International Journal of Information Security (2023) 22:1–27
https://doi.org/10.1007/s10207-022-00616-4

REGULAR CONTRIBUTION

Intrusion detection system over real-time data traffic using machine learning methods with feature selection approaches

Gulab Sah[1] · Subhasish Banerjee[1] · Sweety Singh[1]

Accepted: 14 September 2022 / Published online: 6 October 2022

Abstract
The intrusion detection system (IDS) plays an important role in extracting and analysing the network traffics to detect aberrant activity. However, emerging technologies, like cloud computing, Internet of Things, etc., generate a large volume of traffics, which may carry the irrelevant attributes that do not have any impact on classification or in detection of assaults. Hence, it's became an open challenge for the researchers to extract the meaningful data from huge amounts of traffic and also to examine whether the selected features could increase IDS performance or not. To solve these issues, features selection approaches (FSA) have been used in this research to remove non-relevant features and find the important ones. Later, the various classifiers have been used to investigate the best classifier which could increase the performance of IDS's detection-engine on the NSL-KDD datasets. However, to validate, the investigated best-performing classifier with the suitable features selection technique (FST) has also been implemented on a real-time dataset, i.e. combined CICIDS2017. The experiment results in this research suggest that the acquired subset of relevant features under the proposed model's (Decision Tree + Recursive Feature Elimination) could increase the IDS performance with average accuracy of 99.21% and 99.94% on the well-known NSL-KDD and CICIDS2017 datasets, respectively, and could also minimize the computation cost, in parallel.

Keywords CICIDS2017 · IDS · FSA · NSL-KDD · Classifiers

1 Introduction

In modern society, network-based services are gaining more and more importance. As technology gets advances, such as IoTs, big data, clouds computing, etc., the vast volumes of the traffic are also increasing, rapidly. Therefore, as the network data traffic is growing substantially, updating the attack sign becomes more difficult, time-consuming, and tedious. Hence, as a result of wide internet use and rapid traffic growth, the network security became an emerging field of research among scientists and researchers. In this field, the researchers try to prevent the attackers or intruders who are always looking for finding the flaws in the network or in the system to obtain illegal access (s).

Many solutions exist today to secure a network environment, such as antivirus, firewalls, IDS, etc. However, the IDS is the most prominent mechanism among them to defend a network or individual system.

The IDS also protects the sensitive data during traveling over the networks from being intercepted by attackers or intruders. However, the existing IDS are still not extremely scalable or flexible enough. In the years 2014–2016, two data breaches were reported by Yahoo, impacting 500 million customers' accounts and resulting in a loss of 350 million dollars [1]. The outbreaks are being hammered with the intention of snipping data with the help of intelligent and sophisticated algorithms. Several IDS have been developed since last few years, but determining whether a network is normal or aberrant is not an easy task. Therefore, the several algorithms of machine learning (ML) have been introduced and implemented to boost the intelligence of IDS in order to tackle the challenges [2]. Till date, many types of research have been undertaken which show that ML-based IDS performs better in terms of execution and implementation [3]. However,

✉ Subhasish Banerjee
subhasishism@gmail.com

Gulab Sah
gulabsah15@gmail.com

Sweety Singh
sweetyarti12345@gmail.com

[1] Department of Computer Science and Engineering, National Institute of Technology, Arunachal Pradesh, Jote, India

Table 1 Nomenclature

Abbreviations	Terms	Abbreviations	Terms
ANN	Artificial neural networks	IG	Information gain
ANOVA	Analysis of variance	KNN	K-nearest neighbours
CART	Classification and regression trees	LR	Linear regression
CFS	Correlation-based feature selection	ML	Machine learning
CNN	Convolution neural network	NB	Naive Bayes
DR	Detection rate	NN	Neural networks
DT	Decision tree	PERL	Practical extraction and reporting language
DoS	Denial of services	PHF	Portable hypertext format
DDoS	Distributed denial of service	PCA	Principal component analysis
FN	False negative	PROBE	Probing
FP	False positive	RF	Random forest
FSA	Features selection approaches	R2L	Remote-to-local
FST	Features selection technique	RT	Random tree
FS	Feature selection	RFE	Recursive feature elimination
FPR	False positive rate	SSH	Secure shell
FTP	File transfer protocol	SVM	Support vector machine
GB	Gigabyte	TN	True negative
GHz	Gigahertz	TP	True positive
IMAP	Internet message access protocol	TPR	True positive rate
IDS	Intrusion detection system	U2R	User-to-root
IoT	Internet of Things	XSS	Cross-site scripting

only a few models could combine low computation costs with good detection rates, at the same time.

Therefore, since network data traffic grows at a rapid rate, extracting significant and relevant information from this traffic is a difficult task which must be addressed properly and the computing cost also supposed to be considered, side by side [4].

Moreover, whether the selected features will either improve or not in the performance of IDS is also needed to investigate.

Hence, to minimize the computational cost, one possible solution is, identify and select only relevant features from the dataset that contribute in attacks detection. Thus, a reduction of dataset dimension leads to the requirement of less training time. Simultaneously, it can enhance the performance of the classifier in IDS [5]. Similarly, the other possible solution to minimize the computational cost could be, utilize only the cost-effective algorithms that takes low cost to learn the data [6], for example, K-nearest neighbors (KNN). Therefore, to minimize the computational cost and to increase the performance of IDS, the features selection approaches (FSA) have been used in this research to remove non-relevant features, and various classifiers have also been used to investigate the best-performing classifier for enhancing the performance of IDS. Table 1 provides a list of all the abbreviations used in this paper.

1.1 Contributions

The major contributions in this research are defined as follows:

- The FSA like principal component analysis (PCA) and recursive feature elimination (RFE) have been used to discover and pick up the significant features on the NSL-KDD and CICIDS2017 dataset.
- A smaller and more appropriate subset of features have been identified, i.e. 13 and 8 key features from the NSL-KDD and CICIDS 2017 datasets, respectively.
- A comparative analysis of different FSA with various classifiers such as naive Bayes (NB), decision tree (DT), and KNN on the NSL_KDD dataset has been described.
- Based on the selected decent classifier and features selection technique (FST) on the NSL-KDD dataset, the same has implemented on a real-time dataset, i.e. combined CICIDS2017 and evaluated its performance in terms of F-measure, G-means, recall (sensitivity), precision, specificity, accuracy, testing time, and training time.

1.2 Organization

The related literature review of this research is explained in the next section. In Sect. 3, the proposed framework and its approach are discussed. The experiments and results are demonstrated in Sect. 4. Finally, the conclusion and future work are deliberated in the last section of this article.

2 Literature survey

Many studies have used FSA to reduce the issue of data dimension and improve the IDS detection rate (DR) throughout the previous few decades. However, as networks traffic is growing at a rapid rate, the possible types of threats are also increasing side by side. However, the researchers are still struggling with the issues of dimensionality reduction and computational time [7]. As a result, various ML techniques for IDS with FSA have been proposed, till date.

Mukkamala et al. [8] examined IDS utilizing support vector machine (SVM) and neural networks (NN). The outcome of the experiment found that SVM is highly flexible and suitable for usage with huge datasets where NN needs a lot of learning time. In this context, in 2004, Fleuret et al. utilized the mutual information approach to choose the relevant features; this approach is more effective than SVM when combined with the Bayes network. In general, it emphasizes overall processing time [9]. In 2005, Chebrolu et al. investigated IDS using reverse classification tree and Bayes networks as feature selection methods. Using the proposed FST, they have extracted 12 essential features which are capable of recognizing and detecting various attack types. Unfortunately, the detection rate for User-to-root (U2R) attacks was comparably low [10]. Therefore, in 2008, Chou et al. utilize correlation-based feature selection (CFS) and fast CFS as feature selection (FS) methods to handle high-dimensional data issues such as uncertainty, ambiguity, and redundancy into the collected data items. For obtaining the relevant features, their proposed approach has integrated C4.5 and NB, together. Based on their experiments, they have demonstrated that the detection rate of the proposed fuzzy KNN technique can be improved compared to previous classifiers [11].

In this context, Heba et al. use PCA as a reduction technique in combination with SVM to address the challenges of features dimension reduction and processing costs minimization problem. The experiment demonstrated that the IDS performance can be increased with less computational time [12]. Zainal et al. used a DT classifier with filter-based FSA, such as information gain (IG), Chi-square, and relief-F, to examine the KDDcup99 dataset. Out of a total of 41 attributes, FS methods have been utilized to obtain 5, 10, 15, and 20 relevant features, only. The results showed that "IG" as the FST outperformed other approaches and improved the performance of the model [13]. Revathi et al. has explored the effectiveness of various ML algorithms, including random forest (RF), KNN, and artificial neural networks (ANN). They have identified 15 key features and built the model using RF, KNN, and NN. The results demonstrated that RF performs well in comparison with others, with an accuracy of 98.88%, whereas RF with all features (without using the FST) has an accuracy of 97.94%, only [14]. Using the NSL-KDD dataset, Kim et al. [15] proposed a hybrid approach for intrusion detection. The results of the experiment demonstrate that their proposed approach was more effective in terms of detection rate and time complexity. According to claim [16], the suggested approach is insufficient for time reduction, and as a result, future research will concentrate on developing the decision tree approach. In 2015, Jo et al. suggested a DT model that outperforms the NN model in terms of performance. Finally, they have demonstrated that DT is superior, with a detection rate of 91.37% [17]. In the same year, an approach that combines FS with Fuzzy-genetic IDS was proposed by Jebur et al. [18]. The article uses fuzzy logic to produce rules and used 15 features to represent the rules in order to reduce training time. The complex computing approach generates less efficient rules than soft computing. Over the UNSW-NB dataset, Mishra et al. has proposed program semantic-aware intrusion detection Net-visor security to identify attacks on virtual networks using ML methods such as DT, ANN, linear regression (LR), RF, random tree (RT), and others. However, based on their experiments RF + LR performed well compared to others in terms of accuracy, but has more false positive rate (FPR) than RT + LR [19]. To identify relevant attributes from the KDDcup 99 dataset, Mousavi et al. presented an ant colony algorithm and gradually feature removal method as FST. Then, a model has been built utilizing specified features using an ensemble of decision trees (Ada-Boost classifier). Here, the proposed technique has enhanced accuracy significantly and yielded a Matthew's correlation coefficient value to 0.91 [20]. To select important features from the NSL-KDD dataset, Sah et al. used RFE as the FS method and RF as a classifier to build a model. The proposed method enhanced the model's performance to some extent [21]. As a continuation, in 2021, Ankit et al. investigated the impact of FS methods on overall IDS performance. They have used the NSL-KDD dataset to implement RFE, IG, and chi-square as FST with various Classifiers such as NB, SVM, RF, KNN, Logistic regression, and ANN. Finally, the results have been provided in the form of a comparative study and demonstrated that RFE achieved high performance compared to others FST. However, the entire experiment has been performed and tested only on the NSL-KDD dataset that may not contain modern normal activities as per literature [22]. Hence, future research should focus on developing and evaluate the IDS model using NSL-KDD and

other modern datasets like CICIDS2017. Therefore, in 2021, on the NSL-KDD, CICIDS2017, Kyoto 2006+, and UNSW-NB15 dataset, Gu et al. [23] suggested a practical method for IDS that classifies incursion and regular instances using SVM and the NB classifiers. The proposed technique found that embedding NB with SVM produced the maximum detection accuracy when system results have compared with just one SVM algorithm. The main conclusions of SVM research also showed that SVM requires higher training time.

After studying the related works, it has been observed that the majority of researchers are interested to address the issue of large data dimensions and finding relevant features for the IDS. It's crucial to note that when data dimensions increase, ML approaches processing times grow as well [22]. Multiple FST have been employed in recent years, but still, these FST are not flexible enough to extract meaningful data from huge amounts of traffic and examining whether the selected features could increase IDS performance or not. To overcome these problems, some effective FSA are required that could reduce the features effectively and obtained a suitable set of features. This helps not only in attacks detection but also can enhance the performance of IDS and reduce the computational cost. Therefore, a decision engine approach with a features reduction strategy should be established with maintaining lightweight characteristics.

3 Proposed framework

The trade-off between a low computing cost and a high detection rate, as well as due to high dimensionality of traffic, makes difficult to develop the effective and efficient IDS models. To reduce the computational cost and increase the detection rate of IDS, this study provides a classifier which is an adaptable and effective intrusion detection technique, based on FSA, The major objective of the purposed framework is to provide a high detection rate at a minimal computing cost. The proposed framework involves five main steps, namely: dataset, data pre-processing, feature selection approaches, model building and evaluation, finally, analysis and selection phase, as shown in Fig. 1. The details descriptions of all the individual steps are defined as follows:

3.1 Dataset

A standard dataset is an essential requirement for measuring IDS performance correctly. Moreover, it also helps in evaluating the contrast of several estimators or classifiers in IDS. In the first step, standard datasets (CICIDS2017 and NSL-KDD) have been described in the below subsection before doing the pre-processing steps. These datasets are widely used for IDS and contain a sufficient number of normal activities and attack samples. The following are some detailed descriptions of the NSL-KDD and CICIDS2017 datasets.

3.1.1 NSL-KDD dataset

The NSL-KDD dataset was developed as a modified version of the KDD_1999 dataset [24]. It addresses the drawbacks of the KDD_1999 dataset for example redundant records, duplicate records, etc. In the literature, these datasets are utilized frequently for IDS evaluation. These datasets have already been prepared in two subgroups, namely training set and testing set. The NSL-KDD dataset has 41 attributes, which are divided into four (4) subgroups: basic features, content features, time-based traffic features, and host-based features. It also has five classes in which one is for the normal class, and the rests are attacks classes such as U2R, Remote-to-local (R2L), denial of services (DoS), and probing (PROBE) which are described in Table 2. The NSL_KDD dataset is utilized in this study because of the following reason:

- Redundant records have been removed from the training and testing sets that enable the classifier or estimator to produce unbiased results.
- A sufficient number of objects are available in both the training and testing sets of a dataset, allowing the experiment to be executed on the entire dataset without the requirement to select small parts or portions at random.
- It also offers numerous characteristics such as harmful scenarios, realistic network configuration, full packet capture, labelled observations, etc.

The KDDTrain+ and KDDTest files, which contain 125,973 and 22,544 objects, respectively, have been used in this research work, listed in Table 3, which illustrates the different labels of attacks types in the training and testing sets.

3.1.2 Combined CICIDS 2017 dataset

The Canadian Institute of Cyber-security [25] developed the CICIDS-2017 dataset for IDS. According to a McAfee report [26], the CICIDS-2017 dataset contains a variety of attacks that can be categorized as Web Attack, Infiltration, DoS, Brute Force, Port Scan, Distributed DoS (DDoS), Botnet attacks, etc. and is available in 8 files. The CICIDS-2017 dataset has 79 features which represents the different labels or classes.

Researchers discovered a few flaws in the CICIDS2017 dataset, including the fact that it is easily visible, the size of the dataset is huge, crossed over 8(eight) files that are captured in 5 days, and has many duplicate records which may lead to irrelevant for the IDS training phase. However, many

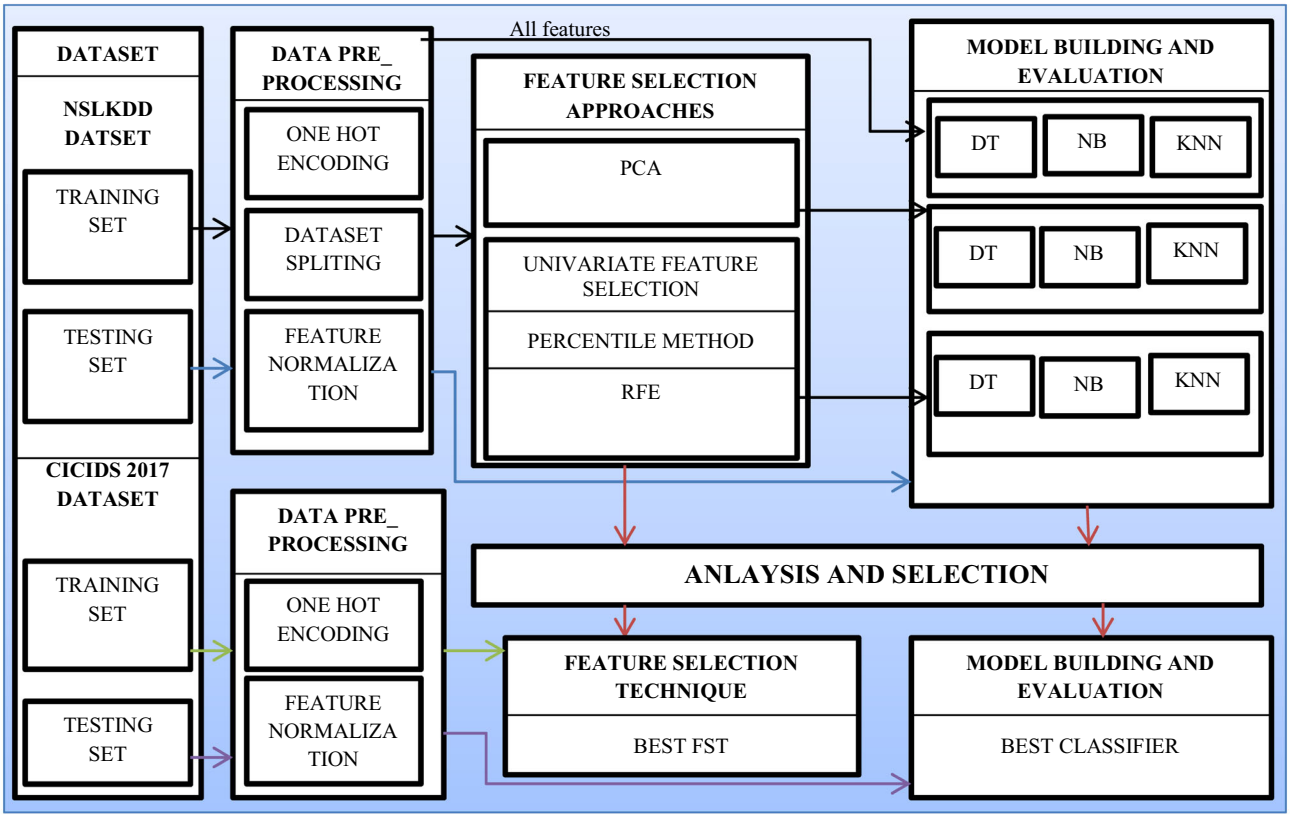

Fig. 1 Proposed framework

Table 2 Attacks and normal classes of NSL_KDD dataset with example

Sl. no.	Classes	Properties and examples
1	U2R	In this class of assault, the attacker tries to log in as a normal user and then tries to acquire access as a super user. Examples: Rootkit. Load-module, Xterm, etc.
2	R2L	In this class, the attacker tries to gain remote access of the victims' computers. Examples: internet message access protocol (IMAP), Spy, Warezclient, etc.
3	PROBE	Here, the attackers try to gain information about the victim's workstation or the network architecture, by finding the loopholes or vulnerabilities in the existing computer networks and systems. Examples: Portsweep, IPsweep, Mscan, etc.
4	DoS	These kinds of outbreaks are frequently carried out by attackers, where they try to block the legitimate users from accessing network resources employing flooding technique, which makes network busy by delivering the excessive number of packets. Examples Back, land, Neptune, etc.
5	Normal	It contains the normal activities or operations in the networks

possible solutions have also been introduced in that context [27]. It also contains an imbalance class in nature [28]. This may result in misleading estimators and influence towards the common class. Some of the shortcomings (limitations) of the dataset are given as:

- *Scattered presence* As the dataset is driven into 8 different files. Working on each individual file is a monotonous task.
- *An enormous volume of data* After integrating all eight files, the resultant set becomes very large and working on it is very tedious as it takes more time for loading and processing of the data.
- *Missing values* The dataset has many missing values that have to be removed before working on it.

The effective IDS model should be capable of the detection of any type of attack. Therefore, in order to design a classic IDS model, the data of all files of the CICIDS2017 dataset are collected and merged into a single dataset to be utilized by IDS. As a result, 3,119,345 total objects are contained in a single dataset, in which 288,602 objects that have

Table 3 Number of the objects and label distribution of different types of attacks in training and testing set

Attack's classes	Label distributions in the training set Attacks types (number of objects)	Label distributions in the testing set Attacks types (number of objects)	Total no. of objects in (training set, testing set) for each category
Normal	Normal (67,343)	Normal (9711)	(67,343, 9711)
U2R	Buffer_overflow (30), rootkit (10), Perl(3), load-module (9)	Buffer-overflow (20), xterm (13), Perl (2), sql-attacks (2), load-module (2), PS (15), rootkit (13)	(52, 67)
R2L	Warezclient (890), guess-pswd (53), waremaster (20), imap(11), ftp-write (8), multihop (7), phf (4), spy (2)	Guess-pswd (1231), waremaster (944), ftp-write (3), multihop (18), phf (2), named (17), xsnoop (4), xlock (9), httptunnel (133), snmpguess(331), sendmail(14), snmpgetattacks(178), imap (1)	(995, 2885)
Probe	Satan (3633), Ipsweep (3599), nmap (1493), port-sweep (2931)	Satan (735), Ipsweep (144), nmap (73), port-sweep (157), mscan (996), saint (319)	(11,656, 2421)
DoS	Back (956), land (18), Neptune (41,214), pod (201), smurf (2646), teardrop (892),	Back (359), land (7), pod (41), Neptune (4657), smurf (665), teardrop (12), mailbomb(293), apache2 (737), udpstrom (2), processtable (685), worm (2)	(45,927,7460)
			Total (125,973, 22,544)

Table 4 All possible types of attacks and normal traffic with new labels in the CICIDS2017 dataset

Sl. no.	New labels	Old labels	No. of objects	Total no. of objects
1	Normal	Benign	525,290	525,290
2	Botnet	Bot	457	457
3	DoS/ DDoS	DDoS	29,751	87,939
		DoS Golden Eyes	2402	
		DoS Hulk	53,159	
		DoS-Slowloris	1333	
		DoS-Slowhttptest	1290	
		Heartbleed	4	
4	Infiltration	Infiltration	8	8
5	Port Scan	Port Scan	36,895	36,895
6	Brute Force	FTP-Patator	1835	3211
		SSH-Patator	1376	
7	Web Attack	Web Attack-Brute Force	381	521
		Web Attack-XSS	138	
		Web Attack-SqlInjection	2	
Total = 654,321				

missing class labels are removed. Finally, the combined dataset left with 2,830,743 objects.

- Dimension of combined CICIDS2017 dataset: (2830743,79).

By merging all traffic files of CICIDS2017 into a single dataset, the scattered presence problem has been solved. The missing values have also been eliminated from the combined CICIDS2017 dataset. The CICIDS2017 dataset becomes enormously large after integrating all 8 files into a single dataset; hence, in this research, a sample of 654,321 records has been picked for experiment purposes. Table 4 provides an illustration of the updated labelling for all attack traffic in the CICIDS2017 dataset. Further, the dimensions of the training and testing dataset are 523,456 and 130,865, respectively, as shown in Table 5. Table 6 describes the dimension of attacks classes from the dataset (sample of combined CICIDS2017) that have been used for implementation.

Table 5 Record of different types of Attack and normal traffic in training and testing set of sample of combined CICIDS2017 dataset

Training set		Testing set	
Attack types	No. of records	Attack types	No. of records
Benign	420,432	Benign	104,934
DoS hulk	42,833	DoS hulk	10,779
Port Scan	29,290	Port Scan	7406
DDoS	23,642	DDoS	5917
DoSGoldenEye	1917	DoSGoldenEye	490
FTP-Patator	1387	FTP-Patator	378
SSH-Patator	1134	SSH-Patator	260
DoSSlowhttptest	968	DoSSlowhttptest	265
DoSslowloris	1079	DoSslowloris	259
Bot	359	Bot	89
Web Attack_Brute Force	283	Web Attack_Brute Force	62
Web Attack_XSS	120	Web Attack_XSS	24
Infiltration	7	Infiltration	1
Heartbleed	1	Heartbleed	-
Web Attack_Sql Injection	4	Web Attack_Sql Injection	1
Total	523,456	Total	130,865

Table 6 Dimension of each attacks class (including normal objects) in the dataset used for the implementation

Sl. no.	Attack classes	Dimension in the training set	Dimension in the testing set
1	Botnet	(420791, 79)	(105023, 79)
2	DoS/ DDoS	(490872, 79)	(122644, 79)
3	Infiltration	(420439, 79)	(104935, 79)
4	Port Scan	(449722, 79)	(112340, 79)
5	Brute Force	(422953, 79)	(105572, 79)
6	Web Attack	(420839, 79)	(105021, 79)

3.2 Data pre-processing

The dataset must be pre-processed in order to verify the models and methods. Several operations are performed in this phase, including the replacement of noise-values such as infinity or null symbols with mean or zeros, feature transformation, normalization, and splitting. Pre-processes are needed for both datasets to verify models and methods.

3.2.1 One-hot- encoding

Basically, one-hot-encoding is used for data transformation from non-numerical to binary vector. As stated before, the CICIDS2017 dataset has one additional column for defining label or class and 78 regular attributes, in which Fwd Header Length, flow packets and bytes are always carried the same entries. As a result, the features like flow packets and bytes are eliminated from the combined CICIDS2017 dataset. Hence, 76 attributes with one label column are left for analysis. As all these 77 attributes represent a numerical type of data; thus data transformation isn't required. On the other hand, the NSL-KDD dataset has both non-categorical (numeric type) and categorical features (non-numeric), so data transformation is required. These categorical features such as flags, services, protocol types of the NSL-KDD dataset have symbolic entities that are transformed into numerical values using *LabelEncoder,* as an example considers the protocol-types feature, which contains three categories (UDP, TCP, and ICMP). These categories are mapping to numeric features as 1, 2 and 3. Later, these obtained numerical values are denoted in binary vector for training and testing purposes using One-Hot-Encoding.

3.2.2 Splitting the datasets

The combined CICIDS2017 dataset has been split into six (6) different parts based on every attacks category, whereas the NSL-KDD dataset has split into four (4) different parts based on attacks types, namely U2R, R2L, probe, and DoS, to train the models for all types of attacks and test the models correctly.

3.2.3 Feature normalization

The next operation after performing the pre-processing is feature normalization using the standardization formula given in Eq. (1) [29], where Z represents as Z-score. Feature normalization makes all attributes within the identical scale and prevents the large numeric value of features, which gives them more importance in classification algorithms. As a result, the classifier allocates the same weight to every feature. Further linear transformation given in Eq. (2) is also used that transforms the feature's value set into a new specific set within the range (0–1)[30].

$$Z = (B - \mu)/\sigma \tag{1}$$

In Eq. (1), the mean value (μ) is subtracted from the feature value (represent by 'B'). The result is divided by their standard deviation (σ). In Eq. (2), the min and max stands for minimum and maximum, respectively.

$$B_{\text{normalization}} = (B - min(B))/(max(B) - min(B)) \quad (2)$$

3.3 FSA

FSA are utilized to remove redundant, irrelevant, or unimportant data. The main purpose of these techniques is to obtain a subset or optimal set of important features from underlying features that can easily classify the given problem or data in different classes or labels. FSA can help in handling the high-dimensional dataset and compute the importance of the feature that supports in data-interpretation.

As stated in Sect. 3.2.1**,** features (flow packets and bytes) have been eliminated from the combined CICIDS2017 dataset, in our experiment. As a result, there are 77 attributes have remained to be analysed. On the other hand, 41 features have remained of the NSL-KDD dataset. In this phase, the PCA, univariate feature selection using analysis of variance (ANOVA) *F test*, followed by RFE approaches have been utilized as FSA to reduce the features and acquire the appropriate subset or optimal set of features from the original set. The detailed descriptions of these approaches are mentioned below.

3.3.1 PCA

The PCA [31] method is similar to clustering, which falls under the category of unsupervised learning. PCA rationalizes the complexity in high-dimensional data by maintaining the tendency and patterns. PCA does this by modifying the data to fewer dimensions, later which act as an outline of features. It identifies the patterns from the data without reference to precision about whether the samples of data come from different treatment groups or have phenotypic differences. PCA is primarily utilized to reduce the number of attributes of the dataset by changing a large set of variables to a smaller one, but it still contains information in the dataset.

3.3.2 Univariate feature selection

In univariate feature selection [32] using ANOVA *F* test [33], each attribute or feature is independently analysed to identify the influence of correlation of an attribute with labels or class. It picks the best attributes based on a univariate statistical-test and works (performed) on them. Here ANOVA is a procedure of comparing each attribute with the target class to understand whether any statistically remarkable connection between them occurs or not. During this procedure, it suppresses other features to obtain the test score for every feature. In the end, all features' scores are compared in order to obtain and select the topped score features.

3.3.3 Percentile method

The Sklearn library's Percentile method or *selectPercentile* [34] is used to select attributes based on the percentile of the highest scores. Furthermore, the default function in selectPercentile is ANOVA, which is solely applicable to the classification task.

3.3.4 RFE

After the Percentile method, RFE [35] is utilized as FST to identify and pick the important features for categorizing (classifying) the network traffic. RFE selects and eliminates the features on the basis of ranks. It starts the feature elimination one attribute at a time that has the lowest rank. The main purpose of RFE is to acquire the best subset of features in terms of performance. RFE [36] method evaluates the performance of an estimator or classifier using elimination properties in iterative manners given as follows:

- Taking a sub-optimal set of features building the model of classification.
- To provide the rank of features, it computes the importance of features.
- Eliminate the lowest rank features based on the relevance of features.

3.4 Model building and evaluation

In this phase, DT, NB, and KNN classifiers have been used to build the models using reduced features (by FSA) and all of the training set's features from the NSL-KDD dataset. Further, on the testing set of the NSL-KDD dataset, the recall, precision, accuracy, and *F*-measure matrices have been calculated to determine the predictions rate of these models. Here, during the procedure of samples considering, learning, and validation, the 10-fold cross-validations have been performed and utilized to measures the performances of the models to unwavering all objects impact.

Later, the best classifier has obtained based on the performance of classifiers on the NSL-KDD dataset and used the same identified classifier with reduced features (by best FST) and all features of the training set into the CICIDS2017 dataset to train the model. Further, this model is evaluated using the testing set of the CICIDS2017 dataset. The detailed descriptions and the working principle of these classifiers are mentioned below.

3.4.1 DT

DT classifier [37] is tree-structured and comprised of edges and nodes. In these trees, every node denotes the problem

category that needs to be classified, whereas every edge represents a decision that is taken based on the evaluated data. These trees can be either regression trees or classification trees. The classifier (DT) can be viewed as a predictive model of ML that illustrates a mapping between dataset attributes and their corresponding value. In DT, every part represents possible values that are considered for a specified (given) category. The tree nodes are recognized using the estimated entropy of dataset properties. The property with the top entropy value is referred to as a root node. The examples of broadly adopted DT models are classification and regression trees (CART), C4.5, and iterative dichotomiser 3.

DT classifier has several advantages: firstly, it is simple and easy to understand with a short-term explanation. Secondly, implications can be consequent, on the basis of different probability-estimation and costs. These Implications can be utilized to obtain detailed outputs. Finally, to obtain correct results, it is flexible to combine with other classification-model, but it also has a limitation while considering the data of similar type, in such cases its accuracy is relatively lower. Moreover, it is not adaptive means minor modifications in data fed to an estimator may lead to a highly unstable decision tree organization.

3.4.2 NB

NB is a supervised learning method based on the Bayes theorem. The NB method is premised on the reality (statement) that the existence of one feature/s is independent of other features of a class. The Bayes theorem has been applied to calculate the posterior_probability P(cl | y) from P(y | cl), P(y), and P(cl). The formula will be given in equation (3) [38]:

$$(\mathrm{cl}|\mathrm{y}) = \frac{(\mathrm{y} \mid \mathrm{cl}).\mathrm{P}(\mathrm{cl})}{\mathrm{P}(\mathrm{y})} \tag{3}$$

where the posterior_probability P(cl | y) of class (cl, target) given predictor (y, features). The prior probability of predictor and class is denoted by P(y) and P(cl), respectively. The likelihood, or probability of the predictor given class, is represented by the expression P(y | cl).

The advantage of the NB algorithm is that it is extremely scalable and speedy in classification. Also, it can utilize for both multi-class and binary problems of classification. As it depends on the theory that any feature is not dependent on others, it cannot establish any relationship among class features. Also, its implementation is more complex with large datasets.

3.4.3 KNN

KNN algorithm is one of the most basic supervised ML classifiers. It supposes that new data and existing data are similar, and new data are allocated to the category that most closely resembles. It puts down all existing data and further categorizes a new data point based on its similarity to the existing data. This means whenever new data arise then they can be effortlessly classified into a pertinent group by utilizing the KNN. It can be utilized for regression as well as for classification, but most over the time it is used for classification problems [39]. It is a nonparametric type of procedure that means, and it does not make any presumption on elementary data.

3.5 Analysis and selection phase

After developing the models, the last phase evaluates the NSL-KDD dataset using FSA such as PCA and RFE with several classifiers such as KNN, DT, and NB. These classifiers with all features have also been considered to build the models for analysis purposes. The performance of these models has been measured using metrics such as recall, precision, *F*-measure, and accuracy to find the best classifier and suitable FST on the NSL_KDD dataset. After obtaining the best classifier and best FST, the same has also been used on CICIDS2017 datasets too, for building the models and analysis purposes.

3.5.1 Evaluation metrics

Primarily evaluation metrics such as *F*-measure, recall, precision, accuracy, training time, testing time, specificity, and, *G*-means: mainly appropriate for imbalanced datasets for analysing or measuring the models' performance. The basic attributes for measuring the performance of the models are defined as follows:

- *True positive (TP)* TP denotes the number of normal objects which are successfully categorized by the model as normal.
- *False positive (FP)* FP indicates a number of normal samples which are wrongly classified by the model as assaults.
- *True negative (TN)* TN specifies the number of attacks trials that are correctly classified (predicted) by a model as attacks.
- *False negative (FN)* FN denotes the number of attacks samples that are mistakenly classified (predicted) by a model as normal.

Accuracy (*A*) Accuracy [40] rate measures the IDS model's accuracy when predicting the traffic as normal or attacks. It

Table 7 Selected features of NSL-KDD dataset using RFE and PCA

FSA	Selected Features for each category sorted by their ranks			
	U2R	R2L	PROBE	DoS
RFE	[(1,'urgent'), (2,'hot'), (3,'root_shell'), (4,'num_file_creations'), (5,'num_shells'), (6,'srv_diff_host_rate'), (7,'dst_host_count'), (8,'dst_host_srv_count'), (9,'dst_host_same_src_port_rate'), (10,'dst_host_srv_diff_host_rate'), (11,'service_ftp_data'), (12,'service_http'), (13,'service_telnet')]	[(1,'src_bytes'), (2,'dst_bytes'), (3,'hot'), (4,'num_failed_logins'), (5,'is_guest_login'), (6,'dst_host_srv_count'), (7,'dst_host_same_src_port_rate'), (8,'dst_host_srv_diff_host_rate'), (9,'service_ftp'), (10,'service_ftp_data'), (11,'service_http'), (12,'service_imap4'), (13, 'flag_RSTO')]	[('logged_in'), (2,'rerror_rate'), (3, 'srv_rerror_rate'), (4,'dst_host_srv_count'), (5,'dst_host_diff_srv_rate'), (6,'dst_host_same_src_port_rate'), (7,'dst_host_srv_diff_host_rate'), (8,'dst_host_rerror_rate'), (9,'dst_host_srv_rerror_rate'), (10,'Protocol_type_icmp'), (11,'service_eco_i'), (12,'service_private'), (13,'flag_SF')]	[(1,'logged_in'), (2, 'count'), (3,'serror_rate'), (4,'srv_serror_rate'), (5,'same_srv_rate'), (6,'dst_host_count'), (7,'dst_host_srv_count'), (8,'dst_host_same_srv_rate'), (9,'dst_host_serror_rate'), (10,'dst_host_srv_serror_rate'), (11,'service_http'), (12,'flag_S0'), (13, 'fag_SF')]
PCA	[(1,'src_bytes'), (2,'dst_bytes'), (3,'hot'), (4,'root_shell'), (5,'num_shells'), (6,'dst_host_count '), (7,'dst_host_same_srv_rate '), (8,'dst_host_srv_diff_host_rat')]	[(1,'duration'), (2,'src_bytes'), (3,'hot'), (4,'num_failed_logins'), (5,'dst_host_same_srv_rate '), (6,'dst_host_same_src_port_rate'), (7,'dst_host_srv_diff_host_rate '), (8,'service_ftp_data')]	[(1,'src_bytes'), (2,'dst_bytes'), (3,'dst_host_same_srv_rate'), (4,'dst_host_diff_srv_rate'), (5,'dst_host_rerror_rate '), (6,'service_ftp_data'), (7,'service_http'), (8,'service_smtp')]	[(1,'src_bytes'), (2,'dst_bytes'), (3,'wrong_fragment'), (4,'num_compromised'), (5,'same_srv_rate'), (6,'dst_host_serror_rate'), (7,'dst_host_srv_serror_rate'), (8,'service_ecr_i')]

Table 8 Performance evaluations for ML classifiers with all features using NSL-KDD dataset

Methods	Attack types	Accuracy	Precision	Recall	F-measure
NB	U2R	97.25	60.13	97.91	66.06
	R2L	93.56	89.09	95.50	91.62
	PROBE	97.89	97.32	96.05	96.65
	DOS	86.73	98.82	70.30	82.14
KNN	U2R	99.60	93.14	85.07	87.83
	R2L	96.73	95.31	95.48	95.38
	PROBE	99.07	98.60	98.50	98.55
	DOS	99.71	99.67	99.66	99.67
DT	U2R	99.66	86.48	91.67	88.62
	R2L	97.45	96.68	96.08	96.38
	PROBE	99.08	98.67	98.46	98.56
	DOS	99.63	99.50	99.66	99.58

Table 9 Performance evaluations for ML classifiers with selected features using RFE on NSL-KDD dataset

Methods	Attack types	Accuracy	Precision	Recall	F-measure
NB	U2R	98.97	67.85	74.76	70.12
	R2L	81.78	74.81	66.76	69.02
	PROBE	94.62	94.30	87.26	90.22
	DOS	87.36	97.24	72.98	83.37
KNN	U2R	99.55	91.36	74.00	79.10
	R2L	94.59	92.09	92.56	92.31
	PROBE	98.85	97.57	97.80	97.68
	DOS	98.14	98.08	97.57	97.82
DT	U2R	99.65	87.74	89.18	87.49
	R2L	97.92	97.15	96.95	97.05
	PROBE	99.57	99.39	99.26	99.32
	DOS	99.73	99.69	99.70	99.69

is represented by the given formula

$$A = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{TP} + \text{FP} + \text{FN}}$$

G-means (G_m) G_m [41] is derived from specificity and sensitivity. It is mainly appropriate for imbalanced datasets. It is computed as

$$G\_\text{m} = \sqrt{(\text{specificity} \times \text{sensitivity})}$$

Specificity (S) Specificity is another name for the true negative rate. It is represented by the given formula

$$S = \frac{\text{TN}}{\text{FP} + \text{TN}}$$

Recall (sensitivity) True positive rate (TPR) or detection rate is other terms for recall. It's calculated using the given formula

$$\text{Recall}(R) = \frac{\text{TP}}{\text{FN} + \text{TP}}$$

Precision (P) Precision is the ratio of true positive (predicted by model correctly) to the total number of positive cases [41]. It is calculated by the given formula

$$P = \frac{\text{TP}}{\text{FP} + \text{TP}}$$

F-measure (F_m) F_m is a weighted harmonic_average of recall and precision. It is mainly suitable for imbalanced datasets. It can be calculated by the given [42] formula

$$F\_\text{m} = 2 * (R * P)/(R + P)$$

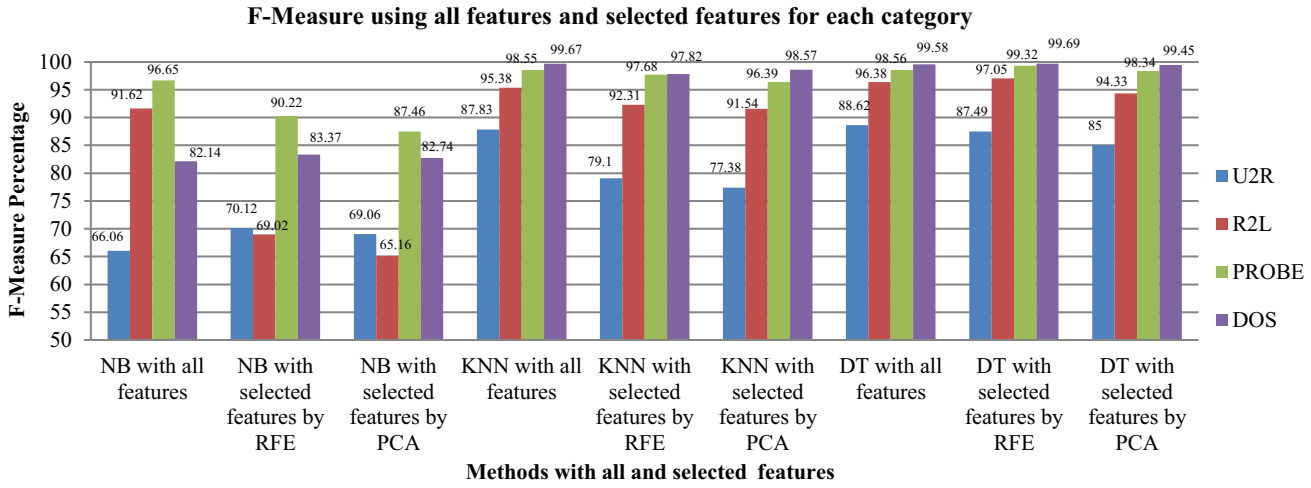

Fig. 2 Different classifiers (NB, DT, and, KNN) *F*-measure using selected features and all features for each category

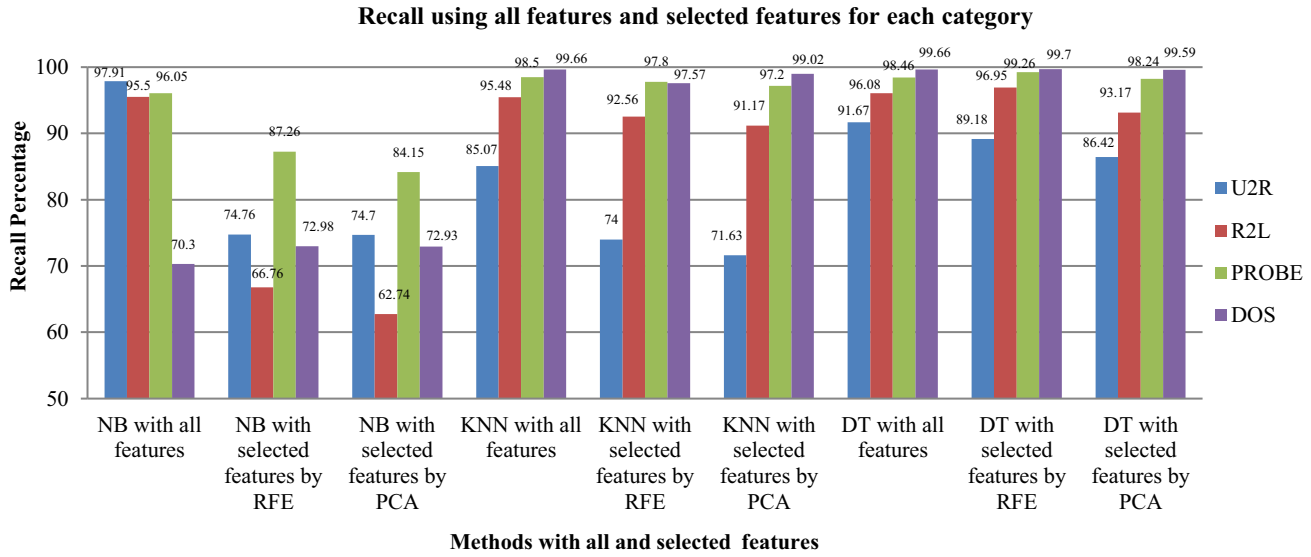

Fig. 3 Different classifiers recall using selected features and all features for each category

Training time (T_1) in seconds (s) T_1 describes the amount of time that a technique utilized for training and building the model using an entire training set of a dataset. It is represented by the given formula [43]

$$T_1 = \text{End}_{\text{training\_time}} - \text{Start}_{\text{training\_time}}.$$

Testing time (T_2) in seconds (s) T_2 describes the amount of time that a technique utilizes to predict the whole testing set of a dataset as either attack or normal. It is computed as [43]

$$T_2 = \text{End}_{\text{testing\_time}} - \text{Start}_{\text{testing\_time}}.$$

4 Experimental setup and results analysis

For the experimental purpose, the *Kaggle* platform has been used in this research, which is a cloud-based online resource where Python programming can be used by utilizing 'Sklearn' (ML library implemented in Python) [44]. *Kaggle* has a maximum memory of 16 Gigabyte (GB) and a storage capacity of 4.9 GB, allowing the users to upload and explore data-analysis models. In this research, the entire experiment has performed through Windows 10 with a quad-core 3.6 Gigahertz (GHz) processor.

4.1 Results and discussion

In this research, to investigate the effectiveness of the proposed models, the CICIDS2017 and NSL-KDD datasets have

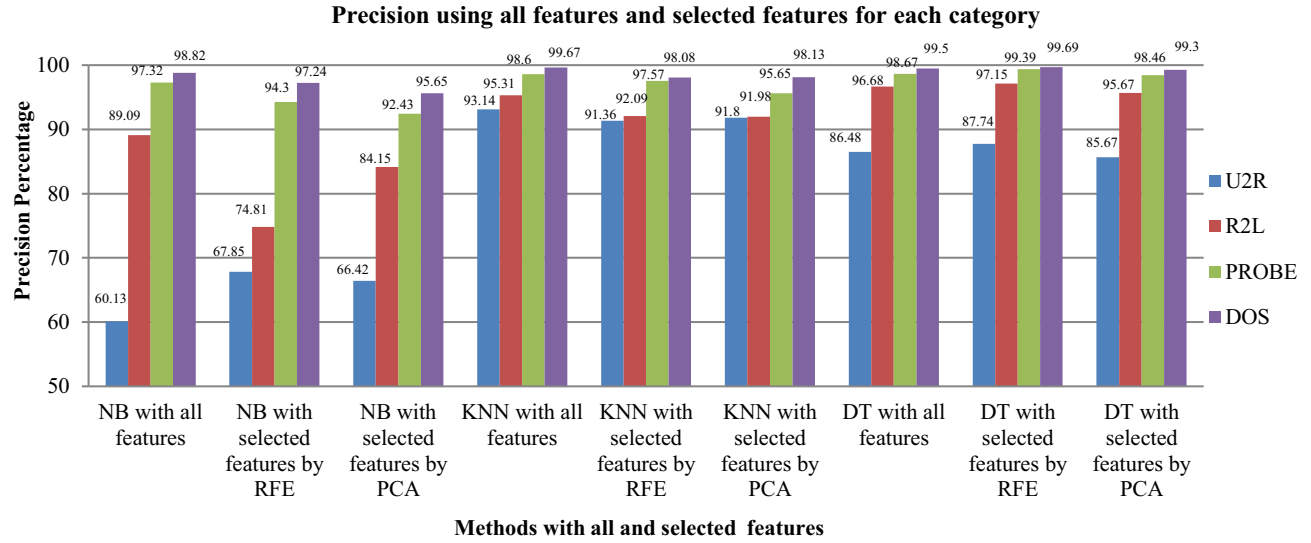

Fig. 4 Different classifiers precision using selected features and all features for each category

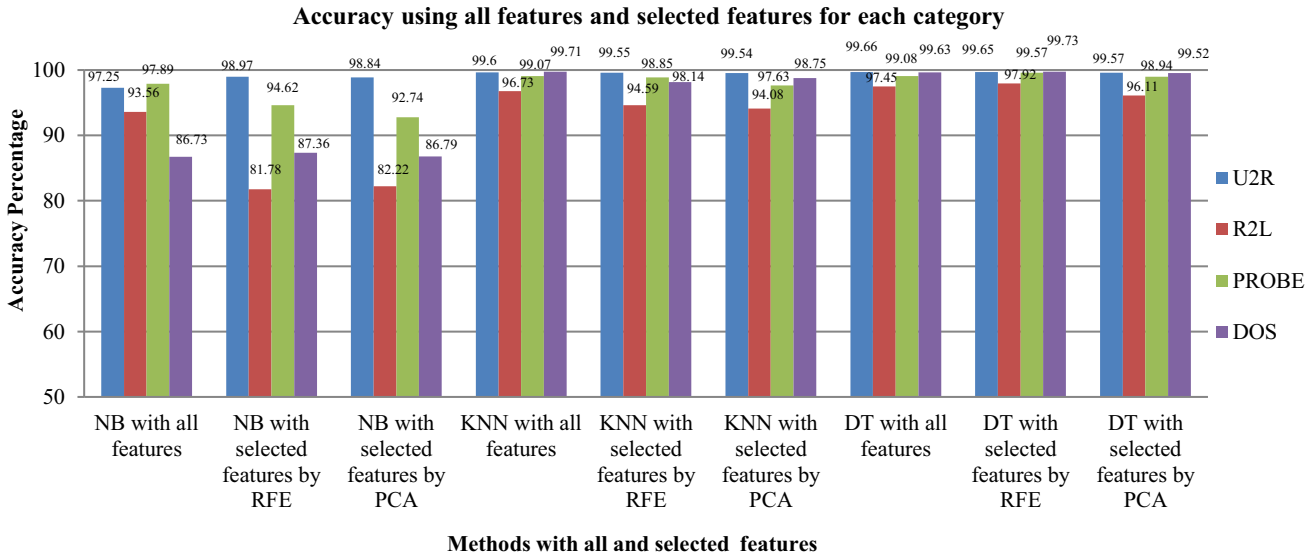

Fig. 5 Different classifiers accuracy using selected features and all features for each category

been used. Initially, to remove the non-relevant features and find the important ones, the RFE and PCA-based FSA has been employed on the dataset. Then, these FSA has been implemented with several ML classifiers for example DT, NB, and, KNN to enhance and measure the model's performance on the NSL-KDD datasets. The accuracy, recall, precision, and *F*-measure metrics stated in Sect. 3.5 have been used to analyse the experiment outcomes. However, based on outcomes, other metrics such as specificity, *G*-means, testing, and training time have also been calculated for identifying the best-performing classifier. Moreover, based on the best-performing classifier with the selected FST, the model has also been examined on a real-time dataset (i.e. CICIDS2017).

4.1.1 Results analysis using NSL-KDD dataset

Since IDS enables early detection of intrusion, feature extraction and selection is always a crucial and difficult task in network security. However, it has a considerable impact on both model performance and computational complexity. The main goal of feature selection approaches is to completely depict a problem by picking a subset of important features from the whole dataset. As a result, working with fewer features may yield better outcomes. Therefore, in this experiment, RFE and PCA have been applied to the NSL-KDD dataset to provide the best set of features.

Table 10 Performance evaluations for ML classifiers with selected features using PCA on NSL-KDD dataset

Methods	Attack type	Accuracy	Precision	Recall	F-measure
NB	U2R	98.84	66.42	74.70	69.06
	R2L	82.22	84.15	62.74	65.16
	PROBE	92.74	92.43	84.15	87.46
	DOS	86.79	95.65	72.93	82.74
KNN	U2R	99.54	91.80	71.63	77.38
	R2L	94.08	91.98	91.17	91.54
	PROBE	97.63	95.65	97.20	96.39
	DOS	98.75	98.13	99.02	98.57
DT	U2R	99.57	85.67	86.42	85.00
	R2L	96.11	95.67	93.17	94.33
	PROBE	98.94	98.46	98.24	98.34
	DOS	99.52	99.30	99.59	99.45

Table 11 Performance evaluations for DT classifier with all and selected features using RFE on NSL-KDD dataset

Methods	Attack type	Recall	Specificity	G-means	Accuracy	Training time (T_1) (s)	Testing time (T_2) (s)
DT with all features	U2R	91.67	46.67	65.41	99.66	0.364	0.273
	R2L	98.08	83.98	89.83	97.45	0.511	0.334
	PROBE	98.46	48.59	69.17	99.08	0.774	0.414
	DOS	99.46	95.62	97.62	99.63	1.138	0.652
					Average (98.96)	Total T_1 (2.787)	Total T_2 (1.673)
DT with selected features	U2R	89.18	75.00	81.78	99.65	0.114	0.103
	R2L	96.95	98.73	97.84	97.92	0.212	0.115
	PROBE	99.26	68.89	82.69	99.57	0.174	0.143
	DOS	99.70	97.80	98.75	99.73	0.251	0.244
					Average (99.21)	Total T_1 (0.751)	Total T_2 (0.605)

In order to reduce the computation cost of proposed models, the RFE and PCA have identified the 13 and 8 important features (rank wise), respectively, from the NSL-KDD datasets for each classifier, which are shown in Table 7. These approaches (PCA and RFE) have provided an appropriate set of attributes or features that are passed over several classifiers (DT, NB, and KNN) for training and testing in order to construct an IDS model. These classifiers have employed to build a model for evaluation and comparison purposes. Tables 9 and 10 demonstrate the performance evaluations of the classifiers with selected features by RFE and PCA, respectively, on NSL-KDD dataset, whereas Table 8 demonstrates the performance of classifiers with all features using NSL-KDD dataset. Here, the performance of the classifiers has been measured based on four matrices namely, recall, precision, F-measure and accuracy.

Figures 2, 3, 4, and 5 illustrate the F-measure, recall, precision, and accuracy of different algorithms (DT, KNN, and NB) using all features and using selected features (via RFE, PCA), respectively. The graph's Y-axis depicts the percentage of performance metrics (F-measure, recall, precision, and accuracy) predicted by various approaches for each category (Probe, DoS, U2R, R2L), while the X-axis depicts various methods such as DT, NB, and KNN.

After analysing the results (Tables 8, 9, 10, Figs. 2, 3, 4, 5) on the NSL-KDD dataset with and without FSA (i.e. using all features), it has been found that the RFE with DT offered superior performance in terms of accuracy. Additionally, the RFE with DT offered higher performance in terms of F-measure, recall, and precision for attack types including DoS, Probe, and R2L. However, the KNN with all features provided better performance in terms of F-measure and precision for U2R attack type. The NB with all features showed better recall performance for U2R attack type.

Further study of the findings for the NSL-KDD (Tables 8, 9, 10, Figs. 2, 3, 4, 5) shows that the RFE performs better

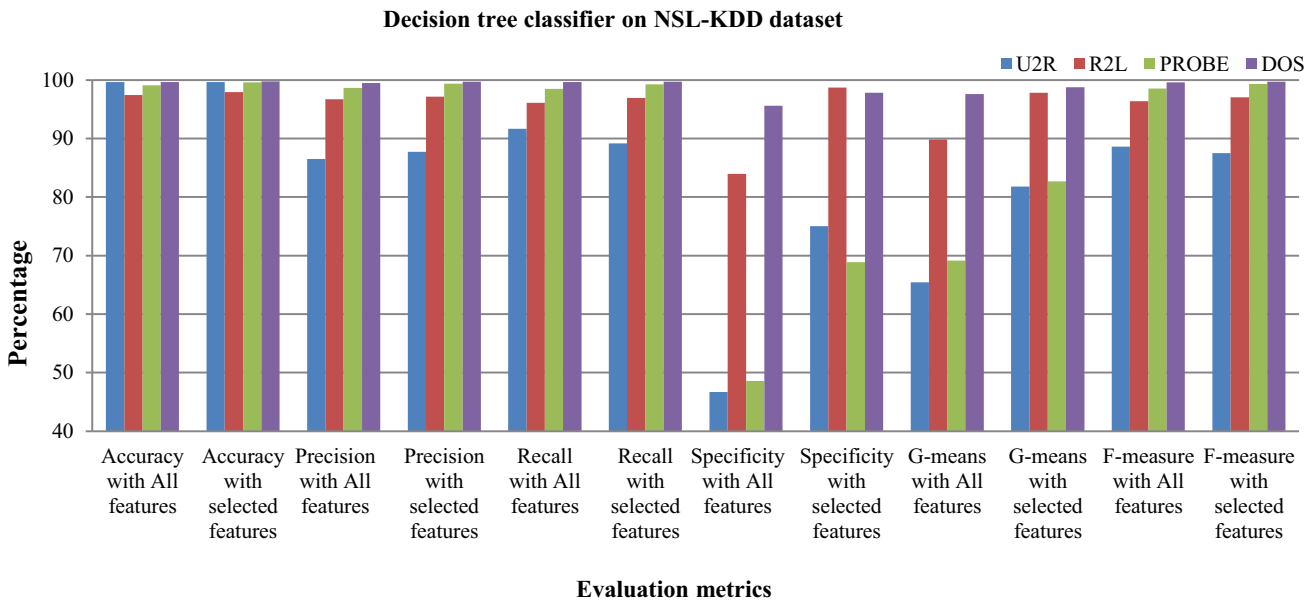

Fig. 6 Evaluation matrices of DT classifiers using selected features (by RFE) and all features for each category on the NSL-KDD dataset

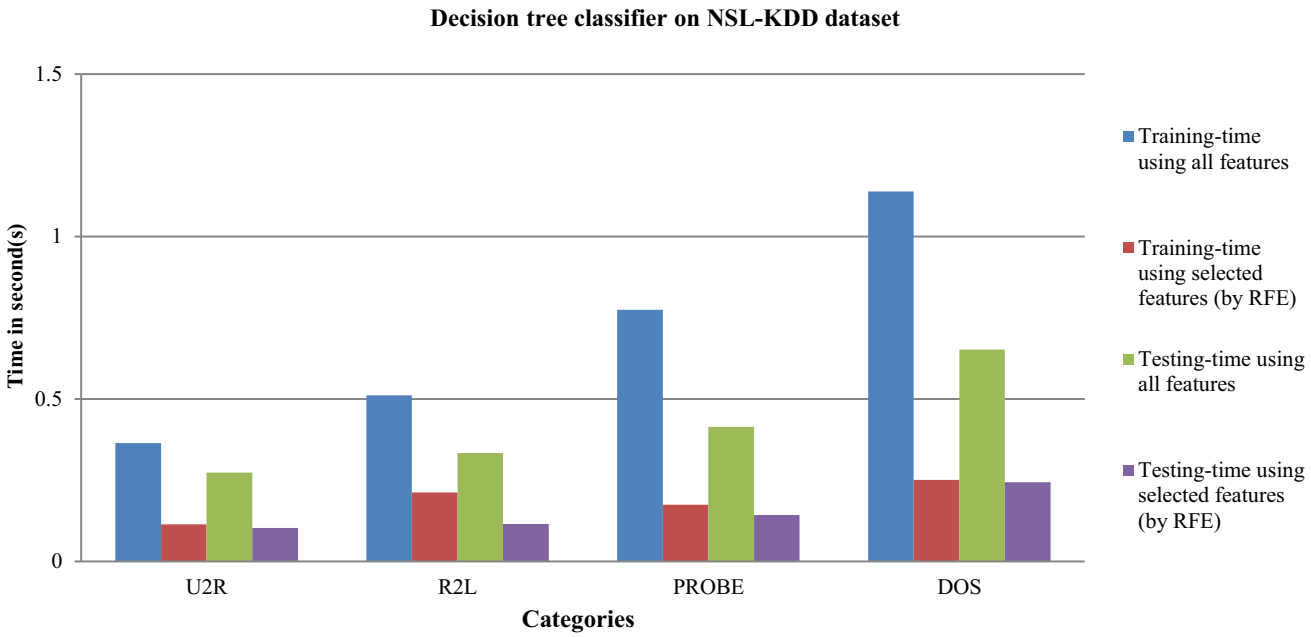

Fig. 7 DT classifier's training and testing time using selected and all features for each category on the NSL-KDD dataset

with DT than PCA with DT, PCA with NB, and PCA with KNN in terms of *F*-measure, precision (with the exception of U2R attack type for PCA with KNN), recall, and accuracy among the FSA.

However, the KNN classifier performs well in terms of Precision under a few specific conditions, specifically those involving U2R attack types. When recall is taken into account, the NB classifier in one instance produces similar results, but only for U2R attack types. Hence, for the majority of the attack types given in the NSL-KDD dataset, the DT classifier using RFE offers higher results in terms of accuracy, *F*-measure, precision, and recall. Therefore, for further investigation, DT as a classifier and RFE as an FST are taken into consideration in the proposed model. Moreover, the other metrics such as specificity, *G*-means, testing, and training time are also calculated for classifier DT with RFE. On the NSL-KDD dataset, Table 11 demonstrates the performance evaluations for the DT classifier with all features and the DT classifier with selected features (by RFE). Figure 6 illustrates the evaluation matrices of DT Classifiers

Table 12 Selected features fromCICIDS2017 combined dataset using RFE

Method	Selected features for each category sorted by their ranks					
	Botnet	DoS	Infiltration	Port Scan	Brute Force	Web_Attack
DT	[(1,'Destination Port'), (2,'Fwd Packet Length Mean'), (3,'Flow IAT Max'), (4,' Flow IAT Min'), (5,'Fwd IAT Std'), (6, 'Bwd IAT Std'), (7,'Bwd Packets/s'), (8,'Init_Win bytes_forward')]	[(1,'Destination Port'), (2, ' Total Backward Packets'), (3, ' Bwd Packet Length Std'), (4, ' Flow IAT Mean'), (5, 'FIN_Flag Count'), (6, ' Fwd Header Length.1'), (7,'Init_Win bytes_forward'), (8,'Init_Win bytes_backward')]	[(1,'Destination Port'), (2, 'Total Length of Fwd Packets)', (3, ' Bwd Packet Length Mean'), (4, ' Fwd IAT Max'), (5, ' Fwd IAT Min'), (6, ' Active Min'), (7, 'Idle Mean'), (8, ' Idle Std')]	[(1,'Destination Port'), (2, ' Flow Duration'), (3, 'Total Length of Fwd Packets'), (4, ' Total Length of Bwd Packets'), (5, ' Bwd Packet Length Min'), (6, ' Fwd IAT Min'), (7, ' Bwd Packets/s'), (8, 'PSH Flag Count')]	[(1,'Destination Port'), (2, 'Total Fwd Packets'), (3, ''Total Length of Fwd Packets'), (4, 'Flow IAT Min'), (5, ' Fwd IAT Min'), (6, 'Bwd Packets/s'), (7,'Init_Win bytes_backward'), (8, 'min_seg_size_forward')]	[(1,' Total Length of Fwd Packets'), (2, 'Bwd Packet Length Std '), (3, ' Flow IAT Min'), (4, ' Fwd IAT Min'), (5, ' Bwd Packets/s'), (6, ' Max Packet Length'), (7, 'Init_Win bytes_forward '), (8,'Init_Win bytes_backward ')]

Table 13 Performance evaluations for DT classifier with all features on CICIDS2017 combined dataset

Method	Attack type	Precision	*F*-measure	Specificity	Recall	*G*-means	Accuracy	Training time (T_1) (s)	Testing time (T_2) (s)
DT	Botnet	79.57	76.16	97.61	75.55	85.87	99.95	8.199	0.361
	DoS/DDos	99.84	99.85	96.53	99.87	98.19	99.93	26.583	0.808
	Infiltration	79.99	79.99	50.00	79.99	63.24	99.90	3.916	0.621
	Port Scan	99.64	99.65	82.89	99.67	90.89	99.91	7.965	0.619
	Brute Force	99.92	99.92	85.98	99.92	92.69	99.90	2.758	0.641
	Web attack	97.17	95.60	52.29	94.54	70.31	99.98	6.189	0.247
							Average (99.92)	Total T_1 (55.61)	Total T_2 (3.29)

employing selected features (by RFE) and all features for each category. Figure 7 shows the testing and training time for the DT classifier with all features and the DT classifier with selected features (by RFE) for each category. In comparison with DT with all features, the DT classifier with selected features (by RFE) requires low training time and testing time. Further, the U2R category requires the lowest training and testing time.

After analysing the results (Table 11, Figs. 6, 7) of DT with selected features (by RFE) and DT with all features on the NSL-KDD dataset, it is observed that the DT classifier with selected features (by RFE) yields higher results in terms of average accuracy, total training time, total testing time, *G*-means, and specificity. Additionally, results analysis for the NSL-KDD shows that DT with RFE increased model performance in terms of precision, *G*-means, accuracy (approximately identical accuracy for U2R), and specificity for each attack category while reducing computational costs in terms of training time and testing time. In terms of recall, and *F*-measure, the DT with RFE also offers higher results for the attack categories Dos, R2L, and Probe. For the assault U2R category, it generates *F*-measure and recall, which have only slight value changes.

4.1.2 Results analysis using combined CICIDS2017 dataset

After analysing the results on the NSL-KDD dataset, the best-performing classifier with the appropriate FST, that is DT + RFE has been adopted and evaluated on a real-time dataset (i.e. CICIDS2017) to examine the model's sustainability on some new real-time datasets for IDS. Table 12 shows "8" important features selected from the CICIDS2017combined dataset using RFE. Then these features have passed over the DT classifier to build a model. Performance evaluations for the DT method with all features and selected features using RFE on the CICIDS-2017 combined dataset are shown in Tables 13 and 14.

On the combined CICIDS2017 dataset, Fig. 8 shows that the DT classifier with selected features (by RFE) and all

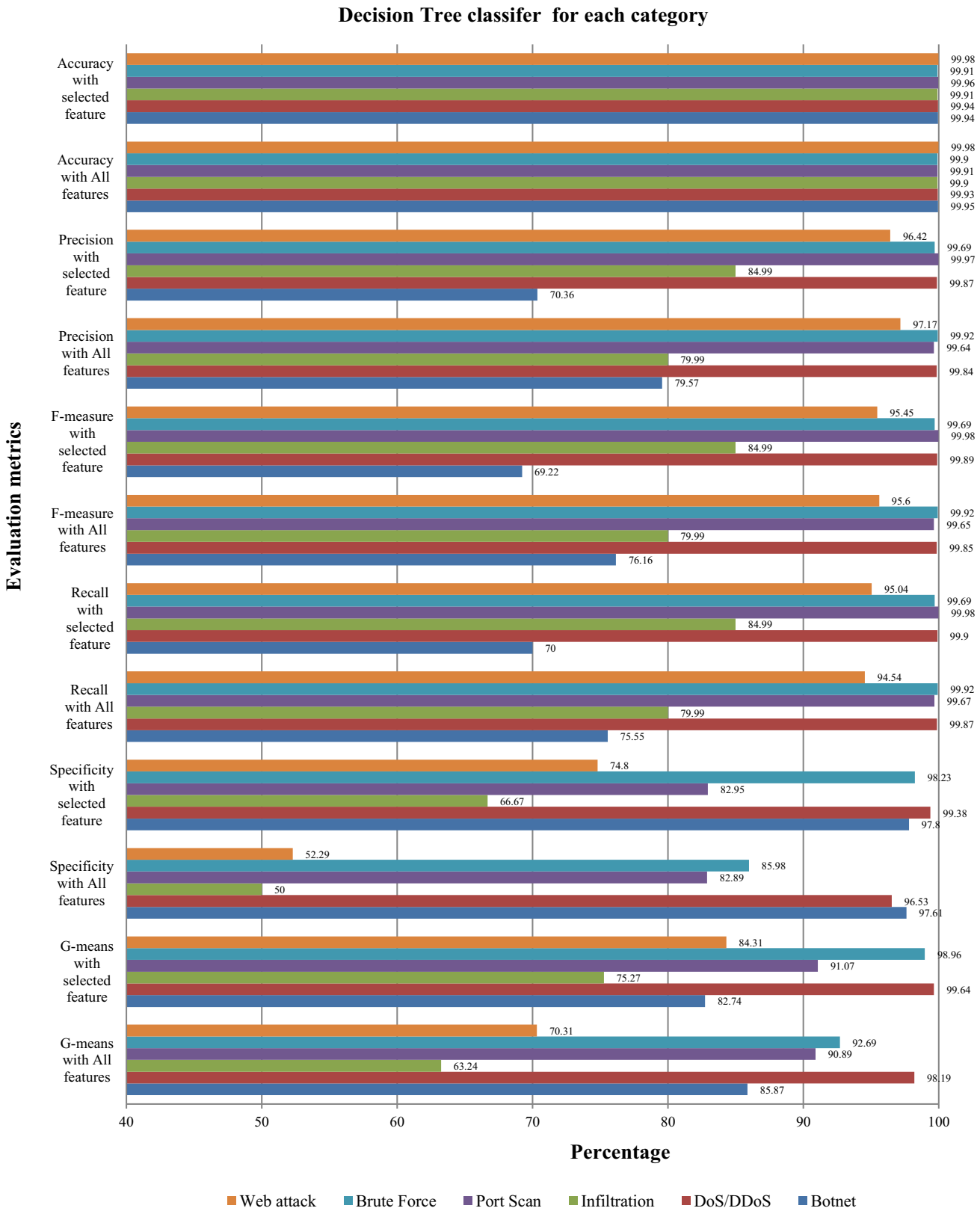

Fig. 8 Evaluation matrices of DT classifier with selected features (by RFE) and all features for each category on combined CICIDS2017 dataset

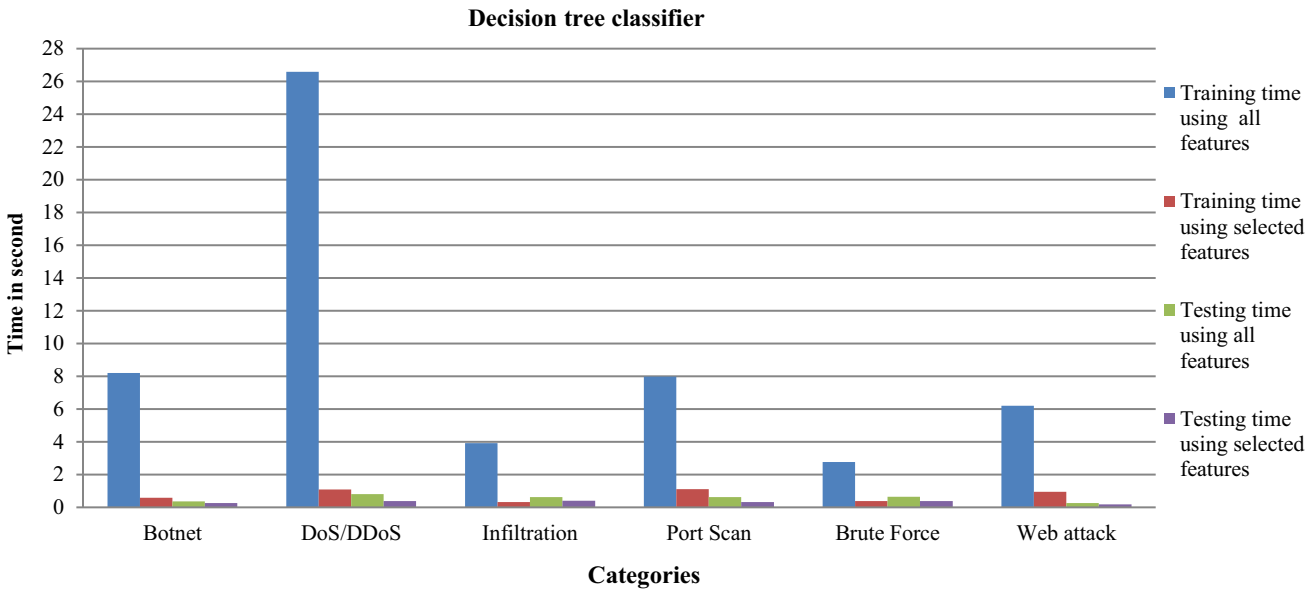

Fig. 9 DT classifier's training and testing time using selected and all features for each category on combined CICIDS2017 dataset

Table 14 Performance evaluations for DT classifier with selected features (by RFE) on CICIDS2017 combined dataset

Method	Attack type	Precision	*F*-measure	Specificity	Recall	*G*-means	Accuracy	Training time (T_1) (s)	Testing time (T_2) (s)
DT	Botnet	70.36	69.22	97.80	70.00	82.74	99.94	0.573	0.250
	DoS/DDos	99.87	99.89	99.38	99.90	99.64	99.94	1.092	0.369
	Infiltration	84.99	84.99	66.67	84.99	75.27	99.91	0.321	0.388
	Port Scan	99.97	99.98	82.95	99.98	91.07	99.96	1.114	0.325
	Brute Force	99.69	99.69	98.23	99.69	98.96	99.91	0.383	0.369
	Web attack	96.42	95.45	74.80	95.04	84.31	99.98	0.941	0.176
							Average (99.94)	Total T_1 (4.42)	Total T_2 (1.88)

features for each category in terms of accuracy, precision, *F*-measure, recall, specificity, and *G*-means. Moreover, Fig. 9 displays the training and testing time for the DT classifier utilizing selected and all features for each category.

After analysing the results (Tables 13, 14, Figs. 8, 9) of DT with selected features (by RFE) and DT with all features on the combined CICIDS2017 dataset, it is observed that the DT classifier with selected features (by RFE) yields higher results in terms of average accuracy, total training time, total testing time, accuracy (approximately identical accuracy for Botnet), *G*-means (with the exception of botnet attack type for DT with all features) and specificity. Additionally, the RFE with DT offered higher performance in terms of *F*-measure, recall, and precision for attack types including web attack, Port Scan, DoS/DDoS, and infiltration.

4.1.3 Comparison

In this study, multiple ML techniques have been utilized with various FSA to reduce the number of attributes in the datasets, which could help for the development of IDSs at a lower cost but with improved performances. The proposed model and various ML classifiers that use FSA for IDS models to detect different sorts of attacks have been thoroughly compared in Tables 15 and 16 using different datasets. Moreover, to compare the results of the proposed model with the others (eg. [45, 47, 56, 59] and [61]; the average accuracy, total training and testing time have been considered using NSL-KDD and CICIDS 2017 which is given in Tables 15 and 16, respectively. In KNN classifier-based model [62], only 4 classes (Brute Force, Cross-site scripting (XSS), SQL injection, BENIGN) of CICIDS 2107 have been considered

Table 15 A comparisons of ML algorithms for the IDS model utilizing FSA on NSL-KDD

References	Algorithms or classifiers for IDS model	Features selection approaches	Attacks types/dataset	Effectiveness						Efficiency	
				Accuracy (%)	Precision (%)	Recall/TPR/DR (%)	F1-measure (%)	Avg. Accuracy (%)	*G*-means (%)	Training time (S)	Testing time (S)
[45]	NB	Sequential search	DoS	99.3	*	*	*		*	4.73	*
			U2R	59.6	*	*	*		*		*
			R2L	95.0	*	*	*		*		*
			PROBE	97.4	*	*	*		*		*
[46]	Self-taught Learning (STL) based on Deep learning	Sparse auto-encoder	NSL_KDD	79.10	85.44	95.95	75.76	79.10	*	*	*
[47]	DT (Modified parallel version of the random forest)	Gain-ratio	KDDcup99	94.4	*	*	*	94.4	*	*	*
[48]	Optimized SVM	PCA	NSL_KDD	99	99.70	99.70	99.70	99	*	1171	
[49]	J48	Correlation-based	DOS	99.1	*	*	*	*	*	*	
			U2R	98.7	*	*	*		*	*	
			R2L	97.9	*	*	*		*	*	
			PROBE	98.9	*	*	*		*	*	
[50]	Meta-heuristic assessment model	Linear canonical correlation	NSL_KDD	90.4	98	98	98	90.4	*	*	*
[51]	SVM	PCA	Probe	99.47	*	98.23	*	91.86	*	*	*
			R2L	69.87	*	93.79	*		*	*	*

Table 15 (continued)

References	Algorithms or classifiers for IDS model	Features selection approaches	Attacks types/dataset	Effectiveness						Efficiency	
				Accuracy (%)	Precision (%)	Recall/TPR/DR (%)	F1-measure (%)	Avg. Accuracy (%)	*G*-means (%)	Training time (S)	Testing time (S)
			U2R	98.97	*	63.11	*		*	*	*
			DoS	99.13	*	99.33	*		*	*	*
[52]	Least square support vector machine	Feature grouping based on multi-variate mutual information	KDDcup99	95.65	*	94.74	*	95.65	*	*	*
			NSL_KDD	98.56	*	98.35	*	98.56	*	*	*
[53]	SVM	Hybrid particle swarm optimization	KDDcup99	90	*	*	*	90	*	*	*
[54]	SVM	Ant colony optimization	KDD99	98.29	*	98.00	*	98.29	*	*	*
[55]	Logistic classifier	Sparse auto-encoder	NSL_KDD	87.2	84.6	92.8	*	87.2	*	*	*
[56]	NB	Parallel binary bat algorithm	Probe	*	*	93.54	*	*	*	312	176
			R2L	*	*	47.41	*	*	*		
			U2R	*	*	74.13	*	*	*		
			DoS	*	*	97.53	*	*	*		
	Proposed model DT + RFE	**RFE**	**DOS**	**99.73**	**99.69**	**99.70**	**99.69**	**99.21**	**98.75**	**0.114**	**0.103**
			U2R	**99.65**	**87.74**	**89.18**	**87.49**		**81.78**	**0.212**	**0.115**
			R2L	**97.92**	**97.15**	**96.95**	**97.05**		**97.84**	**0.174**	**0.143**
			PROBE	**99.57**	**99.39**	**99.26**	**99.32**		**82.69**	**0.251**	**0.244**
										Total (0.751)	Total (0.605)

The results of proposed models are given in bold
Avg. average, *S* second, *not mention/not found

Table 16 A comparisons of ML algorithms for the IDS model utilizing FSA on CICIDS2107

References	Algorithms or classifiers for IDS model	features selection approaches	Attacks types/dataset	Effectiveness						Efficiency	
				Accuracy (%)	Precision (%)	Recall/TPR/ DR ()	F1-measure (%)	Avg. Accuracy (%)	*G*-means (%)	Training time (S)	Testing time (S)
[57]	Event profiling & Fully convolution neural network (CNN)	Term frequency-inverse document frequency & base points	CICIDS2017	99.5	*	98.2	98.7	99.5	*	*	*
[58]	Gaussian Mixture Models	Expectation maximization algorithm	DDoS	*	*	99.90	*	*	*	*	*
[59]	Recurrent neural networks and Blockchain-Based scheme	Back-propagation through time	Port Scan	*	*	98.99	*	98.23	*	102.2 (TPU), 155.9 (GPU), 1772.9 (CPU)	71.39 (TPU), 72.35 (GPU), 123.98 (CPU),
			Web attack	*	*	86.00	*		*		
			Infiltration	*	*	100	*		*		
			Botnet	*	*	97.58	*		*		
			DDoS	*	*	99.89	*		*		
			DoS	*	*	92.72	*		*		
			Brute force	*	*	82.37	*		*		
[60]	Negative selection algorithm and DT (classifier)	Correlation-based FSA	CICIDS2017	98	97	97	97	98	*	27.36	*
[61]	REPTree	J48	Infiltration	99.99	*	99.99	*	99.90	*	*	*
			Web attack	99.61	*	99.99	*		*	*	*
			DoS	99.84	*	99.88	*		*	*	*

Table 16 (continued)

References	Algorithms or classifiers for IDS model	features selection approaches	Attacks types/dataset	Effectiveness						Efficiency	
				Accuracy (%)	Precision (%)	Recall/TPR/ DR ()	F1-measure (%)	Avg. Accuracy (%)	*G*-means (%)	Training time (S)	Testing time (S)
			Brute Force	99.98	*	99.97	*		*	*	*
			DDoS	99.98	*	99.97	*		*	*	*
			Port Scan	98.98	*	99.97	*		*	*	*
			Botnet	99.96	*	99.98	*		*	*	*
[62]	KNN	Random Forest as FST	CICIDS 2017	99.46	99.44	99.45	99.41	99.46	*	2590.6	1358.10
[43]	KNN	Pre-processing	BENIGN	100	100	100	100	99.52	*	11.130	7.92
			Brute Force	73	72	73	73		*		
			XSS (Web attack)	44	41	43	42		*		
			SQL injection	29	25	22	24		*		
[63]	RF	IG	DDoS	86.80	99.63	86.29	*	86.80	*	*	*
[64]	Fuzzy entropy weighted natural nearest neighbour	Fisher score and deep graph feature learning	Port Scan	97	*	*	*	*	*	*	*
			DDoS	92.5	*	*	*		*	*	*
			Botnet	95	*	*	*		*	*	*
			Web attack-XSS	94	*	*	*		*	*	*
			Brute Force	93	*	*	*		*	*	*
[65]	CNN-based model	PCA	Brute Force-SSH	85.2	89.1	92.2	89.4	84.05	*	*	*
			Benign	82.9	88.3	90.1	9.21		*	*	*
	Proposed model DT + RFE	**RFE**	**Port Scan**	**99.96**	**99.97**	**99.98**	**99.98**	**99.94**	**91.07**	**0.573**	**0.250**
			DDoS/DoS	**99.94**	**99.87**	**99.90**	**99.89**		**99.64**	**1.092**	**0.369**
			Infiltration	**99.91**	**84.99**	**84.99**	**84.99**		**75.27**	**0.321**	**0.388**
			Botnet	**99.94**	**70.36**	**70.00**	**69.22**		**82.74**	**1.114**	**0.325**
			Brute Force	**99.91**	**99.69**	**99.69**	**99.69**		**98.96**	**0.383**	**0.369**
			Web attack	**99.98**	**96.42**	**95.04**	**95.45**		**84.31**	**0.941**	**0.176**
										Total (4.42)	Total (1.88)

The results of proposed models are given in bold
Avg. average, *S* second, *not mention/not found

and presented the total training time 11.130 s. However, in the proposed model, 6 classes which cover almost all types' attacks present in CICIDS 2107 dataset have been considered and the total training time of the proposed model is 4.42 s.

5 Conclusion and future work

This paper examines various classifiers with different FSA in order to construct an effective IDS model. Based on the analysis, it has been proved that the dimensional reduction of data in IDS not only decreases processing costs but also enhances the model performance. According to the results of the NSL-KDD dataset, RFE as FST with DT as classifier produces better results in terms of recall, precision (except for U2R attack category), accuracy, and F-measure than other classifiers with FSA. Moreover, the proposed FST identified a smaller, more appropriate subset of features based on information gain and ranking techniques for the classifier. As a consequence, it identified 13 significant features in the NSL-KDD dataset and 8 relevant features in the CICIDS 2017 dataset. It helps to increase the model performance with lower computation cost than to model with all features. The proposed model (RFE + DT) has been evaluated over the combined CICIDS2017 dataset in terms of F-measure, recall, specificity, precision, G-means, accuracy, testing time, and training time. In order to demonstrate the proposed model's efficiency and effectiveness, it has been compared to other well-known models that have been published in the literature. It has been found that using DT as a classification technique and RFE as a feature selection can reduce the computational cost and improves performance.

Future studies may focus on the application of various ML algorithms, such as unsupervised and supervised models, across various IDS-related datasets. The effectiveness of feature selection in selecting features for attack detections utilizing hybrid FST, which includes a number of statistical approaches and meta-heuristics, will also be the subject of future studies because it is a relatively unexplored area of study.

Author contributions All authors contributed to the study conception and design. Material preparation, data collection, and analysis were performed by GS, SB, and SS. The first draft of the manuscript was written by GS and SB, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding This research did not receive any specific funding, and it is carried out as part of the employment and higher degree of the authors.

Availability of data and material The data and material that support the findings of this study are available from the corresponding author, Subhasish Banerjee, upon reasonable request. This research work utilizes the CICIDS2017 and NSL-KDD datasets which are publicly available online.

Code availability The data and material that support the findings of this study are available from the corresponding author, Subhasish Banerjee, upon reasonable request.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent This article does not contain any studies with human participants.

Appendix

Appendix A: the classifiers parameters setting

The ML classifiers utilized in the experiment are displayed in Table 17 along with the parameter settings. In addition, based on the experimental results, a decent classifier with FST that perform well on the NSL-KDD dataset have been selected i.e. DT + RFE and have been used to assess the combined CICIDS 2017 dataset. Therefore, the DT classifier and associated parameter settings are provided only for CICIDS2017 dataset in Table 17.

Table 17 The ML classifiers and their parameters setting

ML classifiers	Parameters setting	Dataset
DT classifier	DecisionTreeClassifier(class_weight = None, criterion = 'gini', max_depth = None, max_features = None, max_leaf_nodes = None, min_impurity_decrease = 0.0, min_impurity_split = None,min_samples_leaf = 1, min_samples_split = 2,min_weight_fraction_leaf = 0.0, presort = False,random_state = 0, splitter = 'best')	NSL-KDD
KNN classifier	KNeighborsClassifier(algorithm = 'auto', leaf_size = 30, metric = 'minkowski',metric_params = None, n_jobs = None, n_neighbors = 5, p = 2,weights = 'uniform')	
NB classifier	GaussianNB(priors = None, var_smoothing = 1e-09)	
DT classifier	DecisionTreeClassifier(ccp_alpha = 0.0, class_weight = None, criterion = 'gini', max_depth = None, max_features = None, max_leaf_nodes = None, min_impurity_decrease = 0.0, min_impurity_split = None, min_samples_leaf = 1, min_samples_split = 2,min_weight_fraction_leaf = 0.0, presort = 'deprecated', random_state = 0, splitter = 'best')	CICIDS2017

Appendix B: the analysis of datasets generated at each phase of the experiment

The datasets (NSL-KDD and CICIDS2017) generated by each phase are shown in Table 18. Further, Table 18 shows that the number of objects in the standard dataset and the reduction dataset is equal because after applying FSA, the number of attributes (column) in standard dataset is reduced. Consequently, the number of rows will remain the same (objects).

Table 18 The explanation of CICIDS2017 and NSL-KDD datasets generated at each phase

Dataset	Number of objects in (training_set, testing_set)	Label-name (objects in training_set, testing_set)	Dimensions	Explanation
Original dataset (NSL-KDD)	(125,973, 22,544)	Normal(67,343, 9711) Probe (11,656, 2421) R2L (995, 2885) U2R (52, 67) DoS (45,927,7460)	41	Without processes
Original dataset (CICIDS2017)	(523,456, 130,865)	Normal (420,432, 104,934) Botnet (359, 89) Brute Force (2521, 638) Infiltration (7, 1) Port-Scan (29,290, 7406) WebAttack (407, 87) DoS/ DDoS (70,440, 17,710)	79	Without processes
Standard dataset (NSL-KDD)	(328,002, 51,677)	Including normal objects in every class Probe:(78,999, 12,132) DoS: (113,270, 17,171) U2R: (67,395, 9778) R2L: (68,338, 12,596)	122	Result of pre-processing steps
Standard dataset (CICIDS2017)	(523,197, 130,806)	Including normal objects in every class Botnet (420,711, 105,007) Brute Force (422,873, 105,556) Infiltration (420,359, 104,919) Port-Scan (449,642, 112,324) Web Attack (420,352, 105,005) DoS/ DDoS (490,613, 122,585)	77	Result of pre-processing steps

Table 18 (continued)

Dataset	Number of objects in (training_set, testing_set)	Label-name (objects in training_set, testing_set)	Dimensions	Explanation
Reduction dataset (NSL-KDD)	(328,002, 51,677)	Probe:(78,999, 12,132) DoS: (113,270, 17,171) U2R: (67,395, 9778) R2L: (68,338, 12,596)	13	After FSA(RFE)
			8	After FSA (PCA)
Reduction dataset (CICIDS2017)	(523,197, 130,806)	Botnet (420,711, 105,007) Brute Force (422,873, 105,556) Infiltration (420,359, 104,919) Port-Scan (449,642, 112,324) Web Attack (420,352, 105,005) DoS/ DDoS (490,613, 122,585)	8	After FSA (RFE)

References

1. Larson, D.: Distributed denial of service attacks–holding back the flood. Netw. Secur. **2016**(3), 5–7 (2016)
2. Almseidin, M., Alzubi, M., Kovacs, S., Alkasassbeh, M.: Evaluation of machine learning algorithms for intrusion detection system. In: 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), pp. 000277–000282. IEEE (2017)
3. Kok, S.H., Abdullah, A., Jhanjhi, N.Z., Supramaniam, M.: A review of intrusion detection system using machine learning approach. Int. J. Eng. Res. Technol. **12**(1), 8–15 (2019)
4. Al-Jarrah, O.Y., Siddiqui, A., Elsalamouny, M., Yoo, P.D., Muhaidat, S., Kim, K.: Machine-learning-based feature selection techniques for large-scale network intrusion detection. In: 2014 IEEE 34th International Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 177–181. IEEE (2014)
5. Thanh, H.N., Van Lang, T.: An approach to reduce data dimension in building effective network intrusion detection systems. EAI Endorsed Trans. Context Aware Syst. Appl. **6**(18), 162633 (2019)
6. Chomboon, K., Chujai, P., Teerarassamee, P., Kerdprasop, K., Kerdprasop, N.: An empirical study of distance metrics for k-nearest neighbor algorithm. In: Proceedings of the 3rd International Conference on Industrial Application Engineering, pp. 280–285 (2015)
7. Wu, S.X., Banzhaf, W.: The use of computational intelligence in intrusion detection systems: a review. Appl. Soft Comput. **10**(1), 1–35 (2010)
8. Mukkamala, S., Sung, A.H.: Feature selection for intrusion detection with neural networks and support vector machines. Transp. Res. Rec. **1822**(1), 33–39 (2003)
9. Fleuret, F.: Fast binary feature selection with conditional mutual information. J. Mach. Learn. Res. **5**(9), 1531–1555 (2004)
10. Chebrolu, S., Abraham, A., Thomas, J.P.: Feature deduction and ensemble design of intrusion detection systems. Comput. Secur. **24**(4), 295–307 (2005)
11. Chou, T.-S., Yen, K.K., Luo, J.: Network intrusion detection design using feature selection of soft computing paradigms. Int. J. Comput. Intell. **4**(3), 196–208 (2008)
12. Heba, F.E., Darwish, A., Hassanien, A.E., Abraham, A.: Principle components analysis and support vector machine based intrusion detection system. In: 2010 10th International Conference on Intelligent Systems Design and Applications, pp. 363–367. IEEE (2010)
13. Zainal, A., Maarof, M.A., Shamsuddin, S.M.: Ensemble classifiers for network intrusion detection system. J. Inf. Assur. Secur. **4**(3), 217–225 (2009)
14. Revathi, S., Malathi, A.: A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. Int. J. Eng. Res. Technol. (IJERT) **2**(12), 1848–1853 (2013)
15. Kim, G., Lee, S., Kim, S.: A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Syst. Appl. **41**(4), 1690–1700 (2014)
16. Kocher, G., Kumar, G.: Machine learning and deep learning methods for intrusion detection systems: recent developments and challenges. Soft. Comput. **25**(15), 9731–9763 (2021)
17. Jo, S., Sung, H., Ahn, B.: A comparative study on the performance of intrusion detection using decision tree and artificial neural network models. J. Korea Soc. Digit. Ind. Inf. Manag. **11**(4), 33–45 (2015)
18. Jebur, S.A., Nasereddin, H.O.: Enhanced solutions for misuse network intrusion detection system using sga and ssga. IJCSNS Int. J. Comput. Sci. Netw. Secur. **15**(5), 12–18 (2015)
19. Mishra, P., Pilli, E.S., Varadharajan, V., Tupakula, U.: PSI-NetVisor: program semantic aware intrusion detection at network and hypervisor layer in cloud. J. Intell. Fuzzy Syst. **32**(4), 2909–2921 (2017)
20. Mousavi, S.M., Majidnezhad, V., Naghipour, A.: A new intelligent intrusion detector based on ensemble of decision trees. J. Ambient Intell. Humaniz. Comput. (2019). https://doi.org/10.1007/s12652-019-01596-5
21. Sah, G., Banerjee, S.: Feature reduction and classifications techniques for intrusion detection system. In: 2020 International Conference on Communication and Signal Processing (ICCSP), pp. 1543–1547. IEEE (2020)
22. Thakkar, A., Lohiya, R.: Attack classification using feature selection techniques: a comparative study. J. Ambient. Intell. Humaniz. Comput. **12**(1), 1249–1266 (2021). https://doi.org/10.1007/s12652-020-02167-9
23. Gu, J., Shan, Lu.: An effective intrusion detection approach using SVM with naïve Bayes feature embedding. Comput. Secur. **103**, 102158 (2021)
24. https://www.unb.ca/cic/datasets/nsl.html
25. Intrusion Detection Evaluation Dataset (CICIDS2017) (2017). https://www.unb.ca/cic/datasets/ids-2017.html

26. Engelen, G., Rimmer, V., Joosen, W.: Troubleshooting an intrusion detection dataset: the CICIDS2017 case study. In: 2021 IEEE Security and Privacy Workshops (SPW), pp. 7–12. IEEE (2021)
27. Panigrahi, R., Borah, S.: A detailed analysis of CICIDS2017 dataset for designing intrusion detection systems. Int. J. Eng. Technol. **7**(3.24), 479–482 (2018)
28. Wang, S., Minku, L.L., Yao, X.: A systematic study of online class imbalance learning with concept drift. IEEE Trans. Neural Netw. Learn. Syst. **29**(10), 4802–4821 (2018). https://doi.org/10.1109/TNNLS.2017.2771290
29. Moustafa, N., Jiankun, Hu., Slay, J.: A holistic review of network anomaly detection systems: a comprehensive survey. J. Netw. Comput. Appl. **128**, 33–55 (2019)
30. Moustafa, N., Slay, J.: The evaluation of network anomaly detection systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Inf. Secur. J. Glob. Perspect. **25**(1–3), 18–31 (2016)
31. Xanthopoulos, P., Pardalos, P.M., Trafalis, T.B. Principal component analysis. In: Robust Data Mining, pp. 21–26. Springer, New York, NY (2013)
32. Saeys, Y., Abeel, T., Van de PeerY.: "Robust feature selection using ensemble feature selection techniques. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 313–325. Springer, Berlin (2008)
33. Doan, D.M., Jeong, D.H., Ji, S.-Y.: Designing a feature selection technique for analyzing mixed data. In: 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0046–0052. IEEE (2020)
34. Powell, A., Bates, D., Van Wyk, C., de Abreu, D.: A cross-comparison of feature selection algorithms on multiple cyber security data-sets. In: FAIR, pp. 196–207 (2019)
35. Chen, X., Jeong, J.C.: Enhanced recursive feature elimination. In: Sixth International Conference on Machine Learning and Applications (ICMLA 2007), pp. 429–435. IEEE (2007)
36. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Mach. Learn. **46**(1), 389–422 (2002)
37. Safavian, S.R., Landgrebe, D.: A survey of decision tree classifier methodology. IEEE Trans. Syst. Man Cybern. **21**(3), 660–674 (1991)
38. Alwateer, M., Almars, A.M., Areed, K.N., Elhosseini, M.A., Haikal, A.Y., Badawy, M.: Ambient healthcare approach with hybrid whale optimization algorithm and Naïve Bayes classifier. Sensors **21**(13), 4579 (2021)
39. Sen, P.C., Hajra, M., Ghosh, M.: Supervised classification algorithms in machine learning: a survey and review. In: Emerging Technology in Modelling and Graphics, pp. 99–111. Springer, Singapore (2020)
40. Chung, Y.Y., Wahid, N.: A hybrid network intrusion detection system using simplified swarm optimization (SSO). Appl. Soft. Comput. **12**(9), 3014–3022 (2012)
41. Espíndola, R.P., Ebecken, N.F.F.: On extending f-measure and g-mean metrics to multi-class problems. WIT Trans. Inf. Commun. Technol. **35** (2005)
42. Sah, G., Goswami, R.S., Nandi, S.K.: Machine learning methods for predicting the popularity of forthcoming objects. Int. J. Innov. Technol. Explor. Eng. (IJITEE) **9**(2S), 645–652 (2019)
43. Maseer, Z.K., Yusof, R., Bahaman, N., Mostafa, S.A., Foozy, C.F.M.: Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. IEEE Access **9**, 22351–22370 (2021)
44. Scikit-Learn (2010). http://scikit-learn.org/stable/index.html. Accessed January 2020
45. Zhang, F., Wang, D.: An effective feature selection approach for network intrusion detection. In: 2013 IEEE Eighth International Conference on Networking, Architecture and Storage, pp. 307–311. IEEE (2013)
46. Javaid, A., Niyaz, Q., Sun, W., Alam, M.: A deep learning approach for network intrusion detection system. 2016 Eai Endorsed Trans. Secur. Saf. **3**(9), 21–26 (2015)
47. Masarat, S., Sharifian, S., Taheri, H.: Modified parallel random forest for intrusion detection systems. J. Supercomput. **72**(6), 2235–2258 (2016)
48. Ikram, S.T., Cherukuri, A.K.: Improving accuracy of intrusion detection model using PCA and optimized SVM. J. Comput. Inf. Technol. **24**(2), 133–148 (2016)
49. Dhanabal, L., Shantharajah, S.P.: A study on NSL-KDD dataset for intrusion detection system based on classification algorithms. Int. J. Adv. Res. Comput. Commun. Eng. **4**(6), 446–452 (2015)
50. Jyothsna, V., Rama Prasad, V.V.: FCAAIS: anomaly based network intrusion detection through feature correlation analysis and association impact scale. ICT Express **2**(3), 103–116 (2016)
51. Subba, B., Biswas, S., Karmakar, S.: Enhancing performance of anomaly based intrusion detection systems through dimensionality reduction using principal component analysis. In: 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1–6. IEEE (2016)
52. Mohammadi, S., Mirvaziri, H., Ghazizadeh-Ahsaee, M.: Multivariate correlation coefficient and mutual information-based feature selection in intrusion detection. Inf. Secur. J. Glob. Perspect. **26**(5), 229–239 (2017)
53. Chahar, V., Chhikara, R., Gigras, Y., Singh, L.: Significance of hybrid feature selection technique for intrusion detection systems. Indian J. Sci. Technol. **9**(48), 1–7 (2017)
54. Mehmod, T., Md Rais, H.B.: Ant colony optimization and feature selection for intrusion detection. In: Advances in machine learning and signal processing, pp. 305–312. Springer, Cham (2016)
55. Gurung, S., Ghose, M.K., Subedi, A.: Deep learning approach on network intrusion detection system using NSL-KDD dataset. Int. J. Comput. Netw. Inf. Secur. **11**(3), 8–14 (2019)
56. Natesan, P., Rajalaxmi, R.R., Gowrison, G., Balasubramanie, P.: Hadoop based parallel binary bat algorithm for network intrusion detection. Int. J. Parallel Prog. **45**(5), 1194–1213 (2017)
57. Lee, J., Kim, J., Kim, I., Han, K.: Cyber threat detection based on artificial neural networks using event profiles. IEEE Access **7**, 165607–165626 (2019)
58. Cepheli, Ö., Büyükçorak, S., Kurt, G.K.: Hybrid intrusion detection system for ddos attacks. J. Electr. Comput. Eng. **2016**, 1–8 (2016)
59. Ferrag, M.A., Maglaras, L.: DeepCoin: a novel deep learning and blockchain-based energy exchange framework for smart grids. IEEE Trans. Eng. Manag. **67**(4), 1285–1297 (2019)
60. Hosseini, S., Seilani, H.: Anomaly process detection using negative selection algorithm and classification techniques. Evol. Syst. **12**(3), 769–778 (2021)
61. Singh Panwar, S., Raiwani, Y.P., Singh Panwar, L.: "Evaluation of network intrusion detection with features selection and machine learning algorithms on CICIDS-2017 dataset. In: International Conference on Advances in Engineering Science Management & Technology (ICAESMT)-2019, Uttaranchal University, Dehradun, India (2019)
62. Alrowaily, M., Alenezi, F., Lu, Z.: Effectiveness of machine learning based intrusion detection systems. In: International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage, pp. 277–288. Springer, Cham (2019)

63. Abdulrahman, A.A., Ibrahem, M.K.: Evaluation of DDoS attacks detection in a CICIDS2017 dataset based on classification algorithms. Iraqi J. Inf. Commun. Technol. (IJICT) **1**(3), 49–55 (2018)
64. Chen, L., Gao, S., Liu, B., Zhigang, Lu., Jiang, Z.: FEW-NNN: a fuzzy entropy weighted natural nearest neighbor method for flow-based network traffic attack detection. China Commun. **17**(5), 151–167 (2020)
65. Wanjau, S.K., Wambugu, G.M., Kamau, G.N.: SSH-brute force attack detection model based on deep learning (2021)