Mesopotamian journal of Cybersecurity
Vol.5,No.1, **pp**. 240–255
DOI: https://doi.org/10.58496/MJCS/2025/016; ISSN: 2958-6542
https://mesopotamian.press/journals/index.php/cybersecurity

Research Article

The Impact of Feature Importance on Spoofing Attack Detection in IoT Environment

Sawsan H. Jadoaa[1], Rasha H. Ali[1,*], Wisal Hashim Abdulsalam[2], Emad M. Alsaedi [3]

[1] *Computer Science Department/College of Education for Women/University of Baghdad, IRAQ.*

[2] *Computer Department/College of Education for Pure Science/Ibn-Al-Haitham/University of Baghdad, IRAQ.*

[3] *Department of Computer Sciences, University of Technology, Baghdad, Iraq.*

ARTICLE INFO

Article history

Received 06 Jan 2025
Accepted 05 Mar 2025
Published 01 Apr 2025

Keywords

Spoofing Attack
IOT
Feature Importance
Decision Tree
Mutual Information
AdaBoosting
XGBoost
CatBoost

ABSTRACT

The Internet of Things (IoT) is an expanding domain that can revolutionize different industries. Nevertheless, security is among the multiple challenges that it encounters. A major threat in the IoT environment is spoofing attacks, a type of cyber threat in which malicious actors masquerade as legitimate entities.

This research aims to develop an effective technique for detecting spoofing attacks for IoT security by utilizing feature-importance methods. The suggested methodology involves three stages: preprocessing, selection of important features, and classification. The feature importance determines the most significant characteristics that play a role in detecting spoofing attacks. This is achieved via two techniques: decision tree (DT) and mutual information (MI). For classification, adaptive boosting (AdaBoost), XGBoost and categorical boosting (CatBoosting) are used to categorize incoming data as normal or spoofing. The experimental results indicate the efficiency of the suggested approach for correctly identifying spoofing attacks with high accuracy, fewer false positives, and reduced time needed. By utilizing feature importance and robust classification algorithms, the system can accurately differentiate between legitimate and malicious IoT traffic, thereby improving the overall security of IoT networks. The CatBoost classifier outperformed the AdaBoost and XGBoost classifiers in terms of accuracy.

1. INTRODUCTION

The Internet of Things (IoT) has fundamentally transformed our interaction with the physical world by linking billions of devices and producing an enormous quantity of data. However, this poses notable security problems. IoT devices may have limited resources and security capabilities, which makes them susceptible to attackers. A spoofing attack is a significant cyber threat that poses a particularly high risk to IoT systems [1, 2].

A spoofing attack occurs when malicious actors masquerade as trustworthy entities, tricking and directing IoT devices or networks into carrying out unauthorized activities [3]. Attackers employ forged identities to illegally gain access to confidential data, disrupt crucial functions, or even execute coordinated attacks on other networks. The increasing number of spoofing attacks emphasizes the necessity for robust mechanisms to detect these attacks, which are specifically designed for the IoT environment [4, 5].

Feature importance approaches have emerged as effective tools for improving the accuracy of spoofing detection in IoT networks. These techniques enable the identification of the most relevant attributes provided by IoT devices, enabling a more targeted and effective analysis. Machine learning (ML) methods can increase accuracy and minimize computational complexity by selecting the most informative features, as demonstrated in [6-8], making them better suited for real-time detection of intrusions in resource-limited IoT scenarios[27].

This work investigates the effectiveness of feature importance methods when combined with categorical boosting (CatBoost), adaptive boosting (AdaBoost) and XGBoost classifiers for spoofing attack detection in the IoT domain. The proposed approach tries to produce an accurate solution by using the strengths of feature importance techniques (decision tree (DT) and mutual information (MI)). This paper offers the following contributions:

1. Application of multiclassification techniques to identify and categorize spoofing attacks.

*Corresponding author. Email: sawsan.h@ coeduw.uobaghdad.edu.iq

2. Two feature selection techniques (decision trees (DTs) and mutual information (MI)) are used to optimize spoofing attack detection.
3. Identification of the Optimal Feature Set: This research finds the optimal features influencing the prediction accuracy for spoofing attacks.
4. This contributes to efficient model development by reducing complexity without sacrificing performance.
5. Comparative analysis of feature selection methods: A comparison between the DT and MI techniques was conducted, highlighting strengths and weaknesses in the spoofing detection context. The selection of appropriate methods for specific applications is critical for practitioners.
6. Performance evaluation: Comparison of the classification techniques AdaBoost, XGBoost, and CatBoost on the basis of performance criteria and computational efficiency.

This study combines innovative approaches to spoofing attack detection, enhancing capabilities by integrating feature selection and classification techniques. It contributes new knowledge on multiclass classification and advanced feature selection methods in cybersecurity.

The remainder of this paper is organized as follows: Section 2 provides a short survey on other approaches to existing spoofing attack detection in IoT networks. Section 3 presents the background of the techniques used in the current work. Section 4 describes the materials and methods employed in the paper. Section 5 presents the evaluation of the proposed framework via benchmark IoT datasets. The evaluation of the framework's performance in terms of accuracy, precision, recall, F1 score, and support. Section 6 presents the conclusion and suggestions for future work.

2. RELATED WORKS

Some of the studies related to the current work are discussed below.

F. Khan et al. [9] presented a technique for detecting and preventing spoofing attacks in IoT networks via the number of connected neighbors (NCN) and received signal strength (RSS) metrics. First, the system uses RSS measurements to identify, detect, and remove spoofing attacks on the intercluster network. However, the RSS is inefficient against intracluster spoofing attacks, necessitating the use of an NCN to successfully detect, identify, and mitigate these threats. The suggested work is carried out in Network Simulator 2 (NS-2) to evaluate its performance in both spoofing-free and spoofing-infested environments.

A. B. Altamimi et al. [10] developed a client-side mechanism for defense based on ML techniques to identify spoofed web pages and protect users from phishing attacks. They created a Google Chrome extension named PhishCatcher. This extension incorporates their ML algorithm, which classifies URLs as either suspicious or trustworthy. The algorithm analyses four distinct types of web features and employs a random forest (RF) classifier to determine whether a login webpage is genuine or spoofed. They conducted extensive testing on actual web applications and yielded an accuracy of 98.5% and a precision of 98.5%, as determined by trials involving 400 classified phishing URLs and 400 legitimate URLs. Additionally, to assess the latency of their tool, they performed experiments on forty phished URLs.

X. Cheng et al. [11] introduced a facial recognition system with an antispoofing method that effectively distinguishes between real and fake faces via optical flow and texture features. The system involves three stages: optical flow field map generation, feature extraction and fusion via a two-channel convolutional neural network (CNN), and liveness classification, which is based on a decision-making process that incorporates texture and motion information. To enhance performance, motion amplification and a lightweight network architecture are employed. An evaluation of the Replay Attack dataset yielded a half-total error rate of 0.66%, demonstrating the method's efficacy in spoofing detection.

X. Wei et al. [12] proposed a novel and lightweight global positioning system (GPS) method to detect spoofing that uses a dynamic threshold and an acquired signal envelope. Validation experiments with real GPS signals and hardware demonstrate its effectiveness. The method hinges on the inherent relationship between signal envelope features and the distance between the receiver and transmitter. Inspired by this relationship, a dynamic threshold approach is developed, replacing the traditional fixed threshold, and it is determined by the signal envelope variance, which enhances the detection performance across various attack scenarios.

M. Shabbir et al. [13] introduced a novel approach to safeguard connected and autonomous vehicles (CAVs) from GPS location spoofing attacks. The proposed work employs a combination of deep learning (DL) algorithms, such as CNNs, and ML algorithms, such as support vector machines (SVMs). The effectiveness of the proposed work is evaluated through real-time simulations in the CARLA simulator. Various learning algorithms have been used to determine the most effective technique among three distinct paths. The training and testing data encompass spoofed coordinates, GPS coordinates, and values of the localization algorithm. The proposed ML algorithm achieved 99% accuracy in the best-case scenario and 96% accuracy in the worst-case scenario. For DL, the accuracy ranged from 99% in the best-case scenario to 82% in the worst-case scenario. A comparison between previous related works and the current work is shown in TABLE I.

.

TABLE I. COMPARISON BETWEEN THE CURRENT WORK AND PREVIOUS STUDIES

Aspect	F. Khan et al.	A. B. Altamimi et al.	X. Cheng et al.	X. Wei et al.	M. Shabbir et al.	Current Research
Focus Area	Detection and prevention of spoofing attacks in IoT networks using RSS and NCN metrics.	Client-side mechanism to identify spoofed web pages using ML techniques (PhishCatcher).	Anti-spoofing method for facial recognition systems using optical flow and texture features.	GPS-based method to detect spoofing using dynamic thresholds and signal envelope features.	Safeguarding Connected and Autonomous Vehicles (CAVs) from GPS location spoofing attacks using DL/ML algorithms.	Multiclass classification techniques for detecting various types of spoofing attacks.
Methodology	Utilizes RSS measurements for intercluster detection and NCN for intracluster spoofing detection.	Developed a Google Chrome extension that uses RF classifier to analyse web features.	Uses a two-channel CNN for feature extraction and liveness classification based on texture and motion information.	Developed a dynamic threshold approach based on signal envelope variance to enhance detection performance.	Combines CNNs with SVMs in real-time simulations to evaluate effectiveness against GPS spoofing attacks.	Employs DT and MI for feature selection, combined with CatBoost, AdaBoost and XGBoost classifiers.
Performance Metrics	Evaluated using NS-2 in both spoofing-free and spoofing-infested environments.	Achieved 98.5% accuracy and precision through testing on classified URLs.	Achieved a half-total error rate of 0.66% on the Replay Attack dataset.	Demonstrated effectiveness through validation experiments with real GPS signals and hardware.	Achieved up to 99% accuracy in the best-case scenario using ML algorithms; DL accuracy ranged from 99% to 82%.	Focuses on enhancing detection accuracy, computational efficiency, and optimal feature selection across diverse scenarios.

3. PRELIMINARY CONCEPTS

The following subsections provide background related to the feature selection techniques and classification algorithms used in the current work.

3.1 Feature selection techniques

The diverse nature of attributes poses a challenge in achieving higher prediction accuracy. To address this, before applying an ML model for prediction, the process of selecting features should be implemented to identify and extract key features. It aids in reducing irrelevant variables, computational costs, and the problem of overfitting, thereby enhancing the performance of the ML model. When the number of attributes used as inputs for an ML model is reduced, the information may not be sufficient for making accurate predictions. Conversely, incorporating many features increases the runtime and diminishes the generalization performance because of the curse of dimensionality. Consequently, selecting only the features that have a significant effect on the results is crucial for achieving successful predictions [14][28]. Two techniques were used in this work: DT and MI.

3.1.1 Decision Tree (DT) Method

Decision trees (DTs) are commonly used for feature selection since they may rank features on the basis of their impact on classification accuracy. The C4.5 technique, an extension of ID3, is frequently used to choose the most crucial features [15]. It is a "wrapper method" for feature selection, and it evaluates the entire model's performance with different feature subsets, selecting the combination that leads to the highest accuracy. This ensures that the selected features not only individually

contribute but also collectively work well for classification [16]. A feature is considered important if splitting on that feature significantly reduces entropy in the dataset [17]. The entropy equation is as follows:

$$\mathrm{H(X)} = -\sum p(x) log_2 p(x) \quad (1)$$

where $p(x)$ is the probability of an outcome in the dataset.

3.1.2 Mutual Information (MI) Method

Feature selection aims to reduce the classification computation time by removing irrelevant characteristics. MI is an ML selection method that shows how significant a feature is in creating an accurate prediction [18]. We use MI, which plays a crucial role in feature selection by quantifying the relationship between features and the response variable. Various methods leverage MI to prioritize features for accurate predictions and reduce costs. It is an information-theoretic metric that measures how much knowledge of one feature decreases uncertainty about another (the target variable).

MI does not presume a linear connection between characteristics and labels, making it adaptable to different types of data. It calculates the statistical dependence between a feature and a class label. Features with higher MI values have a stronger link with the target variable and are deemed more informative for classification [19].
The MI formula is as follows:

$$I(x; y) = H(x) + H(y) - H(x, y) \quad (2)$$

where:

H(x) and *H(y)* are the entropy of feature *x* and target variable *y*, respectively.
H(x,y) is the joint entropy, which is the uncertainty measure when considering both variables together.

3.2 Classification and Performance Evaluation

Three classification algorithms were used in this work, as discussed below:

3.2.1 CatBoost Classifier

The CatBoost classifier is a powerful gradient boosting library based on open-source principles. It is nonlinear, tree-based, and effective with complex datasets. CatBoost outperforms other boosting techniques, showing significant improvements in accuracy and performance. It delivers optimal results quickly, which is valuable for time-sensitive applications such as fraud detection. CatBoost simplifies data preparation by supporting categorical features without preprocessing. Its advanced capabilities, ease of use, and exceptional performance make it a top choice for machine learning tasks [20].

3.2.2 XGBoost Classifier

It is a powerful ML algorithm presented in various research papers for diverse applications. It has been utilized for detecting malware executables with high precision and recall rates. XGBoost stands out for its ability to handle complex problems, achieve high prediction accuracy, and offer efficiency in various domains ranging from optical networks and cybersecurity to waste material identification [21].

3.2.3 AdaBoost Classifier

AdaBoost starts with a collection of 'weak learners' and iteratively improves them. This is accomplished by assigning weights to training examples. The samples misclassified by previous weak learners have higher weights, whereas correctly classified samples have lower weights. In this way, the algorithm focuses on examples that are harder to learn from, eventually building a strong classifier by combining these improved weak learners [22, 23].

4. PROPOSED MODEL

This model contains three stages: preprocessing, feature selection, and classification, as shown below. The feature selection was applied via two techniques for selecting the most important features, which are effective in the prediction of spoofing attacks. While the classification stage was applied via three techniques, Figure 1 shows the architecture of the proposed model.

4.1. Data collection

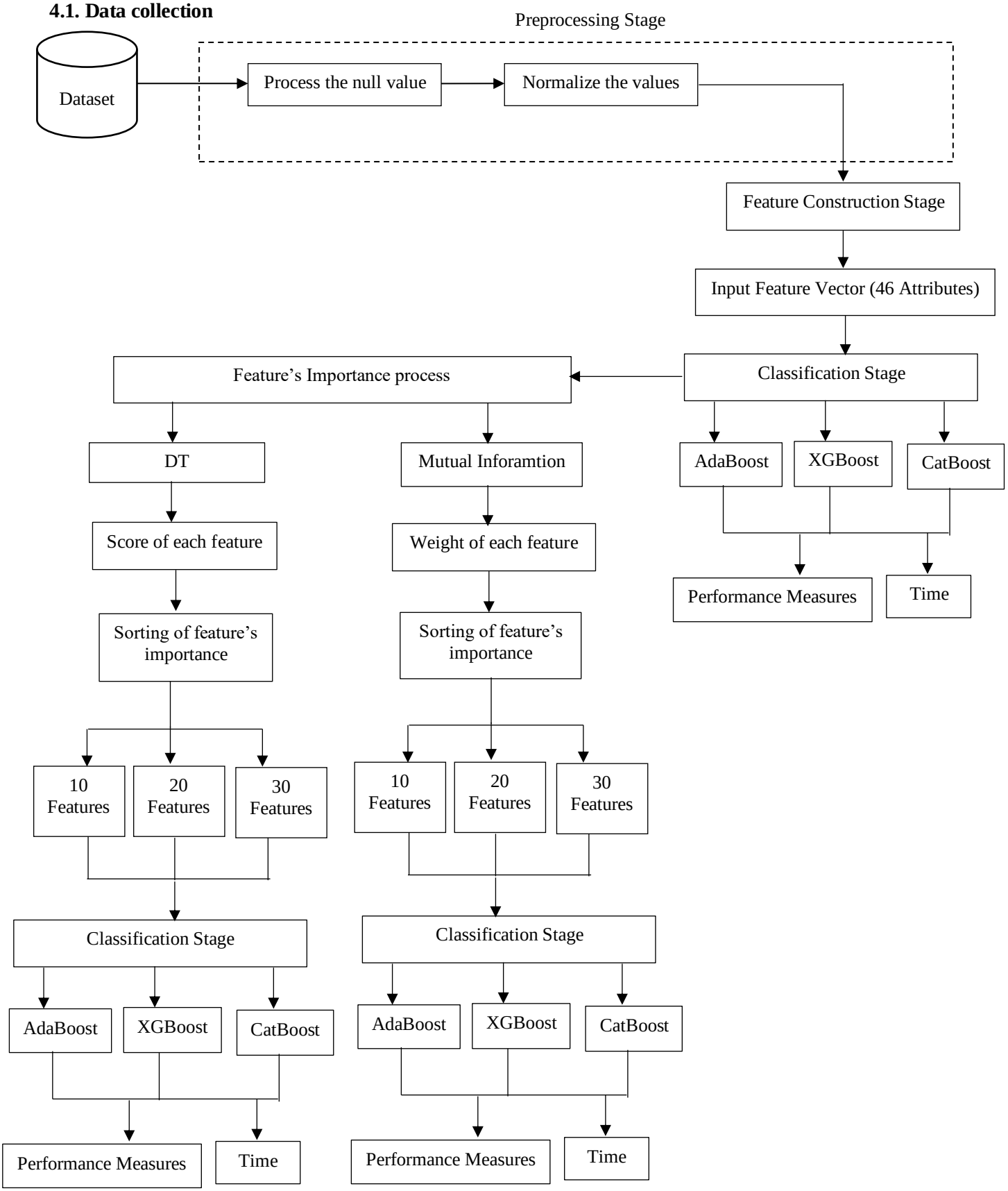

Fig. 1. The architecture of the proposed model.

The CIC-IoT2023 dataset was collected from the website https://www.unb.ca/cic/datasets/iotdataset-2023.html; this dataset contains (33) different attack types. These attacks are classified into seven categories. Two samples (spoofing attack and benign) were selected for this research, where the number of instances of this portion of the dataset was (1584699) and the number of features was (47), as shown in Table 1. The selected dataset contains (3) class labels, which are (307593 for MITM-ArpSpoofing, 178911 for DNS_Spoofing, and 1098195 for Benign_Traffic). Figure 2 shows the histogram of class labels for the spoofing attack [24].

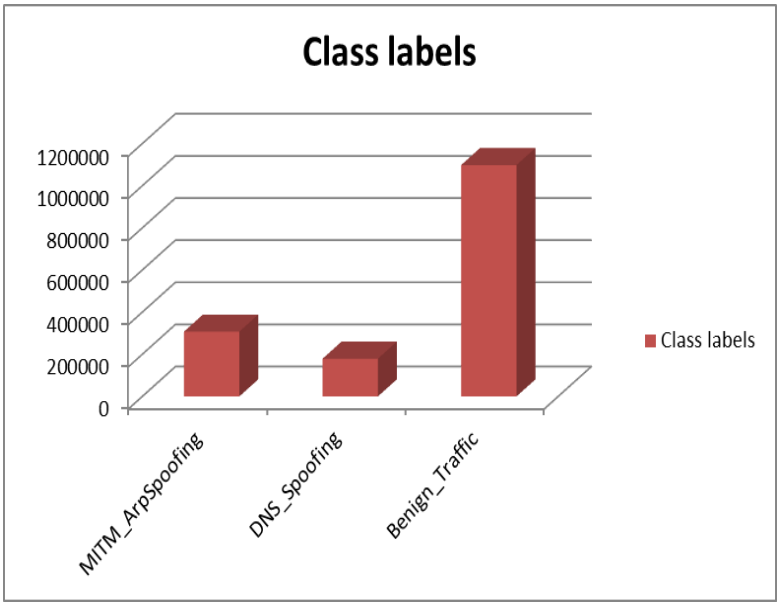

Fig. 2. The histogram of class labels.

4.2 Preprocessing Stage

Preprocessing is vital in the prediction process, as it repairs the data to ensure that it can be used in feature selection and the subsequent classification stage [25, 26]. More than one step has been applied.

a- NaN and infinity values are processed by filling them with the mean value.

b- The timestamp feature is deleted because it does not provide any predictive value for the target variable, the dataset is simplified, and overfitting is avoided by removing nonessential features such as timestamps.

c- The value of each instance is normalized via min–max normalization via equation (1):

$$X_{scaled} = \frac{X - X_{min}}{Xmin_{max}} \quad (3)$$

4.3 Feature Selection Stage

As mentioned previously, the dataset has forty-six features that were directly selected via feature selection methods. More than one model was applied for selecting the features by using the DT method, MI with different levels (all features, 10 features, 20 features, and 30 features). Each attribute has a role in detecting the spoofing attack. Figure 3 presents the attributes and their descriptions.

Feature No.	Feature Name	Feature description
F0	ts	Timestamp
F1	flow_duration	the packet's flow duration
F2	Header_Length	Header Length
F3	Protocol Type	IP, UDP, TCP, IGMP, ICMP, Unknown (Integers)
F4	Duration	Time-to-Live (ttl)
F5	Rate	Rate of packet transmission in a flow
F6	Srate	Rate of outbound packets transmission in a flow
F7	Drate	Rate of inbound packets transmission in a flow
F8	fin_flag_number	Fin flag value
F9	syn_flag_number	Syn flag value
F10	rst_flag_number	Rst flag value
F11	psh_flag_number	Psh flag value
F12	ack_flag_number	Ack flag value
F13	ece_flag_number	Ece flag value
F14	cwr_flag_number	Cwr flag value
F15	ack_count	Number of packets with ack flag set in the same flow
F16	syn_count	Number of packets with syn flag set in the same flow
F17	fin_count	Number of packets with fin flag set in the same flow
F18	urg_count	Number of packets with urg flag set in the same flow
F19	rst_count	Number of packets with rst flag set in the same flow
F20	HTTP	Indicates if the application layer protocol is HTTP
F21	HTTPS	Indicates if the application layer protocol is HTTPS
F22	DNS	Indicates if the application layer protocol is DNS
F23	Telnet	Indicates if the application layer protocol is Telnet
F24	SMTP	Indicates if the application layer protocol is SMTP
F25	SSH	Indicates if the application layer protocol is SSH
F26	IRC	Indicates if the application layer protocol is IRC
F27	TCP	Indicates if the transport layer protocol is TCP
F28	UDP	Indicates if the transport layer protocol is UDP
F29	DHCP	Indicates if the application layer protocol is DHCP
F30	ARP	Indicates if the link layer protocol is ARP
F31	ICMP	Indicates if the network layer protocol is ICMP
F32	IPv	Indicates if the network layer protocol is IP
F33	LLC	Indicates if the link layer protocol is LLC
F34	Tot sum	Summation of packets lengths in flow
F35	Min	Minimum packet length in the flow
F36	Max	Maximum packet length in the flow
F37	AVG	Average packet length in the flow
F38	Std	Standard deviation of packet length in the flow
F39	Tot size	Packet's length
F40	IAT	The time difference with the previous packet
F41	Number	The number of packets in the flow
F42	Magnitude	Average of the lengths of incoming packets in the flow + average of the lengths of outgoing packets in the flow)0.5
F43	Radius	(Variance of the lengths of incoming packets in the flow + variance of the lengths of outgoing packets in the flow)0.5
F44	Covariance	Covariance of the lengths of incoming and outgoing packets
F45	Variance	Variance of the lengths of incoming packets in the flow/ variance of the lengths of outgoing packets in the flow
F46	Weight	Number of incoming packets x Number of outgoing packets

Fig. 3 The attribute and its description

4.4 Classification Stage

After constructing the feature vector, two techniques are used for classifying the feature vectors: AdaBoost and XGBoost and CatBoost. The dataset was split into 70% for training and 30% for testing.

4.5 Importance of features

Two methods were applied for selecting the important features: DT and MI, as shown below:

4.5.1 Decision Tree (DT) Method

The DT was used to determine the importance of each feature. Therefore, each feature has a score and weight. Figure 4 shows the histogram of important features via DT, and Table II shows the values of importance for each feature via DT.

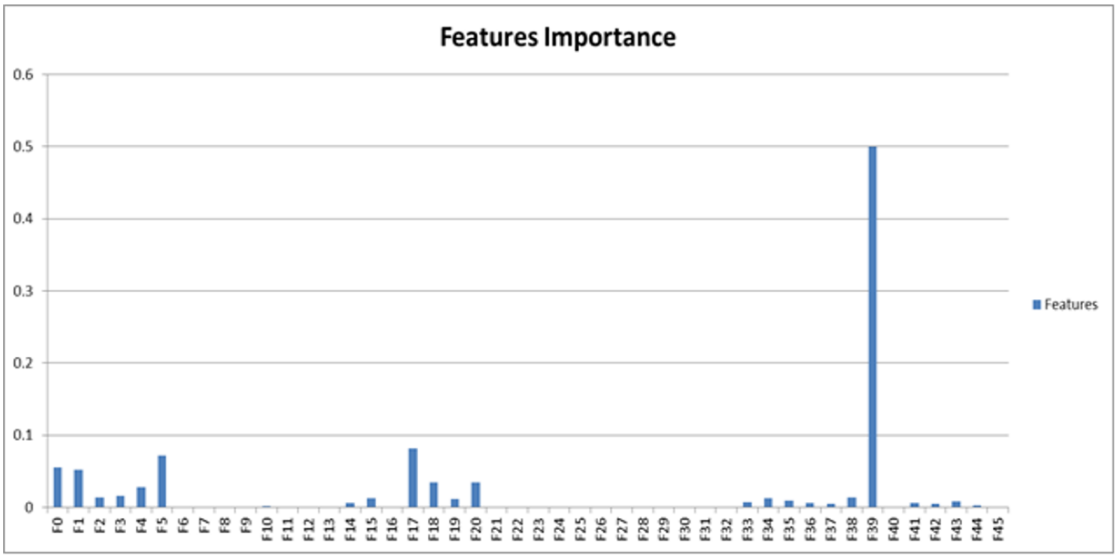

Fig 4. The histogram of important features using DT.

TABLE II. THE SCORE OF EACH ATTRIBUTE VIA THE DT

Feature No.	Feature score	Feature No.	Feature score
F0	0.055778	F23	0.000000
F1	0.052010	F24	0.000000
F2	0.014264	F25	0.000000
F3	0.015649	F26	0.000647
F4	0.028520	F27	0.000697
F5	0.071427	F28	0.000000
F6	0.000000	F29	0.000089
F7	0.000003	F30	0.000000
F8	0.000008	F31	0.000070
F9	0.000003	F32	0.000076
F10	0.001514	F33	0.006989
F11	0.001138	F34	0.012348
F12	0.000006	F35	0.009508
F13	0.000000	F36	0.005768
F14	0.005903	F37	0.005026
F15	0.012432	F38	0.013554
F16	0.001222	F39	0.500488
F17	0.081632	F40	0.000020
F18	0.034911	F41	0.005843
F19	0.011620	F42	0.005315
F20	0.034596	F43	0.007898
F21	0.000090	F44	0.002899
F22	0.000000	F45	0.000039

As shown in Table II and Figure 4, each feature had different values depending on the importance of those features. Additionally, the feature “Tot size” had more importance than the other features did, whereas eight features had no importance, as their values were zero, as shown in Table II.

4.5.2 The mutual information (MI) method

The MI was used to locate the importance of features. Therefore, each feature had a score. Figure 5 shows the histogram of important features via MI, and Table III shows the values of importance for each feature via MI.

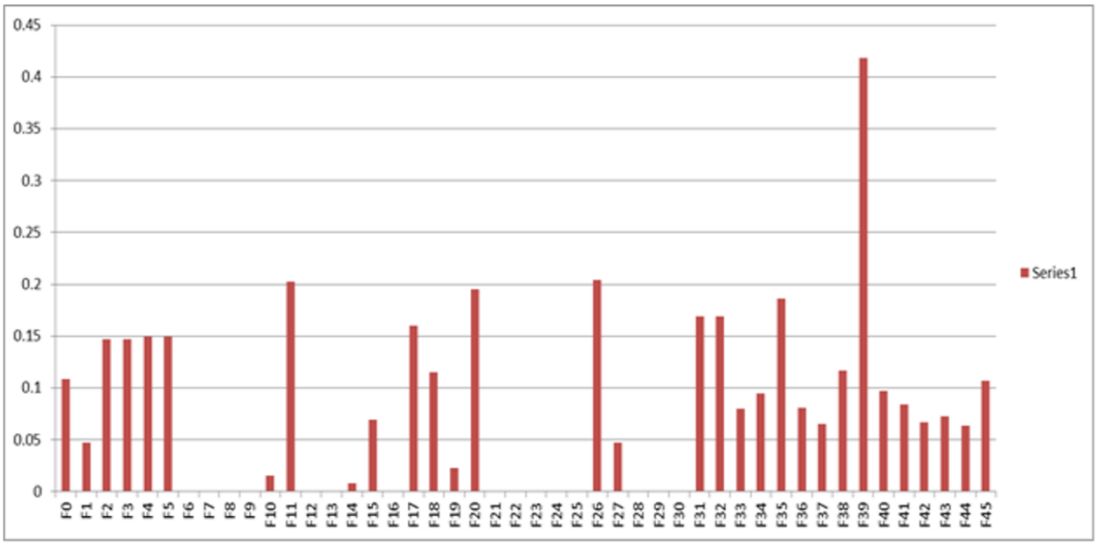

Fig. 5. The histogram of importance using MI

TABLE III. THE SCORE OF EACH ATTRIBUTE USING MI

Feature No.	Feature score	Feature No.	Feature score	Feature No.	Feature score	Feature No.	Feature score
F0	0.108598	F12	0.000000	F24	0.000000	F36	0.080814
F1	0.046902	F13	0.000082	F25	0.000272	F37	0.065374
F2	0.146682	F14	0.008392	F26	0.203900	F38	0.116935
F3	0.146838	F15	0.068966	F27	0.047309	F39	0.417959
F4	0.149632	F16	0.000606	F28	0.000000	F40	0.097364
F5	0.149649	F17	0.160025	F29	0.000098	F41	0.084085
F6	0.000000	F18	0.115367	F30	0.000000	F42	0.067149
F7	0.000000	F19	0.023081	F31	0.168586	F43	0.072808
F8	0.000244	F20	0.195496	F32	0.168724	F44	0.063391
F9	0.000000	F21	0.000000	F33	0.079652	F45	0.107048
F10	0.015161	F22	0.000201	F34	0.094788		
F11	0.202207	F23	0.000000	F35	0.186135		

The proposed work was carried out via Python (V.9.5) as a programming language and Jet Brains PyCharm (V.2018.2) as a framework. As mentioned, the proposed work contains three main stages. First, the data preprocessing involved processing the null values and infinite values and normalizing the data. Second, feature vectors are constructed by selecting important features via two methods of feature selection: DT and MI. Finally, the classification stage was carried out via three techniques (AdaBoost, XGBoost, and CatBoost) for classifying the feature vectors. Therefore, the spoofing attack was classified into four levels (for all features, the thirty most important features, the twenty most important features, and the ten most important features), which were selected via the DT and MI techniques shown below:

5. RESULTS

5.1 Results of Classifying all Features

The accuracies of the model when classifying all the features via the AdaBoost, XGBoost and CatBoost techniques were 86.66%, 94.32%, and 94.63%, respectively, whereas the execution times were 1517.80, 35.1052, and 2321.2705 seconds, respectively. Table IV shows the time and classification results for each technique. Figure 6 shows the accuracy and time for the three techniques (AdaBoost, XGBoost, and CatBoost) for all the features.

TABLE IV. THE TIME AND CLASSIFICATION REPORT OF EACH TECHNIQUE FOR ALL FEATURES

Technique of classification	Time in seconds	Precision	Recall	F1-score	Class label
AdaBoost	1517.8078	0.92	0.98	0.95	BenignTraffic (0)
		0.60	0.49	0.54	DNS_Spoofing (1)
		0.78	0.68	0.73	MITM-ArpSpoofing (2)
				0.87	Accuracy
XGBoost	35.1052	0.95	0.99	0.97	BenignTraffic (0)
		0.89	0.78	0.83	DNS_Spoofing (1)
		0.95	0.87	0.91	MITM-ArpSpoofing (2)
				94.00	Accuracy
CatBoost	2321.2705	95.20	99.05	97.09	BenignTraffic (0)
		89.62	79.76	84.40	DNS_Spoofing (1)
		95.10	87.47	91.12	MITM-ArpSpoofing (2)
				94.63	Accuracy

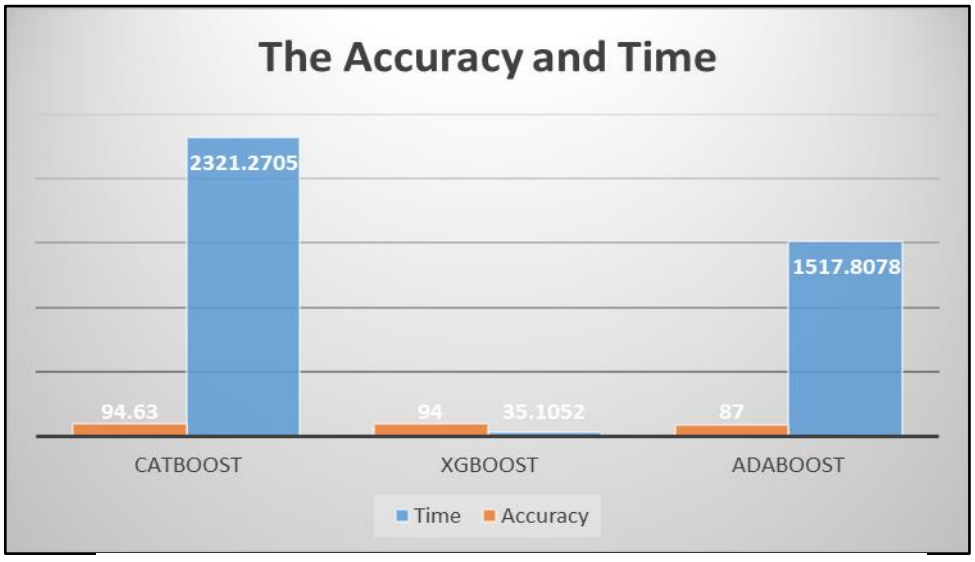

Fig. 6 The accuracy and the time for the three techniques

As shown in Table IV and Figure 6, the results using XGBoost and CatBoost had good accuracy and less time than the AdaBoost technique when classifying all the features.

5.2 Results of Classifying Selected Features via DT

In this section, the accuracy and time of classification using two techniques for selected important features (10, 20, and 30) selected via DT are shown. Table V shows the accuracy and time of the model when DT is used as the feature selection technique, whereas Table VI shows the classification results for each technique when DT is used as the feature selection technique. Figure 7 shows the accuracy and time for three techniques (AdaBoost, XGBoost, and CatBoost) for the important selected features via DT.

TABLE V. THE ACCURACY AND TIME OF EACH TECHNIQUE FOR SELECTED FEATURES USING DT

Technique of Classification	The No. of Selected Features	Accuracy (%)	Time in Seconds
AdaBoost	10	85.75	641.9710
	20	86.48	1148.5588
	30	86.69	1468.4216
XGBoosting	10	94.00	23.5969
	20	94.31	27.7528
	30	94.33	30.2052
CatBoost	10	94.44	812.3928
	20	94.57	1493.9287
	30	94.65	2103.3345

TABLE VI. CLASSIFICATION REPORT OF EACH TECHNIQUE FOR SELECTED FEATURES VIA DT

Technique of Classification	The No. of Selected Features	Precision	Recall	F1-score	Class label
AdaBoost	10	0.91	0.98	0.95	BenignTraffic (0)
		0.57	0.45	0.50	DNS_Spoofing (1)
		0.77	0.64	0.70	MITM-ArpSpoofing (2)
				0.86	Accuracy
	20	0.92	0.98	0.95	BenignTraffic (0)
		0.60	0.49	0.54	DNS_Spoofing (1)
		0.78	0.67	0.72	MITM-ArpSpoofing (2)
				0.86	Accuracy
	30	0.92	0.98	0.95	BenignTraffic (0)
		0.60	0.49	0.54	DNS_Spoofing (1)
		0.78	0.68	0.73	MITM-ArpSpoofing (2)
				0.87	Accuracy
XGBoost	10	0.95	0.99	0.97	BenignTraffic (0)
		0.88	0.77	0.82	DNS_Spoofing (1)
		0.95	0.86	0.90	MITM-ArpSpoofing (2)
				0.94	Accuracy
	20	0.95	0.99	0.97	BenignTraffic (0)
		0.89	0.78	0.83	DNS_Spoofing (1)
		0.95	0.87	0.91	MITM-ArpSpoofing (2)
				0.94	Accuracy
	30	0.95	0.99	0.97	BenignTraffic (0)
		0.89	0.78	0.83	DNS_Spoofing (1)
		0.95	0.87	0.91	MITM-ArpSpoofing (2)
				0.94	Accuracy
CatBoost	10	95.05	99.03	97.00	BenignTraffic (0)
		89.23	79.08	83.85	DNS_Spoofing (1)
		94.89	86.97	90.75	MITM-ArpSpoofing (2)
				94.44	Accuracy
	20	95.16	99.04	97.06	BenignTraffic (0)
		89.53	79.59	84.27	DNS_Spoofing (1)
		95.01	87.30	09.99	MITM-ArpSpoofing (2)
				94.57	Accuracy
	30	95.21	99.06	97.09	BenignTraffic (0)
		89.65	79.92	84.51	DNS_Spoofing (1)
		95.21	87.47	91.18	MITM-ArpSpoofing (2)
				94.65	Accuracy

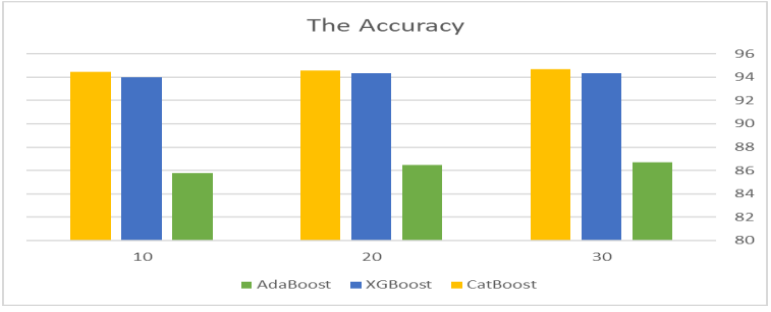

Fig. 7a The Accuracy of three techniques for selected features using DT

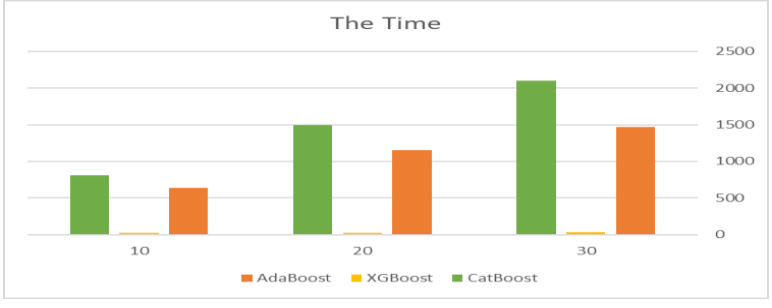

Fig. 7 b The time of three techniques for selected features using DT

Fig. 7 (a,b). The Accuracy and time of the three techniques for selected features using DT

As shown in Tables V and VI and Figure 7, the results using CatBoost and XGBoost had good accuracy and less time than the AdaBoost technique for classifying selected important features via DT for different numbers of important features (ten, twenty, and thirty). A comparison of XGBoost and CatBoost in time revealed that XGBoost required less time than did CatBoost for different numbers of important features.

5.3 Results of Classifying Selected Features via Mutual Information

In this section, the accuracy and time of classification using two techniques for two selected important features (10, 20, and 30) that were selected via MI are shown. Table VII shows the accuracy and time of the model using MI as the feature selection technique, while Table VIII shows the classification report for each technique using MI as the feature selection technique. Figure 8 shows the accuracy and time for three techniques (AdaBoost, XGBoost, and CatBoost) for the important selected features via MI.

TABLE VII. ACCURACY AND TIME OF EACH TECHNIQUE FOR SELECTED FEATURES VIA MUTUAL INFORMATION

Technique of Classification	The No. of Selected Features	Accuracy (%)	Time in Seconds
AdaBoost	10	88.69	1096.9785
	20	85.71	2525.3971
	30	88.72	2639.2806
XGBoosting	10	92.41	22.2754
	20	93.91	26.0892
	30	94.31	30.3309
CatBoost	10	92.45	957.3016
	20	94.19	1125.9692
	30	94.6	1286.4202

TABLE VIII. THE CLASSIFICATION REPORT OF EACH TECHNIQUE FOR SELECTED FEATURES USING MUTUAL INFORMATION

Technique of Classification	The No. of Selected Features	Precision	Recall	F1-score	Class label
AdaBoost	10	0.89	0.98	0.93	BenignTraffic (0)
		0.87	0.54	0.69	DNS_Spoofing (1)
		0.94	0.74	0.80	MITM-ArpSpoofing (2)
				0.89	Accuracy
	20	0.91	0.98	0.94	BenignTraffic (0)
		0.62	0.47	0.54	DNS_Spoofing (1)
		0.75	0.66	0.70	MITM-ArpSpoofing (2)
				0.86	Accuracy
	30	0.92	0.98	0.96	BenignTraffic (0)
		0.61	0.48	0.54	DNS_Spoofing (1)
		0.78	0.68	0.73	MITM-ArpSpoofing (2)
				0.88	Accuracy
	10	0.93	0.99	0.96	BenignTraffic (0)
		0.87	0.70	0.78	DNS_Spoofing (1)
		0.92	0.83	0.88	MITM-ArpSpoofing (2)
				0.92	Accuracy
		0.95	0.99	0.97	BenignTraffic (0)

XGBoost	20	0.89	0.77	0.82	DNS_Spoofing (1)
		0.94	0.86	0.90	MITM-ArpSpoofing (2)
				0.94	Accuracy
	30	0.95	0.99	0.97	BenignTraffic (0)
		0.89	0.78	0.83	DNS_Spoofing (1)
		0.95	0.87	0.91	MITM-ArpSpoofing (2)
				0.94	Accuracy
Catboost	10	93.03	98.47	95.67	BenignTraffic (0)
		87.36	70.99	78.33	DNS_Spoofing (1)
		92.69	83.43	87.82	MITM-ArpSpoofing (2)
				92.45	Accuracy
	20	94.80	98.88	96.80	BenignTraffic (0)
		88.97	78.45	83.38	DNS_Spoofing (1)
		94.61	86.58	90.42	MITM-ArpSpoofing (2)
				94.19	Accuracy
	30	95.11	99.04	97.03	BenignTraffic (0)
		89.65	79.52	84.28	DNS_Spoofing (1)
		95.15	87.33	91.07	MITM-ArpSpoofing (2)
				94.57	Accuracy

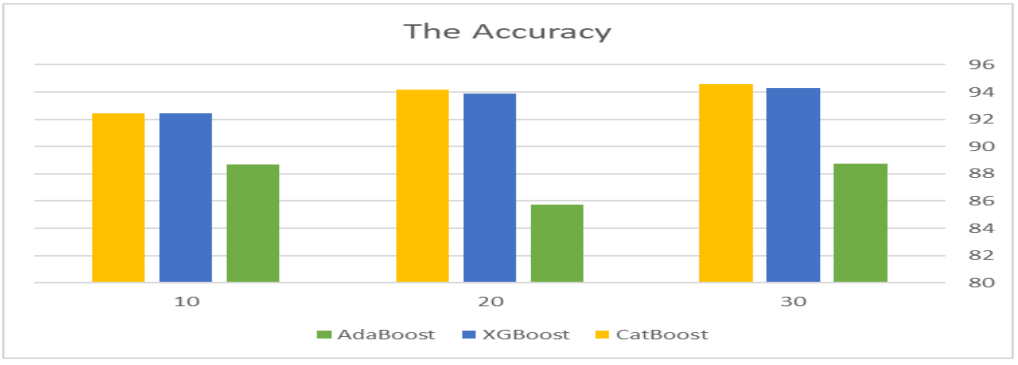

Fig. 8.a The Accuracy of three techniques for selected features using MI

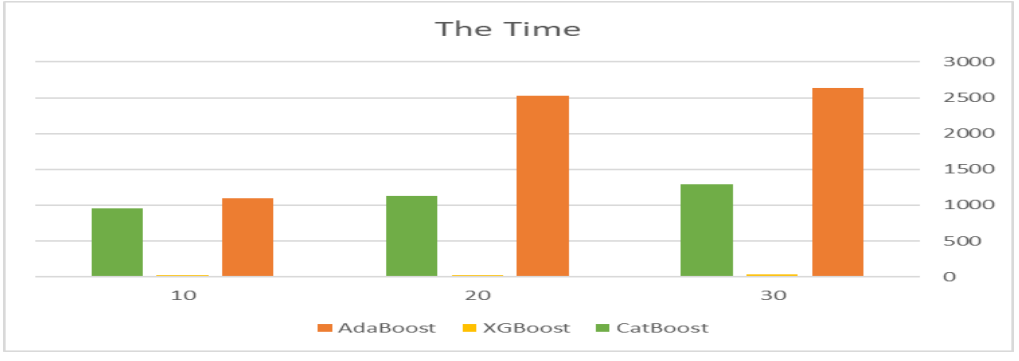

Fig 8.b The Time of three techniques for selected features using MI

Fig. 8 (a,b). The Accuracy and time of two techniques for selected features using MI

As shown in Tables VII and VIII and Figure 8, the results using XGBoost and CatBoost had good accuracy and less time than the AdaBoost technique for classifying selected important features via MI for different numbers of important features (ten, twenty, and thirty). A comparison of XGBoost and CatBoost in time revealed that XGBoost required less time than did CatBoost for different numbers of important features.

5.4 Discussion of the Results

As shown in the previous tables and figures, CatBoost outperforms XGBoost and AdaBoost in terms of accuracy because of its unique ordered boosting approach, efficient handling of categorical features, and strong regularization techniques that prevent overfitting compared with XGBoost and AdaBoost.

The structure of CatBoost has a symmetric tree-building strategy, which ensures better generalization, but these advantages come at the cost of longer training times because the ordered boosting mechanism, combined with automatic feature transformations and additional regularization, requires significantly more computations per iteration than XGBoost does. However, reducing the training time if important and reducing tree depth, using GPU acceleration, and lowering the number of iterations can help speed up training without a significant loss in accuracy.

By comparing the two feature selection methods (DT and MI), the selection of features via DT had good accuracy compared with that via MI because of the ability of the DT to address large and complex datasets and the different natures of the datasets. Figure 9 shows the comparison between all the features and the feature selection techniques in terms of accuracy when three classifier techniques are used.

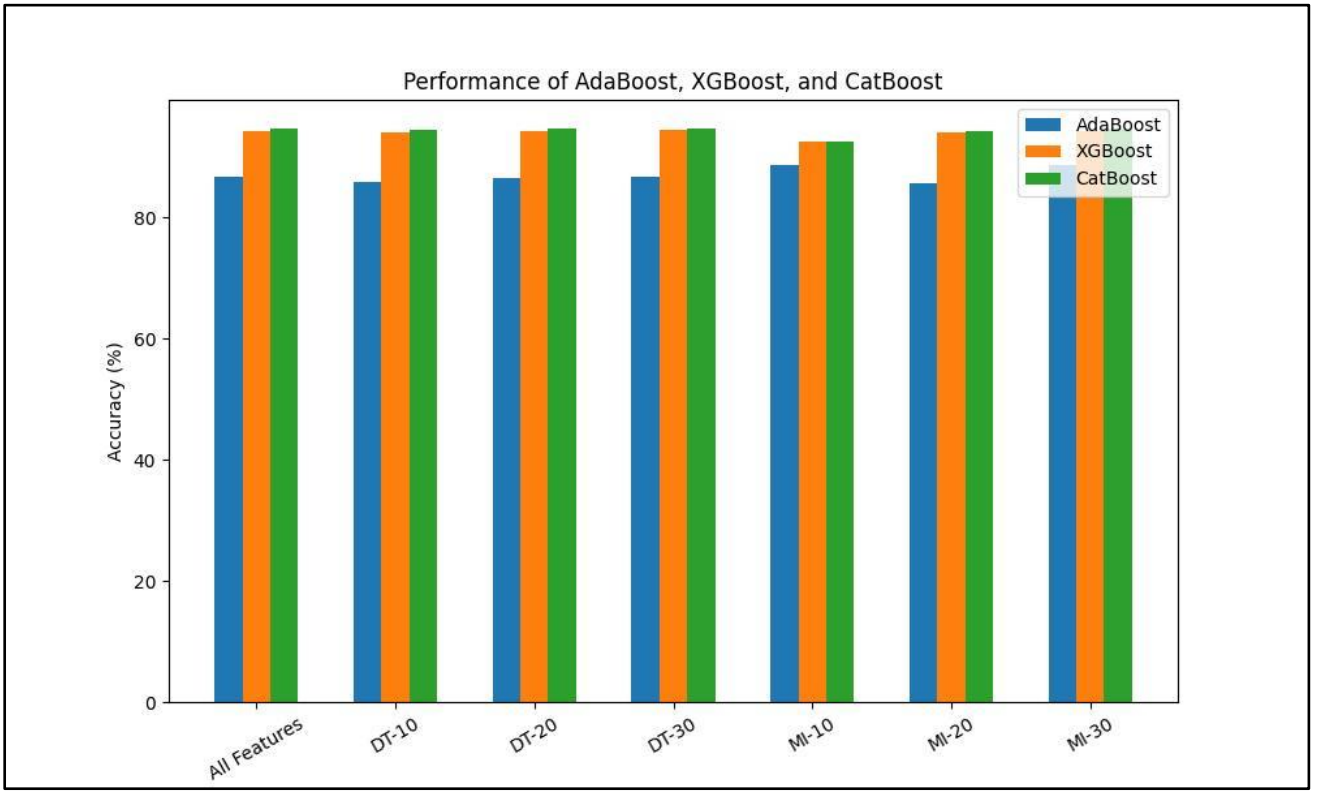

Fig. 9. The Comparison between all features and feature selection techniques in accuracy using three classifier techniques.

The number of selected features affects the resulting accuracy. The best number of features was thirty when important features were selected via DT and MI. As shown in the previous tables, the values of the other performance measures (F1_score, precision, and recall) for the three class labels were the highest for the BenignTraffic class label and the lowest for the DNS_Spoofing class label because the number of BenignTraffic class labels was greater than that of the other class labels, and the number of DNS_Spoofing class labels was less than that of the other class labels. The time required for the selection method, especially DT, was less than the time needed for classifying all the features.

By comparing the related works and the proposed models in this work, the proposed system applied boosting techniques (CatBoost, XGBoost, AdaBoost) with feature selection (DT, MI). The goal is to improve accuracy in classification tasks by selecting the most relevant features. Compared with related studies, which address various spoofing detection techniques across different domains (IoT networks, phishing detection, facial recognition, GPS spoofing, and autonomous vehicles), the proposed system is used for multiclass classification, achieving high accuracy while being computationally efficient for structured data. Compared with related methods, it is more general-purpose than phishing detection, facial antispoofing, and GPS spoofing detection. While deep learning-based models (CNNs) in related works achieve higher accuracy (99%), they require significantly longer training times and large datasets. The proposed system balances accuracy and efficiency well, making it an ideal choice for structured data classification tasks.

5.5 Analysis of the impact of features on model accuracy

In this subsection, we analyse how different features impact model accuracy, focusing on a decision tree (DT) and mutual information (MI) for feature selection. These two methods identify significant features for spoofing attack detection. Not all features contribute equally to accuracy; selecting relevant features improves performance while reducing complexity. A DT assigns importance scores on the basis of its data splitting ability. "Tot size" had the highest score, which is critical for spoofing detection. XGBoost achieved 94.32% accuracy with 46 features. The top 10 features maintained 94.00% accuracy. Increasing the number of features to 20 or 30 slightly improved the accuracy. Selecting fewer features significantly reduces the execution time. MI measures the dependency between features and the target variable. "Tot size" and "Flow duration" had high MI scores. Using all 46 features, XGBoost had 94.32% accuracy. The top 10 features with the MI resulted in 92.41% accuracy. The DT generally outperforms the MI in terms of accuracy. Both methods showed that selecting important features maintains accuracy while reducing cost. "Tot size" was consistently important in spoofing detection. Features with zero importance were irrelevant. Using a fraction of features is sufficient for near-optimal accuracy. DT provides slightly better accuracy than MI with comparable efficiency. Selecting relevant features reduces the execution time while maintaining performance.

5.6 Practical Applications of the Proposed Model

Understanding findings in the real world is crucial for research relevance. Discussing practical applications and considerations for deploying models in advanced IoT systems.

1. Application in IoT Security: Enhances security in vulnerable IoT environments. Integrating the model into security frameworks enables real-time monitoring and automated threat detection.
2. Integration with Existing Security Solutions: Enhances security posture by improving detection rates and reducing false positives.
3. Scalability across diverse environments: The model is flexible and adaptable for various IoT applications.

5. CONCLUSION AND FUTURE WORK

The revolution of the IoT addresses connecting billions of devices and generating an unprecedented volume of data, hence addressing significant security challenges. IoT devices generally possess limited security capabilities, making them vulnerable to cyberattacks. Spoofing attacks are among the cyber threats that target the IoT environment. In this work, the Spoofing attack was detected via two techniques of feature selection, DT and MI, and three classification techniques, AdaBoost, XGBoost, and CatBoost. The proposed work generally contains three main stages. The results showed that the feature selection technique had a positive effect on both time and accuracy without introducing bias. The optimal number of features played an important role in achieving accurate results without bias and in reducing the processing time. The CatBoost classifier outperformed the AdaBoost and XGBoost classifiers in terms of accuracy.

For future work, deep learning could be used for better detection. Test models on diverse datasets for generalizability. Explore advanced feature engineering. Conduct real-world studies in IoT environments and develop user-friendly interfaces for deploying models. For future work, deep learning could be used for better detection. Test models on diverse datasets for generalizability. Explore advanced feature engineering. Conduct real-world studies in IoT environments and develop user-friendly interfaces for deploying models.

Conflicts of interest:
"No conflicts of interest."
Funding
The presented research did not receive any financial support.
Acknowledgments
The authors extend their sincere thanks to the Foundation for its cooperation and provision of the necessary facilities that contributed to the successful completion of this research.

References

[1] Z. Abou El Houda, B. Brik, and S.-M. Senouci, "A novel iot-based explainable deep learning framework for intrusion detection systems," IEEE Internet of Things Magazine, vol. 5, pp. 20-23, 2022. https://doi.org/10.1109/IOTM.005.2200028
[2] S. Abbas, A. Al Hejaili, G. A. Sampedro, M. Abisado, A. Almadhor, T. Shahzad, et al., "A Novel Federated Edge Learning Approach for Detecting Cyberattacks in IoT Infrastructures," IEEE Access, 2023. https://doi.org/10.1109/ACCESS.2023.3318866
[3] W. H. Abdulsalam, S. Mashhadani, S. S. Hussein, and A. A. Hashim, "Artificial Intelligence Techniques to Identify Individuals through Palm Image Recognition," International Journal of Mathematics and Computer Science, vol. 20, no. 1, pp. 165-171, 2025. doi:10.69793/ijmcs/01.2025/abdulsalam
[4] S. Mashhadani, W. H. Abdulsalam, I. Alhakam, O. A. Hassen, and S. M. Darwish, "An enhanced document source identification system for printer forensic applications based on the boosted quantum KNN classifier," Eng. Technol. Appl. Sci. Res., vol. 15, no. 1, pp. 19983–19991, 2025, doi: 10.48084/etasr.9420.
[5] A. S. Wisam, K. J. Khalid,, and K. Q. Luheb, "A Proposed Hybrid Text Cryptographic Method Using Circular Queue," IJCET, vol. 9, no. 7, pp. 1123-1132, 2018.
[6] R. H. Ali and W. H. Abdulsalam, "Attention-Deficit Hyperactivity Disorder Prediction by Artificial Intelligence Techniques," Iraqi Journal of Science, vol. 65, no. 9, pp. 5281-5294, 2024. doi:10.24996/ijs.2024.65.9.39
[7] W. H. Abdulsalam, R. S. Alhamdani, and M. N. Abdullah, "Speech Emotion Recognition Using Minimum Extracted Features," 1st Annual International Conference on Information and Sciences (AiCIS), pp. 58-61, 2018. https://doi.org/10.1109/AiCIS.2018.00023
[8] R. H. Ali, "Artificial intelligence techniques to predict the performance of teachers for kindergarten: Iraq as a case study," Evolutionary Intelligence, vol. 17, pp. 313-325, 2024. https://doi.org/10.1007/s12065-022-00731-0
[9] F. Khan, A. A. Al-Atawi, A. Alomari, A. Alsirhani, M. M. Alshahrani, J. Khan, et al., "Development of a Model for Spoofing Attacks in Internet of Things," Mathematics, vol. 10, no. 19, 2022. https://doi.org/10.3390/math10193686
[10] A. B. Altamimi, M. Ahmed, W. Khan, M. Alsaffar, A. Ahmad, Z. H. Khan, et al., "PhishCatcher: Client-Side Defense Against Web Spoofing Attacks Using Machine Learning," IEEE Access, 2023.
[11] X. Cheng, J. Zhou, X. Zhao, H. Wang, and Y. Li, "A presentation attack detection network based on dynamic convolution and multilevel feature fusion with security and reliability," Future Generation Computer Systems, vol. 146, pp. 114-121, 2023. https://doi.org/10.1016/j.future.2023.04.012
[12] X. Wei, M. N. Aman, and B. Sikdar, "A Light-Weight Technique to Detect GPS Spoofing Using Attenuated Signal Envelopes," IEEE Open Journal of the Computer Society, 2023. https://doi.org/10.1109/OJCS.2023.3278496
[13] M. Shabbir, M. Kamal, Z. Ullah, and M. M. Khan, "Securing Autonomous Vehicles Against GPS Spoofing Attacks: A Deep Learning Approach," IEEE Access, 2023. https://doi.org/10.1109/ACCESS.2023.3319514
[14] H. H. Htun, M. Biehl, and N. Petkov, "Survey of feature selection and extraction techniques for stock market prediction," Financial Innovation, vol. 9, pp. 26, 2023. https://doi.org/10.1186/s40854-022-00441-7
[15] D. K. Ghurkan and A. A. Abdulrahman, "Construct an Efficient DDoS Attack Detection System Based on RF-C4. 5-GridSearchCV," pp. 120-124, 2022. https://doi.org/10.1109/IICCIT55816.2022.10010645
[16] B. H. Majeed, W. H. Abdulsalam, Z. H. Ibrahim, R. H. Ali, and S. Mashhadani, "Digital Intelligence for University Students Using Artificial Intelligence Techniques," International Journal of Computing and Digital Systems, vol. 17, no. 1, pp. 1–16, 2024. http://dx.doi.org/10.12785/ijcds/1571029446
[17] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley, 1991.
[18] A. Maslan, K. M. B. Mohamad, and F. B. M. Foozy, "Feature selection for DDoS detection using classification machine learning techniques," IAES International Journal of Artificial Intelligence, vol. 9, pp. 137, 2020. https://doi.org/10.11591/ijai.v9.i1.pp137-145
[19] V. C.P. and A. A. Chikkamannur, "Feature Selection: An Empirical Study," International Journal of Engineering Trends and Technology, vol. 69, no. 2, pp. 165-170, 2021. https://doi.org/10.14445/22315381/IJETT-V69I2P223

[20] A. S. S. Abinayaa et al., "Securing the edge: Catboost classifier optimized by the Lyrebird algorithm to detect denial of service attacks in Internet of Things-based wireless sensor networks," Future Internet, vol. 16, no. 10, 2024, doi: 10.3390/fi16100381.
[21] R. M. Zaki and I. S. Naser, "Hybrid classifier for detecting zero-day attacks on IoT networks," Mesopotamian J. CyberSecurity, vol. 4, no. 3, pp. 59–74, 2024, doi: 10.58496/MJCS/2024/016.
[22] W. H. Abdulsalam, R. H. Ali, S. H. Jadooa, and S. S. Hussein, "Automated glaucoma detection techniques: A literature review," Eng. Technol. Appl. Sci. Res., vol. 15, no. 1, pp. 19891–19897, 2025, doi: 10.48084/etasr.9316.
[23] K. K.Ruma et al., "The optimization problems for public key cryptosystem," AIP Conf. Proc., vol. 3264, no. 1, 2025, doi: 10.1063/5.0259007.
[24] 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," 2023. https://doi.org/10.20944/preprints202305.0443.v1
[25] I. A. F. Alzuabidi et al., "Deep learning for the recognition of skin cancer," in Proc. 2nd Int. Conf. Adv. Comput. Appl. (ACA), 2023, pp. 1–6, doi: 10.1109/ACA57612.2023.10346847.
[26] A. M. Salman et al., "Enhancing cybersecurity with machine learning: A hybrid approach for anomaly detection and threat prediction," Mesopotamian J. CyberSecurity, vol. 5, no. 1, pp. 202–215, 2025, doi: 10.58496/MJCS/2025/014.
[27] A. Alsajri, A. Steiti, and H. A. Salman , Trans., “Enhancing IoT Security to Leveraging ML for DDoS Attack Prevention in Distributed Network Routing”, BJIoT, vol. 2023, pp. 74–84, Oct. 2023, doi: 10.58496/BJIoT/2023/010.
[28] L. Hussain, “Fortifying AI Against Cyber Threats Advancing Resilient Systems to Combat Adversarial Attacks”, EDRAAK, vol. 2024, pp. 26–31, Mar. 2024, doi: 10.70470/EDRAAK/2024/004.