Computer Networks and Communications
http://ojs.wiserpub.com/index.php/CNC/

Research Article

# Mirai Botnet Multi-Class Attack Detection Through Machine Learning and Feature Selection

**Hayelom Gebrye[1,2*], Yong Wang[1], Fagen Li[1]**

[1] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2] Information Technology, Raya University, Maychew P.O. Box 92, Ethiopia
[*] Correspondence: hmuleta@gmail.com

**Received:** 7 September 2025; **Revised:** 4 December 2025; **Accepted:** 10 December 2025; **Published:** 4 January 2026

**Abstract:** Machine Learning (ML), which provides timely insights for efficient threat identification and prevention, has become a crucial cybersecurity technology. However, the growing number of features in modern datasets increases both processing complexity and computational cost. By concentrating on feature selection and extraction techniques, this study seeks to improve the efficacy of Mirai botnet analysis. A data extraction approach that transforms Internet of Things (IoT) network attack datasets (in Packet Capture (PCAP) format) to flow-driven attributes (in Comma-Separated Values (CSV) format) was presented in our earlier work. A unique framework for effectively developing, assessing, and analyzing Mirai botnet assaults in IoT networks is provided by the obtained and labeled features of the Mirai-based multi-class IoT botnet dataset. In this study, experiments were conducted using the extended Mirai-based multi-class dataset and the widely used Network Security Laboratory-Knowledge Discovery in Databases (NSL-KDD) dataset for comparison. The results of both experiments demonstrate that Random Forest Feature Importance (RFFI) outperforms the Boruta feature selection algorithm. Furthermore, the random forest and decision tree models achieved superior performance in all tests, attaining 100% accuracy in the first experiment using the extended dataset. These findings highlight the importance of selecting relevant features, rather than using all available attributes, to enhance detection performance and computational efficiency.

***Keywords***: Boruta algorithm, Internet of Things (IoT) network, Machine Learning (ML), Mirai botnet, multi-class classification

## 1. Introduction

The Internet of Things (IoT) is currently becoming an essential element of everyday life. However, many users neglect to change the default credentials of their smart devices, leaving default configurations as vulnerabilities that can be exploited by threat actors [1]. Globally, the number of IoT devices is increasing rapidly [2], which heightens security risks and contributes to the growing prevalence of botnet attacks. A botnet is a network of compromised Internet-connected smart devices, each running malicious software (bots) that allow attackers to remotely control the infected hosts. Prominent examples, such as Mirai and BASHLITE, exploit vulnerable device configurations to launch Distributed Denial-of-Service (DDoS) attacks [1]. To detect and prevent such malicious activities or unauthorized access, Intrusion Detection Systems (IDS) play a critical role [3].


DOI: https://doi.org/10.37256/cnc.4120268552

Machine Learning (ML) has emerged as a key component of cybersecurity in the twenty-first century, as it can rapidly generate actionable insights for attack detection and prevention [3]. Network devices typically record security-related data as Packet Capture (PCAP) files, which are subsequently converted into structured formats such as Comma-Separated Values (CSV) for ML-based analysis. These datasets enable the identification of anomalous network activity and provide security experts with timely alerts. However, Mirai botnet detection faces significant challenges due to the enormous volume, high dimensionality, and redundancy of IoT network traffic. Such extensive data often degrade classification performance and slow down the detection process [4]. While earlier ML applications commonly utilized fewer than 40 features [5], modern datasets may contain hundreds of features, substantially increasing computational complexity and resource requirements. Consequently, the "curse of dimensionality" has been highlighted by numerous studies as a major issue in high-dimensional datasets with relatively few samples [6].

The procedure of lowering a dataset's dimensionality by keeping those which are most relevant attributes while removing unnecessary or repetitive components is known as Feature Selection (FS) [7, 8]. FS methods generally evaluate feature relevance in two ways: (i) by assessing the importance of individual features, and (ii) by identifying and removing features that are highly correlated or duplicated. By selecting a smaller yet representative subset of features, FS enhances a classifier's predictive performance—especially in scenarios where training data is limited [9, 10]. In supervised FS, the relationship between features and class labels guides the selection process, ensuring that the chosen subset effectively contributes to accurate classification [8]. Overall, FS plays a role that extends beyond simply building a sophisticated predictive model; models trained on raw or irrelevant data often suffer from reduced accuracy and poor generalization [9].

Many researchers have investigated various approaches to address the challenges of high-dimensional data by identifying the most relevant and non-redundant features [4, 7–10]. FS not only improves precision, accuracy, and recall but also reduces computational costs, simplifies model interpretation, and enhances overall model performance. The main motivations for employing FS include reducing the number of parameters to simplify the model, decreasing training time, preventing overfitting through better generalization, and alleviating the curse of dimensionality. Datasets often contain diverse variables that influence their predictive quality and usefulness during preprocessing and evaluation [11]. Moreover, addressing class imbalance and selecting optimal models that achieve high predictive accuracy with minimal error remain critical challenges in classification tasks.

To address these challenges, FS filters and removes superfluous or irrelevant features, thereby reducing dataset complexity and retaining only the most informative attributes for precise attack classification. FS is typically applied during the preprocessing stage to enhance overall anomaly detection performance and classification accuracy. In this study, we propose a novel framework for IoT botnet detection that integrates multiple classification techniques with advanced feature selection methods. Unlike previous approaches that often rely on a single classifier or feature selection algorithm, our framework combines RFFI and Boruta with four classifiers, Random Forest (RF), K-Nearest Neighbors (KNN), Gaussian Naïve Bayes (GNB), and Decision Tree (DT), to comprehensively evaluate feature relevance and classifier performance. Furthermore, our approach reduces computational cost while maintaining high detection accuracy by emphasizing the effective identification of critical features. The robustness and generalizability of the framework across various IoT traffic scenarios are validated using both the widely adopted NSL-KDD dataset and a recently developed Mirai-based multi-class dataset. Additionally, by incorporating computational efficiency analysis (runtime and memory), our method addresses practical considerations for deployment in resource-constrained IoT systems—an aspect often overlooked in previous studies. The proposed models accurately detect and classify Mirai-based IoT botnet attacks, including Synchronize (SYN)-Flooding, Acknowledgement (ACK)-Flooding, and Hypertext Transfer Protocol (HTTP)-Flooding, distinguishing between benign and malicious network traffic. Moreover, the results demonstrate that classical ML models achieve significantly improved performance when trained on carefully selected subsets of critical features. Experimental comparisons are conducted using the NSL-KDD dataset [12] and a custom Mirai-based dataset developed in our previous work [13], adapted from [14] and available in [15].

The following points represent the findings of the paper.

1. A newly extracted multi-class dataset comprising normal traffic and Mirai-based attacks, specifically SYN-Flooding, ACK-Flooding, and HTTP-Flooding has been developed to support the training and evaluation of machine learning models.

2. Network traffic behavioral patterns have been analyzed to identify and select the most relevant features, thereby reducing dimensionality, enhancing the performance of classifiers and anomaly detection algorithms, and achieving improved overall detection accuracy.

3. A performance comparison was conducted by applying machine learning models to our dataset and the widely used NSL-KDD dataset.

The following is how the paper is structured: The relevant work is reviewed in Section 2. The suggested approach, which includes the procedures and methods employed in the study as well as the materials used, is presented in Section 3. The outcome and a summary of the findings are presented in Section 4. The paper comes to an end and future study are provided in Section 5.

## 2. Related works

Numerous studies have concentrated on enhancing the effectiveness of supervised learning techniques for feature selection-based attack categorization. High-dimensional data often contains attributes that are irrelevant or redundant, which can compromise the reliability of predictions. When training models for classification or anomaly detection, it is crucial to consider the dimensionality of the data (i.e., the number of features) to achieve efficient training, enhance the performance of classifiers or outlier detectors, and ensure correct interpretation of results. FS algorithms provide a systematic approach for selecting attributes based on their relevance [16]. Related studies [17] have demonstrated that traditional detection techniques often fail on high-dimensional data due to the curse of dimensionality. This raises an important question: how can one select an optimal set of critical features and choose the most appropriate FS algorithm when addressing classification and anomaly detection tasks in high-dimensional datasets? The related research works are presented in the following section.

### 2.1 *FS and machine learning*

In studies [18, 19], Principal Component Analysis (PCA) was employed for feature reduction to identify an optimal feature set. During preprocessing, categorical features were mapped to numerical values using feature scaling. The NSL-KDD dataset, originally containing 41 features, was reduced to 23 features after FS. A Radial Basis Function (RBF) was used as the kernel to handle the high-dimensional features, and the model's accuracy was compared with and without FS, showing improved performance when FS was applied. In [18], the KDD CUP 99 dataset was used; preprocessing involved feature normalization, and PCA was combined with Support Vector Machine (SVM) to optimize kernel parameters and enable automatic parameter selection. An ensemble of Decision Tree classifiers and rule-based approaches was proposed in [20], utilizing three classifiers—REPTree, JRip, and the Forest algorithm—for classifying network traffic as normal or malicious. Experiments were conducted using the CICIDS2017 dataset, which contains 14 attack categories. The proposed model achieved a detection rate of 94.4%, accuracy of 96.9%, and a false alarm rate of 1.1%.

Network-based intrusion detection using a Random Forest classifier was investigated in [21]. FS was performed by measuring feature importance with the RF algorithm, and attack classification was subsequently carried out on the reduced feature set. Experiments on the CICIDS2017 dataset demonstrated that the RF classifier achieved an accuracy of 97.34% using the selected features, highlighting the effectiveness of FS in improving classification performance and computational efficiency. Authors produced flooding-based DDoS assault dataset specifically intended for classification into several groups, and they examined the efficacy of flexible ML models. There were three thorough examinations. In every experimental group, they showed that the random forest strategy had a predictive level of over 90.

In [22], Recursive Feature Elimination (RFE) has been applied as an FS mechanism to collect and order properties by relevance. The top five attributes were selected for classifying from the UNSWNB15 dataset. The results demonstrated that preprocessing the data and applying FS improved classifier performance. PCA was also employed for dimensionality reduction on the NSL-KDD and GureKDD datasets in [23], reducing the number of features from 41 to 31, which led to improved classification accuracy and a lower false alarm rate. In [24], RFE was combined with SVM and RF for feature selection. Experiments on the NSL-KDD dataset selected 13 out of 41 attributes for attack classification, and

the study evaluated the effectiveness of anomaly detection using varying sample sizes. Similarly, Ahmad et al. [25] investigated different sample sizes of the NSL-KDD dataset using SVM, RF, and Extreme Learning Machine (ELM) classifiers, showing that ELM outperformed the other methods. In [26], binary classifiers including SVM, Stochastic Gradient Descent, Sequential Model, Logistic Regression (LR), and RF were applied to the NSL-KDD dataset. Experiments were performed both with and without feature encoding, and the results indicated that RF achieved the fewest false negatives and outperformed the other classifiers.

Similarly, a Least Squares Support Vector Machine (LS-SVM) model was developed using FS based on mutual information and tested on three datasets: KDD CUP 99, NSL-KDD, and Kyoto [27]. The results demonstrated that the LS-SVM model built with mutual information FS is computationally more efficient compared to other classification techniques [27]. The detection of various attack types is a complex process that often relies on the analysis of large datasets. These large datasets can be divided into representative subgroups using sampling techniques, also referred to as optimum allocation [28]. In [28], a framework combining optimum allocation with LS-SVM was proposed to extract and validate samples for intrusion detection. Experiments conducted on the KDD CUP 99 dataset showed that the proposed method is effective for both static and incremental datasets.

## 2.2 *Hybrid FS*

A hybrid intrusion detection system model was proposed in [29], integrating DT and one-class Support Vector Machine (SVM) classifiers. In this approach, the DT classifier is used for signature-based attack detection, while the one-class SVM handles anomaly-based attack detection. The model is designed to identify both known and novel attacks. Experiments conducted on the NSL-KDD and AFDA datasets achieved accuracies of 83.24% and 97.04%, respectively. A stacked ensemble classification technique was proposed in [30] for network-based intrusion detection. The stacked model combines RF, LR, KNN, and SVM classifiers to derive optimal predictions based on the collective learning of the classifiers. Experiments were performed using flow-based datasets, namely UNSW-NB15 and UGR-16, and the model achieved an accuracy of 94%. In [31], a binary classification approach using SVM was implemented to distinguish between normal and anomalous network traffic. Feature selection was performed using the information gain technique, selecting 10 out of 41 features for training the SVM with a radial basis function kernel. The proposed approach achieved an accuracy of 96.34%.

To address the increasing volume of network traffic, an ensemble FS method was proposed in [32], which combines the outcomes of multiple filter-based FS techniques to achieve optimal feature selection. In this ensemble-based multi-filter framework, four filter methods—Information Gain, Relief-F, Gain Ratio, and Chi-Square—are applied to reduce the NSL-KDD dataset from 41 attributes to 13. Each FS method ranks the features in the original dataset, and the top features are selected based on these rankings. A threshold is defined to measure the frequency of occurrence of each feature across the methods, and majority voting is used to determine the final threshold value. During the generation of the combined feature subset, a counter identifies features meeting the threshold, resulting in the selection of the most relevant attributes. The ensemble FS approach, combined with a Decision Tree classifier, achieved an accuracy of 99.67%, demonstrating the effectiveness of combining multiple FS methods for high-dimensional network traffic datasets.

Despite numerous studies addressing the attack classification problem, existing models exhibit several limitations. For instance, Kabir et al. [28], Rajagopal et al. [30], Kumar et al. [31], and Osanaiye et al. [32] do not address the class imbalance commonly present in intrusion detection datasets. Furthermore, the selection of training examples is often performed randomly rather than using a systematic approach. Many related works are also constrained by the use of outdated datasets, such as NSL-KDD and KDD CUP 99. Additionally, reported results are typically evaluated on a single dataset rather than being validated across multiple datasets, limiting the generalizability of the findings.

Recent advances in IoT and network anomaly detection have leveraged deep learning and federated learning techniques. A resilience baseline for distributed detection pipelines was presented by the authors [33], robustness diagnostics for ML security were introduced by [34], and defense-oriented ML techniques were examined by [35]. In addition to these efforts, current IoT botnet detection techniques often reduce high-dimensional traffic data using feature-selection or dimensionality-reduction techniques like Boruta, RFE, Mutual Information, LASSO, and PCA. To enhance model privacy and efficiency across distributed IoT devices, Xia et al. [36] and Gebrye et al. [37] proposed an abnormal traffic detection

approach that integrates depthwise separable CNNs with federated learning. In their survey of CNN-based Android malware detection methods, Shu et al. [38] highlighted the importance of feature representation and model design in malware and network threat detection. Similarly, Dong et al. [39] demonstrated the benefits of combining limited labeled data with reinforcement signals by introducing a semi-supervised deep reinforcement learning model for abnormal traffic detection. Although these studies achieved promising results, they primarily focus on distributed frameworks and deep learning approaches. In contrast, our study emphasizes robust feature selection using Boruta and ensemble classifiers, providing a highly interpretable, computationally efficient, and accurate method for detecting Mirai-based and other network attacks, thereby complementing and extending the existing literature.

# 3. The proposed approach

This section describes the suggested research technique, comprising the general procedure and resources used in the study.

## 3.1 *Research workflow*

Figure 1 explains the methods and materials of the study. The input data, consisting of three Mirai-based attacks, were first converted from PCAP to CSV (a format better suited for machine learning) and then merged with benign data. Further, this work uses the popular and most common dataset called NSLKDD [12] to conduct the experiment and compare the performance with the newly generated dataset during the study. Second, our effort utilizes two feature selection methods to handpicked significance attributes. The success is the contrast among the different supervised learning models for classification analysis. Selecting an optimal subset of features from a larger set is a computationally challenging task, particularly when the evaluation process becomes complex without applying specific assumptions or trade-offs. Different models will excel in classification and data analysis in different ways. We will compare the detector method using different attributes in order to choose the finest detector method grounded on accuracy.

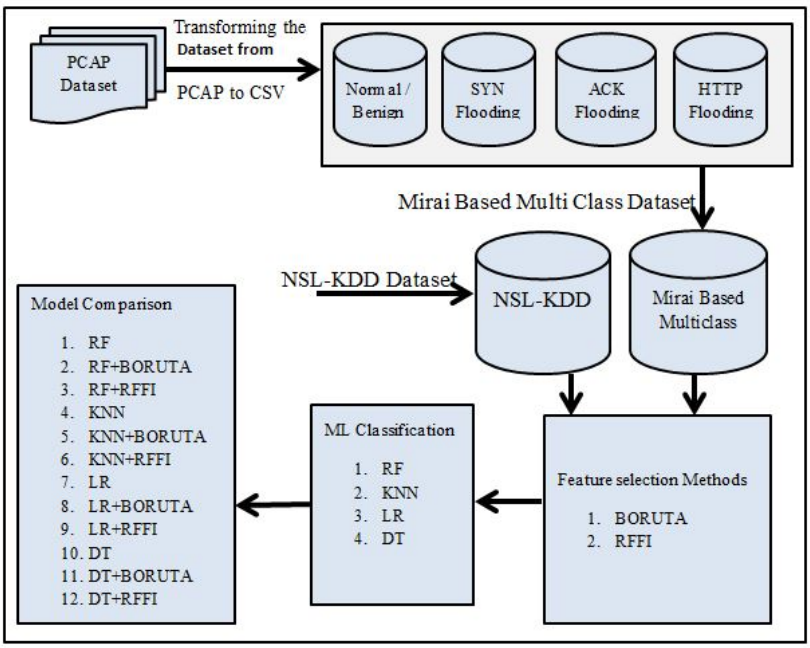



**Figure 1.** Research workflow

### 3.1.1 *Transforming the dataset from PCAP to CSV*

Initially, the IoT Network Intrusion Dataset was captured and saved in PCAP format using Wireshark network traffic analyzer. Although each PCAP file contains numerous packet records, machine learning algorithms cannot directly process this raw data. Therefore, it is necessary to extract and convert the packet information into a machine-readable format. Initially, Wireshark filtering rules provided by the dataset creators were applied to accurately classify packets, as the dataset contains both attack and normal traffic [14]. After preprocessing, two types of packets were identified: attack packets (SYN, ACK, and HTTP flooding attacks) and normal traffic. Subsequently, a custom PCAP extraction and processing tool was developed using Python and the Scapy module to automatically convert the categorized PCAP files into CSV format. Python was chosen for its flexibility, object-oriented features, and efficient text-parsing capabilities. The tool processes attack and normal files from different directories, extracts relevant features, and generates structured CSV datasets suitable for machine learning analysis. The resulting labeled dataset is publicly available on the IEEE Dataport repository [15]. For classification, only the most informative features were selected from the CSV data. Linear Discriminant Analysis (LDA) [40] was employed to evaluate and validate feature significance. Comparative studies using LDA demonstrated that these extracted features significantly enhance classification performance compared to other datasets, such as the full version of our dataset and IoTID20 [41]. Accuracy metrics for both training and testing datasets are presented in Figure 2.

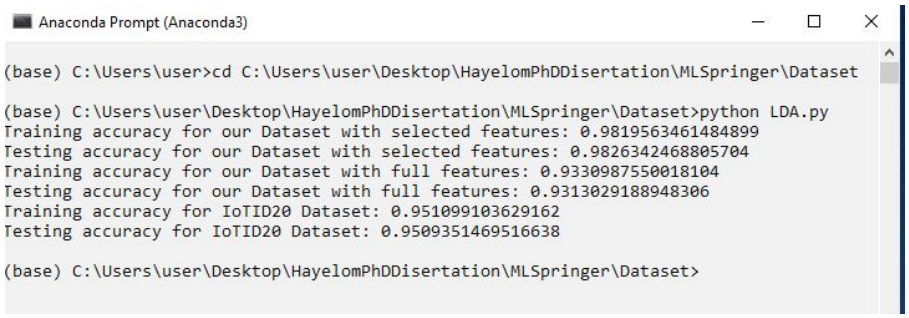



**Figure 2.** Accuracy evaluation of features employing LDA

### 3.1.2 *Mirai-based multi-class dataset*

The dataset used in the study is composed of various IoT data on network traffic archives; every one of which contains benign, i.e. typical network activity data, and fraudulent traffic data associated with the Mirai botnet, which is the most common IoT botnet attack. We focus on SYN-Flooding, ACK-Flooding, HTTP-Flooding, and normal/benign traffic data from these various forms of attacks by IoT botnets for the study. The dataset utilized in this study was retrieved and adapted from [14] and generated in our earlier work [13] aimed at binary classification, and it allows for multi-class classification in addition to identifying an example as benign, SYN-Flooding, ACK-Flooding, or HTTP-Flooding assault. During the inquiry, network traffic data was used to detect three types of Mirai-based botnet assaults. The newly created dataset, which had 16 features, was obtained from the initially created dataset, that was saved in PCAP type. The full set of characteristics of the newly produced dataset utilized in the present study for multi-class attack categorization will be presented in the experiment's results assessment and discussion portion of the study.

### 3.1.3 *NSL-KDD dataset*

To evaluate the usefulness of the newly constructed dataset, we also utilized the NSL-KDD dataset. Developed and curated by the Canadian Institute for Cybersecurity over a nine-week period [42], NSL-KDD represents an improved and more concise version of the original KDD Cup 99 dataset. The NSL-KDD dataset's data format and attack categorization remains same from KDD Cup 99, with the exception of the removal of replicated entries. As a result, the detectors won't have any prejudices in favor of more prevalent recordings. NSL-KDD data is divided into two files: the KDD Train+

dataset for training and the KDD Test+ dataset for evaluation. The NSL-KDD datasets include 37 categories of attack [43] with 41 features discussed later.

### 3.1.4 *FS algorithms*

In light of the large scale of many intrusion detection tasks, feature-based techniques are in plentiful supply. In the present study, we apply two choice of characteristics methods: Boruta and RFFI, which are discussed in detail in the following section. Boruta and random forest feature significance were chosen because they provide model-based significance of features metrics capable of capturing interaction between features and nonlinear associations, both of which are critical in complex network invasion data. Through its intrinsic feature importance ranking, RFFI provides interpretability [44]. Boruta expands this capability by carrying out an all-relevant feature selection procedure to guarantee that no potentially helpful features are prematurely omitted [45]. On the other hand, unsupervised techniques like Principal Component Analysis (PCA) can decrease interpretability by converting original features into latent components [46], and mutual information techniques, while useful, might ignore multivariate dependencies between features [47].

**Boruta algorithm**: The Boruta method is a feature selection and ranking tool based on the RF methodology. We utilized Python's Boruta library to validate the feature selection [45]. This set of components is based on the wrapper, which employs the RF classification technique to find important characteristics. It uses a consequence value that captures all of the important and significant properties in every compilation of datasets.

Random forest feature importance method was used to identify significant characteristics for network intrusion identification. In numerous domains, RFFI is frequently explored as a robust learner [44, 48]. The goal of selecting characteristics is to determine the most important features of an issue's domain. It aids in increasing computing speed and forecast accuracy [49]. The RF technique can accommodate a large number of duplicate characteristics while avoiding model over-fitting. According to the research conducted in [50], the optimal features for improving the efficiency of models are critical.

### 3.1.5 *Machine learning classification*

The technique of anticipating a category or class determined by observed values or points of data is known as categorization. In our circumstance, the multi-class outcome might be normal or ACK-Flooding, SYN-Flooding, or HTTP-Flooding. Classification of $(Y)$ refers to the process of estimating a graphing function $(f)$ between input data $(X)$ to output variables. Because target variables are provided along with the input portion of the dataset, it is classified as supervised machine learning. As a result, on the created dataset, we used four supervised methods for categorization.

Random forest comprised of many decision trees. It optimizes the ability to discriminate of an individual tree classifiers through the integration of the bootstrap aggregating approach and randomness in choosing a set of input components throughout the creation of a decision tree [51]. A decision tree with $M$ branches partitions the feature region into $M$ regions $Rm$, where $1 <= m <= M$. Every tree's forecasting parameter $f(x)$ is specified as

$$f(x) = \sum_{m=1}^{M} C_m \prod(x, R_m), \tag{1}$$

where

$M$ = is the number of regions in the feature space;

$R_m$ = is a region appropriate to m;

$C_m$ = is a constant suitable to m;

$$\prod(x, R_m) = \begin{cases} 1, & \text{if } x \in R_m \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The ultimate classification choice is made by obtaining the majority vote from all trees in the ensemble.

K-Nearest Neighbor [52, 53] performs under the assumption that samples of an indistinguishable category are typically followed by instances that belong to the same category. Consequently, it is assumed a subgroup of training cases in the attribute space as well as the scalar k. A particular unlabeled instance is identified by assigning the label that appears the most frequently among the k training examples closest to it. The Euclidean distance is the furthermost frequently used metric for the distance between instances, according to several different measures [54]. Previous investigations on KNN can be found in [55–57]. This approach employs the Euclidean distance metric, which is represented in the equation below:

$$L(x_i, x_j) = \left( \sum_{i,j=1}^{n} \left( \left( \left| x_i - x_j \right| \right) \right)^2 \right)^{\frac{1}{2}} X \in R^n. \tag{3}$$

Gaussian Naive Bayes the most basic classifier used is Gaussian Naïve Bayes (GNB), which assumes that the features for each class follow a Gaussian distribution. Naïve Bayes is a probabilistic classification approach based on Bayes' theorem, together with the strong assumption that all features are conditionally independent given the class label. Furthermore, the availability of one feature is thought to be independent to the availability of another inside the same group, disregarding the fact that all features participate in the procedure for categorizing.

Decision Tree a decision tree is a classification model that organizes instances based on their feature values. Each internal node represents a feature, and each branch corresponds to a possible value that the feature can take. Starting from the root node, instances are progressively routed through the tree according to their feature values until they reach a leaf node, which provides the final predicted class [58]. Decision trees are widely used in machine learning and data mining to map observations about an instance to conclusions about its target label. Classification trees [59] specifically refer to tree-based models designed for categorical outcomes. A tree consists of two main components: decision nodes, where data are split, and leaf nodes, which provide the classification result. In this context, the cost function evaluates the quality of splits for the categorical target classes, Normal, ACK-Flooding, SYN-Flooding, and HTTP-Flooding, ensuring that the tree identifies the most informative partitions in the dataset.

## 4. Experimental result analysis and discussion

The experiments were performed using a Python implementation running on an HP laptop with an Intel Core i7-7500U (2.9 GHz) processor and 8 GB of RAM. To assess model performance, we employed common classification metrics such as accuracy, precision, recall, the confusion matrix, and the F1-score, following the criteria described in [60]. The corresponding equations for these metrics are provided in the following section.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{6}$$

$$F1 - \text{Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{7}$$

where $TP$ is True Positives, $TN$ is True Negatives, $FP$ is False Positives and $FN$ is False Negatives.

## 4.1 *Dataset descriptions*

This study utilizes two datasets. The first is a Mirai-based multi-class dataset containing 287,230 instances and 16 features, collected and generated in earlier work [13]. The second is the widely used and publicly available NSL-KDD dataset, hosted by the Canadian Institute for Cybersecurity [12]. Developed in 2009, NSL-KDD comprises a total of 147,976 training and testing samples. Both datasets represent multi-class attack scenarios but differ in the number of instances and feature dimensions. Table 1 contains an overview regarding both datasets.

**Table 1.** Dataset file description

| No. | Dataset name | Instances | Features | Year |
|---|---|---|---|---|
| 1 | Mirai Based Dataset | 287,230 | 16 | 2023 |
| 2 | NSl-KDD Dataset | 147,976 | 41 | 2009 |

The next Table 2 displays the label distribution of the newly extracted dataset, which is the multi-class dataset based on Mirai. The NSL-KDD dataset has 37 attack class labels overall, but we eliminate attack labels with fewer than 100. The purpose of this exercise is to improve the classifiers' performance. The class label distribution of the NSL-KDD dataset utilized in the investigation is described in Table 3.

**Table 2.** Label distribution of Mirai-based multi-class dataset

| Label encoded | Label name | No of instances |
|---|---|---|
| 0 | ACK Flooding | 75,632 |
| 1 | HTTP Flooding | 10,464 |
| 2 | Normal | 136,488 |
| 3 | SYN Flooding | 64,646 |

**Table 3.** Label distribution of NSL-KDD dataset

| Encoded | Label name | No of instances | Encoded | Label name | No of instances |
|---|---|---|---|---|---|
| 0 | apache2 | 228 | 8 | pod | 236 |
| 1 | back | 1,183 | 9 | portsweep | 3,302 |
| 2 | guess_passwd | 464 | 10 | processtable | 211 |
| 3 | ipsweep | 4,078 | 11 | satan | 4,331 |
| 4 | mscan | 310 | 12 | smurf | 3,186 |
| 5 | neptune | 47,868 | 13 | teardrop | 996 |
| 6 | nmap | 1,699 | 14 | warezclient | 997 |
| 7 | normal | 78,588 | 15 | warezmaster | 299 |

Table 4 provides descriptions of the features of the Mirai-based multi-class dataset and Table 5 provides descriptions of the features of NSL-KDD dataset.

**Table 4.** Mirai-based multi-class dataset description

| No | Feature name | Feature description |
|---|---|---|
| 1 | IP src | Source IP address |
| 2 | IP dst | Destination IP address |
| 3 | Iflags | IP flags |
| 4 | Tflags | TCP flags |
| 5 | Sport | Source port number |
| 6 | Dport | Destination port number |
| 7 | Frag | IP fragment |
| 8 | Ttl | IP ttl |
| 9 | Ichksum | IP checksum |
| 10 | Len | IP length |
| 11 | Ack | TCP acknowledgment |
| 12 | Dataofs | TCP Dataofs |
| 13 | Seq | TCP Sequence |
| 14 | Window | TCP window size |
| 15 | Tchksum | TCP checksum |
| 16 | Label | Multi-class label (Normal/SYN/ACK/HTTP) |

**Table 5.** NSL-KDD dataset feature descriptions

| No | Feature name | Feature description |
|---|---|---|
| 1 | duration | Length of the connection in seconds |
| 2 | protocol_type | Type of protocol used |
| 3 | service | Network service on the destination host |
| 4 | flag | Status flag of the connection |
| 5 | src_bytes | Bytes sent from source to destination |
| 6 | dst_bytes | Bytes sent from destination to source |
| 7 | land | connection is from/to the same host/port |
| 8 | wrong_fragment | # of incorrect IP fragments |
| 9 | urgent | # of urgent packets |
| 10 | hot | # of hot indicators |
| 11 | num_failed_logins | # of failed login attempts |
| 12 | logged_in | logged in status |
| 13 | num_compromised | # of compromised conditions |
| 14 | root_shell | root shell obtained or not |
| 15 | su_attempted | su root command attempted |
| 16 | num_root | # of root-level accesses |
| 17 | num_file_creations | # of file creation operations |
| 18 | num_shells | # of shell prompts accessed. |
| 19 | num_access_files | # of access-control file operations |
| 20 | num_outbound_cmds | # of outbound commands in an FTP session |
| 21 | is_host_login | host login status |
| 22 | is_guest_login | guest login status |
| 23 | count | Connections from same source to destination |
| 24 | srv_count | Connections to the same service |
| 25 | serror_rate | % of connections with SYN errors |
| 26 | srv_serror_rate | % of service connections with SYN errors |
| 27 | rerror_rate | % of connections with reject errors |

**Table 5.** (cont.)

| No | Feature name | Feature description |
|---|---|---|
| 28 | srv_rerror_rate | % of service connections with reject errors |
| 29 | same_srv_rate | % of connections to the same service |
| 30 | diff_srv_rate | % of connections to different services |
| 31 | srv_diff_host_rate | % of connections to d/t destination for same service |
| 32 | dst_host_count | # of connections to the same destination host |
| 33 | dst_host_srv_count | # of connections to same service on destination host |
| 34 | dst_host_same_srv_rate | % of connections to same service on destination host |
| 35 | dst_host_diff_srv_rate | % of connections to different services on destination |
| 36 | dst_host_same_src_port_rate | % of connections from the same source port |
| 37 | dst_host_srv_diff_host_rate | % of service connections to different destination hosts |
| 38 | dst_host_serror_rate | % of destination-host connections with SYN errors |
| 39 | dst_host_srv_serror_rate | % of same-service destination-host with SYN errors |
| 40 | dst_host_rerror_rate | % of destination-host connections with reject errors |
| 41 | dst_host_srv_rerror_rate | % of same-service destination-host with reject errors |
| 42 | labels | label category |

The performance of the models trained on the Mirai-based dataset and the NSL-KDD dataset depends heavily on the selection of hyperparameters. Accordingly, the tuning process was carried out with care to ensure effective model behavior and improved optimization. The GNB classifier does not require hyperparameter adjustments, while the parameters used for the remaining three ML models are summarized in Table 6.

**Table 6.** Hyper-parameter setting for ML models

| ML model | Hyper-parameter name | value |
|---|---|---|
| KNN | n_neighbors | 8 |
| | metric | euclidean |
| | p | 2 |
| DT | random_state | 123 |
| | max_depth | 30 |
| | criterion | gini |
| RF | random_state | 123 |
| | max_depth | 30 |
| | n_estimators | 500 |

## 4.2 *Mirai based multi class dataset*

This study performs two major operations on two independent datasets, as previously described. First, we use the suggested methods for selecting features (RF, and Boruta,) to identify key traits for detecting Mirai botnet attacks in IoT networks. Second, we confirm and verify what we found by comparing the effectiveness of four predictor types (RF, KNN, GNB, and DT) prior to and following the methods for selecting features.

In the first experiment, RF and Boruta used for selecting features on a mirai-based multi-class datase. Figure 3 displays the Boruta analysis, which includes comparisons of shadow features and statistically significant choices, while Figure 4 displays the features ranked according to Random Forest relevance scores. To begin, in RF, the solution procedure at every parent node depends on the efficacy of the splitting requirement, and this is dependent on the purpose of impurity. The RF feature choice method identifies the 11 most significant characteristics, as seen in the picture beneath.

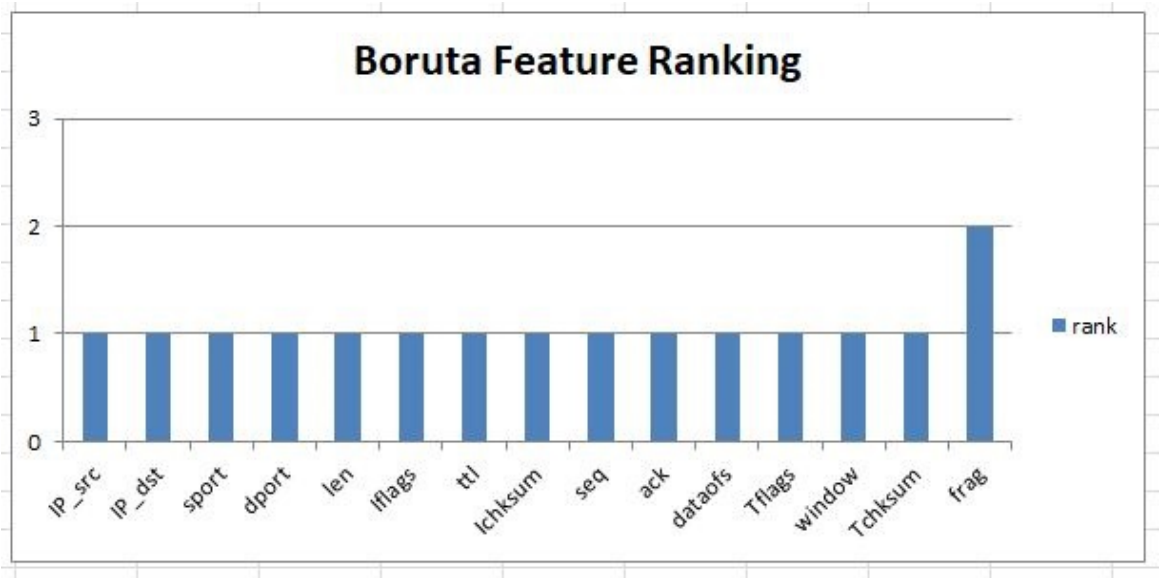



**Figure 3.** Mirai-based multi-class dataset feature selection using Boruta

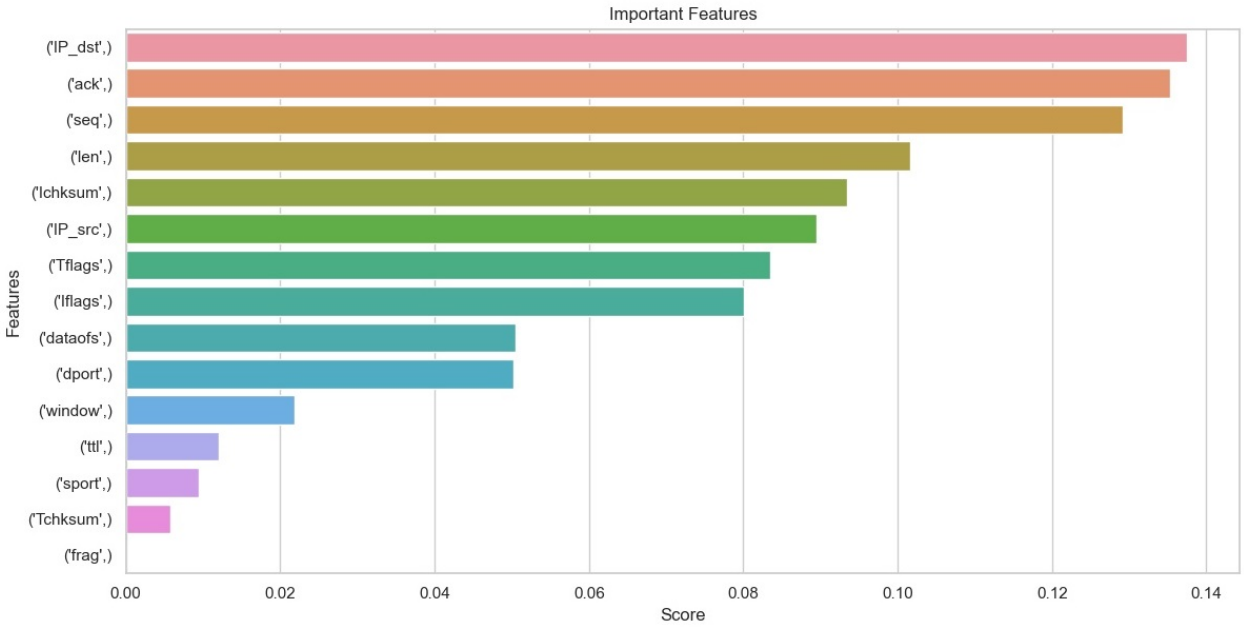
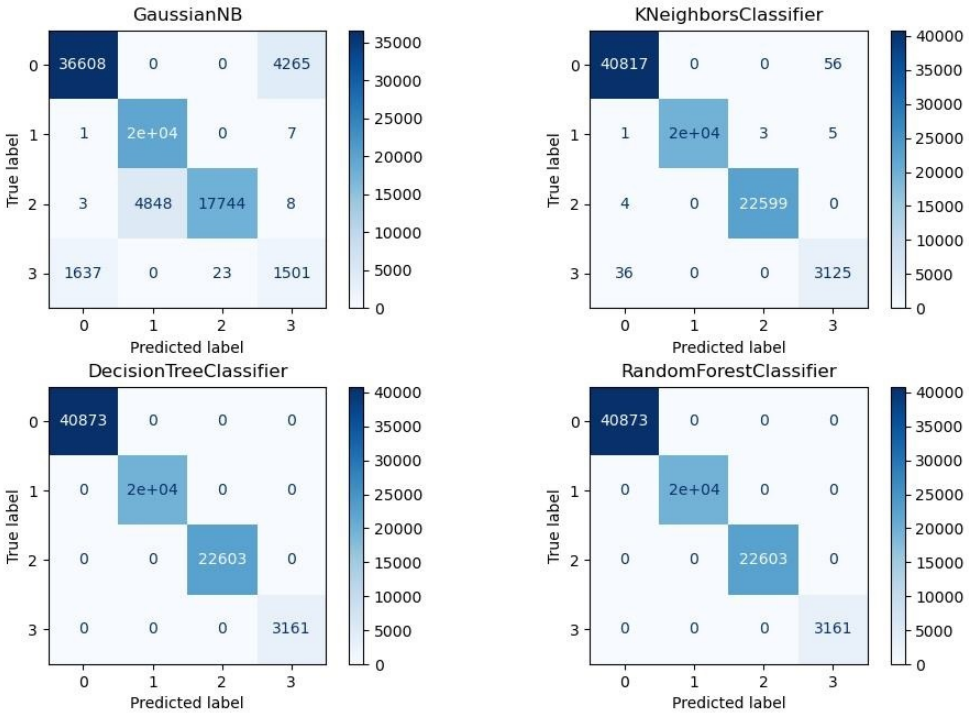


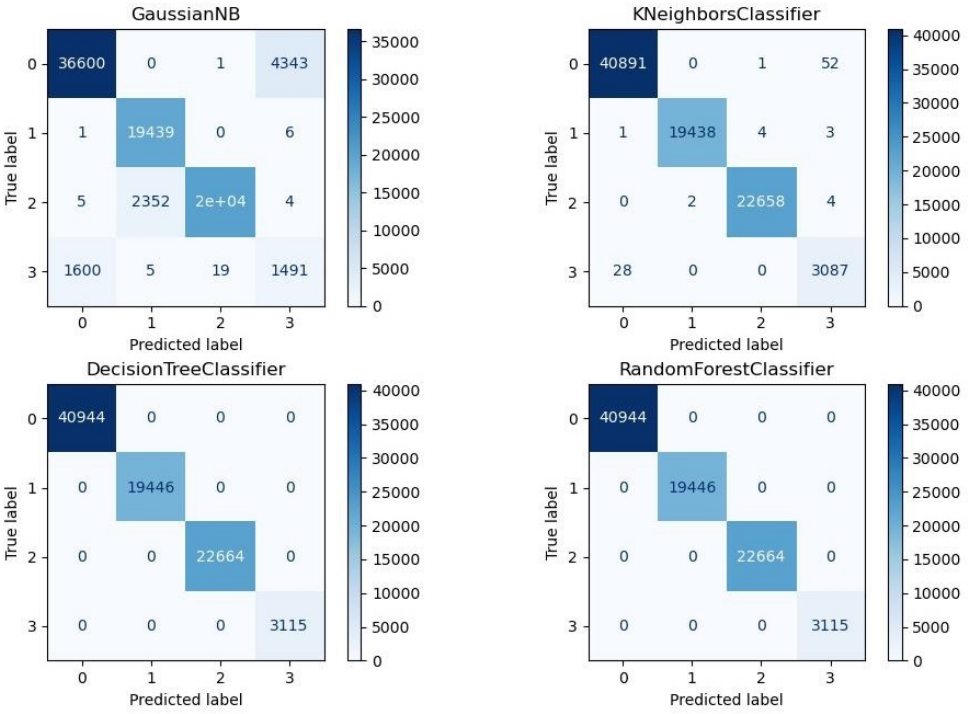

**Figure 4.** Feature selection using RFFI

Succeeding, we used the Boruta selection of features approach using a mirai-based multi-class dataset to obtain the characteristic score shown below. Using machine learning categorization, we employed the top 14 characteristics that were categorized first or regarded as among the most significant characteristics for multi-class attack categorization. Figure 5 presented the confusion matrix result of different classifiers with/without feature selection methods. Tables 7 and 8 describe the accuracy, precision, f1 score and recall outcomes of the experiment utilizing the mirai-based multi-class dataset containing and without selecting features using several machine learning approaches.

(a)


GaussianNB
36608 0 0 4265
1 2e+04 0 7
3 4848 17744 8
1637 0 23 1501
KNeighborsClassifier
40817 0 0 56
1 2e+04 3 5
4 0 22599 0
36 0 0 3125
DecisionTreeClassifier
40873 0 0 0
0 2e+04 0 0
0 0 22603 0
0 0 0 3161
RandomForestClassifier
40873 0 0 0
0 2e+04 0 0
0 0 22603 0
0 0 0 3161
True label
Predicted label

(b)


GaussianNB
36600 0 1 4343
1 19439 0 6
5 2352 2e+04 4
1600 5 19 1491
KNeighborsClassifier
40891 0 1 52
1 19438 4 3
0 2 22658 4
28 0 0 3087
DecisionTreeClassifier
40944 0 0 0
0 19446 0 0
0 0 22664 0
0 0 0 3115
RandomForestClassifier
40944 0 0 0
0 19446 0 0
0 0 22664 0
0 0 0 3115
True label
Predicted label

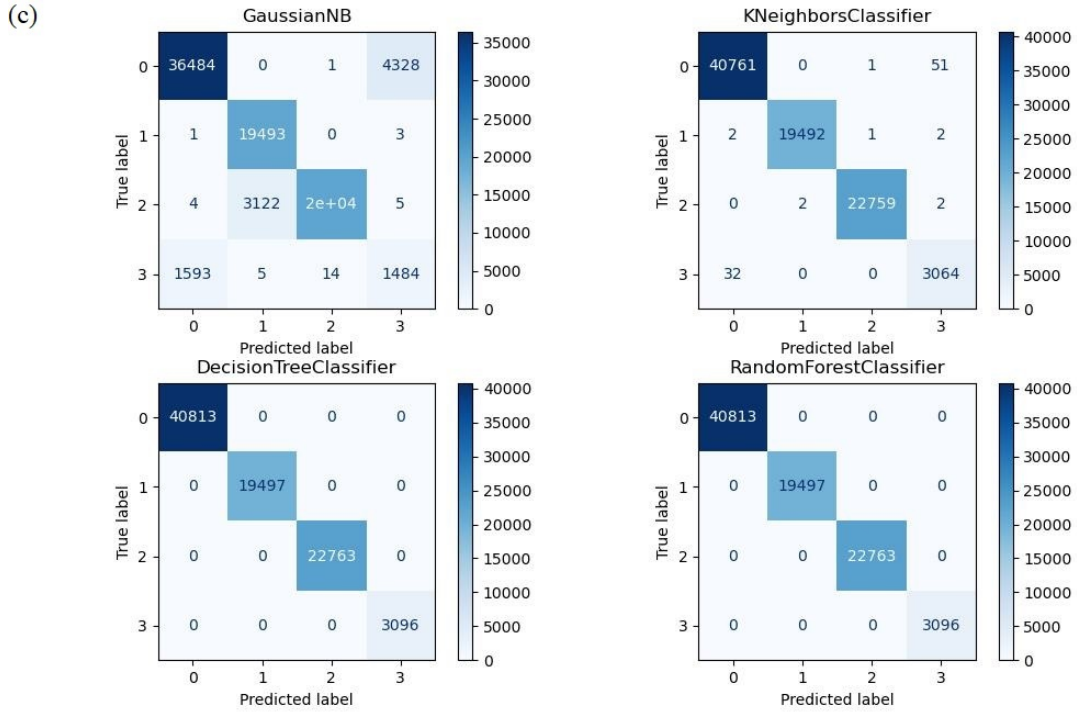



**Figure 5.** Confusion matrix: (a) All features; (b) Using Boruta and (c) Using RFFI

The prediction performance of four classifiers, GNB, KNN, DT, and RF, was evaluated using 5-fold cross-validation in order to guarantee reliable model evaluation and reduce the possibility of overfitting. The findings, which are displayed in Figure 6, demonstrate that GNB had a mean cross-validation accuracy of 0.8741 (± 0.0042), indicating a moderate capacity for prediction but a limited ability to capture the nonlinear feature dependencies present in network intrusion data. KNN, on the other hand, achieved an exceptionally high accuracy of 0.9975 (± 0.0002), demonstrating its capacity to take advantage of local data structures and similarity patterns. Furthermore, flawless cross-validation accuracies of 1.0000 (± 0.0000) were attained by both DT and RF, indicating that these tree-based techniques successfully captured intricate decision boundaries. All models exhibit similar performance, as seen by the small variance across folds. Overall, the cross-validation results show that when modeling the intricate structure of intrusion detection data, tree-based and instance-based learning algorithms (RF, DT, KNN) perform better than probabilistic models (GNB).

**Table 7.** Accuracy of different classifiers with Mirai multi-class dataset

| Method | Accuracy | No. of features |
|---|---|---|
| RF | 100 | 16 |
| RF+RFFI | 100 | 11 |
| RF+Boruta | 100 | 14 |
| KNN | 99.52 | 16 |
| KNN+RFFI | 99.56 | 11 |
| KNN+Boruta | 99.51 | 14 |
| GNB | 66.34 | 16 |
| GNB+RFFI | 91.24 | 11 |
| GNB+Boruta | 87.22 | 14 |
| DT | 100 | 16 |
| DT+RFFI | 100 | 11 |
| DT+Boruta | 100 | 14 |

**Table 8.** Precision and recall of classifiers with Mirai multi-class dataset

| Method | Precision | Recall | F1 score | No.of features |
|---|---|---|---|---|
| RF+RFFI | 100 | 100 | 100 | 11 |
| KNN+RFFI | 100 | 100 | 199 | 11 |
| GNB+RFFI | 91 | 87 | 88 | 11 |
| DT+RFFI | 100 | 100 | 100 | 11 |
| RF+Boruta | 100 | 100 | 100 | 14 |
| KNN+Boruta | 100 | 100 | 100 | 14 |
| GNB+Boruta | 94 | 91 | 92 | 14 |
| DT+Boruta | 100 | 100 | 100 | 14 |

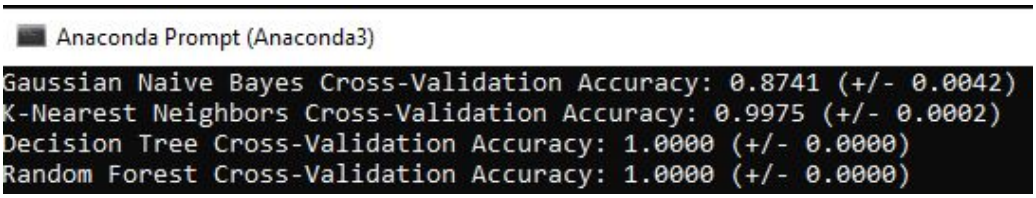
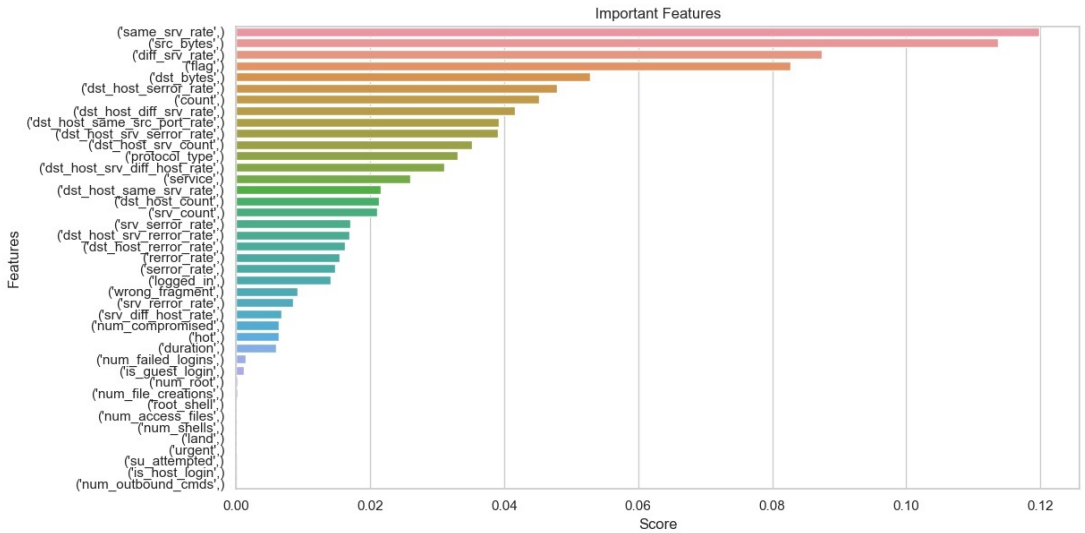



**Figure 6.** 5-fold mean accuracy for Mirai-based multi-class dataset

## 4.3 *NSL-KDD dataset*

In the subsequent experiment, the NSL-KDD dataset was put through to attribute selection using RF and Boruta. Figure 7 shows a correlation heatmap between the chosen features using Boruta and the features that were kept versus those that were rejected. The interpretability and robustness of the feature selection process are supported by these visualizations, which offer a clear grasp of how each feature contributes to the prediction task. To begin, in RF, the solution procedure at every parent node relies on the accuracy of the splitting criterion, and this is depending on the purpose of impurity. The RF choice of characteristics method chooses the 17 most important characteristics, as illustrated in the picture below.

(a)



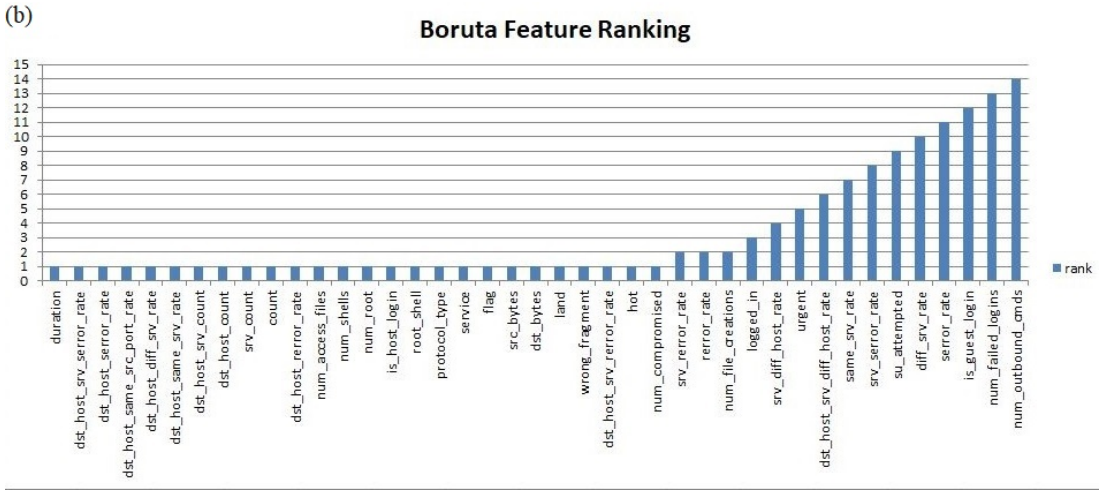


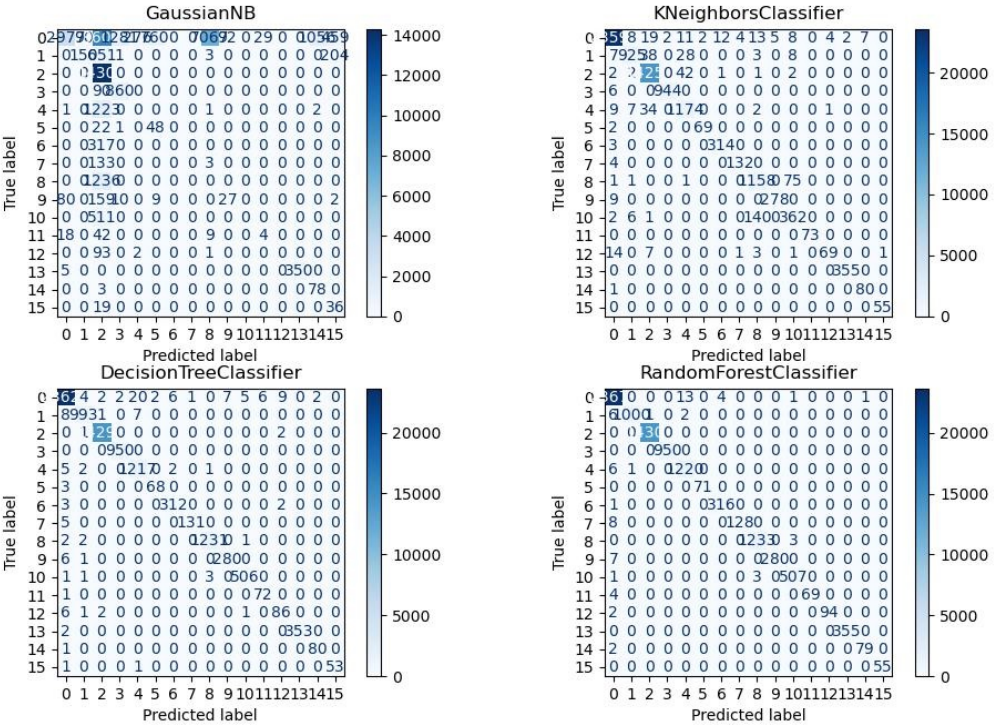

**Figure 7.** Selected features using NSL-KDD: (a) Using RFFI; (b) Using Boruta

Following, we used the Boruta selection of features methodology using the NSL-KDD dataset to obtain the characteristic score shown below. Using machine learning detectors, we employed the best 26 characteristics that were either ranked first or regarded as among the most pertinent characteristics for multi-class attack categorization. Figure 8 presented the confusion matrix result of different classifiers with/without feature selection methods. Tables 9 and 10 describe the accuracy, precision, and recall outcomes of the study using the NSL-KDD dataset both with and without selecting features using several machine learning approaches.

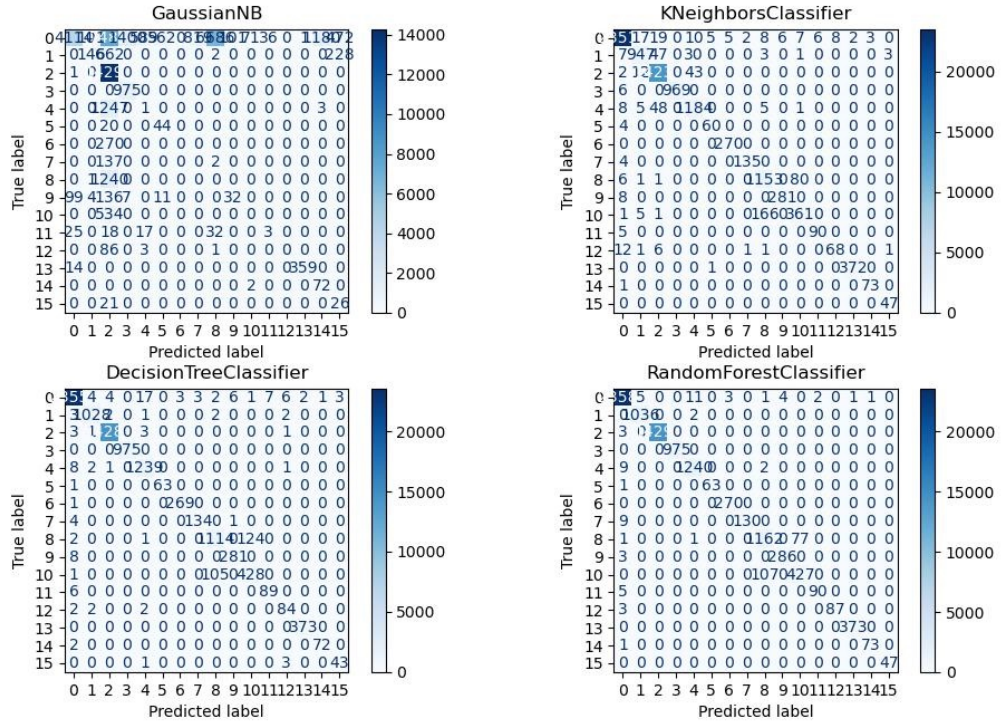
**Table 9.** Accuracy of different classifiers with NSL-KDD dataset

| Method | Accuracy | No. of features |
|---|---|---|
| RF | 99.67 | 41 |
| RF+RFFI | 99.69 | 17 |
| RF+Boruta | 99.31 | 26 |
| KNN | 97.91 | 41 |
| KNN+RFFI | 97.91 | 17 |
| KNN+Boruta | 97.73 | 26 |
| GNB | 39 | 41 |
| GNB+RFFI | 41.77 | 17 |
| GNB+Boruta | 37.81 | 26 |
| DT | 99.51 | 41 |
| DT+RFFI | 99.54 | 17 |
| DT+Boruta | 99.10 | 26 |

(a)


GaussianNB
KNeighborsClassifier
DecisionTreeClassifier
RandomForestClassifier
True label
Predicted label

(b)


GaussianNB
KNeighborsClassifier
DecisionTreeClassifier
RandomForestClassifier
True label
Predicted label

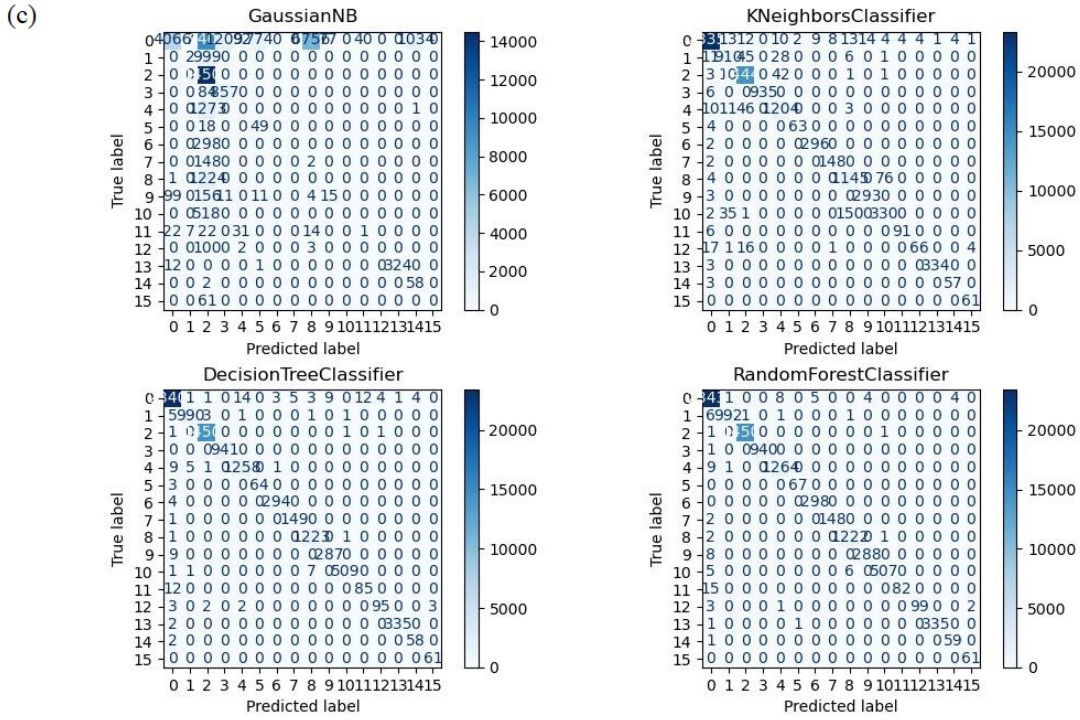



**Figure 8.** Confusion matrix: (a) All features; (b) Using Boruta and (c) Using RFFI

**Table 10.** Precision and recall of classifiers with NSL-KDD dataset

| Method | Precision | Recall | F1 score | No.of features |
|---|---|---|---|---|
| RF+RF | 100 | 100 | 100 | 17 |
| KNN+RF | 98 | 98 | 98 | 17 |
| GNB+RF | 71 | 42 | 39 | 17 |
| DT+RF | 99 | 100 | 98 | 17 |
| RF+Boruta | 99 | 99 | 99 | 26 |
| KNN+Boruta | 98 | 98 | 98 | 26 |
| GNB+Boruta | 71 | 38 | 45 | 26 |
| DT+Boruta | 99 | 99 | 99 | 26 |

## 4.4 *Performance evaluation and discussion*

### 4.4.1 *Performance evaluation*

Compared to the number usually chosen using raw random forest feature importance, the Boruta method chose 14 features for the Mirai dataset and 26 features for NSL-KDD. This discrepancy results from Boruta's all relevant feature selection procedure, which involves statistically comparing each original feature with randomized shadow features and keeping any that exhibit noteworthy predictive value. On the other hand, weaker features may be ignored by RF significance alone, even if they make a significant contribution. Further investigation showed that adding the additional characteristics Boruta chose had no detrimental effect on classifier performance, but rather enhanced robustness across cross-validation folds by identifying more subtle patterns in network traffic data. These results show that Boruta offers a more complete feature collection without compromising prediction accuracy.

We built a smart PCAP extractor and processing solution as part of the prior research that can transform the data in PCAP file into an appropriate machine-learning processable format. We created a new dataset utilizing the newly developed technology that will be useful in deploying mirai-based multi-class identification of attacks in IoT networks. In the present study, we employed the dataset to identify multi-class attacks. For multi-class assault categorization, we used the four most prevalent machine learning frameworks (RF, KNN, GNB, and DT). It was additionally contrasted with the efficiency of the datasets when methods of feature selection were used to choose the key characteristics. In Figure 9, we evaluated the results of the contrast between the two experiments done using the mirai-based multi-class dataset and the NSL-KDD dataset utilizing two approaches to selecting features (random feature and boruta feature selection techniques) including four machine learning strategies.

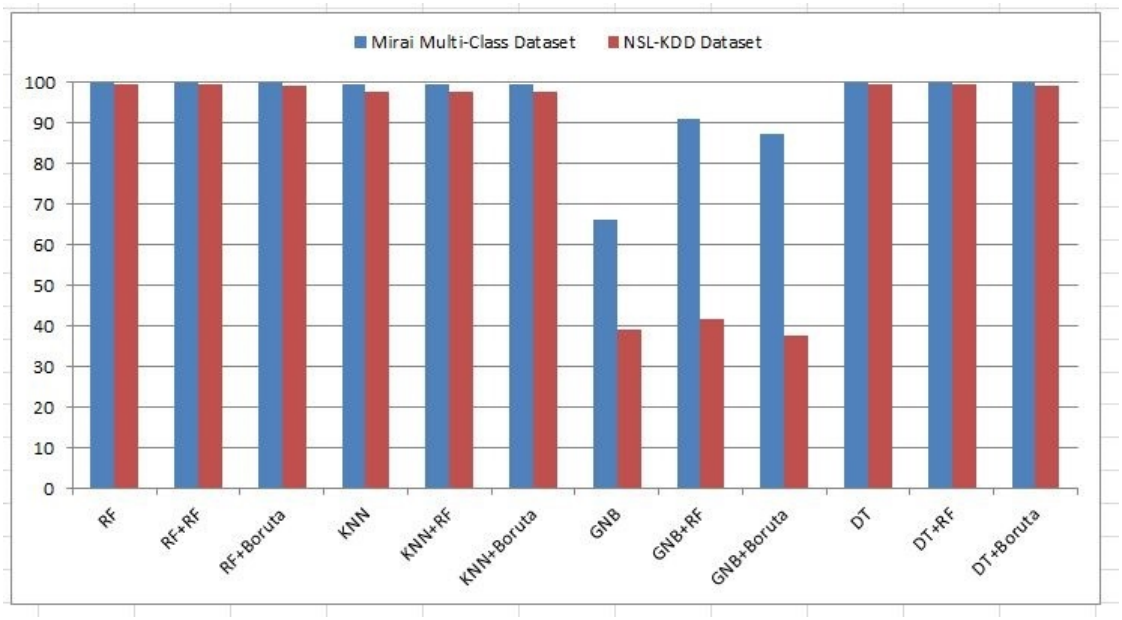



**Figure 9.** Accuracy result comparison between the two datasets

### 4.4.2 *Computing time evaluation*

We implemented four baseline classifiers—GNB, KNN, DT, and RF—to examine computation time and accuracy as part of assessing the proposed approach. Both raw and modified versions of the datasets (with RFFI and Boruta applied) were evaluated. The results indicate that dataset size, in terms of feature count and number of records, has a notable impact on computational cost. Compared with models trained without feature reduction, the use of RFFI and Boruta generally enhanced performance and lowered processing time. Table 11 summarizes the computation time outcomes, highlighting the benefits of applying feature-selection methods. We measured wall-clock training and inference times, along with peak memory usage, for each pipeline paired with four classifiers (RF, DT, KNN, and GNB) on both the Mirai-based multi-class and NSL-KDD datasets in order to compare the computational efficiency of our proposed RFFI against well-known feature selection and dimensionality-reduction methods, Boruta. The mean ± standard deviation was calculated by repeating the experiments five times on a normal workstation with an Intel i7 CPU and 16 GB of RAM. These results demonstrate that RFFI delivers competitive predictive performance while dramatically reducing both computational cost and memory footprint. Consequently, our hypothesis, that lightweight, randomized feature selection performs well on modern, high-dimensional cyber-security datasets, has been confirmed. These results demonstrate that lightweight, randomized feature selection achieves a satisfying compromise between processing efficiency and prediction accuracy for modern high-dimensional cyber-security datasets, supporting the design thinking of our approach.

**Table 11.** Execution time comparison

| | Mirai-based multi-class dataset | | | | | |
|---|---|---|---|---|---|---|
| | Original dataset | | Boruta | | RFFI | |
| Models | Train time | Test time | Train time | Test time | Train time | Test time |
| GNB | 1.175 | 0.376 | 0.138 | 0.251 | 0.075 | 0.156 |
| KNN | 0.049 | 0.002 | 0.34 | 0.001 | 0.02 | 0.001 |
| DT | 0.792 | 0.019 | 0.56 | 0.015 | 0.437 | 0.013 |
| RF | 8.446 | 0.844 | 7.12 | 0.74 | 8.1 | 0.82 |
| | NSL-KDD dataset | | | | | |
| GNB | 0.11 | 0.093 | 0.109 | 0.078 | 0.094 | 0.062 |
| KNN | 1.42 | 0.001 | 1.265 | 0.001 | 0.937 | 0.001 |
| DT | 0.48 | 0.001 | 0.45 | 0.001 | 9.328 | 0.001 |
| RF | 11.11 | 0.69 | 10.72 | 0.64 | 10.37 | 0.51 |

### 4.4.3 *Ablation study*

We assess each classifier's performance both with and without feature selection. To evaluate the contribution of each design choice in our proposed pipeline, we conducted a controlled ablation study utilizing the NSL-KDD and Mirai-based multi-class datasets. We specifically looked at two components: (i) the feature selection strategy, which required comparing RFFI with Boruta; and (ii) the classifier family, which comprised RF, DT, KNN, and GNB. Following five evaluation runs, each configuration's mean ± standard deviation for accuracy, precision, recall, F1-score, wall-clock training, inference time, and peak memory use was reported. When the feature selection method was isolated, Tables 12 and 13 summarized the controlled ablation study using the two datasets.

The findings show that using RFFI consistently increases classification recall, accuracy, and precision for all classifiers. Although RFFI shows a superior mix between performance gains and computational efficiency, Boruta also offers improvement. In every experiment, RF and DT consistently achieve the highest accuracy and stability, outperforming KNN and GNB. Due to its simplifying assumptions about feature independence, GNB exhibits the lowest performance, while KNN performs quite well but is sensitive to attribute dimensionality. The ablation investigation unequivocally demonstrates that the best overall performance is achieved by combining RFFI with RF (RF+RFFI) and DT (DT+RFFI), supporting the design decision to incorporate feature selection with the classifier. The study further emphasizes the significance of finding pertinent features in IoT botnet detection by showing that feature selection correlates more to improvements in performance than classifier choice alone. This research enables the practical implementation of the suggested approach in resource-constrained IoT contexts and offers insights into which components are essential for high performance and robustness. By separating the contributions, we show that RF guarantees consistent and dependable classification across datasets, and RFI greatly improves model accuracy while lowering computational cost. The ablation investigation demonstrates that a robust classifier (RF and DT) in conjunction with efficient feature selection is the primary driver of the suggested approach's higher performance. This strengthens the evidence for the efficacy and generalizability of the method by validating our design and offering deeper insights into the relative importance of each component.

**Table 12.** Mirai multi-class: performance + efficiency (mean ± std over 5 runs)

| FS | Models | Acc | Pre | Rec | F1 | Training time (s) | Testing time (s) | Memory usage (MB) |
|---|---|---|---|---|---|---|---|---|
| Full Features | RF | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 49.8620 ± 1.1047 | 3.0212 ± 0.0951 | 26.74 ± 0.01 |
| | DT | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 0.6623 ± 0.0125 | 0.0094 ± 0.0077 | 20.97 ± 0.00 |
| | KNN | 99.89 ± 0.00 | 99.53 ± 0.02 | 99.70 ± 0.03 | 99.62 ± 0.02 | 1.6332 ± 0.0413 | 7.9513 ± 0.3236 | 80.05 ± 0.00 |
| | GNB | 71.61 ± 9.76 | 53.85 ± 15.35 | 63.91 ± 9.42 | 53.75 ± 12.89 | 0.4186 ± 0.0153 | 0.0687 ± 0.0076 | 23.14 ± 0.00 |
| Boruta | RF | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 52.5314 ± 0.6848 | 3.1555 ± 0.0280 | 25.98 ± 0.00 |
| | DT | 100.00 ± 0.00 | 100.00 ± 0.01 | 100.00 ± 0.01 | 100.00 ± 0.01 | 0.6249 ± 0.0099 | 0.0094 ± 0.0077 | 20.20 ± 0.00 |
| | KNN | 99.89 ± 0.02 | 99.48 ± 0.06 | 99.72 ± 0.05 | 99.60 ± 0.05 | 1.5340 ± 0.0918 | 7.5232 ± 0.3910 | 75.54 ± 0.00 |
| | GNB | 79.49 ± 10.57 | 62.25 ± 16.68 | 71.54 ± 9.96 | 64.71 ± 14.31 | 0.3687 ± 0.0077 | 0.0593 ± 0.0062 | 21.60 ± 0.00 |
| RFFI | RF | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 49.2238 ± 0.5118 | 3.1086 ± 0.0171 | 23.67 ± 0.01 |
| | DT | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 100.00 ± 0.00 | 0.5436 ± 0.0207 | 0.0062 ± 0.0077 | 17.90 ± 0.00 |
| | KNN | 99.90 ± 0.01 | 99.55 ± 0.04 | 99.71 ± 0.04 | 99.63 ± 0.03 | 1.2778 ± 0.0762 | 6.4248 ± 0.1902 | 62.02 ± 0.00 |
| | GNB | 80.22 ± 11.31 | 70.13 ± 11.83 | 72.41 ± 8.20 | 68.07 ± 11.78 | 0.3898 ± 0.0553 | 0.0531 ± 0.0077 | 17.00 ± 0.00 |

**Table 13.** NSL-KDD dataset: performance + efficiency (mean ± std over 5 runs)

| FS | Models | Acc | Pre | Rec | F1 | Training time (s) | Testing time (s) | Memory usage (MB) |
|---|---|---|---|---|---|---|---|---|
| Full Features | RF | 99.80 ± 0.01 | 99.28 ± 0.13 | 97.80 ± 0.27 | 98.50 ± 0.21 | 42.2456 ± 0.8886 | 4.1600 ± 0.1386 | 48.61 ± 0.00 |
| | DT | 99.70 ± 0.01 | 97.87 ± 0.36 | 97.81 ± 0.32 | 97.82 ± 0.27 | 0.8748 ± 0.0872 | 0.0219 ± 0.0077 | 48.61 ± 0.00 |
| | KNN | 98.74 ± 0.05 | 95.17 ± 0.43 | 93.61 ± 0.42 | 94.19 ± 0.46 | 0.2156 ± 0.0269 | 15.7775 ± 2.6577 | 64.81 ± 0.00 |
| | GNB | 44.04 ± 0.62 | 25.60 ± 0.41 | 35.98 ± 0.33 | 20.66 ± 0.35 | 0.3218 ± 0.0212 | 0.3655 ± 0.0377 | 66.91 ± 0.04 |
| Boruta | RF | 99.46 ± 0.03 | 98.10 ± 0.14 | 97.00 ± 0.14 | 97.52 ± 0.10 | 41.8917 ± 0.7574 | 4.0991 ± 0.0723 | 30.82 ± 0.00 |
| | DT | 99.30 ± 0.03 | 96.51 ± 0.17 | 96.03 ± 0.32 | 96.25 ± 0.24 | 0.7530 ± 0.0651 | 0.0156 ± 0.0000 | 30.83 ± 0.00 |
| | KNN | 98.68 ± 0.05 | 95.33 ± 0.46 | 93.70 ± 0.26 | 94.33 ± 0.34 | 0.2031 ± 0.0242 | 13.2375 ± 2.2992 | 41.10 ± 0.00 |
| | GNB | 46.00 ± 3.06 | 25.25 ± 0.20 | 36.01 ± 0.34 | 20.93 ± 0.44 | 0.2687 ± 0.0062 | 0.2281 ± 0.0125 | 42.47 ± 0.01 |
| RFFI | RF | 99.83 ± 0.01 | 99.59 ± 0.10 | 98.51 ± 0.33 | 99.03 ± 0.18 | 40.3113 ± 1.1582 | 4.0959 ± 0.1000 | 18.97 ± 0.00 |
| | DT | 99.66 ± 0.04 | 97.18 ± 0.39 | 97.31 ± 0.44 | 97.23 ± 0.38 | 0.5530 ± 0.0212 | 0.0125 ± 0.0062 | 18.97 ± 0.00 |
| | KNN | 98.58 ± 0.03 | 94.47 ± 0.37 | 92.90 ± 0.41 | 93.43 ± 0.24 | 0.1906 ± 0.0229 | 11.5473 ± 1.9971 | 25.29 ± 0.00 |
| | GNB | 46.74 ± 3.97 | 21.88 ± 0.55 | 30.51 ± 0.50 | 18.48 ± 0.83 | 0.2406 ± 0.0076 | 0.1406 ± 0.0140 | 26.16 ± 0.02 |

RFFI consistently achieved the best predictive accuracy on both datasets while maintaining a significantly lower computational overhead. Despite minor differences in predicting performance, all classifiers maintained short inference times when combined with RFFI. This suggests that while classifier selection can further maximize efficiency or marginally increase resilience, RFFI causes the majority of accuracy gains. The ablation analysis concludes that the recommended RFFI approach is the primary factor determining accuracy, robustness, and computational efficiency. Although preprocessing and classifier selection provide ancillary advantages that enhance stability and fine-tuning performance, the reported enhancements are mostly due to the randomized feature selection itself. These results demonstrate that lightweight, randomized feature selection achieves a satisfying compromise between processing efficiency and prediction accuracy for modern high-dimensional cyber-security datasets, supporting the design thinking of our approach.

#### 4.4.4 *Comparison with related works*

Several machine learning approaches performed badly for classification with multiple classes compared with our work and comparable efforts. It was additionally contrasted with additional comparable works to verify the validity and acceptability of the suggested approach as well as the recently generated dataset in our prior study employing the tool. We take out, handled, and suggested a characteristic picking and machine learning approach that outperforms previous work. In Table 12, we summarised the comparison of our study with related works. In particular, we have included research that makes use of cutting-edge feature selection methods and contemporary classification models. For example, Liu and Du [61] proposed a feature selection strategy based on genetic algorithms that achieved 99.98% detection accuracy on the Bot-IoT dataset. Furthermore, Hossain and Islam [62] demonstrated notable gains in botnet detection performance by introducing a hybrid feature selection and ensemble learning approach. Additionally, Saied et al. [63] carried out a thorough analysis of filtering-based feature selection methods, offering insightful information on how well they work for botnet detection in IoT environments. The performance of the suggested method in contrast to previous studies is summarized in Table 14. It is crucial to remember that stated performance can be greatly impacted by variations in datasets, feature sets, and attack types. For example, some research uses synthetic or small-scale datasets with fewer attack kinds, which could make the categorization work easier. On the other hand, the current work captures a variety of DDoS attack behaviors using a Mirai-based dataset with 16 carefully chosen flow-based variables. The computational cost and generalizability of the model may be impacted by the use of packet-level features or higher-dimensional feature sets in other works. By giving these specifics, the comparison puts performance differences in perspective and emphasizes that the high accuracy of the suggested models is a result of both the efficacy of the chosen feature set and the uniqueness of the Mirai traffic patterns, not just variations in experimental design.

**Table 14.** Performance comparison with related works

| No. | Paper | # Dataset | # FS | # ML | Accuracy |
|---|---|---|---|---|---|
| 1 | [64] | CS-SCADA | Chi | RF | 99.28 |
| 2 | [65] | KDD99 | IG | RF | 99.96 |
| 3 | [66] | CICIDS2017 | RF | RF | 97.46 |
| 4 | [67] | new IDS dataset | Boruta | RF | 99.99 |
| 5 | [68] | CICDDOS2019 | RF | RF | 99.82 |
| 6 | [69] | KDDCUP | Boruta | RF | 99 |
| 7 | [70] | NSL-KDD | Boruta | RF | 74.44 |
| 8 | Our work | NSL-KDD& Mirai | RF | RF | 100 |

The four types of traffic in the Mirai-based dataset are not exactly the same as those in the NSL-KDD dataset. It encompasses both regular traffic and a wider range of assault categories (37 attack types). When possible, we aligned the NSL-KDD attack categories in our trials with the four-class framework for comparative assessment. As intended, the algorithm concentrates on the four main classes. It is possible for an assault type that is entirely novel or unseen to be

mistakenly categorized as one of the recognized classifications. In order to address undiscovered attack types, we intend to investigate anomaly-based or incremental learning methodologies in future work. This limitation is addressed in the study. We recognize that the NSL-KDD dataset predates the Mirai botnet from 2016. It was added to confirm the generalizability of our method and to offer a well-known standard for comparison. This justification and the age disparity in the dataset are now specifically mentioned in the publication.

According to recent research, dual-focused and variation-based attacks can alter estimations in graph neural networks [71, 72], while selective and multi-targeted perturbations can deceive audio and text systems [73–77]. These references place our IoT botnet detection approach into the larger framework of adversarial resilience research by highlighting the generality of adversarial vulnerabilities and supporting the significance of resilient feature selection and classifier design.

## 5. Conclusion and future work

Our earlier work [13] presented a data extraction program that converts IoT network attack datasets (in PCAP format) to flow-based (CSV) parameters. Despite introducing a data extraction program and creating a dataset in our earlier work was created especially for binary classification. We use the same extraction technique in this study, but we create a new, larger multi-class dataset that contains both regular traffic and many Mirai-based attack types (SYN-Flooding, ACK-Flooding, and HTTP-Flooding). This new dataset was developed to facilitate multi-class machine learning assessment and was not present in our previous work. The annotated features in the Mirai-based multi-class IoT botnet dataset offer a structured basis for developing and assessing attacks related to Mirai within IoT environments. Accuracy, recall, and precision were used to evaluate the experimental results. Four classifiers—RF, KNN, GNB, and DT—were tested, and each was combined with two different feature-selection methods, RFFI and Boruta, to identify the most effective model configuration. The experiments highlight the importance of selecting relevant features for classification tasks. As seen in Figure 9, RF regularly demonstrated superior performance throughout the various measurement circumstances. In our experiments with both the Mirai multi-class dataset and the NSL-KDD dataset, RFFI generally selected more impactful features than Boruta. The findings also indicate that using the full feature set is unnecessary and may not be optimal. As a result, it will have an impact on processing speed, accuracy, and more characteristics that are of greater depth of data. According to our assessment results, our recommended approach overtakes other techniques across every dataset.

By methodically combining feature selection and classification techniques, the suggested framework increases IoT botnet detection. Our findings demonstrate that the combination of RFFI and RF consistently produces better outcomes in terms of computing efficiency and accuracy. The technique improves classification reliability while cutting down on needless computational expenses by isolating important features. In contrast to previous research, our method shows: (i) Methodological innovation by combining the assessment of several feature selection techniques and classifiers. (ii) Improvement in practice by specifically taking computing efficiency into account for IoT implementation. (iii) Sturdiness and generalizability among conventional and modern datasets, such as NSL-KDD and Mirai-based datasets. All things considered, the study not only shows increases in empirical accuracy but also offers a methodical, effective, and deployable methodology that raises the bar for IoT botnet identification.

In future work, we plan to explore unsupervised learning methods to enhance anomaly detection for Mirai-related DDoS activity in IoT environments, leveraging the newly released IoT botnet dataset. The second purpose of the forthcoming study is to test gradient boosting and other boosted algorithm families to see whether they can increase the algorithm's predicted accuracy. Certain enhancing techniques, such as XGBoost [78], AdaBoost [79], and Gentle Boost [80, 81] contain mathematical formulas and vary in complexity. Gradient Boosting's notion is in its advancement, resulting in a new dimension that contributes to the requirement fitting.

## Author contributions

H.G.: Conceptualization, Methodology, Software, Validation, Investigation, Data Curation, Writing—Original Draft, Writing—Review & Editing and Visualization; Y.W.: Conceptualization, Review & Editing and Supervision. F.L: Review & Editing and Supervision.

## Confilict of interest

The authors declare no competing financial interest.

## References


[1] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80-84, 2017, https://doi.org/10.1109/MC.2017.201.

[2] L. Sun and Q. Du, "A review of physical layer security techniques for internet of things: challenges and solutions," *Entropy*, vol. 20, no. 10, p. 730, 2018, https://doi.org/10.3390/e20100730.

[3] S. Mukkamala, A. Sung, and A. Abraham, "Cyber security challenges: designing efficient intrusion detection systems and antivirus tools," in *Enhancing Computer Security with Smart Technology*, V. R. Vemuri, Ed., 2005, pp. 125-163.

[4] W. Wang, X. Du, and N. Wang, "Building a cloud IDS using an efficient feature selection method and SVM," *IEEE Access*, vol. 7, pp. 1345-1354, 2018, https://doi.org/10.1109/ACCESS.2018.2883142.

[5] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, no. 3, pp. 1157-1182, 2003.

[6] M. Zekić-Sušac, S. Pfeifer, and N. Šarlija, "A comparison of machine learning methods in a high-dimensional classification problem," *Business Systems Research: International Journal of the Society for Advancing Innovation and Research in Economy*, vol. 5, no. 3, pp. 82-96, 2014, https://doi.org/10.2478/bsrj-2014-0021.

[7] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245-271, 1997, https://doi.org/10.1016/S0004-3702(97)00063-5.

[8] J. Kim, J. Yoon, E. Park, and S. Choi, "Patent document clustering with deep embeddings," *Scientometrics*, vol. 123, pp. 563-577, 2020, https://doi.org/10.1007/s11192-020-03396-7.

[9] X.-W. Chen and M. Wasikowski, "Fast: a ROC-based feature selection metric for small samples and imbalanced data classification problems," In Proc. 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, Aug. 24-27, 2008, pp. 124-132, https://doi.org/10.1145/1401890.1401910.

[10] M. R. Segal, "Machine learning benchmarks and random forest regression," *Open Access Publications from the University of California*. [Online]. Available: https://escholarship.org/uc/item/35x3v9t4. [Accessed Aug. 23, 2025].

[11] J. K. Jaiswal and R. Samikannu, "Application of random forest algorithm on feature subset selection and classification and regression," In Proc. 2017 World Congress on Computing and Communication Technologies, Tiruchirappalli, India, Feb. 2-4, 2017, pp. 65-68, https://doi.org/10.1109/WCCCT.2016.25.

[12] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD Cup 99 data set," In Proc. 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, Jul. 8-10, 2009, pp. 1-6, https://doi.org/10.1109/CISDA.2009.5356528.

[13] H. Gebrye, Y. Wang, and F. Li, "Traffic data extraction and labeling for machine learning based attack detection in IoT networks," *International Journal of Machine Learning and Cybernetics*, 2023, pp. 1-16, https://doi.org/10.1007/s13042-022-01765-7.

[14] H. Kang, D. H. Ahn, G. M. Lee, J. D. Yoo, K. H. Park, and H. K. Kim, "IoT network intrusion dataset," *IEEE Dataport*, 2019, https://dx.doi.org/10.21227/q70p-q449.

[15] H. Gebrye, "Mirai-based multi-class dataset," *IEEE Dataport*, 2023, https://dx.doi.org/10.21227/h4ac-wr38.

[16] L. C. Molina, L. Belanche, and À. Nebot, "Feature selection algorithms: a survey and experimental evaluation," In Proc. 2002 IEEE International Conference on Data Mining, Maebashi City, Japan, Dec. 9-12, 2002, pp. 306-313, https://doi.org/10.1109/ICDM.2002.1183917.

[17] L. Gao, J. Song, X. Liu, J. Shao, J. Liu, and J. Shao, "Learning in high-dimensional multimedia data: the state of the art," *Multimedia Systems*, vol. 23, pp. 303-313, 2017, https://doi.org/10.1007/s00530-015-0494-1.

[18] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of PCA and optimized SVM," In Proc. 2014 International Conference on Contemporary Computing and Informatics, Mysore, India, Nov. 27-29, 2014, pp. 879-884, https://doi.org/10.1109/IC3I.2014.7019692.

[19] B. Subba, S. Biswas, and S. Karmakar, "Enhancing performance of anomaly based intrusion detection systems through dimensionality reduction using principal component analysis," In Proc. 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems, Bangalore, India, Nov. 6-9, 2016, pp. 1-6, https://doi.org/10.1109/ANTS.2016.7947776.

[20] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, "A novel hierarchical intrusion detection system based on decision tree and rules-based models," In Proc. 2019 15th International Conference on Distributed Computing in Sensor Systems, Santorini, Greece, May 29-31, 2019, pp. 228-233, https://doi.org/10.1109/DCOSS.2019.00059.

[21] C. Ambikavathi and S. K. Srivatsa, "Predictor selection and attack classification using random forest for intrusion detection," *Journal of Scientific & Industrial Research*, vol. 79, pp. 365-368, 2020.

[22] S. Meftah, T. Rachidi, and N. Assem, "Network based intrusion detection using the UNSW-NB15 dataset," *International Journal of Computing and Digital Systems*, vol. 8, no. 5, pp. 478-487, 2019.

[23] S. T. Ikram and A. K. Cherukuri, "Improving accuracy of intrusion detection model using PCA and optimized SVM," *Journal of Computing and Information Technology*, vol. 24, no. 2, pp. 133-148, 2016, https://doi.org/10.20532/cit.2016.1002701.

[24] R. Patgiri, U. Varshney, T. Akutota, and R. Kunde, "An investigation on intrusion detection system using machine learning," In Proc. 2018 IEEE Symposium Series on Computational Intelligence, Bangalore, India, Nov. 18-21, 2018, pp. 1684-1691, https://doi.org/10.1109/SSCI.2018.8628676.

[25] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789-33795, 2018, https://doi.org/10.1109/ACCESS.2018.2841987.

[26] F. Yihunie, E. Abdelfattah, and A. Regmi, "Applying machine learning to anomaly-based intrusion detection systems," In Proc. 2019 IEEE Long Island Systems, Applications and Technology Conference, Farmingdale, NY, USA, May 3, 2019, pp. 1-5, https://doi.org/10.1109/LISAT.2019.8817340.

[27] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986-2998, 2016, https://doi.org/10.1109/TC.2016.2519914.

[28] E. Kabir, J. Hu, H. Wang, and G. Zhuo, "A novel statistical technique for intrusion detection systems," *Future Generation Computer Systems*, vol. 79, pp. 303-318, 2018, https://doi.org/10.1016/j.future.2017.01.029.

[29] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, A. Alazab, "Hybrid intrusion detection system based on the stacking ensemble of C5 decision tree classifier and one class support vector machine," *Electronics*, vol. 9, no. 1, p. 173, 2020, https://doi.org/10.3390/electronics9010173.

[30] S. Rajagopal, P. P. Kundapur, and K. S. Hareesha, "A stacking ensemble for network intrusion detection using heterogeneous datasets," *Security and Communication Networks*, vol. 2020, pp. 1-9, 2020, https://doi.org/10.1155/2020/4586875.

[31] T. Kumar, V. K. Verma, and S. Tyagi, "On (r, s)-norm entropy of intuitionistic fuzzy sets," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*, S. S. Dash, C. Lakshmi, S. Das, and B. K. Panigrahi, Eds, Singapore: Springer, 2020, pp. 785-796.

[32] O. Osanaiye, H. Cai, K.-K. R. Choo, A. Dehghantanha, Z. Xu, and M Dlodlo, "Ensemble-based multi-filter feature selection method for DDoS detection in cloud computing," *EURASIP Journal on Wireless Communications and Networking*, vol. 2016, no. 1, pp. 1-10, 2016, https://doi.org/10.1186/s13638-016-0623-3.

[33] H. Kwon, Y. Kim, H. Yoon, and D. Choi, "Optimal cluster expansion-based intrusion tolerant system to prevent denial of service attacks," *Applied Sciences*, vol. 7, no. 11, p. 1186, 2017, https://doi.org/10.3390/app7111186.

[34] H. Kwon, Y. Kim, H. Yoon, and D. Choi, "Classification score approach for detecting adversarial example in deep neural network," *Multimedia Tools and Applications*, vol. 80, no. 7, pp. 10339-10360, 2021, https://doi.org/10.1007/s11042-020-09167-z.

[35] H. Kwon and J. Lee, "Advguard: fortifying deep neural networks against optimized adversarial example attack," *IEEE Access*, vol. 12, pp. 5345-5356, 2020, https://doi.org/110.1109/ACCESS.2020.3042839.

[36] Q. Xia, S. Dong, and T. Peng, "An abnormal traffic detection method for IoT devices based on federated learning and depthwise separable convolutional neural networks," In Proc. 2022 IEEE International Performance, Computing, and Communications Conference, Austin, TX, USA, Nov. 11-13, 2022, pp. 352-359, https://doi.org/10.1109/IPCCC55026.2022.9894354.

[37] H. Gebrye, Y. Wang, and F. Li, "Computer vision based distributed denial of service attack detection for resource-limited devices," *Computers and Electrical Engineering*, vol. 120, p. 109716, 2024, https://doi.org/10.1016/j.compeleceng.2024.109716.

[38] L. Shu, S. Dong, H. Su, and J. Huang, "Android malware detection methods based on convolutional neural network: a survey," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 5, pp. 1330-1350, 2023, https://doi.org/10.1109/TETCI.2023.3281833.

[39] S. Dong, Y. Xia, and T. Peng, "Network abnormal traffic detection model based on semi-supervised deep reinforcement learning," *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 4197-4212, 2021, https://doi.org/10.1109/TNSM.2021.3120804.

[40] A. Izenman, "Linear Discriminant Analysis," in *Modern Multivariate Statistical Techniques*, New York: Springer, 2013, pp. 237-280.

[41] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Advances in Artificial Intelligence*, Cham: Springer, 2020, pp. 508-520.

[42] S. Sapre, P. Ahmadi and K. Islam, "A robust comparison of the KDDCup99 and NSL-KDD IoT network intrusion detection datasets through various machine learning algorithms," *arXiv:1912.13204*. 2019, https://doi.org/10.48550/arXiv.1912.13204.

[43] M. A. M. Hasan, M. Nasser, S. Ahmad, and K. I. Molla, "Feature selection for intrusion detection using random forest," *Journal of Information Security*, vol. 7, no. 3, pp. 129-140, 2016, https://doi.org/10.4236/jis.2016.73009..

[44] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," *Expert Systems with Applications*, vol. 39, no. 1, pp. 424-430, 2012, https://doi.org/10.1016/j.eswa.2011.07.032.

[45] M. B. Kursa and W. R. Rudnicki, "Feature selection with the boruta package," *Journal of Statistical Software*, vol. 36, pp. 1-13, 2010, https://doi.org/10.18637/jss.v036.i11.

[46] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016, https://doi.org/10.1098/rsta.2015.0202.

[47] J. R. Vergara and P. A. Estévez, "A review of feature selection methods based on mutual information," *Neural Computing and Applications*, vol. 24, no. 1, pp. 175-186, 2014, https://doi.org/10.1007/s00521-013-1368-0.

[48] T. M. Khoshgoftaar, M. Golawala, and J. Van Hulse, "An empirical study of learning from imbalanced data using random forest," In Proc. 19th IEEE International Conference on Tools with Artificial Intelligence, Patras, Greece, Oct. 29-31, 2007, pp. 310-317, https://doi.org/10.1109/ICTAI.2007.46.

[49] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "Hybrid feature selection by combining filters and wrappers," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8144-8150, 2011, https://doi.org/10.1016/j.eswa.2010.12.156.

[50] C. Dewi and R.-C. Chen, "Random forest and support vector machine on features selection for regression analysis," *International Journal of Innovative Computing, Information and Control*, vol. 15, no. 6, pp. 2027-2037, 2019, c10.24507/ijicic.15.06.2027.

[51] V. Vapnik, *The Nature of Statistical Learning Theory*, New York: Springer, 1999.

[52] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175-185, 1992, https://doi.org/10.1080/00031305.1992.10475879.

[53] P. Cunningham and S. J. Delany, "k-nearest neighbour classifiers—a tutorial," *ACM Computing Surveys*, vol. 54, no. 6, pp. 1-25, 2021, https://doi.org/10.1145/3459665.

[54] M. R. Bonyadi, Q. M. Tieng, and D. C. Reutens, "Optimization of distributions differences for classification," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 511-523, 2018, https://doi.org/10.1109/TNNLS.2018.2844723.

[55] M.-L. Zhang and Z.-H. Zhou, "A k-nearest neighbor based algorithm for multi-label classification," In Proc. 2005 IEEE International Conference on Granular Computing, Beijing, China, Jul. 25-27, 2005, pp. 718-721, https://doi.org/10.1109/grc.2005.1547385.

[56] L. E. Peterson, "k-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009, https://doi.org/10.4249/scholarpedia.1883.

[57] D. J. Hand and V. Vinciotti, "Choosing k for two-class nearest neighbour classifiers with unbalanced classes," *Pattern Recognition Letters*, vol. 24, no. 9-10, pp. 1555-1562, 2003, https://doi.org/10.1016/s0167-8655(02)00394-x.

[58] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas, "Supervised machine learning: a review of classification techniques," *Emerging Artificial Intelligence Applications in Computer Engineering*, vol. 160, no. 1, pp. 3-24, 2007.

[59] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., New York: Springer, 2009.

[60] S. Ni, Q. Qian, and R. Zhang, "Malware identification using visualization images and deep learning," *Computers & Security*, vol. 77, pp. 871-885, 2018, https://doi.org/10.1016/j.cose.2018.04.005.

[61] X. Liu and Y. Du, "Towards effective feature selection for IoT botnet attack detection using a genetic algorithm," *Electronics*, vol. 12, no. 5, p. 1260, 2023, https://doi.org/10.3390/electronics12051260.

[62] M. A. Hossain and M. S. Islam, "A novel hybrid feature selection and ensemble-based machine learning approach for botnet detection," *Scientific Reports*, vol. 13, no. 1, p. 21207, 2023, https://doi.org/10.1038/s41598-023-48230-1.

[63] M. Saied, S. Guirguis, and M. Madbouly, "Review of filtering based feature selection for botnet detection in the internet of things," *Artificial Intelligence Review*, vol. 58, no. 4, p. 119, 2025, https://doi.org/10.1007/s10462-025-11113-0.

[64] L. A. C. Ahakonye, C. I. Nwakanma, J.-M. Lee, and D.-S. Kim, "SCADA intrusion detection scheme exploiting the fusion of modified decision tree and chi-square feature selection," *Internet of Things*, vol. 21, p. 100676, 2023, https://doi.org/10.1016/j.iot.2022.100676.

[65] M. A. Hussein, and S. Q. Mohammed, "Performance analysis of different machine learning models for intrusion detection systems," *Journal of Engineering*, vol. 28, no. 5, pp. 61-91, 2022, https://doi.org/10.31026/j.eng.2022.05.05.

[66] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, "A new ensemble-based intrusion detection system for internet of things," *Arabian Journal for Science and Engineering*, pp. 1-15, 2021, https://doi.org/10.1007/s13369-021-06086-5.

[67] N. Farhana, A. Firdaus, M. F. Darmawan, and M. F. Ab Razak, "Evaluation of boruta algorithm in DDoS detection," *Egyptian Informatics Journal*, vol. 24, pp. 27-42, 2022, https://doi.org/10.1016/j.eij.2022.10.005.

[68] Z. A. Ibrahim and I. J. Mohammed, "Analysis of features selection effects on different classification algorithms with performance metrics improvement based on portscan-attack of CICDDoS2019 dataset," *Journal of Algebraic Statistics*, vol. 13, no. 3, pp. 1712-1723, 2022.

[69] S. Subbiah, K. S. M. Anbananthen, S. Thangaraj, S. Kannan, and D. Chelliah, "Intrusion detection technique in wireless sensor network using grid search random forest with boruta feature selection algorithm," *Journal of Communications and Networks*, vol. 24, no. 2, pp. 264-273, 2022, https://doi.org/10.23919/jcn.2022.000002.

[70] M. A. F. A. Fida, T. Ahmad, and M. Ntahobari, "Variance threshold as early screening to boruta feature selection for intrusion detection system," In Proc. 2021 13th International Conference on Information & Communication Technology and System, Surabaya, Indonesia, Oct. 19-20, 2021, pp. 46-50, https://doi.org/10.1109/ICTS52701.2021.9608852.

[71] H. Kwon and D.-J. Kim, "Dual-targeted adversarial example in evasion attack on graph neural networks," *Scientific Reports*, vol. 15, no. 1, p. 3912, 2025, https://doi.org/10.1038/s41598-025-85493-2.

[72] H. Kwon and J.-W. Baek, "Targeted discrepancy attacks: crafting selective adversarial examples in graph neural networks," *IEEE Access*, vol. 13, pp. 13700-13710, 2025, https://doi.org/10.1109/access.2024.3456728.

[73] K. Ko, S. Kim, and H. Kwon, "Multi-targeted audio adversarial example for use against speech recognition systems," *Computers & Security*, vol. 128, p. 103168, 2023, https://doi.org/10.1016/j.cose.2023.103168.

[74] K. Ko, S. Kim, and H. Kwon, "Selective audio perturbations for targeting specific phrases in speech recognition systems," *International Journal of Computational Intelligence Systems*, vol. 18, no. 1, p. 103, 2025, https://doi.org/10.1007/s44196-025-00844-1.

[75] H. Kwon and S.-H. Nam, "Audio adversarial detection through classification score on speech recognition systems," *Computers & Security*, vol. 126, p. 103061, 2023, https://doi.org/10.1016/j.cose.2022.103061.

[76] T. Lee, S. Lee, and H. Kwon, "Multi-targeted textual backdoor attack: model-specific misrecognition via trigger position and word choice," *IEEE Access*, 2025, https://doi.org/10.1109/access.2024.3474088.

[77] H. Kwon and S. Lee, "Detecting textual adversarial examples through text modification on text classification systems," *Applied Intelligence*, vol. 53, no. 16, pp. 19161-19185, 2023, https://doi.org/10.1007/s10489-022-03313-w.

[78] R.-C. Chen, R. E. Caraka, N.E.G. Arnita, S. Pomalingo, A. Rachman, T. Toharudin, et al., "An end to end of scalable tree boosting system," *Sylwan*, vol. 165, no. 1, pp. 1-11, 2020.

[79] R. E. Schapire, "Explaining adaboost," in *Empirical Inference: Festschrift in Honor of Vladimir N. Vapnik*, Berlin: Springer, 2013, pp. 37-52.

[80] M. Cabezas, A. Oliver, S. Valverde, B. Beltran, J. Freixenet, , J. C. Vilanova, et al., "Boost: a supervised approach for multiple sclerosis lesion segmentation," *Journal of Neuroscience Methods*, vol. 237, pp. 108-117, 2014, https: //doi.org/10.1016/j.jneumeth.2014.08.024.

[81] A. Kubankova, D. Kubanek, and J. Prinosil, "Digital modulation classification based on characteristic features and gentleboost algorithm," In Proc. 34th International Conference on Telecommunications and Signal Processing, Budapest, Hungary, Aug. 18-20, 2011, pp. 448-451, https://doi.org/10.1109/TSP.2011.6043692.