ICI) for competing risk models," *Diagnostic Prognostic Res.*, vol. 6, no. 1, p. 1–2, Dec. 2022.

[67] C. F. Dormann, "Calibration of probability predictions from machine-learning and statistical models," *Global Ecology Biogeography*, vol. 29, no. 4, pp. 760–765, Apr. 2020.

[68] G. Meyfroidt, F. Güiza, D. Cottem, W. De Becker, K. Van Loon, J.-M. Aerts, D. Berckmans, J. Ramon, M. Bruynooghe, and G. Van den Berghe, "Computerized prediction of intensive care unit discharge after cardiac surgery: Development and validation of a Gaussian processes model," *BMC Med. Informat. Decis. Making*, vol. 11, no. 1, p. 64, Dec. 2011.

[69] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "NetFlow datasets for machine learning-based network intrusion detection systems," in *Big Data Technologies and Applications* (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), Z. Deze, H. Huang, R. Hou, S. Rho, and N. Chilamkurti, Eds. Cham, Switzerland: Springer, 2020.

[70] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: Statistical decision-making using finite Dirichlet mixture models," in *Data Analytics and Decision Support for Cybersecurity*, H. K. Y. P. Carrascosa and I. Kalutarage, Eds. Cham, Switzerland: Springer, 2017, pp. 127–156.

[71] N. Moustafa, J. Slay, and G. Creech, "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE Trans. Big Data*, vol. 5, no. 4, pp. 481–494, Dec. 2019.

[72] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J., Global Perspective*, vol. 25, nos. 1–3, pp. 18–31, Apr. 2016.

[73] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, 2015, pp. 1–6.

[74] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Advances in Artificial Intelligence* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2020, pp. 508–520.

[75] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. 4th Int. Conf. Inf. Syst. Secur. Privacy*, 2018, pp. 108–116.

[76] R. Panigrahi and S. Borah, "A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems," *Int. J. Eng. Technol.*, vol. 7, no. 3, pp. 479–482, 2018.

[77] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 23, no. 13, p. 5941, Jun. 2023.

[78] C. S. K. Dash, A. K. Behera, S. Dehuri, and A. Ghosh, "An outliers detection and elimination framework in classification task of data mining," *Decis. Analytics J.*, vol. 6, Mar. 2023, Art. no. 100164.

[79] M. Ahsan, M. Mahmud, P. Saha, K. Gupta, and Z. Siddique, "Effect of data scaling methods on machine learning algorithms and model performance," *Technologies*, vol. 9, no. 3, p. 52, Jul. 2021.

[80] A. Kaur, K. Guleria, and N. Kumar Trivedi, "Feature selection in machine learning: Methods and comparison," in *Proc. Int. Conf. Advance Comput. Innov. Technol. Eng. (ICACITE)*, Mar. 2021, pp. 789–795.

[81] S. Venkatesan, "Design an intrusion detection system based on feature selection using ML algorithms," *Math. Statistician Eng. Appl.*, vol. 72, no. 1, pp. 702–710, 2023.

[82] Z. K. Ibrahim and M. Y. Thanon, "Performance comparison of intrusion detection system using three different machine learning algorithms," in *Proc. 6th Int. Conf. Inventive Comput. Technol. (ICICT)*, Jan. 2021, pp. 1116–1124.

[83] M. H. Nasir, S. A. Khan, M. M. Khan, and M. Fatima, "Swarm intelligence inspired intrusion detection systems—A systematic literature review," *Comput. Netw.*, vol. 205, Mar. 2022, Art. no. 108708.

ERDAL OZDOGAN received the B.Sc. degree in astronomy and space science from Ankara University, in 2000, the master's degree from the Department of Computer Science, Gazi University, in 2012, the B.Sc. degree in management information systems from Anadolu University, Turkey, in 2019, and the Ph.D. degree from the Department of Information Systems, in 2020. He serves as an Instructor training to public institutions and organizations about computer networks, cybersecurity, and the IoT with the Cisco Networking Academy. He is currently a Lecturer with Gazi University and teaching computer networks, cryptography, and machine learning courses with the University of Turkish Aeronautical Association. His current research interests include computer networks, network security, cybersecurity, and the IoT.