mathematics

MDPI

Article

Performance Analysis of Feature Subset Selection Techniques for Intrusion Detection

Yousef Almaghthawi, Iftikhar Ahmad * and Fawaz E. Alsaadi

Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia
* Correspondence: iakhan@kau.edu.sa

Abstract: An intrusion detection system is one of the main defense lines used to provide security to data, information, and computer networks. The problems of this security system are the increased processing time, high false alarm rate, and low detection rate that occur due to the large amount of data containing various irrelevant and redundant features. Therefore, feature selection can solve this problem by reducing the number of features. Choosing appropriate feature selection methods that can reduce the number of features without a negative effect on the classification accuracy is a major challenge. This challenge motivated us to investigate the application of different wrapper feature selection techniques in intrusion detection. The performance of the selected techniques, such as the genetic algorithm (GA), sequential forward selection (SFS), and sequential backward selection (SBS), were analyzed, addressed, and compared to the existing techniques. The efficiency of the three feature selection techniques with two classification methods, including support vector machine (SVM) and multi perceptron (MLP), was compared. The CICIDS2017, CSE-CIC-IDS218, and NSL-KDD datasets were considered for the experiments. The efficiency of the proposed models was proved in the experimental results, which indicated that it had highest accuracy in the selected datasets.

Keywords: intrusion detection; genetic algorithm; greedy search; backward elimination learning; NSL-KDD; CIC-IDS-2017; CIC-IDS2018

MSC: 68M25

Citation: Almaghthawi, Y.; Ahmad, I.; Alsaadi, F.E. Performance Analysis of Feature Subset Selection Techniques for Intrusion Detection. *Mathematics* **2022**, *10*, 4745. https://doi.org/10.3390/math10244745

Academic Editor: Wei Fang

Received: 30 October 2022
Accepted: 6 December 2022
Published: 14 December 2022

1. Introduction

Currently, the internet is necessary for storing and transferring the diverse information of users, companies, and governments. Protecting and securing systems and information is necessary. One of the most efficient existing systems used to secure systems and control intrusion activities is the intrusion detection system (IDS). In recent years, several IDSs have been proposed. These security systems have many problems such as an increasing processing time, high false positive rate (FPR), and low detection rate (DR), which are caused by the large amount of data containing various irrelevant and redundant features [1,2]. Feature selection (FS) can solve these problems by reducing the number of features and selecting only useful features. Several feature selection methods are available, but the task of finding which one is suitable for IDS that provides the minimum number of features with the maximum accuracy is a major challenge [2,3]. This challenge motivated us to investigate the application of different FS techniques in intrusion detection. The performance of the selected techniques, such as the GA [4], SFS [5], and SBS [3] were analyzed, addressed, and compared with existing techniques. This study used the recent CICIDS2017 and CSE-CIC-IDS218 [6] datasets as well as the NSL-KDD [7] dataset.

1.1. Intrusion Detection System

IDS is a software or hardware that monitors activities inside and outside the network to detect abnormal ones [3]. It is one of the most important mechanisms that protect

systems and networks against malicious activities [6,8]. This system generates alarms if any abnormal pattern is recognized. IDSs have two types: host-based IDS, which focuses on individual computers, and network-based IDS, which focuses on the traffic between computers. Based on the detection method, IDS can be classified into two categories, namely signature-based and anomaly-based detection [1,9].

1.1.1. Signature-Based Detection

This detection method is also known as misuse detection. It uses the attack signature database to identify intrusions or abnormal activities. When a packet signature matches with a signature in the database, IDSs detect that packet as an intrusion. This detection type can detect known attacks only [1].

1.1.2. Anomaly-Based Detection

In this detection type, IDSs build profiles of the normal network packets. They then analyze and monitor the network packets. When there is any deviation from the normal profile, IDSs detect the abnormal activity. This detection type can detect known and unknown attacks [1].

1.2. Feature Selection Methods

Feature selection methods select the relevant and useful features from a dataset. The goal of FS techniques is to increase the accuracy and detection rate and decrease the execution time and false positive rate. To achieve this goal, FS measures the importance of features and only allows the most important features to enter the classification. The number of features is then reduced as well as the classification time. There are two types of FS methods based on evaluation criteria [8,10]. The first type is the filter method, which it is independent of the classifier. It analyzes each single feature and decides which features are useful and should be trained based on statistical measures [8,11,12]. The second type is the wrapper method, which is dependent on the classifier. In contrast to filter methods, wrapper methods use machine learning algorithms to determine the best feature subset to provide a high classification performance [8,10–12]. The filter and wrapper FS methods are based on two components: a search strategy (e.g., GA, SFS, and SBS) and an objective function or fitness function (e.g., classifier performance in wrapper methods and statistical measure in filter methods). The search strategy determines the optimal subset of features that provide high accuracy and low false alarms results based on the classifier's performance (in wrapper methods) or statistical measures such as information gain (in filter methods) [13]. The search strategy can be made up of exhaustive, heuristic, and random searches [14]. Exhaustive searches are time consuming and impractical because of the large number of combinations to evaluate. For example, a search of an 2^{n-1} possible feature subsets in n features dataset becomes an NP-hard problem as the number of features grow [13]. Heuristic searches such as the forward sequential search and fuzzy systems and random searches such as the GA perform better in large datasets [15–17].

Filter selection methods are faster and have a lower level of complexity than wrapper methods, but the latter are more accurate [18]. Therefore, we used wrapper methods in our study to build an efficient IDS model that provides a low FPR and high ACC with the minimum number of features.

Several features selection methods are available such as the GA, SBS, and SFS, but the problem is to find which one is more suitable for an intrusion detection system (IDS) that provides a minimum number of features with maximum accuracy [2,3].

1.2.1. Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS)

SFS and SBS are simple search techniques that run in iterations and make a greedy decision to select the best local solution in each iteration based on the objective function.

The SFS algorithm starts with an empty set and adds one feature to the subset at each iteration until a stopping criterion is met such as the search is completed or the desired

number of subset features is reached. After that, it returns the local optimal solution among iterations or the solution in the desired number of subset features. Figure 1a shows an SFS methodology in which a dataset has three features: F1, F2, and F3. In the first iteration, SFS generates and evaluates several subsets that contain only one feature from the complete set of features. The feature that has the maximum objective function is selected as a local optimal solution in this iteration. In the second iteration, it also generates and evaluates several subsets, in which one feature is added to the selected feature from the previous iteration. Subsequently, each subset is evaluated to select the best local optimal solution. In the third iteration, a feature is added to the selected features from the previous iteration; in this case, the complete set of features is evaluated. Finally, it returns the best local optimal solution among these iterations as the output. By contrast, the SBS algorithm starts from a complete set of features and iteratively removes one feature until a stopping criterion is met. After that, it returns the local optimal solution among iterations or the solution in the desired number of subset features. Figure 1b shows an SBS methodology in which a dataset has three features: F1, F2 and F3.

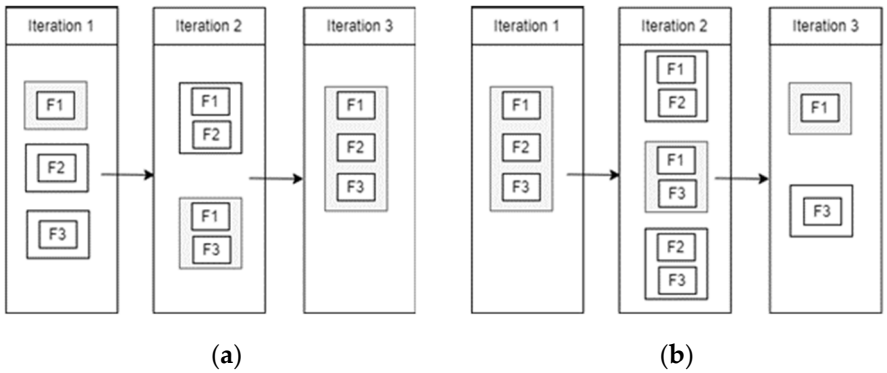

Figure 1. (**a**) An example of SFS; (**b**) an example of SBS.

1.2.2. Genetic Algorithm

The GA is a metaheuristic search and work algorithm based on a direct Darwinian natural selection analogy and genetics in biological systems [12,13]. It is composed of four components: a population of individuals, a fitness function, a selection function, and a genetic operator (e.g., crossover and mutations). The GA randomly generates a population of individuals or chromosomes in which each chromosome represents a solution to the problem [12,13]. The fitness function determines the chromosome's chance of being chosen to create the offspring or the next generation individuals. The selection function selects the parents of the offspring from the current generation, and then the crossover and mutation are applied to the selected parents to generate the offspring or the next generation of individuals. Several selection functions are available, such as the tournament selection method, roulette wheel, and rank selection [13]. Crossover and mutation operators are then applied to the selected individuals or chromosomes to create offspring or the generation of new individuals. Crossover is the exchange of individual bits between two randomly selected parents. Mutation is the alteration of individual bits to generate new individuals. New and different individuals are generated from the current generation after applying crossover and mutation operators. The evaluation of individuals, selection, crossover, and mutation are repeated in a predefined number of generations until a stopping criterion is met.

2. Related Work

Sarvari et al. [19] used the wrapper cuckoo search algorithm (CSA) as an FS technique to build an efficient IDS model. They applied the FS method and trained an artificial neural network (ANN) using a multiverse optimizer (MVO). The proposed model, called MCF &

MVO-ANN, was evaluated using the NSL-KDD dataset. As a result, their model achieved a high ACC (98.16%), a high DR (96.83%), and a low FPR (0.03%).

Saleh et al. [8] presented an IDS approach based on the wrapper NBFS technique to select the best feature subsets. The proposed model combines optimized support vector machines (OSVMs) and prioritized k-nearest neighbors (PKNN) techniques. They used OSVM for rejecting outliers and PKNN to detect attacks. The experimental results indicate that NBFS achieves a higher DR (approximately 90.28%) than other techniques using an NB classifier with 18 selected features. The proposed model also gives a higher DR (95.77% using the NSL-KDD dataset and 93.28% using KDD Cup99) than other techniques.

Ates et al. [5] used the greedy search (GS) algorithm and SVM to detect distributed denial of service (DDoS) attacks. They used GS for FS and SVM as a classifier. The proposed model calculates the distances between the probability distributions of header information using the GS algorithm and Kullback–Leibler divergence. In the testing phase, they used a dataset collected from the MIT Darpa 2000 dataset and a university network and achieved a high AC of 99.99% and a low FPR of 0.001% for the MIT Darpa 200 dataset. However, the proposed model needs to be evaluated using a standard dataset to compare its performance results with recent models.

Asdaghi et al. [20] proposed a new backward elimination (BE) method called Smart-BT for web spam detection. They used index of balance accuracy values as a performance metric. The proposed model uses the chi-square as a pre-processing process. Afterward, Smart-BT selects the relevant and useful features by eliminating a set of features from the initial set. The experimental results show that the Smart-BT gives better classification results than other existing FS techniques such as the ranker search algorithm and particle swarm optimization (PSO) techniques.

Tao et al. [4] introduced an FS technique based on the GA and SVM to improve the IDS. They presented a new fitness function for the chosen FS method to determine the optimal feature subset. The experimental results indicate that their model succeeds in minimizing the number of features to 19 features and achieves a high DR and low FPR using the KDD Cup99 dataset. However, the determination of the weight values for the TPR and selected feature number is carried out manually.

Thakkar and Lohiya [3] presented a performance analysis of FS techniques in IDSs. Chi-square, IG, and recursive feature elimination (REF) were implemented separately as FSs with several ML classifiers such as SVM and ANN. For determining the best combination performance in terms of FS technique and machine learning classifier using the NSL-KDD dataset, they conducted several experiments and reported that the combination of SVM and REF performs well compared with other techniques. The average ACC, precision, recall, and F-score rates were 98.95%, 99.2%, 99.75%, and 98.40%, respectively, for the SVM-REF model.

G Suseendran and T. Nathiya [10] presented a GS FS method to increase the accuracy and decrease the false alarm rate in IDSs. They used correlation feature selection (CFS) to evaluate the selected feature subsets. The RF classifier is used in their model. The experiments showed that their model achieves 98.32% in terms of ACC and 0.40% in terms of the false alarm rate using the NSL-KDD dataset.

Aslahi et al. [21] used the GA for FS and SVM as a classifier for building a new IDS model. Their model reduces the amount of features to 10 features from the original 41 features. The KDD dataset was used in their experiments. Their model achieves 97.30% in terms of accuracy and 1.70% in terms of false alarm rate.

J. Lee et al. [22] proposed a sequential forward floating search (SFFS) method to increase the ACC and decrease the false alarm rate in IDSs. They used a SFFS to select the optimal feature set and RF classifier for the evaluation. The NSL-KDD dataset was used in the experiments for the proposed model. The experiments proved that their model achieves a 0.40% false alarm rate and 99.89% ACC with 10 selected features.

Li et al. [15] proposed a modified random mutation hill climbing (MRMHC) method as a wrapper FS technique. They used a modified linear SVM as an evaluation criterion.

MRMHC generates an initial subset from the complete set of features. Then, the MLSVM evaluates the selected subset and select the best subset. Their experiments on the KDD Cup dataset showed that their approach has a high ACC and low detection time.

Li Y et al. [23] used a gradual feature removal method to select the important features in IDSs. This method deletes the less important features gradually. They used SVM classifier as an objective function in their proposed model. Their model selected 19 features from the KDD Cup dataset, which was used in the experiments. The results obtained an ACC of 98.62%.

Raman et al. [24] proposed a hypergraph-based GA (HG-GA) to build an adaptive IDS with a high ACC and low false alarm rate. HG-GA is used for FS in their proposed model, and SVM is used for classification. In their experiments, they used the NSL-KDD dataset. The proposed model achieves an ACC of 97.14% and a false alarm rate of 0.83%.

Khammassi and Krichen [13] used the GA for FS and linear regression as a classification algorithm. This wrapper approach selects the important features from the original datasets (KDD99 and UNSW-NB15). Then, the DT classifier uses the selected features to measure the efficiency of the selected features. An accuracy of 99.90% and false alarm rate of 0.11% were obtained in their experiments for the KDD99 dataset with 18 features.

Zhou and Cheng [25] developed an IDS model based on correlation-based FS (CSF) and the bat algorithm (BA). They combined RF, C4.5, and forest attributes to build an ensemble approach. The proposed model was evaluated using several IDS datasets such as the CICIDS2017 and NSL-KDD datasets. Their model obtains 94.04% in terms of the detection rate, 2.38% in terms of the FPR and 96.76% in terms of ACC.

Hua Y. [26] developed a hybrid filter and wrapper FS method based on IG and LightGBM. An under-sampling technique was used to balance the used CICIDS2017 dataset. The proposed model achieves an ACC of 98.37%, a precision of 98.17%, and a recall of 98.37% with 10 selected features.

Sugandh Seth et al. [27] used random forest (RF) and principal component analysis (PCA) as a hybrid feature selection method. A light gradient boosting machine (LightGBM) classifier was used to classify the instances of the CIC-IDS-2018 dataset. The authors used an under-sampling technique to balance the dataset. Their model achieves 97.73% in terms of accuracy and a 97.57% F1-Score with 24 selected features.

Alazzam et al. [28] used the pigeon inspired optimizer (PIO) technique. They designed two models of the POI such as the sigmoid PIO and binary cosine PIO to determine the optimal number of features. In the binary PIO, the calculation of the velocity of pigeons is based on the cosine similarity. Their models were evaluated using the decision tree (DT) technique. Through experiments, the models were evaluated using the UNSW-NB15, NSL-KDD, and KDDCUP99 datasets. The result indicate that the proposed model (cosine PIO) outperforms several proposed FS algorithms in terms of AC, F-score, FPR, and TPR.

Mazini et al. [29] used the wrapper artificial bee colony (ABC) algorithm for FS to build an efficient anomaly IDS. The AdaBoost classifier is used for evaluation and classification. Through simulation, the model was evaluated using the ISCXIDS2012 and NSL-KDD datasets. As a result, the proposed model achieves a high DR (99.61%), low FPR (0.01%), and high ACC (98.90%). However, the parameter settings for FS are determined manually.

Aween Saeed and Noor Jameel [30] used the particle swarm optimization algorithm with a decision tree (DT) to build a wrapper feature selection model for IDS. The authors trained and tested their model using a DT classifier. Their model selected 19 features from the CIC-IDS-2018 dataset, which was used in the experiments. The results obtained an ACC of 99.52%.

Jahed Shaikh and Deepak Kshirsagar [31] used information gain (IG) and a correlation attribute evaluation as a feature selection method to select the optimal number of features from the CIS-IDS-2017 dataset. A PART rule-based machine learning classifier was used to evaluate the proposed model. An accuracy of 99.98% and false alarm rate of 1.35% were obtained in their experiments with 56 features.

Patil and Kshirsagar [32] presented an IDS model based on information gain (IG) and ranker method as a feature selection method. The J48 classifier was used to evaluate the selected features. Their model selected 75 features from the CIC-IDS-2017 dataset, which was used in the experiments. The results obtained an ACC of 87.44%.

Many of the above FS techniques have not been evaluated on high-dimension datasets such as CICI-IDS-2017 and CSE-CIC-IDS-2018, which contain more features (80 features) compared with KDD Cup99 and NSL-KDD, which contain 41 features [17]. In addition, certain researchers did not mention the obtained number of selected features, which affects the execution time in terms of classification [2]. Moreover, BE, GA, and GS techniques achieved high results in previous works [3–5,31]. Hence, the present study investigated these FS techniques using several standard datasets to determine the most appropriate technique that provides the minimum number of relevant features with maximum accuracy. A summary of the literature review is shown in Table 1.

Table 1. Summary of the literature review.

Ref.	Feature Selection Algorithm	Classification Algorithm	Dataset	Number of Features	Result (%)
[3]	REF	SVM	NSL-KDD	-	ACC: 98.95 F-score: 99.75
[4]	GA	SVM	KDD Cup99	19	
[5]	Greedy Search	SVM	MIT Darpa 2000	-	ACC: 99.99 FPR: 0.001
[8]	NBFS	PKNN + OSVMs	NSL-KDD	-	DR: 95.77
[10]	Greedy Search + CFS	RF	NSL-KDD	-	ACC: 98.32 FPR: 0.40
[13]	GA	DT	KDD99	18	ACC: 99.90 FPR: 0.11
[15]	MRMHC	MLSVM	KDD Cup	4	TPR: 80.00 FPR: 3.65
[19]	CSA	MCF & MVO-ANN	NSL-KDD	22	ACC: 98.81 DR: 97.25 FPR: 0.03
[21]	GA	SVM	KDD Cup	10	ACC: 97.30 FPR: 1.70
[22]	SFFS	RF	NSL-KDD	10	ACC: 99.89 FPR: 0.40
[23]	Gradual feature removal	SVM	KDD Cup	19	ACC: 98.62
[24]	HG-GA	SVM	NSL-KDD	-	ACC: 97.14 FPR: 0.83
[25]	CSF + BA	RF + C4.5 + FOREST ATTRIBUTE	NSL-KDD CICIDS2017	-	ACC: 96.76 DR: 94.04 FPR: 2.38
[26]	IG	LightGBM	CICIDS2017	10	ACC: 98.37
[27]	Hybrid Feature Selection (RF + PCA)	Light GBM	CICIDS2018	24	ACC: 97.73
[28]	Sigmoid POI	DT	NSL-KDD	18	ACC: 86.90 FPR: 6.40
[28]	Cosine POI	DT	NSL-KDD	5	ACC: 86.90 FPR: 8.80

Table 1. *Cont.*

Ref.	Feature Selection Algorithm	Classification Algorithm	Dataset	Number of Features	Result (%)
[29]	ABC	AdaBoost	NSL-KDD	25	ACC: 98.90 FPR: 0.01
[30]	Binary-particle swarm optimization	Decision tree	CICIDS2018	19	ACC: 99.52
[31]	IG and correlation attribute evaluation methods	PART	CICIDS2017	56	ACC: 99.98 FPR: 1.35
[32]	Information gain and ranker algorithm	J48	CICIDS2017	75	ACC: 87.44

3. Methodology

Figure 2 shows the methodology which includes: database selection, pre-processing, feature selection, classification, evaluation, as well as the analysis and comparison of results steps. We will explain these steps in the following subsections.

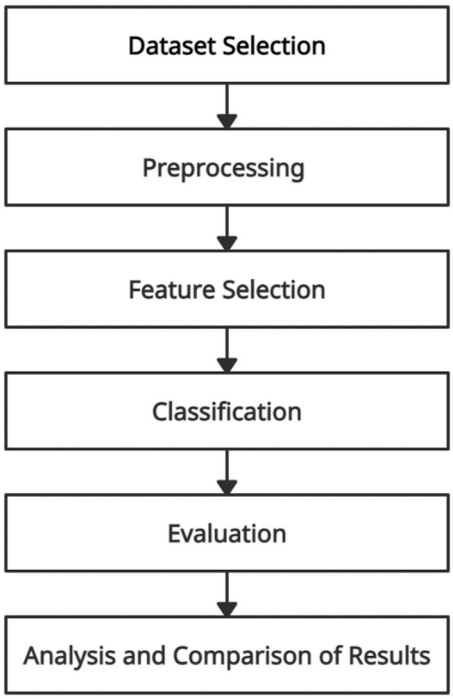

Figure 2. The methodology.

3.1. Dataset Selection

New cybersecurity datasets are available, so this work used three different datasets, namely NSL-KDD, CIC-IDS-2017, and CIC-IDS-2018, for the experiments. All of these datasets were explored to determine which was the most suitable dataset for building an efficient IDS.

3.1.1. NSL-KDD

The NSL-KDD dataset is one of the most widely used benchmarks for IDSs [8,19,28]. It is an upgraded version of the old KDD Cup99 dataset [7]. It has four attack categories: user to root attack (U2R), denial of service attack (DoS), probing attack, and remote to local attack (R2L). The full NSL-KDD training dataset contains 125,973 records, whereas the full

NSL-KDD testing dataset contains 22,254 records. There are 41 features in the NSL-KDD dataset: three of them are nominal or symbolic features, and the rest are numeric features.

3.1.2. CIC-IDS-2017

The CIC-IDS-2017 dataset is a network traffic dataset that consists of both normal and a variety of attack data developed by the Faculty of Computer Science, University of New Brunswick and the Canadian Institute of Cybersecurity (CIC) in 2017. This dataset was captured over a duration of five days, and it uses a large variety of attack types [25,33,34]. In the CIC 2017 dataset, the attack simulation is divided into seven categories namely, botnet, brute force attack, DoS attack, DdoS attack, infiltration attack, web attack, and heart bleed attack [25,34]. This dataset contains over 2.5 million records and 78 features, including the label column, and 19.70% of CIC-IDS-2017 is attack traffic. The dataset has a class imbalance. An unequal distribution between majority and minority classes in databases is known as class imbalance, which affects the performance of classification [34].

3.1.3. CIC-IDS-2018

The CIC-IDS-2018 dataset is a network traffic dataset consisting of both normal and a variety of attack data developed by the Faculty of Computer Science, University of New Brunswick, the CIC, and the Communications Security Establishment (CSE) in 2018 [6]. The structure of this dataset is similar to the previous dataset, and both have a class imbalance. Roughly 17% of the total number of records, which is 16,233,002 records, is attack traffic.

3.2. Preprocessing

The datasets must be preprocessed to avoid inconsistent, irrelevant, and missing data that affect the performance of the IDS. This step may include several tasks such as removing null or missing values, resampling, and scaling.

Certain datasets have missing, null, or symbolic values. These types of data structures make the classification algorithms difficult to handle. Therefore, missing values or null values are removed, and symbolic values are mapped to numeric values. Duplicated records and features should be removed to prevent the classifiers from being biased to the most frequent records [28]. Certain machine learning classifiers such as SVM and MLP are sensitive to feature scaling and require the scaling of a dataset because these classifiers provide weights to the input features according to their data points and inferences for output. Therefore, scaling all the used datasets before the FS, training, and testing phases is highly recommended.

The highly unbalanced datasets in the CIC-IDS-2017 and CIC-IDS-2018 datasets affect the performance of the classifier. The random under-sampling (RUS) technique is used to balance the class distribution. The RUS technique, in which specific majority instances are removed to balance a dataset, provides a good result compared with other sampling methods in the context of the CIC-IDS-2017 dataset [35].

3.3. Feature Subset Selection

This work focused on FS techniques and considered different wrapper FS techniques such as SFS, SBS, and the GA. Different machine learning techniques such as MLP and SVM were applied as the objective functions for the chosen FS techniques.

Not all features of the NSL-KDD, CIC-IDS-2017, and CIC-IDS-2018 datasets are important to build an efficient IDS. A subset of these features can achieve high a ACC and low FPR. Moreover, eliminating certain features using FS techniques is necessary to build an efficient IDS with a high accuracy and low false alarms. In this study, the performance of SFS, SBS, and GA FS techniques were explored. We used the wrapper FS approach. This approach is based on three components: a search strategy, a classifier, and an evaluation function [13,14]. We used three search strategies (SFS, SBS, and the GA). In each search strategy, we used SVM and MLP as classifiers. The evaluation of the feature subsets was a

fitness function based on a custom score of the cross-validation of the previous classifiers. Cross-validation is a resampling technique that divides the dataset into equal different portions to train and test a model on different iterations. Cross-validation gives an insight on how the model can be generalized to an unknown dataset, and it is a useful technique to identify the overfitting of a model. In the FS phase, two-fold, five-fold, and ten-fold cross validation are used. Our goal in this study was to propose an FS method that increases the ACC and decreases the false alarm rate with the minimum number of features. Therefore, our objective function or fitness function was based on three criteria: the classification accuracy of SVM or MLP, the false alarm rate or FPR, and the number of selected features. Hence, the subset having the smallest number of features, the highest ACC, and the lowest FPR produces the highest objective function.

Several objective functions are available. Hence, we investigated several objective functions in the FS phase. Bamakan et al. [36] proposed an objective function based on the detection rate, the false alarm rate of the classifier, and the number of selected features, as shown in Equation (1). Hang and Wang [37] proposed an objective function based on the accuracy and the number of selected features, as shown in Equation (2).

$$Objective\ Function(X) = DR(X) * WA + (1 - FPR) * WF + (1 - N * WN) \quad (1)$$

$$Objective\ Function(X) = ACC(X) * WA + (1 - N * WN) \quad (2)$$

where N is the number of selected features in the subset, WA is a predefined weight for the accuracy score of the subset, WF is a predefined weight for the FPR of the subset, WD is a predefined weight for the DR of the subset, and WN is a predefined weight for the number of selected features in the subset. All the weights should be in range [0–1].

In the following, we explain the selected FS techniques in this study, which were the SFS, SBS, and GA feature methods.

3.3.1. Sequential Forward Selection (SFS)

SFS is a wrapper FS that selects k features from an initial d features dataset where k < d through a number of iterations. It selects the best feature subset by starting from an empty dataset and adding one feature at a time based on the classifier performance in an iterative process until a stopping criterion is met, such as a feature subset of the specified size k being reached. If the desired number of features is in a range (e.g., 1–40), SFS selects the feature subset that contains the highest objective function (e.g., accuracy) in that range. SFS flowchart is shown in Figure 3.

3.3.2. Sequential Backward Selection (SBS)

This method is essentially the reverse of the above method. It selects the best feature subset by starting from the original full feature dataset and removes one feature at a time based on the classifier performance in an iterative process until a stopping criterion is met, such as a feature subset of the specified size k being reached. If the desired number of features is in a range (e.g., 1–40), SBS selects the feature subset that contains the highest objective function (e.g., accuracy) in that range. SBS flowchart is shown in Figure 4.

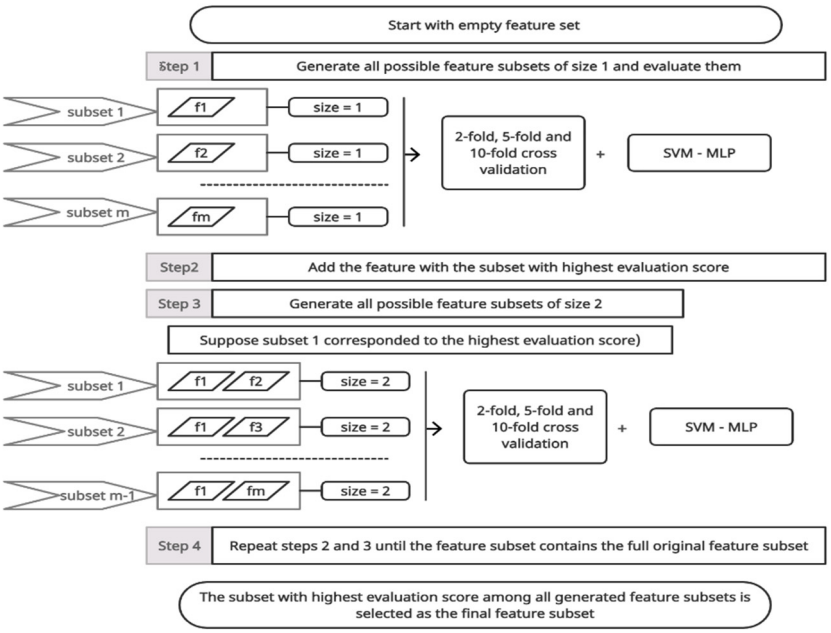

Figure 3. SFS flowchart.

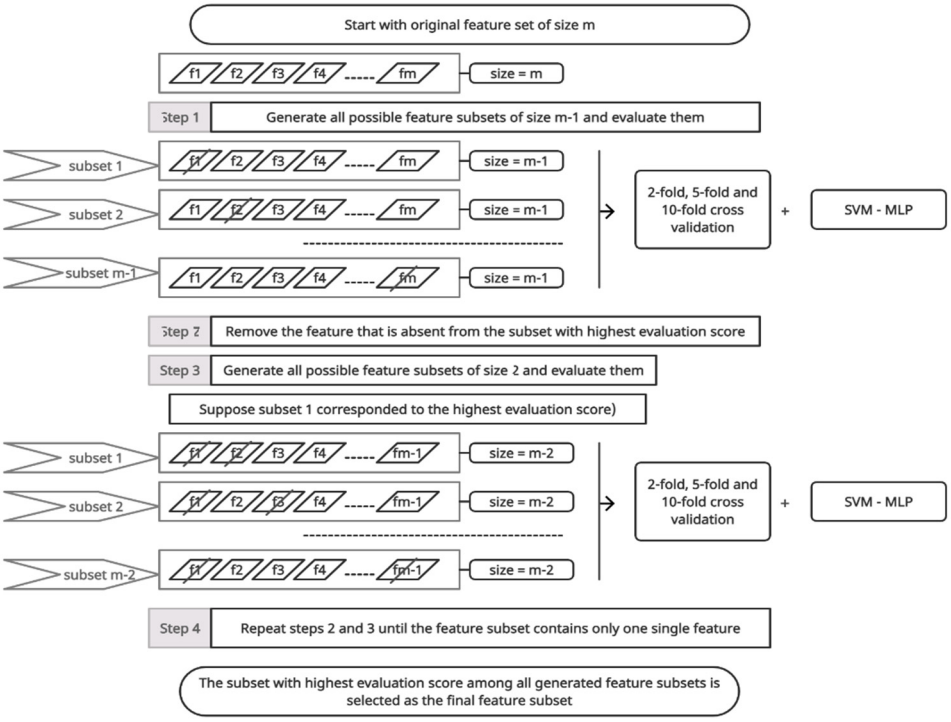

Figure 4. SBS flowchart.

3.3.3. Genetic Algorithm (GA)

The GA is composed of five components: the initiation of the population, a fitness function, a selection function, genetic operators (e.g., crossover and mutations), and stopping criteria. Below, we explain each component in a separate section.

Population Initiation

First, the GA randomly generates a population of individuals or chromosomes in which each chromosome represents a solution for the problem [12,13]. Each chromosome or individual is encoded as binary; a feature is either included or not in the subset. The num-

ber of individuals or the population size is defined by the user (n_population). A large population size provides a large search space to the GA but it increases the required time to evaluate all the individuals of the population. In contrast, a small population size provides a small search space to the GA but the required time to evaluate all the individuals of population is less than in the first situation.

Fitness Function

The fitness function determines the chromosome's chance of being chosen to create the offspring or the next generation. The fitness function can be any metric of the classifier performance, such as accuracy. For example, if the classifier accuracy is chosen to be the fitness function, the chromosome having the highest accuracy produces the highest fitness value [13].

Selection Function

The selection function selects the parents of offspring or the next generation of individuals from the current generation. In this study, we used the tournament selection method that selects the best individuals from a population of individuals. It randomly selects a predefined number of individuals (tournament_size) and runs a tournament among them. The individual or chromosome with the best fitness is the winner of the tournament and it is selected for generating the next generation. The tournament selection is repeated several times, as specified by the user. The best induvial with the highest fitness function among other individuals in each generation is added to the population of the next generation. Hence, each generation in the GA has an individual that has the highest fitness function among its previous generations.

Crossover Process

The crossover operator is used to generate new offspring or solutions from the current solutions (the parents). For each two randomly selected individuals or chromosomes, a crossover is performed with a predefined probability in a range [0–1]. Several crossover operators are available, such as one-point, two-point, and uniform crossovers. In this study, we used a uniform crossover in which each bit in a parent's chromone is swapped based on a predefined probability and a random number in a range [0–1]. For instance, if the predefined probability is 0.5 and the random number is 0.7, in this case the random number is equal to or greater than the predefined probability. Therefore, bit swapping occurs. This process is repeated for all bits in the parents' chromosome. Hence, we have two probabilities: a probability for applying crossover between two individuals and a probability for exchanging the bit value between two individuals.

Mutation Process

Mutation, which is applied after the crossover process, is the alteration of individual bits to generate new individuals from the current individuals. Several mutation operators are available, such as inversion mutation, insertion mutation, and flip-bit mutation [38]. In this study, we used flip-bit mutation, which switches certain bits from 1 to 0 or vice versa. Two probabilities are used in mutation: the probability of applying mutation on the individual and the probability of mutation or flipping the bit value of the individual. New different individuals are generated from the current generation after applying crossover and mutation operators.

Stopping Criteria

The evaluation of individuals, selection, crossover, and mutation are repeated in a predefined number of generations until a stopping criterion is met (e.g., the maximum number of generation or a specified number of generations is reached when the objective function cannot improve).

3.4. Classification

The selected feature subsets are then fed to a classifier such as MLP and SVM to classify the inputs as normal or attack traffic. Different classifiers are investigated to identify the most suitable one for training and testing the IDS model.

3.4.1. Support Vector Machine (SVM)

SVM is a supervised learning model that provides a good level of accuracy in classification problems [4,8,10,21,33]. SVM builds one or more hyperplanes by using the nearest training data points of each data (called support vectors). Then, it tries to maximize the margin between the data points. In the implementation of SVMs, there are certain kernel functions such as polynomial, sigmoid, and radial kernel functions (RBF). Making the trade-off between constant C and the type of kernel function is crucial to achieve a good result for the classification [8].

3.4.2. Artificial Neural Network Multi Perceptron (ANN-MLP)

MLP is also a supervised learning model. It is a class of feedforward ANN. There are at least three layers in MLP: an input, hidden, and output layer [19]. Each layer contains one or more neurons. Each neuron connects with a specific weight to every neuron in the following layer [19]. The learning in MLP occurs by changing the weights after each input of data.

3.5. Evaluation

The performance of each FS technique is evaluated on the basis of performance measures such as ACC, FPR and F1 score as well as the number of selected features. Several metrics are available for evaluating feature selection algorithms such as accuracy, false positive rate, detection rate, and precision. These metrics can be calculated using the confusion matrix that is represented by the following four main parameters:

- True positive (TP): represents number of attack samples classified correctly.
- True negative (TN): represents number of normal samples classified correctly.
- False positive (FP): represents number of normal samples classified wrongly.
- False negative (FN): represents number of attack samples classified wrongly.

In binary classification, a confusion matrix is a table or matrix of size 2×2 that is used to describe the performance of machine learning classifiers. A confusion matrix is shown in Table 2, where each column represents the predictive records and each row represents the actual records.

Table 2. Confusion matrix.

	Predictive Records	
Actual records	TP	FP
	FN	TN

The definition and formulas of the performance metrics are as follows [28]:

- Accuracy: represents the proportion of correct classified instances to the total number of classifications, as in Equation (3).

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (3)$$

- FPR (a.k.a. false alarms): represents the proportion of the normal instances that are identified as attack or abnormal instances, as in Equation (4).

$$FPR = FP/(TN + FP) \quad (4)$$

- Precision: represents the ratio of correctly predicted positive instances to the total predicted positive instances, as in Equation (5).

$$Percision = TP/TP + FP \tag{5}$$

- Recall (a.k.a. detection rate (DR)): represents the ratio of correctly predicted positive instances to the overall number of actual positive instances, as in Equation (6).

$$Recall = TP/TP + FN \tag{6}$$

- F1 score: represents the weighted average of precision and recall values, as in Equation (7).

$$F1\ score = 2 * \frac{percision * recall}{percision + recall} \tag{7}$$

3.6. Analysis and Comparison of Results

The results obtained from the previous step were analyzed and compared with one another to select an efficient IDS model with a high accuracy, low false alarm, and a small number of features.

4. Implementation

In this section, the experimental settings are presented, and the effectiveness of the SFS, SBS, and GA methods are illustrated. The experimental environment was set up as follows: a desktop computer running Windows 10 on an Intel Core i9-12th Gen with 48 GB RAM, Anaconda 3 with Python 3 distribution, the Scikit-learn library, and the Jupyter notebook [39]. Figure 5 illustrates and summarizes the steps of implementation, which are presented in detail.

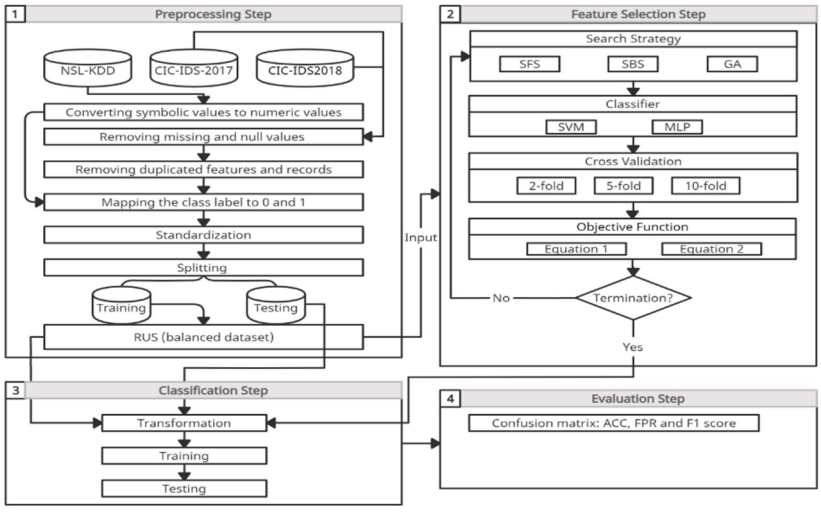

Figure 5. Implementation flowchart.

4.1. Preprocessing

The processing phase we used consist of eight steps which are:

1. In the NSL-KDD dataset, three categorical features, which are flag, service, and protocol_type features, are mapped to numeric values ranging from 0 to N − 1, where N is the number of symbols in the feature.
2. Missing values or null values are removed from the CIC-IDS-2017 and CIC-IDS-2018 datasets. A script written in Python is used for removing these records.

3. A duplicate feature in CIC-IDS-2017, namely Fwd Header Length, is removed manually. The timestamp feature is removed manually in CIC-IDS-2018. In addition, ten features are removed manually in CIC-IDS-2017 and CIC-IDS-2018, as they have zero values.
4. Duplicated records are removed in all datasets. A script written in Python is used for removing these records.
5. The class label is mapped to 0 for normal class and 1 for attack. As we use binary classification in this study, all the sub-category attack labels are mapped to 1. The resulted label feature contains 0 for normal records and 1 for attack records.
6. The StandardScaler method from the sklearn library in Python is applied to standardize the feature variance in all the used datasets.
7. All datasets are split into 70% training and 30% testing datasets.
8. The random under-sampling (RUS) technique is applied in all training datasets.

4.2. Feature Selection

The scikit-learn library and Jupyter notebook are used to implement SFS, SBS, and GA FS methods. The performance evaluation of the selected features of the subsets in each iteration in SFS, SBS, and the GA is carried out using the cross-validation technique in which the dataset is randomly divided into several K subsets. One subset is used for testing the model, and the remaining subsets are used for training. This process is iterated K times, and the testing subset is different in each iteration [13]. Table 3 shows the selected weights in the selected fitness functions.

Table 3. Parameters of fitness functions.

Objective Function	Parameter	Value
Equation (1)	WD	0.45
	WF	0.45
	WN	0.1
Equation (2)	WA	0.94
	WN	0.06

Implementation of SFS, SBS and the GA

The MLxtend library in Python is used to implement SFS and SBS, while the Sklearn-genetic, which is a genetic FS module for scikit-learn, is used to implement the GA [40,41]. SVM and MLP are used separately as estimators in SFS, SBS and the GA. To evaluate the subsets of selected features in each iteration, two-fold, five-fold, and ten-fold cross validation are used separately in the implementation of SFS, SBS and the GA. The parameters of SFS and SBS are configured as shown in Table 4, while the parameters of the GA are configured as shown in Table 5.

Table 4. Parameters of SFS and SBS.

Parameter	Value	Parameter	Value
Estimator	SVM	n_jobs	−1
	MLP	floating	False
Scoring	Equation (1)	cv	2
	Equation (2)		5
k_features	NSL-KDD (1,40)		10
	CIC-IDS-2017 (1,66)	forward	True (SFS)
	CIC-IDS-2018 (1,67)		False (SBS)

Table 5. Parameters of GA.

Parameter	Value	Parameter	Value
Estimator	SVM	crossover_proba	0.6
	MLP	crossover_independent_proba	0.6
Scoring	Equation (1)	mutation_proba	0.1
	Equation (2)	mutation_indenpdent_proba	0.1
Max_features	NSL-KDD (40)	caching	True
	CIC-IDS-2017 (66)	n_gen_no_change	NSL-KDD (50)
	CIC-IDS-2018 (67)		CIC-IDS-2017 (65)
n_population	NSL-KDD (60)		CIC-IDS-2018 (65)
	CIC-IDS-2017 (80)	n_jobs	−1
	CIC-IDS-2018 (80)	verbose	1
n_generations	NSL-KDD (50)	cv	2
	CIC-IDS-2017 (65)		5
	CIC-IDS-2018 (65)		10

4.3. Training and Testing

For each selected subset, we transform the original training dataset to have the same number of selected features from the FS phase. The selected feature subsets from the previous FS methods are fitted into the same used classifier in the FS phase to determine and evaluate the performance of the selected feature subset. Hence, SVM is used to evaluate the selected feature subsets selected from the FS phase when SVM is used as an objective function. MLP is used to evaluate the selected feature subsets selected from the FS phase when MLP is used an objective function.

4.4. Results and Discussion

The experimental results of the chosen feature selection methods are presented in this section.

Table 6 shows the result of SFS, SBS and the GA based on SVM and MLP in the NSL-KDD dataset using an objective function based on accuracy and the number of selected features. As shown in the table, the results are presented and compared using different cross validations such as two-fold, five-fold, and ten-fold cross validation for each combination of FS method and classifier. The highest accuracy among all these combinations is 99.23% with the GA+SVM, with a 0.77% false alarm rate and 29 selected features using two-fold cross validation in the feature selection phase. This model also has the highest F1 score, of 99.20%, among the models. The lowest false alarm rate among these combinations is 0.73% with the GA+SVM, with an accuracy of 99.06%. SBS+SVM performs well with a different number of folds. It obtained 98.92%,98.95%, and 98.99% in terms of accuracy and 1.20%, 1.36%, and 1.16% in terms of FPR, respectively, with the same selected number of features (10 features). A graphical illustration of Table 6 is shown in Figure 6.

Table 7 shows the result of SFS, SBS, and the GA based on SVM and MLP in the CIC-IDS-2017 dataset using an objective function based on accuracy and the number of selected features. As shown in the table, the highest accuracy among all these combinations is 99.96% with the GA+MLP, with a 0.03% false alarm rate and 40 selected features using five-fold cross validation in the feature selection phase. The lowest false alarm rate among all these combinations is 0.01% with SFS+MLP and the GA+SVM, with accuracies of 88.74% and 99.9%, respectively. SBS+SVM performs well with a different number of folds. It obtained 99.65%, 99.81%, and 99.75% in terms of accuracy and 0.50%, 0.13%, and 0.27% in terms of FPR, with the six, five, and five selected features, respectively. An illustration of Table 7 is shown in Figure 7.

Table 6. Performance of SFS, SBS and the GA using objective function based on accuracy and number of selected features in NSL-KDD.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	98.94	97.83	98.95	98.25	99.23	99.01
	5	98.80	97.66	98.82	98.81	99.16	99.02
	10	98.89	98.78	98.88	98.16	99.06	99.06
Number of selected features	2	10	9	10	16	29	38
	5	9	9	10	14	27	35
	10	10	13	10	13	29	36
FPR (%)	2	1.14	2.51	0.99	1.50	0.77	0.95
	5	1.44	2.84	1.18	1.26	0.84	1.20
	10	1.39	1.59	1.26	1.84	0.73	1.16
F1 (%)	2	98.90	97.75	98.81	98.17	99.20	98.98
	5	98.76	97.59	98.77	98.77	99.13	98.98
	10	98.85	98.83	98.84	98.09	99.03	99.02

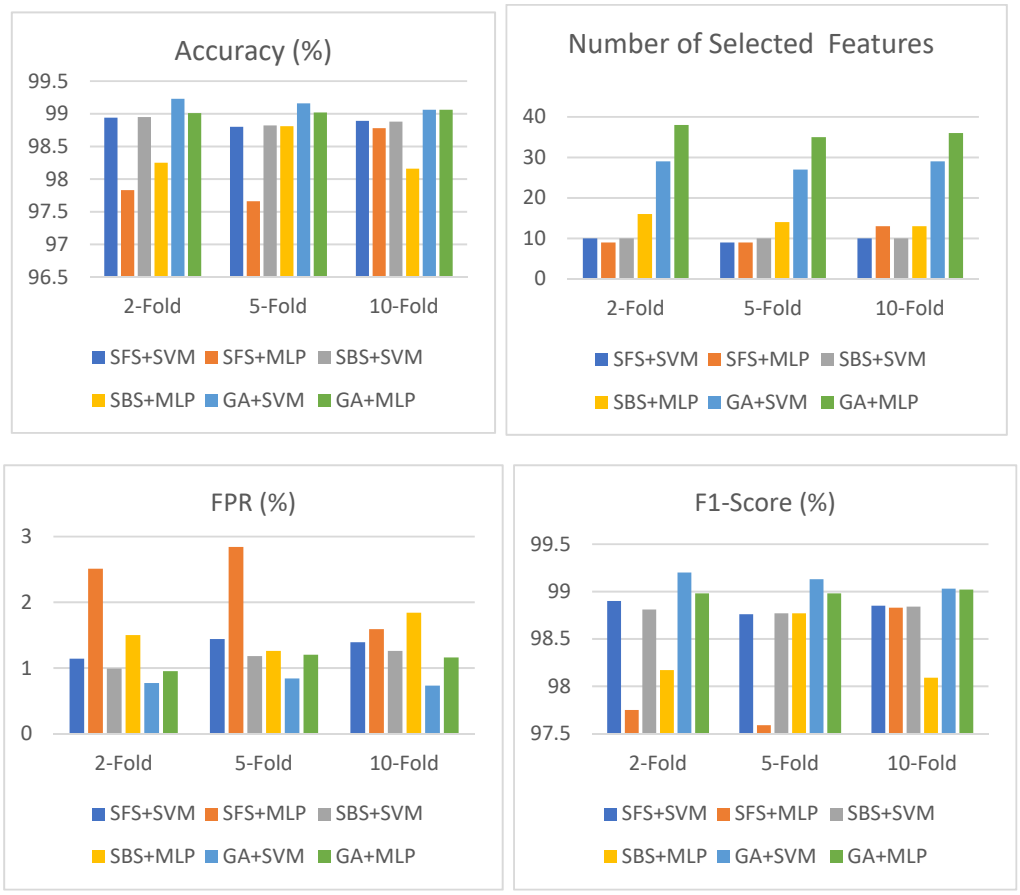

Figure 6. Performance of SFS, SBS and GA using objective function based on accuracy and number of selected features in NSL-KDD.

Table 7. Performance of SFS, SBS, and GA using objective function based on accuracy and number of selected features in CIC-IDS-2017.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	99.75	99.83	99.65	94.79	99.94	99.93
	5	99.78	89.61	99.81	99.84	99.94	99.96
	10	99.28	88.74	99.75	99.83	99.9	99.91
Number of selected features	2	5	7	5	6	42	44
	5	5	5	6	5	39	40
	10	5	5	5	6	45	38
FPR (%)	2	0.27	0.04	0.5	1.12	0.02	0.02
	5	0.22	0.03	0.13	0.17	0.02	0.03
	10	1.44	0.01	0.27	0.12	0.01	0.03
F1 (%)	2	99.78	99.86	99.69	95.24	99.95	99.96
	5	99.81	89.93	99.83	99.86	99.93	99.94
	10	99.36	89.00	99.78	99.85	99.93	99.39

Figure 7. Performance of SFS, SBS, and GA using objective function based on accuracy and number of selected features in CIC-IDS-2017.

Table 8 shows the result of SFS, SBS, and the GA based on SVM and MLP in the CIC-IDS-2018 dataset using an objective function based on accuracy and the number of selected features. As shown in the table, the highest accuracy among all these combinations is 99.87% with the SFS+MLP and SBS+MLP models. SFS+MLP obtains a lower FPR, of 0.1%,

than SBS+MLP with eight selected features using five-fold cross validation in the feature selection phase. Three models, which are SFS+MLP, SBS+SVM, and the GA+MLP obtain the lowest false alarm rate, of 0.1%, among the models. A demonstration of Table 8 is shown in Figure 8.

Table 8. Performance of SFS, SBS, and GA using objective function based on accuracy and number of selected features in CIC-IDS-2018.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	99.46	99.87	99.78	99.87	99.71	99.69
	5	97.67	99.87	99.64	99.51	99.82	99.83
	10	97.72	99.80	99.69	99.67	99.8	99.78
Number of selected features	2	21	11	7	8	18	23
	5	6	8	7	10	21	26
	10	6	8	8	11	24	25
FPR (%)	2	0.32	0.90	0.10	0.90	0.16	0.21
	5	0.18	0.10	0.10	0.84	0.16	0.10
	10	0.30	0.30	0.20	0.90	0.19	0.17
F1 (%)	2	99.46	99.88	99.77	99.88	99.71	99.69
	5	97.61	99.85	99.64	99.51	99.82	99.80
	10	97.63	99.80	99.71	99.74	99.80	99.75

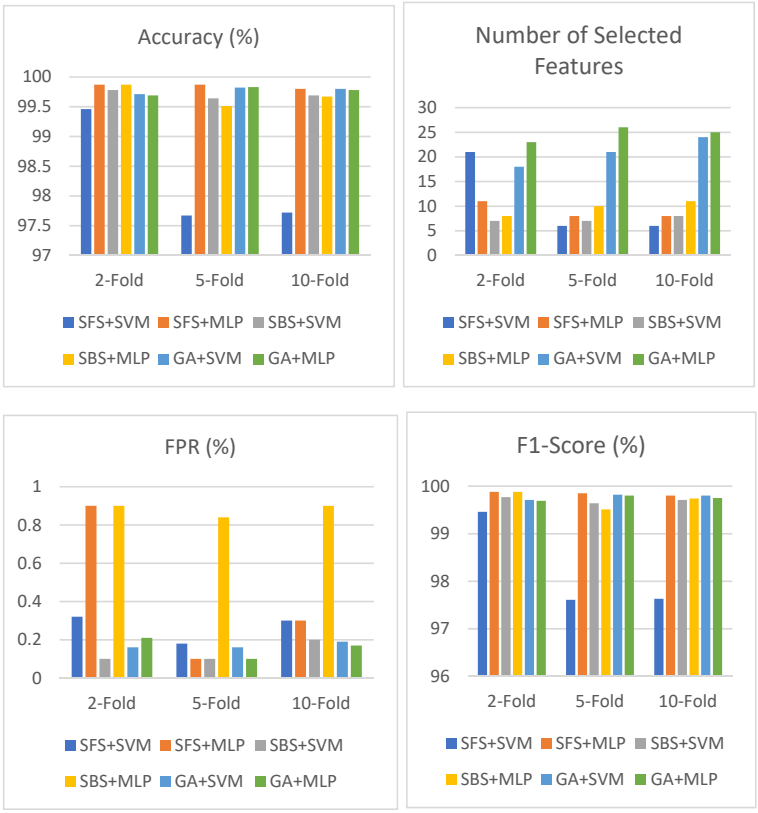

Figure 8. Performance of SFS, SBS, and GA using objective function based on accuracy and number of selected features in CIC-IDS-2018.

Table 9 shows the result of SFS, SBS, and the GA based on SVM and MLP in the NSL-KDD dataset using an objective function based on detection rate, false positive rate, and number of selected features. As shown in the table, the GA+SVM obtains the highest accuracy and lowest false alarm rate among all these combinations. It obtains an accuracy of 99.21% and an FPR of 0.83% with 30 selected features using 10-fold cross validation in the feature selection phase. In addition, this model obtains an accuracy of 99.19% and an FPR of 0.81% with 33 selected features using five-fold cross validation in the feature selection phase. Table 9 is depicted in Figure 9.

Table 9. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in NSL-KDD.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	98.77	97.83	97.76	97.7	99.18	99.11
	5	98.19	97.66	98.82	97.99	99.19	98.98
	10	98.19	98.82	98.65	97.93	99.21	98.93
Number of selected features	2	9	9	6	9	27	37
	5	7	9	10	10	33	28
	10	7	12	8	10	30	39
FPR (%)	2	1.46	2.51	1.94	2.76	0.89	0.85
	5	1.85	2.84	1.81	1.76	0.81	1.17
	10	1.85	1.65	1.82	2.64	0.83	1.28
F1 (%)	2	98.72	97.96	97.66	97.63	99.15	99.08
	5	98.12	97.59	98.78	97.91	99.16	98.92
	10	98.12	98.09	98.61	98.01	99.18	98.88

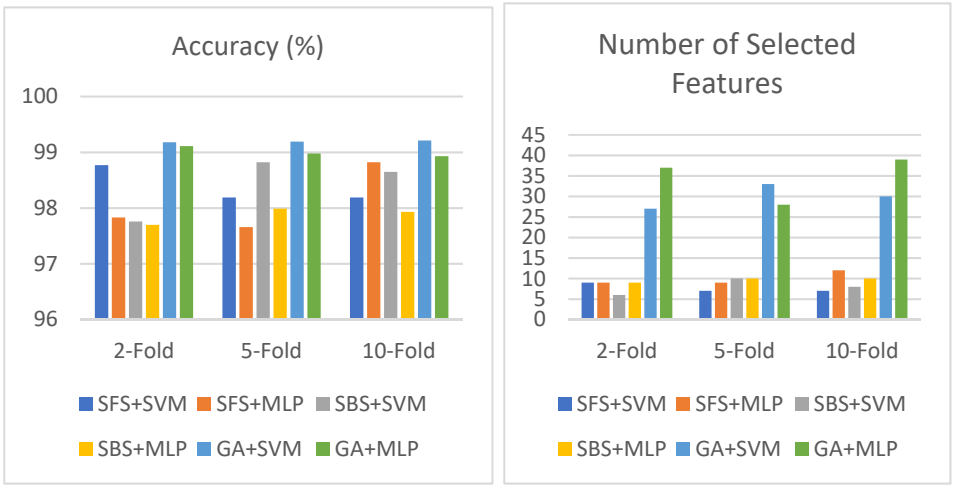

Figure 9. *Cont.*

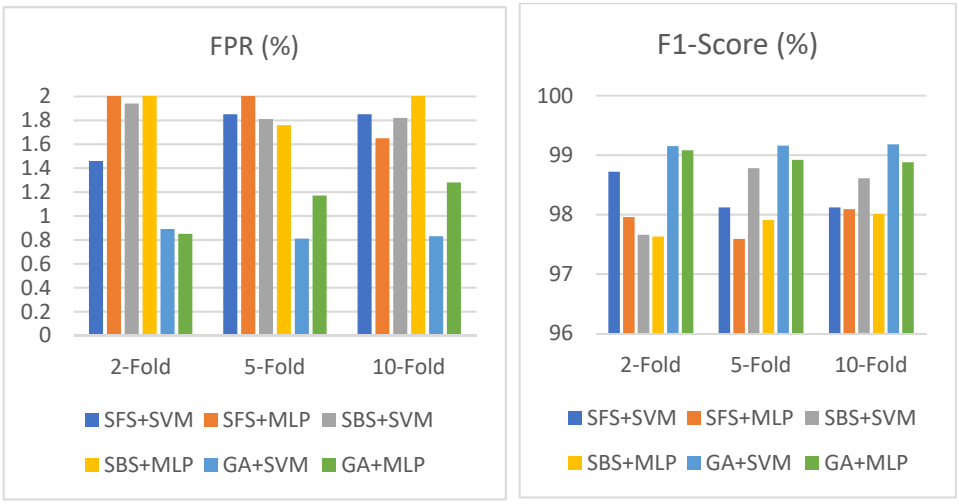

Figure 9. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in NSL-KDD.

Table 10 shows the results of SFS, SBS, and the GA based on SVM and MLP in the CIC-IDS-2017 dataset using an objective function based on the detection rate, false positive rate, and number of selected features. As shown in the table, GA+MLP obtains the highest accuracy, of 99.95%, among all these combinations and a low false alarm rate of 0.007% with 35 selected features using two-fold cross validation in the feature selection phase. The GA models obtain a higher accuracy than the SFS and SBS models and give a low false alarm rate (less than 0.035%) in all their combinations. Figure 10 displays Table 10 in its entirety.

Table 10. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in CIC-IDS-2017.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	97.86	98.07	99.86	96.41	99.94	99.95
	5	99.87	97.9	99.9	98.17	99.93	99.91
	10	99.87	97.62	99.89	94.72	99.94	99.93
Number of selected features	2	27	17	16	10	38	35
	5	31	24	14	14	42	40
	10	29	24	19	9	41	33
FPR (%)	2	0.06	0.05	0.03	0.03	0.003	0.007
	5	0.09	0.003	0.05	0.05	0.03	0.01
	10	0.06	0.05	0.04	0.04	0.02	0.005
F1 (%)	2	98.08	98.28	99.88	96.74	99.93	99.92
	5	99.89	98.12	99.92	98.36	99.93	99.93
	10	99.89	97.86	99.91	95.12	99.92	99.89

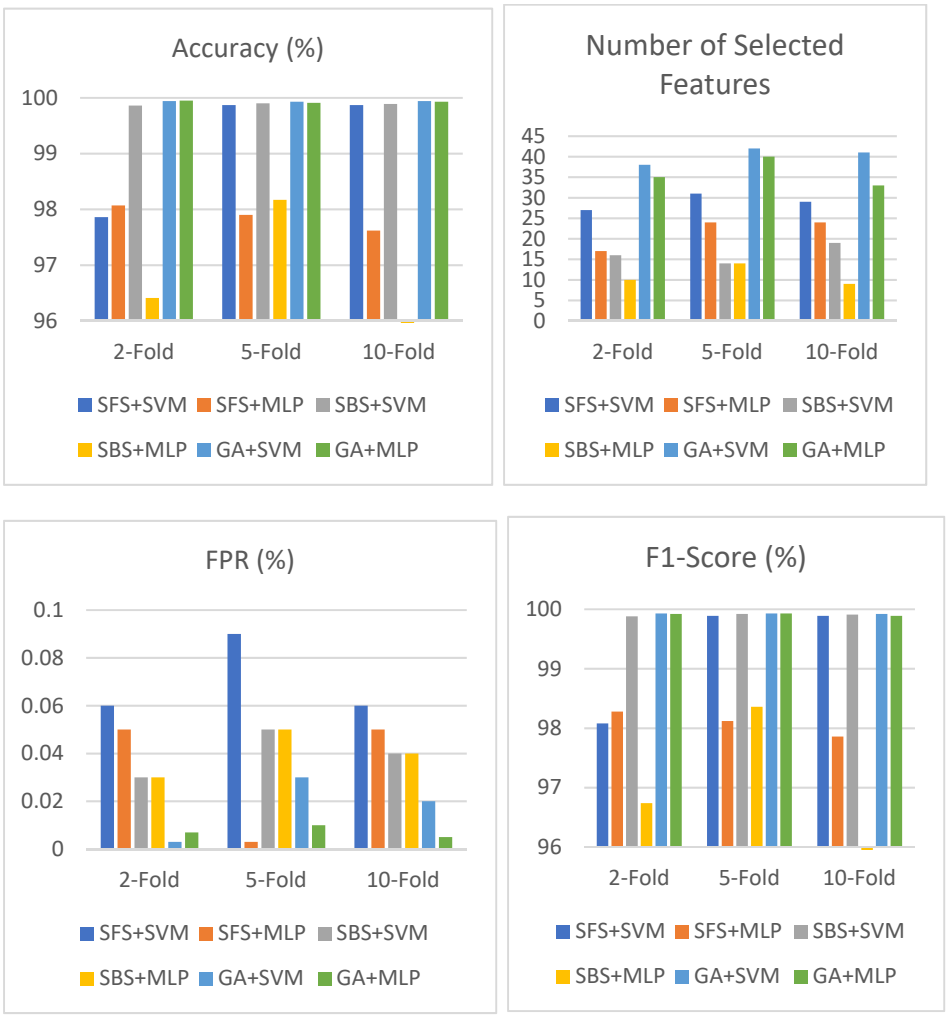

Figure 10. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in CIC-IDS-2017.

Table 11 shows the result of SFS, SBS, and the GA based on SVM and MLP in the CIC-IDS-2018 dataset using an objective function based on the detection rate, false positive rate, and number of selected features. As shown in the table, the GA+MLP obtains the highest accuracy, of 99.92%, and lowest false alarm rate, of 0.07%, with 47 selected features using two-fold cross validation in the feature selection phase. The GA models obtain a higher accuracy than the SFS and SBS models and give a low false alarm rate (less than 0.20%) in all their combinations. Figure 11 depicts Table 11 in detail.

Table 11. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in CIC-IDS-2018.

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Accuracy (%)	2	97.41	97.57	98.2	99.56	99.56	99.92
	5	97.41	99.78	99.6	99.09	99.8	99.89
	10	97.38	98.21	99.01	99.16	99.74	99.82

Table 11. *Cont.*

Metric	Fold	SFS+SVM	SFS+MLP	SBS+SVM	SBS+MLP	GA+SVM	GA+MLP
Number of selected features	2	6	8	6	8	14	47
	5	4	7	5	7	15	41
	10	6	7	6	9	17	38
FPR (%)	2	0.35	0.19	0.36	0.15	0.13	0.07
	5	0.30	0.16	0.19	1.31	0.19	0.11
	10	0.33	0.19	0.20	1.22	0.15	0.19
F1 (%)	2	97.35	97.51	98.91	99.55	99.56	99.93
	5	97.34	99.78	99.60	99.07	99.80	99.89
	10	98.85	98.83	98.84	98.09	99.03	99.02

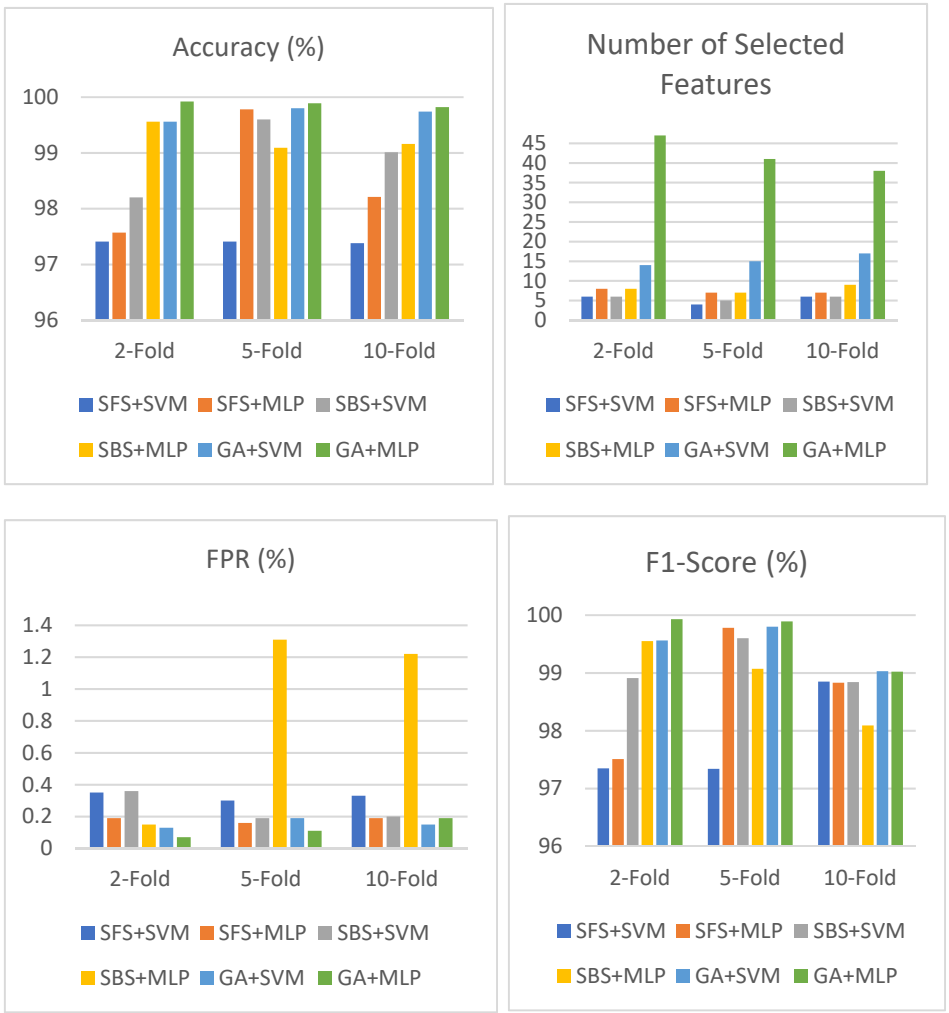

Figure 11. Performance of SFS, SBS, and GA using objective function based on DR, FPR, and number of selected features in CIC-IDS-2018.

4.5. Performance Comparison with the Recent Methods

Tables 12–14 show the comparison between our best obtained results and those of other recent methods in the NSL-KDD, CIC-IDS-2017, and CIC-IDS-2018 datasets.

Table 12. Performance comparison with recent methods in NSL-KDD.

Ref.	FS Tech.	Classifier	Number of Selected Features	ACC (%)	FPR (%)
[19]	CSA	MCF+MVO-ANN	22	98.81	0.02
[28]	Sigmoid POI	DT	18	86.90	6.40
[28]	Cosine POI	DT	5	88.30	8.80
[29]	ABC	AdaBoost	25	98.90	0.01
Proposed model	GA	SVM	29	99.23	0.77

Table 13. Performance comparison with recent methods CIC-IDS-2017.

Ref.	FS Tech.	Classifier	Number of Selected Features	ACC (%)	FPR (%)
[31]	IG and correlation attribute evaluation methods	PART	56	99.98	1.35
[32]	Information gain and ranker algorithm	J48	75	87.44	-
[26]	IG	LightGBM	10	98.37	-
Proposed model	GA	MLP	35	99.95	0.007
Proposed model	GA	SVM	38	99.94	0.003

Table 14. Performance comparison with recent methods in CIC-IDS-2018.

Ref.	FS Tech.	Classifier	Number of Selected Features	ACC (%)	FPR (%)
[27]	Hybrid feature selection (RF + PCA)	Light GBM	24	97.73	-
[30]	Binary-particle swarm optimization	Decision tree	19	99.52	-
Proposed model	SFS	MLP	11	99.87	0.90
Proposed model	SBS	MLP	8	99.87	0.90
Proposed model	GA	MLP	47	99.92	0.07

5. Conclusions

This paper has presented a comparative study of sequential forward selection, sequential backward selection, and genetic algorithm feature selection methods in intrusion detection systems to select an efficient IDS model that provides a high accuracy and low false alarm with a minimum number of features. The efficiencies of the three feature selection techniques with two classification methods, namely SVM and MLP, were compared. These methods were applied to three publicly available intrusion detection system data sets, namely NSL-KDD, CICIDS2017, and CICIDS2018. This paper has presented an assessment of these datasets, identifying their limitations and providing solutions to overcome these limitations.

The performance of the proposed models was analyzed, addressed, and compared to existing techniques. The efficiencies of the proposed models were proven in the experimental results, which indicated that the highest accuracy in the NSL-KDD dataset was 99.23%, achieved using the GA+SVM, with a 0.77% false alarm rate and 29 selected features using two-fold cross validation in the feature selection phase. This model also has the highest F1 score, of 99.20%, among the models. In the CICIDS2017 dataset, the highest accuracy among the proposed models is 99.96%, achieved with the GA+MLP, with a 0.03% false alarm rate and 40 selected features using five-fold cross validation in the feature

selection phase. The lowest false alarm rate among the proposed models is 0.01% in the case of SFS+MLP and the GA+SVM, with accuracies of 88.74% and 99.9%, respectively. In the CSE-CIC-IDS218 dataset, SFS+MLP and SBS+MLP achieved an accuracy of 99.87%. SFS+MLP obtains a lower FPR, of 0.10%, than SBS+MLP with eight selected features using five-fold cross validation in the feature selection phase.

Author Contributions: Conceptualization, I.A. and Y.A.; methodology, I.A.; software, Y.A.; validation, I.A., Y.A. and F.E.A.; formal analysis, I.A. and Y.A.; investigation, Y.A.; resources I.A. and F.E.A.; data curation, I.A.; writing—original draft preparation, Y.A.; writing—review and editing, I.A., Y.A. and F.E.A.; visualization, I.A.; supervision, I.A. and F.E.A.; project administration, I.A.; funding acquisition, I.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research work was funded by Institutional Fund Projects under grant no. (IFPRC-076-611-2020). The authors acknowledge technical and financial support from the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This research work was funded by Institutional Fund Projects under grant no. (IFPRC-076-611-2020). Therefore, the authors gratefully acknowledge technical and financial support from the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Conflicts of Interest: There is no conflict of authors.

References

1. Thakkar, A.; Lohiya, R. A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions. *Artif. Intell. Rev.* **2022**, *55*, 453–563. [CrossRef]
2. Alhakami, W.; Alharbi, A.; Bourouis, S.; Alroobaea, R.; Bouguila, N. Network Anomaly Intrusion Detection Using a Nonparametric Bayesian Approach and Feature Selection. *IEEE Access* **2019**, *7*, 52181–52190. [CrossRef]
3. Thakkar, A.; Lohiya, R. Attack classification using feature selection techniques: A comparative study. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *12*, 1249–1266. [CrossRef]
4. Tao, P.; Sun, Z.; Sun, Z. An Improved Intrusion Detection Algorithm Based on GA and SVM. *IEEE Access* **2018**, *6*, 13624–13631. [CrossRef]
5. Ates, C.; Ozdel, S.; Anarim, E. A New Network Anomaly Detection Method Based on Header Information Using Greedy Algorithm. In Proceedings of the 6th International Conference on Control, Decision and Information Technologies (Codit 2019), Paris, France, 23–26 April 2019; IEEE: New York, NY, USA, 2019; pp. 657–662. [CrossRef]
6. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the International Conference on Information Systems Security and Privacy, Funchal, Portugal, 22–24 January 2018; pp. 108–116.
7. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
8. Saleh, A.I.; Talaat, F.M.; Labib, L.M. A hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. *Artif. Intell. Rev.* **2019**, *51*, 403–443. [CrossRef]
9. Leevy, J.L.; Khoshgoftaar, T.M. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *J. Big Data* **2020**, *7*, 104. [CrossRef]
10. Wang, W.; Du, X.; Wang, N. Building a Cloud IDS Using an Efficient Feature Selection Method and SVM. *IEEE Access* **2019**, *7*, 1345–1354. [CrossRef]
11. Guyon, I.; Elisseeff, A. An Introduction to Variable and Feature Selection. *J. Mach. Learn. Res.* **2003**, *3*, 1157–1182.
12. Thangavel, N.S.G. Building an Efficient of Feature Selection Using Greedy Search Method for HNIDS in Cloud Computing. *J. Adv. Res. Dyn. Control Syst.* **2019**, *11*, 307–316.
13. Khammassi, C.; Krichen, S. A GA-LR wrapper approach for feature selection in network intrusion detection. *Comput. Secur.* **2017**, *70*, 255–277. [CrossRef]
14. Kohavi, R.; John, G.H. Wrappers for feature subset selection. *Artif. Intell.* **1997**, *97*, 273–324. [CrossRef]
15. Li, Y.; Wang, J.-L.; Tian, Z.-H.; Lu, T.-B.; Young, C. Building lightweight intrusion detection system using wrapper-based feature selection mechanisms. *Comput. Secur.* **2009**, *28*, 466–475. [CrossRef]

16. Mohammadzadeh, A.; Taghavifar, H. A robust fuzzy control approach for path-following control of autonomous vehicles. *Soft Comput.* **2020**, *24*, 3223–3235. [CrossRef]
17. Varma, P.R.K.; Kumari, V.V.; Kumar, S.S. Feature Selection Using Relative Fuzzy Entropy and Ant Colony Optimization Applied to Real-time Intrusion Detection System. *Procedia Comput. Sci.* **2016**, *85*, 503–510. [CrossRef]
18. Mohammadi, S.; Mirvaziri, H.; Ghazizadeh-Ahsaee, M.; Karimipour, H. Cyber intrusion detection by combined feature selection algorithm. *J. Inf. Secur. Appl.* **2019**, *44*, 80–88. [CrossRef]
19. Sarvari, S.; Sani, N.F.M.; Hanapi, Z.M.; Abdullah, M.T. An Efficient Anomaly Intrusion Detection Method with Feature Selection and Evolutionary Neural Network. *IEEE Access* **2020**, *8*, 70651–70663. [CrossRef]
20. Asdaghi, F.; Soleimani, A. An effective feature selection method for web spam detection. *Knowl.-Based Syst.* **2019**, *166*, 198–206. [CrossRef]
21. Aslahi-Shahri, B.M.; Rahmani, R.; Chizari, M.; Maralani, A.; Eslami, M.; Golkar, M.J.; Ebrahimi, A. A hybrid method consisting of GA and SVM for intrusion detection system. *Neural Comput. Appl.* **2016**, *27*, 1669–1676. [CrossRef]
22. Lee, J.; Park, O. Feature Selection Algorithm for Intrusions Detection System using Sequential forward Search and Random Forest Classifier. *KSII Trans. Internet Inf. Syst.* **2017**, *11*, 5132–5148. [CrossRef]
23. Li, Y.; Xia, J.; Zhang, S.; Yan, J.; Ai, X.; Dai, K. An efficient intrusion detection system based on support vector machines and gradually feature removal method. *Expert Syst. Appl.* **2012**, *39*, 424–430. [CrossRef]
24. Raman, M.G.; Somu, N.; Kirthivasan, K.; Liscano, R.; Sriram, V.S. An efficient intrusion detection system based on hypergraph —Genetic algorithm for parameter optimization and feature selection in support vector machine. *Knowl.-Based Syst.* **2017**, *134*, 1–12. [CrossRef]
25. Zhou, Y.; Cheng, G.; Jiang, S.; Dai, M. Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Comput. Netw.* **2020**, *174*, 107247. [CrossRef]
26. Hua, Y. An Efficient Traffic Classification Scheme Using Embedded Feature Selection and LightGBM. In Proceedings of the Information Communication Technologies Conference (ICTC), Nanjing, China, 29–31 May 2020; pp. 125–130. [CrossRef]
27. Seth, S.; Singh, G.; Chahal, K.K. A novel time efficient learning-based approach for smart intrusion detection system. *J. Big Data* **2021**, *8*, 1–28. [CrossRef]
28. Alazzam, H.; Sharieh, A.; Sabri, K.E. A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer. *Expert Syst. Appl.* **2020**, *148*, 113249. [CrossRef]
29. Mazini, M.; Shirazi, B.; Mahdavi, I. Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms. *J. King Saud Univ.-Comput. Inf. Sci.* **2019**, *31*, 541–553. [CrossRef]
30. Saeed, A.A.; Jameel, N.G.M. Intelligent feature selection using particle swarm optimization algorithm with a decision tree for DDoS attack detection. *Int. J. Adv. Intell. Inform.* **2021**, *7*, 37. [CrossRef]
31. Shaikh, J.M.; Kshirsagar, D. Feature Reduction-Based DoS Attack Detection System. In *Next Generation Information Processing System*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 170–177. [CrossRef]
32. Patil, A.; Kshirsagar, D. Towards Feature Selection for Detection of DDoS Attack. *Comput. Eng. Technol.* **2019**, 215–223. [CrossRef]
33. Ahmad, I.; Hussain, M.; Alghamdi, A.; Alelaiwi, A. Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components. *Neural Comput. Appl.* **2014**, *24*, 1671–1682. [CrossRef]
34. He, H.; Garcia, E.A. Learning from Imbalanced Data. *IEEE Trans. Knowl. Data Eng.* **2009**, *21*, 1263–1284. [CrossRef]
35. Ho, Y.B.; Yap, W.S.; Khor, K.C. The effect of sampling methods on the cicids2017 network intrusion data set. In IT Convergence and Security. In *IT Convergence and Security*; Springer: Singapore, 2021; pp. 33–41.
36. Bamakan, S.M.H.; Wang, H.; Yingjie, T.; Shi, Y. An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization. *Neurocomputing* **2016**, *199*, 90–102. [CrossRef]
37. Huang, C.-L.; Wang, C.-J. A GA-based feature selection and parameters optimizationfor support vector machines. *Expert Syst. Appl.* **2006**, *31*, 231–240. [CrossRef]
38. Gen, M.; Cheng, R. *Genetic Algorithms and Engineering Optimization*; John Wiley & Sons: Hoboken, NJ, USA, 1999; Volume 7.
39. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
40. Raschka, S. Mlxtend: Providing machine learning and data science utilities and extensions to python's scientific computing stack. *J. Open Source Softw.* **2018**, *3*, 638. [CrossRef]
41. Calzolari, M. Manuel-Calzolari/Sklearn-Genetic: Sklearn-Genetic 0.5.1 (0.5.1). Zenodo. 2022. Available online: https://zenodo.org/record/5854662#.Y5knyH1ByUk (accessed on 18 January 2022).