electronics

Article

Security of Things Intrusion Detection System for Smart Healthcare

Celestine Iwendi [1,2,†], Joseph Henry Anajemba [3,*,†], Cresantus Biamba [4,*,†] and Desire Ngabo [5,6,†]

[1] School of Creative Technologies, University of Bolton, Bolton BL3 5AB, UK; celestine.iwendi@ieee.org
[2] Department of Mathematics and Computer Science, Coal City University, Enugu 400231, Nigeria
[3] Department of Communication Engineering, College of Internet of Things, Hohai University, Changzhou 230001, China
[4] Faculty of Education and Business Studies, University of Gävle, 80176 Gävle, Sweden
[5] Department of Computer Science and Electronics Engineering, Hunan University, Changsha 410000, China; dngabo@hnu.edu.cn
[6] African Center of Excellence in the Internet of Things, University of Rwanda, Kigali P.O. Box 3900, Rwanda
* Correspondence: herinopallazo@ieee.org (J.H.A.); cresantus.biamba@hig.se (C.B.)
† These authors contributed equally to this work.

Abstract: Web security plays a very crucial role in the Security of Things (SoT) paradigm for smart healthcare and will continue to be impactful in medical infrastructures in the near future. This paper addressed a key component of security-intrusion detection systems due to the number of web security attacks, which have increased dramatically in recent years in healthcare, as well as the privacy issues. Various intrusion-detection systems have been proposed in different works to detect cyber threats in smart healthcare and to identify network-based attacks and privacy violations. This study was carried out as a result of the limitations of the intrusion detection systems in responding to attacks and challenges and in implementing privacy control and attacks in the smart healthcare industry. The research proposed a machine learning support system that combined a Random Forest (RF) and a genetic algorithm: a feature optimization method that built new intrusion detection systems with a high detection rate and a more accurate false alarm rate. To optimize the functionality of our approach, a weighted genetic algorithm and RF were combined to generate the best subset of functionality that achieved a high detection rate and a low false alarm rate. This study used the NSL-KDD dataset to simultaneously classify RF, Naive Bayes (NB) and logistic regression classifiers for machine learning. The results confirmed the importance of optimizing functionality, which gave better results in terms of the false alarm rate, precision, detection rate, recall and F1 metrics. The combination of our genetic algorithm and RF models achieved a detection rate of 98.81% and a false alarm rate of 0.8%. This research raised awareness of privacy and authentication in the smart healthcare domain, wireless communications and privacy control and developed the necessary intelligent and efficient web system. Furthermore, the proposed algorithm was applied to examine the F1-score and precisionperformance as compared to the NSL-KDD and CSE-CIC-IDS2018 datasets using different scaling factors. The results showed that the proposed GA was greatly optimized, for which the average precision was optimized by 5.65% and the average F1-score by 8.2%.

Keywords: intrusion detection; smart healthcare; SoT; deep learning; web security

Citation: Iwendi, C.; Anajemba, J.H.; Biamba, C.; Ngabo, D. Security of Things Intrusion Detection System for Smart Healthcare. *Electronics* **2021**, *10*, 1375. https://doi.org/10.3390/electronics10121375

Academic Editors: Abayomi Otebolaku, Gyu Myoung Lee, Edward Meinert, Asiya Khan and Gloria Iyawa

Received: 27 April 2021
Accepted: 1 June 2021
Published: 8 June 2021

1. Introduction

The origin of smart healthcare can be traced back to the idea of smart Earth projected by the International Business Machines Corporation (IBM) in 2009, which made reference to the unified utilization of artificial intelligence, big data, cloud computing and, mostly, the Internet of Things (IoT) to develop an interactive framework for distributing medical-related data and allowing communication among medical equipment (such as smart devices, applications and local networks), medical staff/institutions and patients (https://www.ibm.com/midmarket/us/en/article_Industries_1209.html (accessed on 12 January 2021)). Some of the most promising technologies that enable the implementation of smart healthcare systems include Wireless Sensor Networks (WSNs) and Ultra-High-Frequency (UHF) Radio Frequency Identification (RFID). However, since the inception of the smart healthcare concept, securing smart devices and confidential hospital/patient data has been one of the major challenges facing the industry. On the other hand, Machine Learning (ML), which is closely associated with computational statistics and one major aspect of the Security of Things (SoTs), has been presented to several applications to cyber-security for the analysis of hybrid networks, comprised of both anomaly detection and the detection of data mismanagement [1]. The ML technique is apparently becoming the most promising approach to deal with security glitches and several hidden (otherwise known as zero-day) attacks in healthcare systems [2]. The technique can recognise attacks merely by monitoring the modifications of data or simply by detecting alterations in the features of the network's traffic [3]. Even though ML may not be appropriate for issues that necessitate a prescribed descriptive solution, the technique can realize vigorous outcomes in problems that are ambiguous for human formalization. In a smart healthcare system (just as other smart systems), the performance of most Internet-based security frameworks is in line with compiling a list of malicious features to wedge them. Nevertheless, intruders repeatedly utilize ingenuity in refining and altering their procedures, which makes it extremely difficult to forecast their bad features being injected into the security black-list [4]. One little change of the security protocol can permit an intruder to inject unwanted packets or gain access to confidential information unnoticed. These undesirable frameworks relating to all hypothetically destructive packets and incessantly apprising the ruleset are unrealistic and enormously capital intensive [5]. Therefore, at the moment, the ML technique can play a substantial part in understanding the good packets, thus producing a prototype of them in a manner such that data packets that do not match them are measured as glitches, which are probably attacks or intrusions [6]. Thus, the performance of ML is thriving in the fields of data clustering and classification, which are both key in data security applications. A survey carried out on the latest intrusion detection system used in IoT models showed how improving ML's efficiency and reliability is on important task [7].

The Intrusion Detection System (IDS) is one of the most popular SoT paradigms for detecting anomaly or intrusion in live network traffic [8]. Several scholars have highlighted different issues relating to the Network Intrusion Detection System (NIDS) in the past due to its increasing importance in the current era of intelligent cyber-attacks. Recently, various Machine-Learning (ML) and feature-selection methods were widely implemented to develop the NIDS. One of the first attempts to achieve the detection rate and false alarm rate was performed using the 1998 DARPA dataset in the study of [9]. The algorithm utilized Principal Component Analysis (PCA) to select 22 features and neural networks for classification. Although PCA provides an optimal feature set, it compromises the training efficiency [10]. Some experiments employed multiple techniques for feature selection. Hee-su et al. [11] utilized four feature-selection techniques. The placement of the IDS is also a significant challenge. Many organizations implement or purchase the IDS, but they do not know where the IDS should be placed and where it can detect anomalies with perfection. According to Abhijit Sarmah [12], the IDS's success is dependent on how it is deployed. Furthermore, great effort is required to design and implement perfect IDS and implement a new state-of-the-art network-based IDS dataset. Other issues are the lack of efficient detection algorithms that have a very high Detection Rate (DR) and low False Alarm Rate (FAR).

The primary motivation of this paper was centred on designing a feature optimization method and developing a primal IDS with a high DR and more accurate FAR system.

The main objective of this research was that in network traffic, when we inspect a network packet to determine whether it is normal or an anomaly, it takes a huge amount of time to examine a significant number of attributes. Therefore, through optimized attribute selection, we identified and proposed only those features that play an essential role in an intrusion detection system and ignored all other irrelevant features. This has the practical implication of using fewer system resources with an increased DR and a reduced FAR.

The contribution of this study can be summarized as follows:

1. In this research, after performing preprocessing on the NSL-KDD and CSE-CIC-IDS2018 datasets, all the features of both datasets were passed into the genetic algorithm fitness function. The Random Forest (RF) entropy function was utilized in the genetic algorithm fitness function to calculate the fitness values of the features. After the selection of optimal features, these features were again passed to an RF classifier to predict the network attacks. The newly proposed approach had the ability to use RF inside the genetic algorithm. Our main goal was to find the optimal feature subset using the genetic algorithm, and for this, we proposed a new fitness function, which was based on RF;
2. This study also proposed the following weights for the genetic algorithm to obtain the optimal features from the NSL-KDD and CSE-CIC-IDS2018 datasets used in wireless communications systems. The parameters we updated were: SPX-crossover, crossover probability 0.7 and random-init. These were implemented as an initialization operator for the genetic algorithm in this study. Moreover, bit-flip was employed as mutation operator with a mutation probability rate of 0.5 and a population size of 200. The generational parameter was used as the replacement operator; the report frequency was set at 200; and the selection operator was the tournament selection, which was employed in this research;
3. The experimental results upheld the importance of feature optimization, which yielded better results considering the precision, recall, F1-measure, DR and FAR;
4. The study of J. Ren et al achieved a 92.8% accuracy and a 33% false alarm rate with the DO-IDS method [13]. Their findings also indicated an 86.5% accuracy and a 12.4% false alarm rate using the traditional genetic algorithm and RF classifier. However, our proposed model, which was a combination of the genetic algorithm and RF (GA-RF), outperformed these results at 98.81% DR and 0.8% FAR, respectively;
5. The results showed that the proposed GA was greatly optimized, in which the average precision was optimized by 5.65%, and the average F1-score was optimized by 8.2%.

Research Organization

The remaining part of this research is structured as follows: In Section 2, several related literature works are collected, reviewed and analysed in comparison to the current study. The proposed models coupled with the selected datasets are analysed and established in Section 3. Several numerical experiments to prove the performance of the proposed model are performed and discussed in Section 4. The results of the performances are also extensively analysed in this section. Finally, conclusions and findings are outlined in Section 5.

2. Literature Review

Several scholars highlighted the NIDS's issues in the past due to its increased importance in the current era of intelligent cyber-attacks. A mobile agent-based IDS was designed to secure the network of interconnected medical gadgets [14]. Numerous solutions regarding privacy, security, authorization and authentication and making use of blockchain technology for secure data sharing in smart healthcare have been discussed [15]. Smart health solutions have great market potential considering that the expenditure made on them by the EU makes up about 10% of GDP [16]. In the past, various ML methods such as nearest neighbour, RF, logistic regression, Support Vector Machines (SVMs), and *k*-means were widely used to develop the NIDS. Manjula C. Belavagi and Balachandra Muniyal [17], in the measurement of the accuracy of different models on the NSL-KDD dataset, used a supervised learning approach.

They used SVM, RF logistic regression and Gaussian Naive Bayes (NB). An accuracy of 84% was achieved using logistic regression, an accuracy of 79% using Gaussian NB and an accuracy of 75% and 99% by the SVM and RF classifiers, respectively.

Almeida [18] compiled and made available the NSL-KDD dataset to be used for future research in the intrusion detection domain. Chen [19] combined deep learning and the IDS model and proposed a recurrent neural network model for the NIDS. Eduardo DelaHoz et al. [20] detected anomalies using the statistical and self-organizing maps methods. Optimal features were selected using PCA and Fisher′s discriminant ratio method. After removing noise and using probabilistic self-organizing maps, the classification was performed by classifying normal or anomalous traffic. S.K. Wagh [21] proposed various machine learning techniques for the NIDS. The NIDS's performance of C.Qiu [22] was improved by using various feature extraction techniques. Taher et al. [23] used a two-feature selection technique for feature selection. One was the chi-squared, and the other one was correlation based. Seventeen features were selected using the correlation technique, and thirty-five features were selected using the chi-square method. After optimal feature selection, SVM and ANN were applied to the selected optimal feature subset for classification. The overall accuracy with SVM was 82.34%, and with Artificial Neural Networks (ANNs), they achieved a 94.02% classification accuracy. The detection accuracy with SVM and ANN using the correlation method was 81.78% and 94.02%, respectively. With chi-squared using SVM and ANN, they achieved an accuracy 82.34% and 83.68%, respectively.

Mukherjee et al. [24] used the Feature Vitality-Based Reduction Method (FVBRM) for feature selection from the NSL-KDD dataset, and twenty-four optimal features were selected from the dataset out of 41 features. The NB classifier was used for classification with an accuracy of 97.78%. However, they did not perform extensive results to find the precision, recall, F1 scores, DR and FAR. Although NB produces good results with a structured dataset, it has strong feature independence. The NB classifier considers that all the attributes in the dataset are not correlated with each other. The feature removal technique was used by Yinhui and Jingbo [25] for optimal feature selection. Different feature subsets were selected, and the experiment was performed using the KDD99 dataset. For classification, they used SVM. They achieved a 98.62% classification accuracy. They did not find the confusion matrix and other results in the form of the DR and FAR. They also failed to mention the computational cost of the proposed method.

The Cuttlefish Optimization Algorithm (CFA) was used to select an optimal feature subset by Adel Sabry Eesa [26]. The KDD99 dataset was used to perform all the experiments. The detection rate and false positive rate were used as the evaluation measures, and they achieved a DR of 91% and an FPR of 3.91% with five features. Kumar et al. [27] used three hybrid feature selection models, CFS + best-first, gain ratio + ranker and info-gain + ranker. They used a modified version of the NB classier, which had fewer feature independence assumptions. They selected fifteen features from the dataset using three types of hybrid feature selection techniques, and they classified the data using a modified version of the NB classifier. In their research, they achieved a good accuracy of around 94% to 98% for different attacks, but they did not mention false alarms and the detection rate in their research. The PSO algorithm was used by Syarif et al. [28] to select optimal features from the KDDCUP99 dataset after the selection of the 25 best subsets of features, The K-nearest neighbour classifier was used for detection. The ant colony optimization technique was used for optimal feature selection. Different features of the subsets were selected. For the normal class, five features were selected, and for the DoS attack, four features were selected. Four and three features were selected for U2R and R2L, respectively. Eight optimal features were selected for the probe attack. Two datasets, KDD99 and NSL-KDD, were used in the research. An accuracy of 98.80% and a 2.59% false-positive rate were achieved. Accuracies were reported for DoS of 99.78%, U2R of 93.51%, R2L of 99.17%, probe of 74.65% and the normal class of 97.41%, respectively [29]. Likewise, many early studies reported the results of several traditional ML approaches. The main objective of this research was that in network traffic, when we inspected a network packet to determine it to be normal or an anomaly, it took much time to inspect a significant number of attributes. Therefore, through optimized attribute selection, we identified and proposed only those features that played an important role in the intrusion detection system and ignored all irrelevant features. A survey of some of these features and classification is shown in Table 1. This had the practical implication of using fewer system resources with an increased DR and a reduced FAR.

Table 1. Survey on feature selection and classification techniques.

Author	Year	Feature Selection Approach	Feature Type	Dataset	Classifier
Mahmood et al. [30]	2016	Information Gain Genetic Algorithm	Single	NSLKDD	Naive Bayes
K. Rai et al. [31]	2016	Information Gain	Single	NSLKDD	Decision Tree
Thaseen et al. [32]	2016	PCA	Single	NSLKDD UNSW-NB	SVM LDC QDC WMA
M. Ambusaidi et al. [33]	2016	FMIFS	Single	KDD99 NSLKDD Kyoto2006	LSSVM
Bamakan et al. [34]	2016	TVCPSO	Single	NSLKDD	SVM
Thaseen et al. [35]	2017	Chi	Single	NSLKDD	SVM
Pajouh et al. [36]	2017	-	-	NSLKDD	Deep Learning
Shone et al. [37]	2018	-	-	NSLKDD	RNN
Naseer et al. [38]	2018	-	-	NSLKDD	LSTM
Woo et al. [39]	2019	Correlation Method	Ensemble	NSLKDD	NN

3. Proposed Methodology

This section outlines the methodology used in realizing our proposed optimization method. The workflow of the proposed scheme is shown in Figure 1. For feature optimization, a weighted genetic algorithm and RF were integrated to produce an optimal subset of features that could be used to achieve a high detection rate and a low false alarm rate. Classification was performed on RF, NB and logistic regression. The NSL-KDD dataset was used in this research, which is a benchmark dataset for IDS, mostly used in recent studies. This research had 6 phases. Phase 1 was the data collection. Phase 2 was the data pre-processing. Missing and duplicates values were replaced and removed, respectively. Outliers were also checked. To bring a dataset down to one standard scale, data normalization was performed using min-max normalization. Data encoding was also performed. Phase 3 was a feature optimization step where the weighted genetic algorithm was used with RF. Phase 4 was the splitting of the data into two sets for training and testing. In Phase 5, different machine learning classifiers were utilized for classification, which included RF, NB and logistic regression. Phase 6 was a comparison of our proposed model with existing research, as shown in Figure 1.

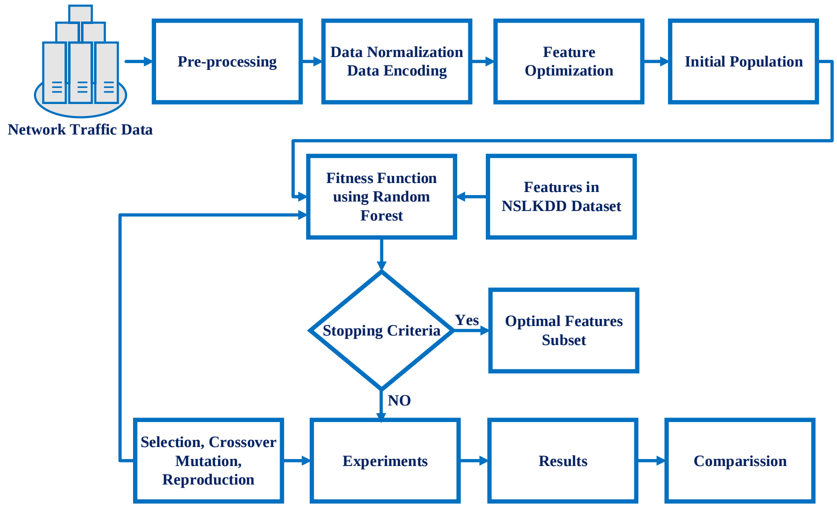

Figure 1. Proposed algorithm workflow.

3.1. NSL-KDD Dataset

Intrusion detection systems are those systems that can detect abnormal traffic over the Internet and are trained on information from Internet traffic records. The most famous dataset for the IDS is NSL-KDD (https://www.unb.ca/cic/datasets/vpn.html (accessed 12 January 2021)), and it is used as a benchmark for modern-day network traffic. This dataset contains 43 attributes per file, and out of the 43 attributes, forty-one features are on traffic input, while two characteristics indicate whether the packet is standard or malicious. The malicious class has 4 different types of attacks in it. They include probe, Remote to Local (R2L), Denial-of-Service (DoS) and User to Root (U2R). In a DoS attack, the hacker tries to stop the flow of network traffic to and from the target system. A considerable amount of abnormal packets are sent towards the victim system, and the IDS is flooded with these malicious packets. As a result, the system fails to handle the vast amounts of packets and shut downs to protect itself. When the system shuts down, normal traffic is also disturbed, and a user cannot access anything over the network In the probe attack, all the information is gathered from the web. The purpose of a probe attack is to act as a thief and collect useful information about a person or bank, then launch the attack. In the U2R attack, the attacker gains access as a normal user and then tries to access the system or network as an administrator. The attacker takes advantage of vulnerabilities inside the system to gain root access. In the R2L type of attack, the attacker tries to gain local access to a remote machine. The attacker does not have the right to access the local system/network, so he/she decides to hack the system to enter the system and perform the desired operations. The total size of the dataset is 148,517 with 43 features. There are 4 different types of features inside the dataset, and they are categorized as below:

- 10 continuous (features: $1, 5, 6, 10, 11, 13, 16, 17, 18, 19$);
- 4 categorical (features: $2, 3, 4, 42$);
- 23 discrete (features: $8, 9, 15, 23$ to $41, 43$);
- 6 binary (features: $7, 12, 14, 20, 21, 22$).

There are 3 possible values for the protocol type, 60 possible values for service and 11 possible values for the flag. In this research, different pre-processing methods were applied to the NSL-KDD dataset to clean the dataset and obtain optimal results. For this purpose, missing values were checked whether they existed or not. Then, features were brought into one order (features were normalized using the min-max technique. Outliers were also tested in this study. The last step of pre-processing in this research was feature encoding to change the categorical variables in the continuous process. Next was the optimal feature selection using the genetic algorithm and then passing the optimal feature set to the machine learning model to obtain the results. The dataset was divided into a 70:30 training and testing ratio.

3.2. Data Normalization (Min-Max Method)

Min–max normalization is the most popular and state-of-the-art method to scale down the features between 0 and 1. For every attribute, minimum and maximum values are transformed between 0 and 1, respectively. The min-max equation is given below in Equation (1):

$$z_i = \frac{D_i - \min(D)}{\max(D) - \min(D)} \tag{1}$$

Z_i and $\mathrm{X} = (\mathrm{x}_1, \ldots\ldots, \mathrm{x}_\mathrm{n})$ were now the normalized data, while the min and max were the range that we wanted to set for scaling (M) in this research: min was 0, and max was 1.

3.3. Optimal Feature Selection Using Evolutionary Search

Genetic Algorithms (GAs) are part of evolutionary computation, a field of artificial intelligence that is rapidly growing. Genetic algorithms are optimization and searching algorithms based on the principle of natural selection and genetics [40]. A chromosome population in GAs shows candidate solutions to the problem. Every chromosome has fixed-length bits. The primary chromosome population is formed by randomly distributing 1s and 0s. Values between 0 and 1 are assigned to different distributions. Chromosomes in this encoding scheme are bits of a string (1 s and 0 s) whose length is determined by the number of principal components in the main space. Each chromosome suggests a solution for the candidate or a subset of significant elements. The population grows by using genetic operators to look for the optimal solution. The flow of GA for feature selection is shown in Figure 2.

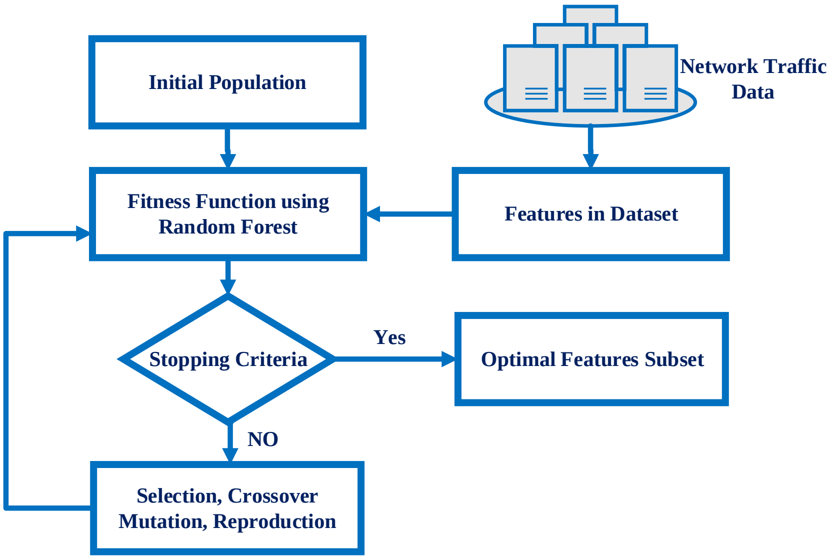

Figure 2. Genetic algorithm workflow.

In the above genetic algorithm (Algorithm 1) , we calculated the fitness function by using RF conditional entropy, as shown in the equation:

$$\text{Fitness } (G|a) = \text{sum } x\text{in a}Ga(x)/G * H(Ga(x)) \tag{2}$$

where $Ga(x)/S$ is the relationship among the number of entities in the dataset, vector a has the value x and $H(Ga(x))$ is the entities' group entropy where the variable has the value x. The other parameters we updated were: SPX-crossover, crossover probability 0.7 and random-init, used as an initialization operator for the genetic algorithm in this study. Moreover, bit-flip was used as the mutation operator; the mutation probability was 0.5; the population size was 200; generational was used as the replacement operator; the report frequency was 200; and the selection operator was tournament selection.

Algorithm 1: Pseudocode of the genetic algorithm.

```
Begin
Set parameters
Generation of initial population
while $i$ <Max iteration and optimal fitness <max fitness do
    Fitness calculation using RF entropy function
    Selection
    Crossover
    Mutation
    End while loop
end
Return the optimal features subset
End
```

3.4. Machine Learning Algorithms

The advantages of machine learning algorithms are numerous. They have been used by several researchers in several domains such as healthcare, banking, finances, the stock market, intrusion detection, etc. Several scholars highlighted NIDS's issues in the past due to its increased importance in the current era of intelligent cyber-attacks. In the past, various ML methods such as nearest neighbour, RF, logistic regression, support SVM and *k*-means were widely used to develop the NIDS. In this study, the following machine learning classifiers were used.

- Random forest:
  Multiple decision trees are combined to build an RF classifier. The purpose of the RF classier is that it assembles numerous decision trees to give more meaningful and precise results. For regression, it calculates the mean of every decision tree and assigns the mean value to the predicted variable. For classification cases, the RF employs a majority vote approach. For example, if 3 trees predicated yes and 2 trees predicated no, then it will assign yes to the predicated variable. Entropy and information gain are used for deciding the root or parent node of the tree [41] and give the following known equations.

$$\text{Entropy:H}(x_1, x_2, \ldots\ldots, x_z) = \sum_{i=1}^{s} x_i \log \frac{1}{x_i} \tag{3}$$

  where $x_1, x_2, \ldots, x_n$ represents the probabilities of the class labels:

$$\text{Information Gain } (I, S) = H \sum_{i=1}^{s} X(l_i)H(l_i) \tag{4}$$

- Logistic regression:
  Another popular and well-known machine learning algorithm used for classification is logistic regression. Typically, logistic regression produces more dichotomous results. The working methodology of logistic regression is that it finds a correlation among final output values and also provides the characteristics. In logistic regression, the log odds function is used for prediction.
- Naive Bayes:
  NB is a supervised machine learning technique. NB is a probability machine learning model used for classification. In this research, we used a traditional NB classifier for the predication of various attacks.

3.5. Evaluation Metrics

Various performance metrics as shown from the equations below can be applied to evaluate the proposed solution, including the accuracy, recall, F1-measurement, false alarm rate (FAR) and detection rate (DR). The above performance measurements were based on True Positives (TPs), False Positives (FPs), False Negatives (FNs), and True Negatives (TNs). The FAR is a combination of the total instances that are normal, but classified as the attack class.

$$\text{False Alarm Rate } (FAR) = \frac{FP}{FP + TN} \tag{5}$$

$$\text{Accuracy } = \frac{TP + TN}{TP + FP + FN + TN} \tag{6}$$

$$\text{Detection Rate } (DR) = \frac{TP}{TP + FN} \tag{7}$$

$$\text{Precision } = \frac{TP}{TP + FP} \tag{8}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

$$F1 - \text{Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

4. Experiments and Results

4.1. Random Forest Attack Experiment Results

The red curve represents the training and testing scores, while the green curve represents the cross-validation scores. Figure 3 depicts the training learning curve for the RF, while Figure 4 represents the testing curve for the RF. From Figure 3, we can see that the training rate for RF classifiers was 99.5%, and the validation score for RF was 98.75%. Figure 4 shows that the testing rate for the RF classifiers was approximately 99%, while the validation score for testing was 98.5%.

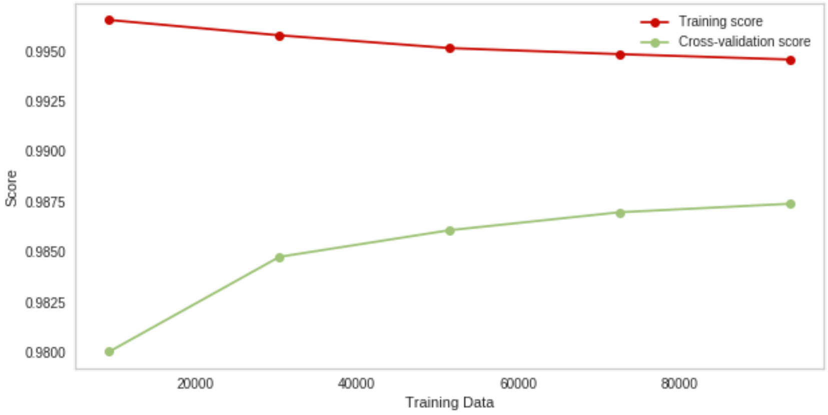

Figure 3. Random forest training learning curve.

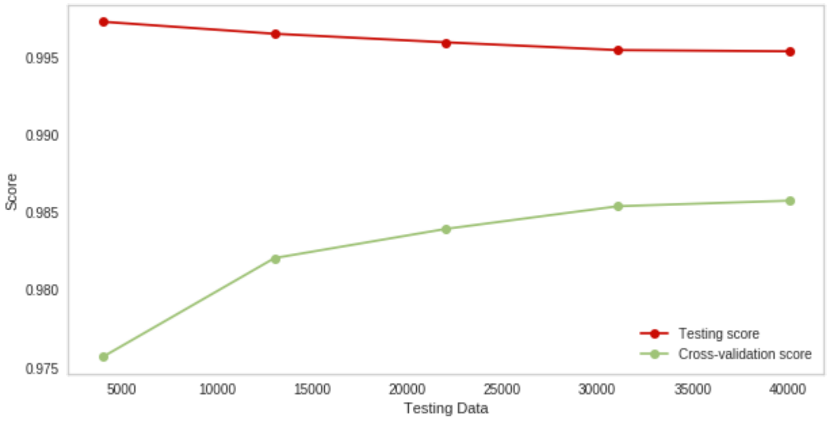

Figure 4. Random forest testing learning curve.

Figure 5 represents the confusion matrix for RF. The prediction was made on five classes. Four classes contained attacks, and one class was normal. From Figure 6, we can see that the detection rate for the DoS attack was 99.7%, which means that out of 16,039 DoS attacks. there were 15,984 attacks detected correctly by our proposed model, which are quite high numbers. Ninety-nine percent of probe attacks were identified correctly; a total of 4209 packets had probe attacks in them, and four-thousand one-hundred sixty-six packets were correctly detected as probe attacks. Similarly, for the R2L and U2R attacks, out of 1192 packets, one-thousand twenty-six packets were detected correctly for the R2L attack, and eight packets were identified correctly out of 28 for the U2R attack. Ninety-six-point-eight percent of packets were detected correctly as normal packets that had no anomaly. Out of 23,087 packets, twenty-two-thousand eight-hundred and four packets were identified correctly with a percentage of 98.8%. The reason for the low detection for U2R was because it had a very low amount of packets in it, and it was mostly machine learning that performed very well when it had more data for training. It could be tested on new data when we compared DoS and U2R, and we can see that DoS had 16,039 packets, while U2R only had 28 packets. That was the reason for its low performance.

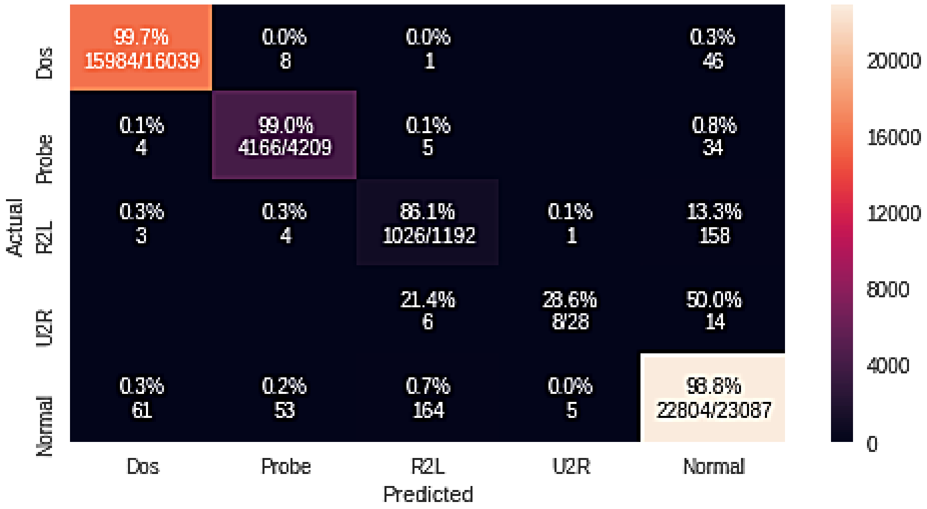

Figure 5. Random forest testing learning curve.

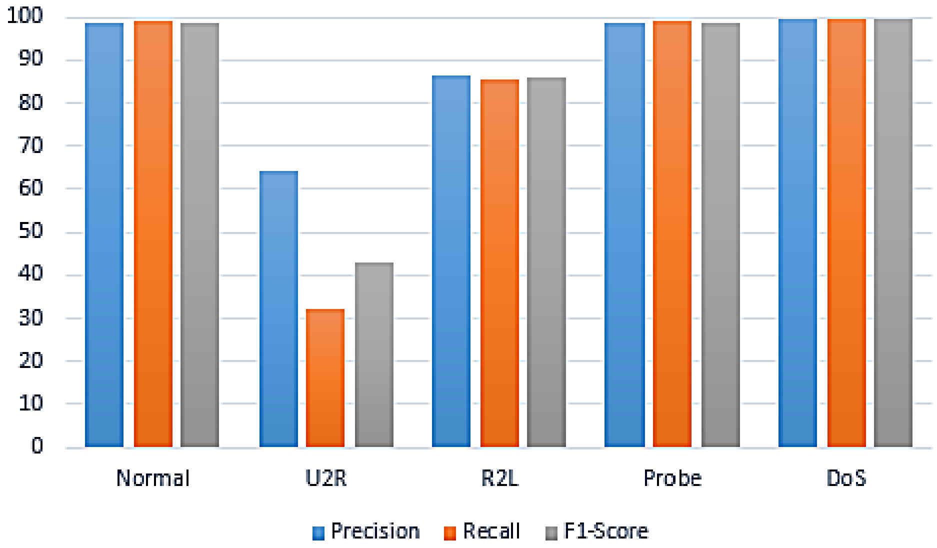

Figure 6. Classification report of random forest.

Figure 6 represents the classification report for various attacks mentioned in this research. For normal packets, precision was 98.8%; for U2R and R2L, the precision scores were 64.30% and 86.50%, respectively. The probe attack precision score was 98.70%, and the DoS attack had the highest precision score of 99.6%. Recall and the F1-measure for the normal packets were 98.90% and 98.80%, respectively. Recall for both U2R and R2L were 32.1% and 85.30%, respectively. The F1-measure for U2R was 42.90% and for R2L was 85.90%, respectively. The probe attack had a 98.90% recall and a 98.80% F1-measure. The DoS attack had a 99.6% recall and a 99.6% F1-measure, respectively.

4.2. Naive Bayes Classifier Experimental Results

The training curve for NB is represented in Figure 7. The training score for NB started from 83.31% and went to 83.45%, while the cross-validation score started from 83.41% for NB and went to 83.44%, respectively.

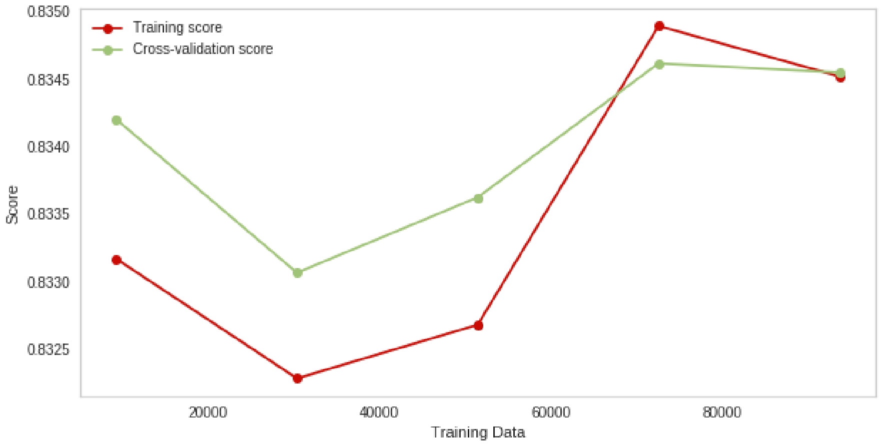

Figure 7. Naive Bayes training learning curve.

In Figure 8, the red line represents the testing, while the green line represents the cross-validation score for the NB algorithm. The testing score of NB was 83.45%, and the validation score of NB was 83.44%.

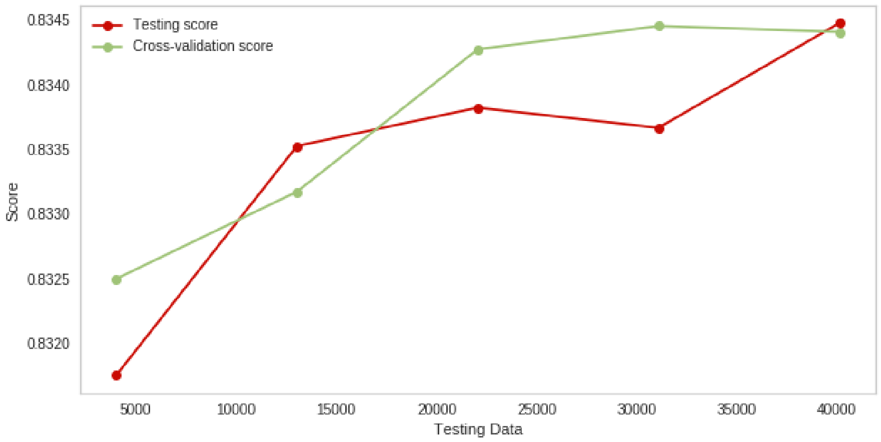

Figure 8. Naive Bayes testing learning curve.

Figure 9 represents the confusion matrix for NB. From Figure 9, we can see that the detection rate for the DoS attack was 86.3%, which means that out of 16,039 DoS attacks, there were 13,849 attacks detected correctly. Fifty-seven-point-one probe attacks were detected correctly using NB classifiers. A total of 4209 packets had probe attacks in them. Two-thousand four-hundred and eight packets were correctly identified as probe attacks. Similarly, for the R2L and U2R attacks out of 1192 packets, no packet was detected successfully for the R2L attack, and fourteen packets were detected correctly out of 28 packets for the U2R attack. Ninety-point-seven percent of packets were identified correctly as normal packets with no anomaly in them, which means that 20,933 out of 23,087 packets were detected successfully with a 90.7% accuracy for the normal class. NB achieved very low per-class accuracy for R2L and U2R, and the reason for this was that both had low data for the training set and testing set compared to the DoS, probe, and normal classes. These affected the testing accuracy for R2L and U2R.

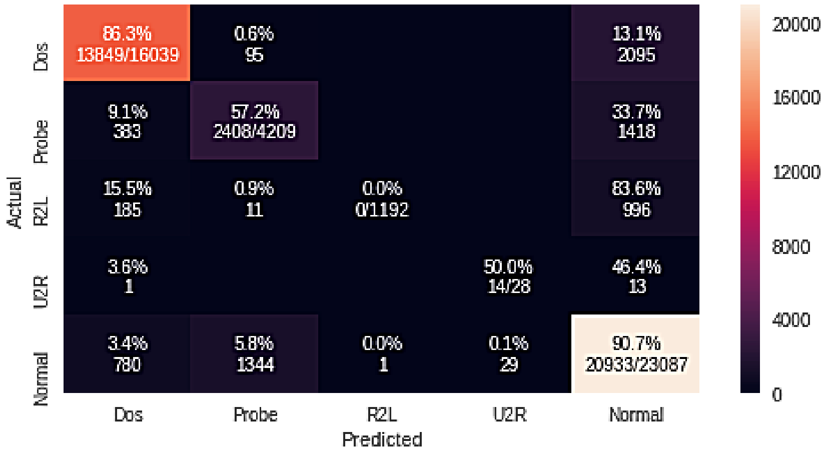

Figure 9. Naive Bayes confusion matrix.

Figure 10 represents the classification report for NB. For normal packets, the precision was 82.20%. Similarly, for U2R and R2L, the precision scores were 32.60% and 0%, respectively. The probe attack precision score was 62.40%, and the DoS attack had a precision score of 91.10%. The recall and F1-measure for the normal packet were 90.70% and 86.20%, respectively. Recall for both U2R and R2L were 50.00% and 0%, respectively. The F1-measure for U2R was 39.40% and for R2L was 0.0%, respectively. The probe attack had a 57.20% recall and a 59.70% F1-measure. The DoS attack had an 86.30% recall and an 88.70% F1-measure, respectively.

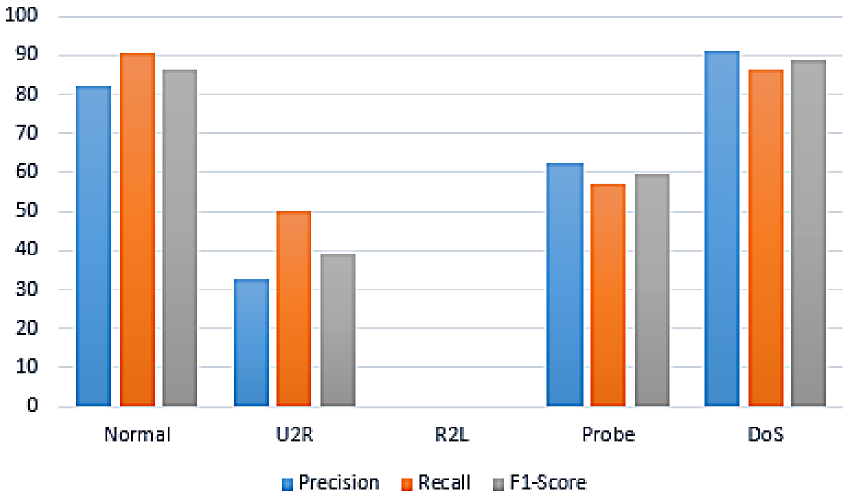

Figure 10. Classification report of Naive Bayes.

4.3. Logistic Regression Classifier Experiment Results for Multiple Attacks

In Figures 11 and 12, red lines represent the training and testing scores for logistic regression, while the green curve represents the cross-validation scores. Figure 11 depicts the training learning curve for logistic regression, while Figure 12 represents the testing curve for logistic regression. From Figure 11, we can see that the training rate for the logistic regression classifiers was 88.70%, and the validation score for logistic regression was also 88.70%. Figure 12 shows that the testing rate for the logistic regression classifier was approximately 88.47%, while the validation score for testing was 88.42%, respectively.

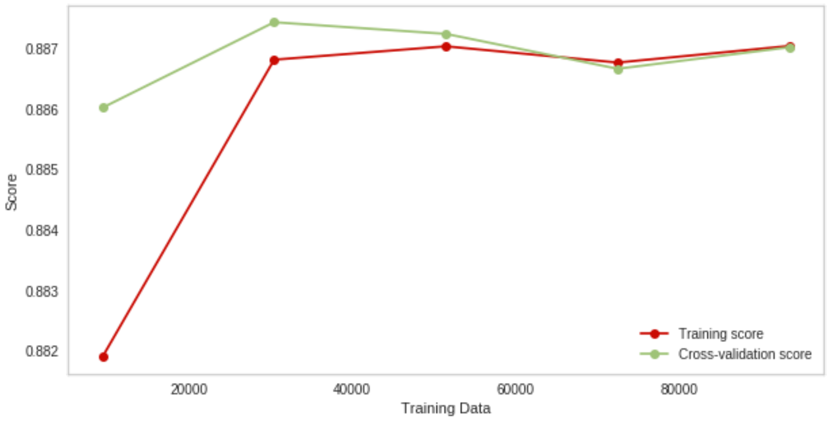

Figure 11. Logistic regression training learning curve.

Figure 13 represents the confusion matrix for logistic regression. From Figure 13, we can see that the detection rate for the DoS attack was 90.10%, which means that out of 16,039 DoS attacks, there were 14,456 attacks detected correctly. Seventy-seven-point-six percent of probe attacks were identified correctly; a total of 4209 packets had probe attacks in them, and three-thousand two-hundred sixty-five packets were correctly identified as probe attacks. Similarly, for R2L attacks, out of 1192 packets, two-hundred and one packets were detected successfully. Ninety-three-point-seven percent of packets were detected correctly as normal packets with no anomaly in them. There were 21,622 packets detected correctly out of 23,087, which was a high ratio for detection.

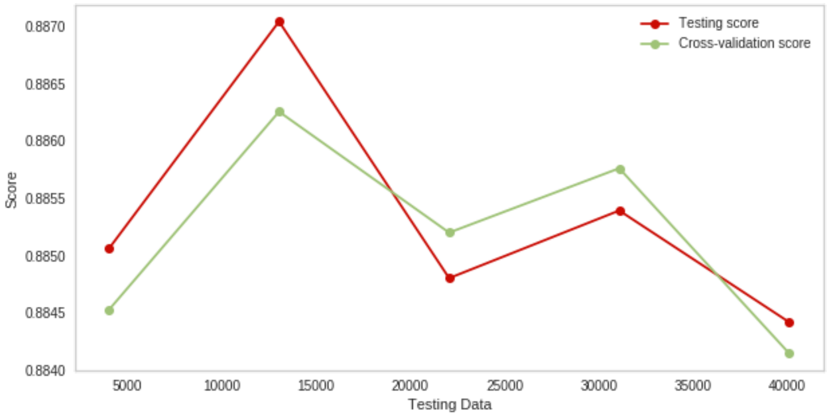

Figure 12. Logistic regression testing learning curve.

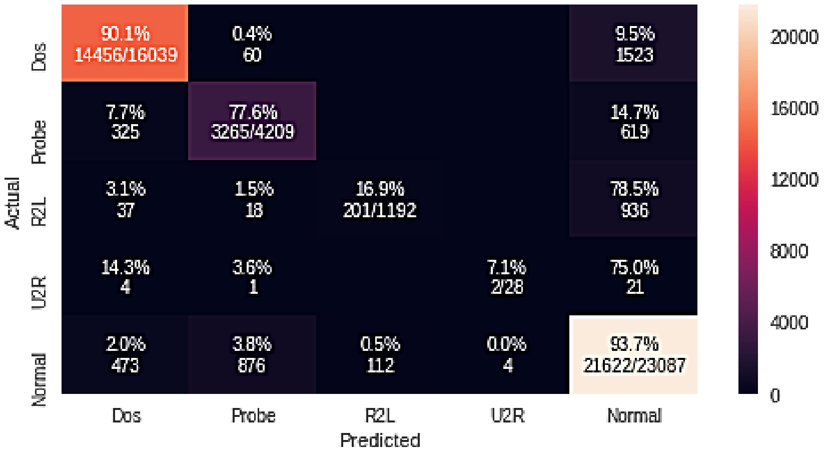

Figure 13. Logistic regression confusion matrix.

Figure 14 represents the classification report for logistic regression. For normal packets, the precision was 87.5%, and for U2R and R2L, the precision scores were 33.3% and 64.2%, respectively. The probe attack precision score was 77.4%, and the DoS attack had a precision score of 94.5%. The recall and F1-measure for the normal packet were 93.7% and 90.5%, respectively. The recall for both U2R and R2L were 7.10% and 16.9%, respectively. The F1-measure for U2R was 11.80% and for R2L was 26.70%, respectively. The probe attack had a 77.60% recall and a 77.50% F1-measure. The DoS attack had a 90.10% recall and a 92.30% F1-measure, respectively. We show below in Tables 2–4 the average detection rate and the training and testing accuracies of the different proposed models.

Table 2. Average detection rate of different proposed models.

Classifier Name	DoS	Probe	R2L	U2R	Normal
Random Forest + GA	99.7	99.0	86.1	28.6	98.8
Naive Bayes + GA	86.3	57.2	0.0	50.0	90.7
Logistic Regression + GA	90.1	77.6	16.9	7.1	93.7

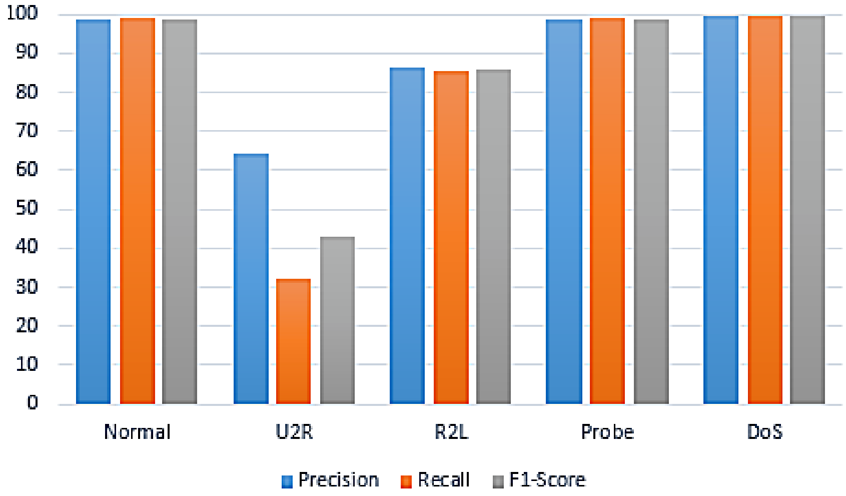

Figure 14. Logistic regression classification report.

Table 3. Average training and testing accuracies of the proposed models.

Classifier Name	Training Accuracy	Testing Accuracy
Random Forest + GA	99.4	98.7
Naive Bayes + GA	83.4	83.5
Logistic Regression + GA	88.8	88.7

Table 4. Average classification report comparison of the proposed models.

Classifier	Attacks	Precision	Recall	F1-Measure
RF + GA	DoS	99.60	99.60	99.60
	Probe	98.70	98.90	98.80
	R2L	86.50	85.30	85.90
	U2R	64.30	32.10	42.90
	Normal	98.80	98.90	98.80
NB + GA	DoS	91.10	86.30	88.70
	Probe	62.40	57.20	59.70
	R2L	0.0	0.0	0.0
	U2R	32.60	50.00	39.40
	Normal	88.20	90.70	86.20
LR + GA	DoS	94.50	90.10	92.30
	Probe	77.40	77.60	77.50
	R2L	64.20	16.90	26.70
	U2R	33.33	7.10	11.80
	Normal	87.50	93.70	90.50

4.4. Binary Class Experimental Results

4.4.1. Random Forest Classifier Experiment Results for the Binary Class

Figures 15 and 16 represent the RF binary class training learning curve and testing learning curve. From Figure 15, we can see that the training rate for the RF classifiers was99.70%, and the validation score for RF was 99.00%. Figure 16 shows that the testing rate for the RF classifiers was approximately 99.75%, while the validation score for testing was 99.00%, respectively.

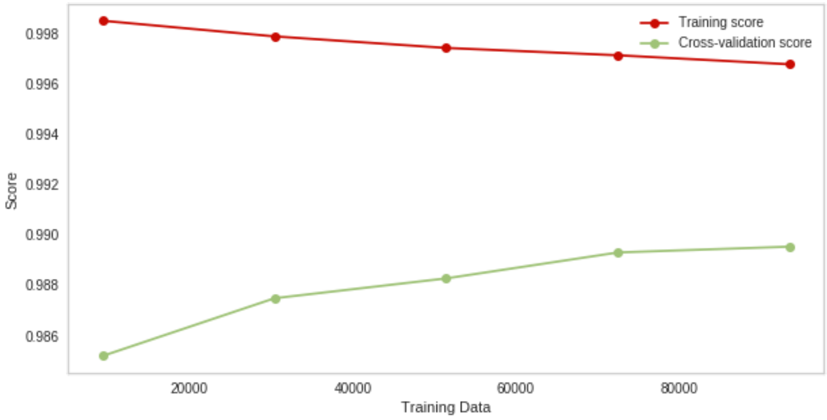

Figure 15. Binary class random forest training and validation learning.

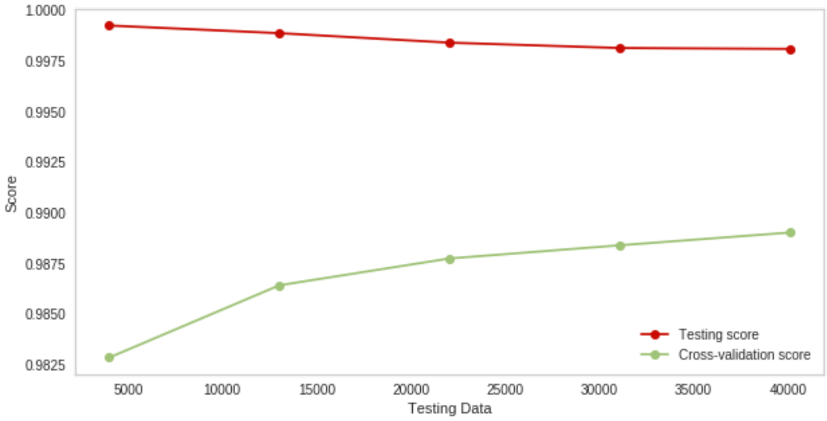

Figure 16. Binary class random forest testing and validation learning.

From Figure 17, we can see that 99.10% of attacks were detected correctly; out of 21,468 packets, there were 21,274 packets identified correctly as anomaly packets. There were 194 packets detected as normal packets, but in reality, they were anomaly packets. Similarly, there were 256 packets detected as anomaly packets, but they were normal packets. Ninety-eight-point-nine percent of packets were correctly identified as normal packets. Out of 23,087 normal packets, there were 22,831 packets detected successfully.

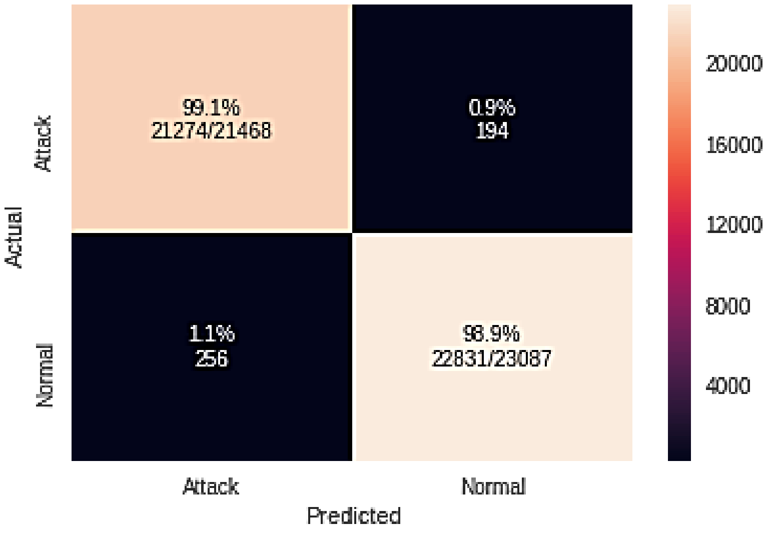

Figure 17. Random forest classification report.

Figure 18 represents the RF classification report for the binary class. For the normal class, the precision was 99.10%, recall 99.10%, and F1-measure also 99.10%, respectively. The precision, recall, and F1-measure scores for the attack class were 99.00%, 99.00%, and 99.00%, respectively.

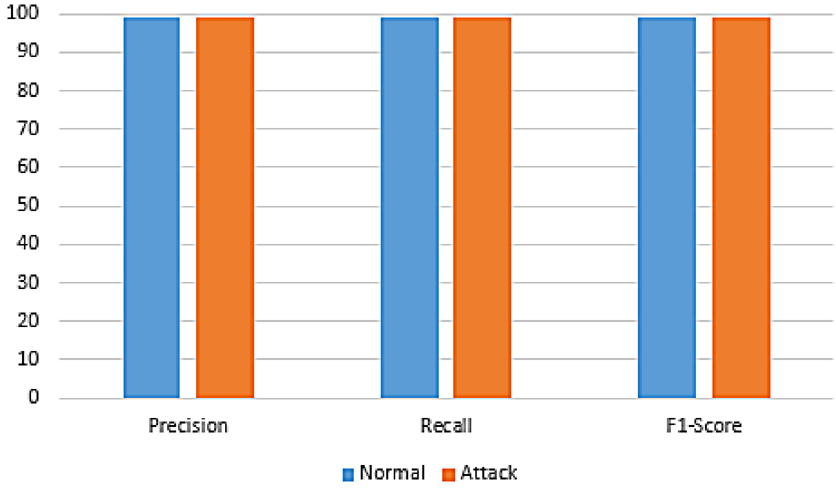

Figure 18. Random forest binary class classification report.

4.4.2. Naive Bayes Experiment Results for the Binary Class

The training curve for the binary class with NB is represented in Figure 19. The training and cross-validation score for NB was more than 88%.

Figure 20 shows the binary class testing and validation curves for NB. Both had the same scores for training and validation, which was 86.18%, respectively.

Figure 21 represents the binary class confusion matrix for the NB classifier. It can be seen from Figure 21 that 80.8% of attacks were detected correctly. Out of 21,468 packets, there were 17,345 packets detected correctly as anomaly packets. There were 4123 packets identified as normal packets, but in reality, they were anomaly packets. Similarly, there were 1887 packets detected as anomaly packets, but in reality, they were normal packets. Likewise, ninety-one-point-eight percent of packets were correctly recognized as normal packets. Out of 23,087 normal packets, there were 21,200 packets identified correctly.

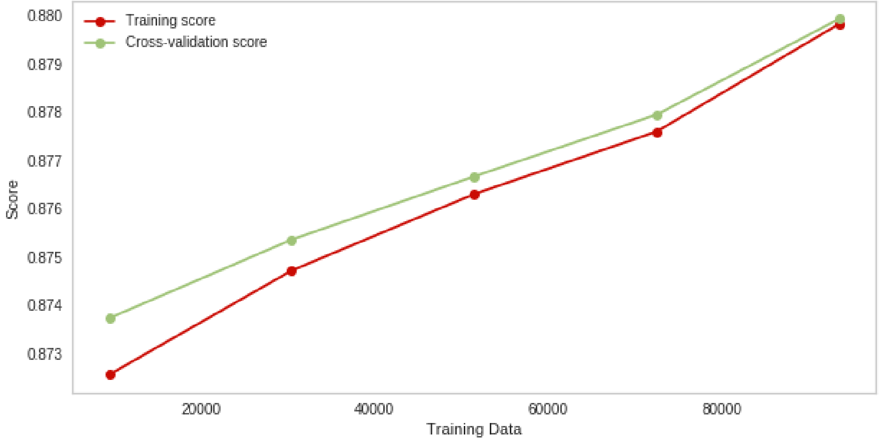

Figure 19. Binary class naive Bayes training and validation learning curve.

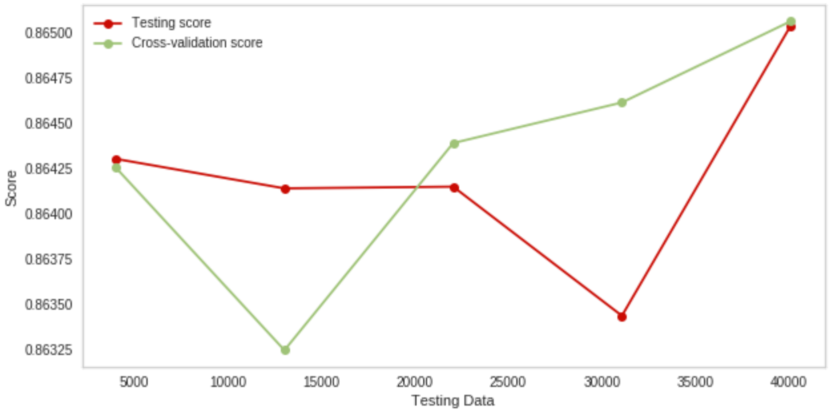

Figure 20. Binary Class naive Bayes testing and validation learning curve.

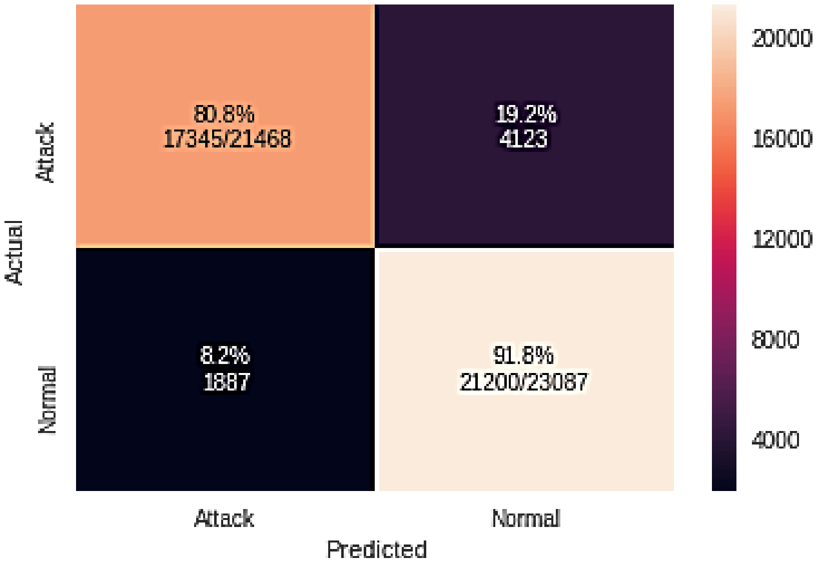

Figure 21. Naive Bayes binary class confusion matrix.

Figure 22 represents the normal class with 83.7% precision, 91.8% recall and an 87.6% F1-measure. Precision for the attack class was 90.2%, recall 80.8% and F1-measure 85.2%, respectively.

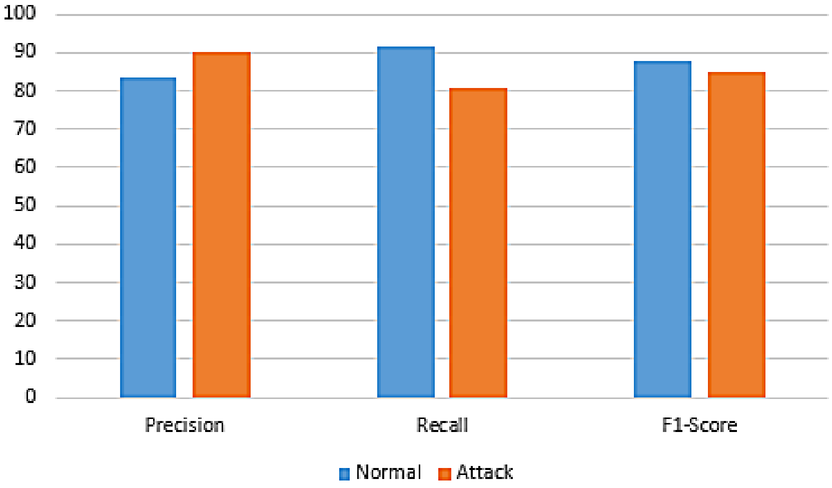

Figure 22. Naive Bayes class classification report.

4.4.3. Logistic Regression Experiment Results for the Binary Class

In Figure 23, the green curve represents the validation curve for logistic regression, and its score was 89.45%, while the red curve represents the training score, which was 89.50%.

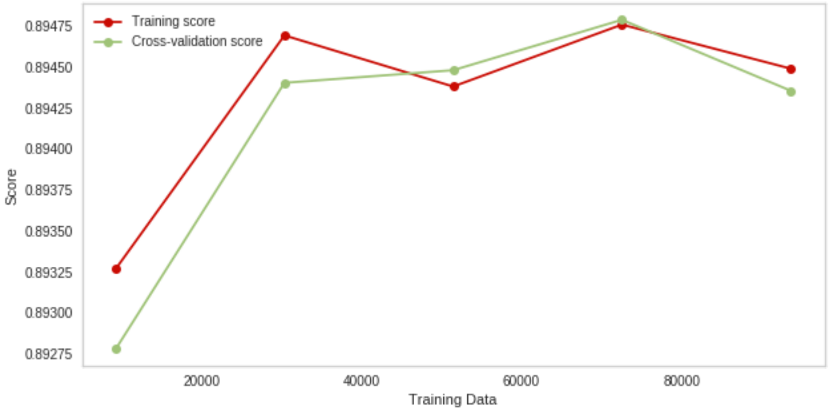

Figure 23. Binary class logistic regression training and validation learning curve.

From Figure 24, it can be seen that the testing score for logistic regression started from 89.60% and went down to 89.55%. Similarly, the cross-validation score began at 89.45% and went up to 89.55%.

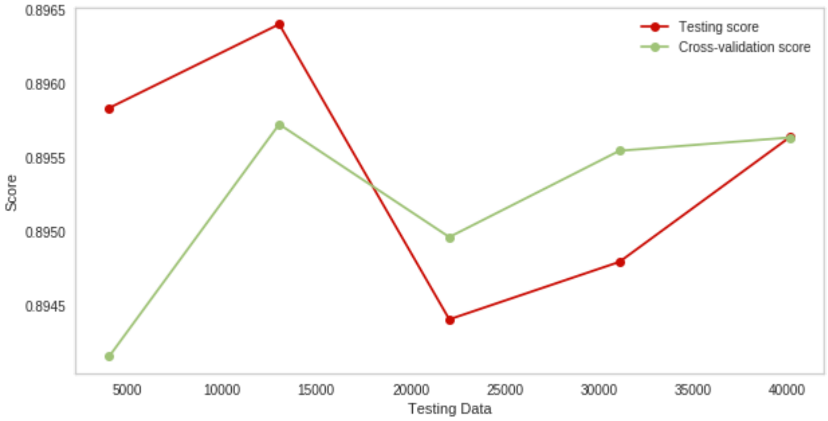

Figure 24. Binary class logistic regression testing and validation learning curve.

Figure 25 represents the binary class confusion matrix for the logistic regression classifier. It can be seen from Figure 25 that 87.3% of attacks were detected correctly; out of 21,468 packets, there were 18,732 packets detected correctly as anomaly packets. There were 2736 packets identified as normal packets, but in reality, they were anomaly packets. Similarly, there were 1793 packets detected as anomaly packets, but in reality, they were normal packets. Ninety-two-point-two percent of packets were correctly detected as normal packets. Out of 23,087 normal packets, there were 21,294 packets identified correctly.

Figure 26 represents the logistic regression classification report for the binary class. For the normal class, the precision was 88.10%, recall 92.40% and F1-measure also 90.20%, respectively. The precision, recall and F1-measure scores for the attack class were 91.40%, 86.60% and 88.90%, respectively. Table 5 shows the average comparison of the classification reports of the proposed model, while Table 6 shows the average comparison of the confusion matrices for the proposed models.

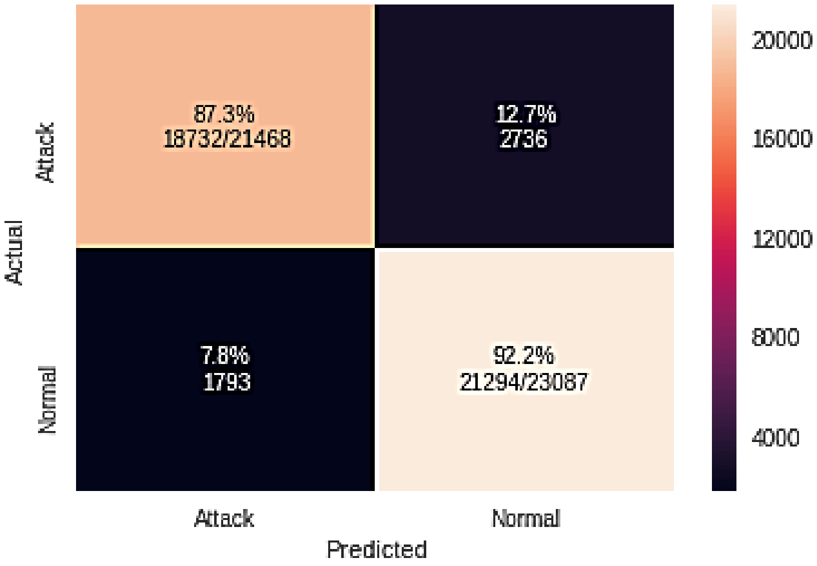

Figure 25. Logistic regression binary class confusion matrix.

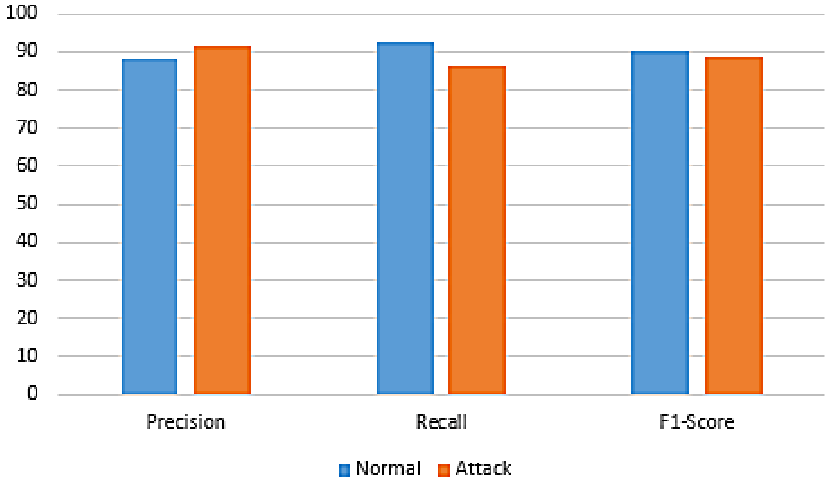

Figure 26. Logistic regression binary class confusion matrix.

Table 5. Average classification report comparison of the proposed models.

Classifier Name	Class Label	Precision	Recall	F1-Measure
GA + RF	Normal Attack	99.10 99.00	99.10 99.00	99.10 99.00
GA + NB	Normal Attack	83.70 90.20	91.80 80.80	87.60 85.20
GA + LG	Normal Attack	88.10 91.40	92.40 86.60	90.20 88.90

Table 6. Average comparison of the confusion matrix for the proposed models.

Classifier Name	TP	FN	FP	TN
GA + RF	21274	256	194	22,831
GA + NB	17345	1887	4123	21,200
GA + LG	18732	1973	2736	21,294

From Table 7, we can conclude that our proposed GA-RF model achieved a high detection rate and false alarm rate. The GA-RF model outperformed the other two proposed models. The DR and FAR for the GA-RF model were 98.81% and 0.8%, respectively.

Table 7. Average comparison of different evaluation metrics of the proposed models.

Model	Training Accuracy %	Testing Accuracy %	FAR	DR %
GA + RF	99.73	99.10	0.008	98.81
GA + NB	88.00	86.50	00.16	90.18
GA + LG	89.65	91.31	00.11	90.47

From Table 8, we can conclude that the performance of RF was evident, so we applied the weighted genetic algorithm to optimize our feature selection techniques further and then passed these best feature subsets to RF for classification; as a result, our proposed model outperformed other similar methods for the IDS, and we achieved a high DR and FAR of 98.81% and 0.8%, respectively.

Table 8. Comparison of our proposed model with similar approaches used in other research.

Authors	Detection Rate %	FAR %
Mahmood et al. [30]	98.00	1.96
Bamakan et al. [34]	97.03	0.87
Pajouh et al. [36]	84.86	4.86
Shone et al. [37]	85.43	4.84
Woo et al. [39]	98.02	NA
H. H. Pajouh et al. [36]	81.97	5.44
Muhammad Shakil Pervez et al. [42]	82.37	15.00
Navaneeth et al. [43]	85.05	12.20
MR Gauthama et al. [44]	97.14	0.83
Kuang et al. [45]	95.26	1.03
Singh et al. [46]	97.67	1.74
De la Hoz et al. [47]	93.4	14
Tavallaee et al. [48]	80.67	NA
Tsang et al. [49]	92.76	NA
Kayacik et al. [50]	90.60	1.57
Gauthama et al. [51]	97.56	1.22
Proposed Model	98.81	0.80

4.5. Discussion

The experiments were performed on the NSL-KDD dataset for a two-class training set and a five-class training set separately. Three classifiers were used for classification, and a genetic algorithm was used for optimal feature selection. The DoS and normal classes had a high amount of packets in them, while R2L, U2R and probe had fewer packets. Since the DoS attack had more packets, it outperformed the other attacks. R2L, U2R and probe had a very low amount of packets in them, so it was a challenge to achieve a high detection rate for these attacks. To achieve this, we selected optimal features to achieve high detection accuracy. Our proposed model not only achieved high accuracy, but it also achieved a very promising FAR. Table 5, indicates that the GA-RF model outperformed the other two models in terms of the precision, recall and F1-measure. For the normal class GA-RF, the model had a 99.10% precision, 99.10% recall and 99.10% F1-measure, respectively. For the attack class, it had a 99.00% precision, 99.00% recall and 99.00% F1-measure, respectively. The GA-NB model had a 83.70% precision, 91.80% recall and 87.60% F1-measure for the normal class and attack class, and it had a 90.20% precision, 80.80% recall and 85.20% F1-measure, respectively.

The proposed GA-LG model performed well as it had an 88.10% precision, 92.40% recall and 90.20% F1-measure for a normal class. For the attack class GA-LG model, it had a 91.40% precision, 86.60% recall and 88.90% F1-measure. Similarly, Table 6 presents different models used in this research and there true-positive, true-negative, false-positive and true-negative values. For GA-RF, out of 21,468 packets, there were 21,274 packets detected correctly as TPs. There were 256 packets detected as FNs, and similarly. there were 194 packets detected as FPs, but in a reality, they were normal packets. Out of 23,087 normal packets, there were 22,831 packets correctly identified as TNs. Similarly, for the GA-NB model, there were 17,345 packets detected as TPs, and there were 1887 and 4123 packets detected as FNs and FPs, respectively. There were 21,200 packets detected as TNs. For GA-LG, there were 18,732 packets detected as TPs, 1973 packets as FNs, similarly, 2736 packets as FP and 21,294 packets as TNs, respectively. Table 7 shows that our proposed GA-RF model achieved a high detection rate and false alarm rate. The GA-RF model outperformed the other two proposed models. The DR and FAR for the GA-RF model were 98.81% and 0.8%, respectively. The reason for achieving such a high accuracy and low FAR was the optimal feature selection and RF, which is a bagging technique and outperforms other traditional machine learning algorithms.

Our investigation also analysed the performance of the classifier with regard to the training set, which was picked with diverse deflation factors and setting the scaling factor as M. This paper examined the F1-score and precision performance as compared to the NSL-KDD and CSE-CIC-IDS2018 datasets. Thus, we explored the security performance on the NSL-KDD dataset, which guaranteed that the data selection was valuable and devoid of extreme noise. The results indicated that the classifiers attained an outstanding average performance at $M = 60$. Similarly, on the CSE-CIC-IDS2018 dataset, the classifiers attained exceptional average performance $M = 10$. Consequently, with respect to the average F1-score, on the NSL-KDD dataset, $M = 60$ was utilized as the performance scaling factor, where the complex models in the normal class with DoS and probe were flattened, and the complex models in R2L and U2R were amplified with the available data. $M = 20$ was utilized as the scaling factor on the CSE-CIC-IDS2018 dataset, while a comparable action was executed on the NSL-KDD dataset for the complex models. After the execution, we generated the new training set and present the results in Table 9.

Table 9. The newly generated training set class distribution treated by the proposed genetic algorithm.

Dataset	Type	Original Training Set	Realized Training Set
NSL-KDD ($M = 60$)	Normal	78,232	70,803
	DoS	54,816	51,536
	R2L	878	13,792
	Probe	9767	6459
	U2R	85	3751
CSE-CIC-IDS2018 ($M = 20$)	Benign	31,000	26,719
	Bot	14,000	16,329
	DDoS attack-LOIC-UDP	1273	2600
	DDoS attack-HOIC	15,000	14,980
	DDoS attacks-LOIC-HTTP	15,000	13,901
	DoS attacks-GoldenEye	15,000	13,332
	DoS attacks-Hulk	15,000	14,721
	Dos attacks-Slowloris	8128	11,949
	SSH-Brute force	15,000	14,474
	FTP-Brute force	75	548
	Infiltration	15,000	12,851
	Brute Force-web	620	2006
	Brute Force-XSS	202	1651
	SQL Injection	95	802

Furthermore, we utilized diverse values of the scaling factors to treat the training set in both the CSE-CIC-IDS2018 and NSL-KDD datasets. We executed tests on the six projected classifiers. By utilizing the average F1-score of each classifier, we estimated their performance and present their results in Figures 27 and 28.

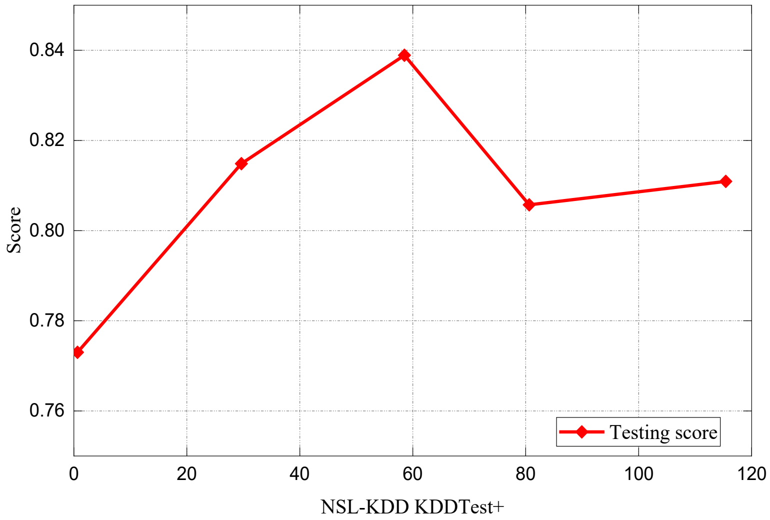

Figure 27. F1-score of the genetic algorithm with $(M = 60)$.

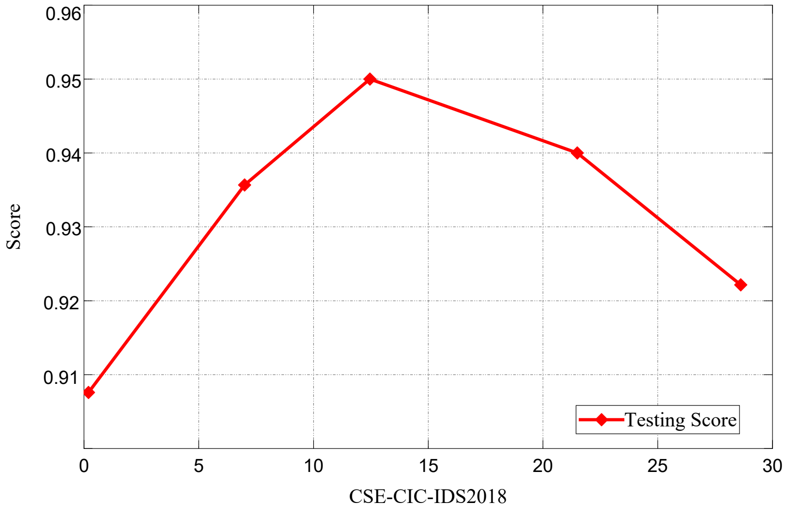

Figure 28. F1-score of genetic algorithm with $(M = 20)$.

As illustrated in Figure 29, for each sampling technique, we calculated the F1-score and average precision of the classifier. The performance of the sampling algorithms in the NSL-KDD dataset utilized RUS, ROS and SMOTE. The results showed that all performed with better quality as compared to the initial algorithm. Regarding the forecasting of the F1-score and precision, the enhancement was a little higher. The proposed GA was greatly optimized, for which the average precision was optimized by 5.65%, and the average F1-score was optimized by 8.2%. On the other hand, the performance improvements were insignificant or even worse after utilizing the RUS, ROS and SMOTE sampling algorithms for the CSE-CIC-IDS2018 dataset. After the training set with genetic algorithm sampling, which was projected in this research, the average precision was optimized by 3.65%, while the average F1-score was optimized by 4.24%. Considering that the F1-score model is a harmonic average of the prediction and recall rates, it is a decent indicator of the performance of classification models. Subsequently, the average precision and F1-score were accepted in this research as the metrics to measure the different approaches proposed by previous researchers regarding disparities in network traffic.

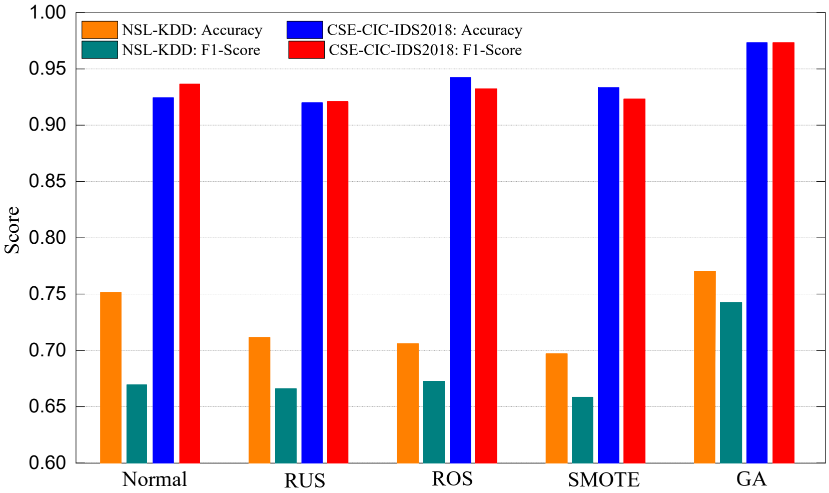

Figure 29. Comparison of different sampling techniques' performance (using the F1-score and precision as the average of each classifier).

In summary, conventional sampling approaches decreased the disparity in the training set and synthesized it with good proximity to the actual data; they did not create a resultthat fit the actual data. The RUS algorithm resulted in the loss of valid data, while the ROS algorithm resulted in overfitting and data redundancy. SMOTE interruption produced data intersection and noise traffic, as well as increased the number of complex samples in the training set. In all, the major target of our proposed genetic algorithm was to compress and improve complex information from a disproportionate training set. This allowed the classifier to attain a better distribution of the data, therefore enhancing the performance of the classification.

5. Conclusions

Considering the recent increase of security threats in smart systems and particularly in the smart healthcare industry, machine learning security approaches are considered as a very promising and viable technique towards combating these unrelenting intruding vulnerabilities. Therefore, this study performed an intrusion detection analysis for false alarm rate detection using machine learning feature selection, which was integrated in a genetic algorithm with the combination of random forest in a SoT model, and we further proposed a secured communication system suitable for smart healthcare. The research employed various machine learning algorithms on the NSL-KDD dataset for the experiments. Firstly, the utilized dataset was transformed into two binary classifications, namely the attack class and the normal class. The analysis performed pre-processing on the selected dataset, and non-numeric values are replaced with numeric encoding. Secondly, the data were normalized using min-max normalization. Then, the study performed feature selection using a Genetic Algorithm (GA), and the features with optimal performance were selected. After feature selection, the research employed different machine learning algorithms to analyse the selected dataset. For all selected ML algorithms, Random Forest (RF) outperformed all others in terms of the accuracy, detection rate and false alarm rate. Furthermore, the proposed technique was compared with other recent work, and the experimental results proved that our method performed better in terms of the detection rate, false alarm rate and accuracy. With respect to the FAR, the proposed model achieved 0.8% on the NSL-KDD dataset, which was improved from the base case, which had an FAR of 0.6% using the NSL-KDD dataset. Similarly, we delivered, on average, a 98.81% detection rate. In the future, attack classification accuracy based on high data and low data implementation will be improved. For example, there is a vast difference between the data attack size of Neptune with forty-five-thousand eight-hundred seventy-one packetsand the SQL attack, which has only two packets with almost zero classifications. We intend to use SMOTE, which is a data-balancing technique, to solve this anomaly. Finally, this study employed deep neural network models such as CNN and RNN to test the model. Another area of interest would be to auto-learn the pattern of the packet and then detect the anomaly by self-learning. Furthermore, the proposed algorithm was applied to examine the F1-score and precisionperformance as compared to the NSL-KDD and CSE-CIC-IDS2018 datasets using different scaling factors. Results showed that the proposed GA was tremendously optimized, in which the average precision was optimized by 5.65% and the average F1-score was optimized by 8.2%. Finally, the proposed GA-LG model performed well as it had an 88.10% precision, 92.40% recall and 90.20% F1-measure for the normal class. For the attack class GA-LG model, it had a 91.40% precision, 86.60% recall and 88.90% F1-measure.

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

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, S.; Xie, X.; Xu, Y. A Brute-Force Black-Box Method to Attack Machine Learning-Based Systems in Cybersecurity. *IEEE Access* **2020**, *8*, 128250–128263. [CrossRef]
2. Hady, A.A.; Ghubaish, A.; Salman, T.; Unal, D.; Jain, R. Intrusion Detection System for Healthcare Systems Using Medical and Network Data: A Comparison Study. *IEEE Access* **2020**, *8*, 106576–106584. [CrossRef]
3. Ullah, A.; Azeem, M.; Ashraf, H.; Alaboudi, A.A.; Humayun, M.; Jhanjhi, N. Secure Healthcare Data Aggregation and Transmission in IoT—A Survey. *IEEE Access* **2021**, *9*, 16849–16865. [CrossRef]
4. Anajemba, J.H.; Tang, Y.; Iwendi, C.; Ohwoekevwo, A.; Srivastava, G.; Jo, O. Realizing Efficient Security and Privacy in IoT Networks. *Sensors* **2020**, *20*, 2609. [CrossRef] [PubMed]
5. Anajemba, J.H.; Yue, T.; Iwendi, C.; Chatterjee, P.; Ngabo, D.; Alnumay, W.S. A Secure Multi-user Privacy Technique for Wireless IoT Networks using Stochastic Privacy Optimization. *IEEE Internet Things J.* **2021**, 1. [CrossRef]
6. Hussain, F.; Abbas, S.G.; Shah, G.A.; Pires, I.M.; Fayyaz, U.U.; Shahzad, F.; Garcia, N.M.; Zdravevski, E. A Framework for Malicious Traffic Detection in IoT Healthcare Environment. *Sensors* **2021**, *21*, 3025. [CrossRef] [PubMed]
7. Elrawy, M.F.; Awad, A.I.; Hamed, H.F.A. Intrusion detection systems for IoT-based smart environments: A survey. *J. Cloud Comput.* **2018**, *7*, 21–29. [CrossRef]
8. Bhattacharya, S.; Ramakrishnan, S.S.; Maddikunta, P.K.R.; Kaluri, R.; Singh, S.; Gadekallu, T.R.; Alazab, M.; Tariq, U. A Novel PCA-Firefly based XGBoost classification model for Intrusion Detection in Networks using GPU. *Electronics* **2020**, *9*, 219. [CrossRef]
9. Liu, G.; Yi, Z.; Yang, S. A hierarchical intrusion detection model based on the PCA neural networks. *Neurocomputing* **2007**, *70*, 1561–1568. [CrossRef]
10. Heba, F.E.; Darwish, A.; Hassanien, A.E.; Abraham, A. Principle components analysis and support vector machine based intrusion detection system. In Proceedings of the 2010 10th International Conference on Intelligent Systems Design and Applications, Cairo, Egypt, 29 November–1 December 2010; pp. 363–367.
11. Chae, H.S.; Jo, B.O.; Choi, S.H.; Park, T.K. Feature selection for intrusion detection using NSL-KDD. *Recent Adv. Comput. Sci.* **2013**, *32*, 184 –187.
12. Sarmah, A. *Intrusion Detection Systems: Definition, Need and Challenges*; SANS Institute: Bethesda, MD, USA, 2001.
13. Ren, J.; Guo, J.; Qian, W.; Yuan, H.; Hao, X.; Jingjing, H. Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms. *Secur. Commun. Netw.* **2019**, *2019*, doi:10.1155/2019/7130868. [CrossRef]
14. Thamilarasu, G.; Odesile, A.; Hoang, A. An Intrusion Detection System for Internet of Medical Things. *IEEE Access* **2020**, *8*, 181560–181576. [CrossRef]
15. Vaiyapuri, T.; Binbusayyis, A.; Varadarajan, V. Security, Privacy and Trust in IoMT Enabled Smart Healthcare System: A Systematic Review of Current and Future Trends. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*. [CrossRef]
16. Cook, D.J.; Duncan, G.; Sprint, G.; Fritz, R.L. Using Smart City Technology to Make Healthcare Smarter. *Proc. IEEE* **2018**, *106*, 708–722. [CrossRef]

17. Belavagi, M.C.; Muniyal, B. Performance evaluation of supervised machine learning algorithms for intrusion detection. *Procedia Comput. Sci.* **2016**, *89*, 117–123. [CrossRef]
18. Almseidin, M.; Alzubi, M.; Kovacs, S.; Alkasassbeh, M. Evaluation of machine learning algorithms for intrusion detection system. In Proceedings of the 2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY), Subotica, Serbia, 14–16 September 2017; pp. 277–282.
19. Chen, L.S.; Syu, J.S. Feature extraction based approaches for improving the performance of intrusion detection systems. In Proceedings of the International MultiConference of Engineers and Computer Scientists, Hong Kong, China, 18–20 March 2015; Volume 1, pp. 18–20.
20. De la Hoz, E.; De La Hoz, E.; Ortiz, A.; Ortega, J.; Prieto, B. PCA filtering and probabilistic SOM for network intrusion detection. *Neurocomputing* **2015**, *164*, 71–81. [CrossRef]
21. Wagh, S.K.; Pachghare, V.K.; Kolhe, S.R. Survey on intrusion detection system using machine learning techniques. *Int. J. Comput. Appl.* **2013**, *78*, 30–37.
22. Qiu, C.; Shan, J.; Shandong, B. Research on intrusion detection algorithm based on BP neural network. *Int. J. Secur. Its Appl.* **2015**, *9*, 247–258. [CrossRef]
23. Taher, K.A.; Jisan, B.M.Y.; Rahman, M.M. Network intrusion detection using supervised machine learning technique with feature selection. In Proceedings of the 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 10–12 January 2019; pp. 643–646.
24. Mukherjee, S.; Sharma, N. Intrusion detection using naive Bayes classifier with feature reduction. *Procedia Technol.* **2012**, *4*, 119–128. [CrossRef]
25. 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]
26. Eesa, A.S.; Orman, Z.; Brifcani, A.M.A. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. *Expert Syst. Appl.* **2015**, *42*, 2670–2679. [CrossRef]
27. Kumar, K.; Batth, J.S. Network intrusion detection with feature selection techniques using machine-learning algorithms. *Int. J. Comput. Appl.* **2016**, *150*, 1–13. [CrossRef]
28. Syarif, A.R.; Gata, W. Intrusion detection system using hybrid binary PSO and K-nearest neighbourhood algorithm. In Proceedings of the 2017 11th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 31 October 2017; pp. 181–186.
29. Aghdam, M.H.; Kabiri, P. Feature Selection for Intrusion Detection System Using Ant Colony Optimization. *IJ Netw. Secur.* **2016**, *18*, 420–432.
30. Mahmood, D.I.; Hameed, S.M. A Feature Selection Model based on Genetic Algorithm for Intrusion Detection. *Iraqi J. Sci.* **2016**, 168–175.
31. Rai, K.; Devi, M.S.; Guleria, A. Decision tree based algorithm for intrusion detection. *Int. J. Adv. Netw. Appl.* **2016**, *7*, 2828.
32. Thaseen, S.; Kumar, C.A. Intrusion Detection Model using PCA and Ensemble of Classiers. *Adv. Syst. Sci. Appl.* **2016**, *16*, 15–38.
33. Ambusaidi, M.A.; He, X.; Nanda, P.; Tan, Z. Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Trans. Comput.* **2016**, *65*, 2986–2998. [CrossRef]
34. 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]
35. Thaseen, I.S.; Kumar, C.A. Intrusion detection model using fusion of chi-square feature selection and multi class SVM. *J. King Saud Univ. Comput. Inf. Sci.* **2017**, *29*, 462–472.
36. Pajouh, H.H.; Dastghaibyfard, G.; Hashemi, S. Two-tier network anomaly detection model: a machine learning approach. *J. Intell. Inf. Syst.* **2017**, *48*, 61–74. [CrossRef]
37. Shone, N.; Ngoc, T.N.; Phai, V.D.; Shi, Q. A deep learning approach to network intrusion detection. *IEEE Trans. Emerg. Top. Comput. Intell.* **2018**, *2*, 41–50. [CrossRef]
38. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced network anomaly detection based on deep neural networks. *IEEE Access* **2018**, *6*, 48231–48246. [CrossRef]
39. Woo, J.H.; Song, J.Y.; Choi, Y.J. Performance Enhancement of Deep Neural Network Using Feature Selection and Preprocessing for Intrusion Detection. In Proceedings of the 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Okinawa, Japan, 11–13 February 2019; pp. 415–417.
40. Tao, P.; Sun, Z.; Sun, Z. An improved intrusion detection algorithm based on GA and SVM. *IEEE Access* **2018**, *6*, 13624–13631. [CrossRef]
41. Negandhi, P.; Trivedi, Y.; Mangrulkar, R. Intrusion Detection System Using Random Forest on the NSL-KDD Dataset. In *Emerging Research in Computing, Information, Communication and Applications*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 519–531.
42. Pervez, M.S.; Farid, D.M. Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs. In Proceedings of the 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014), Dhaka, Bangladesh, 18–20 December 2014; pp. 1–6.
43. Kanakarajan, N.K.; Muniasamy, K. Improving the accuracy of intrusion detection using GAR-Forest with feature selection. In Proceedings of the 4th International Conference on Frontiers in Intelligent Computing: Theory and Applications (FICTA), Durgapur, India, 16–18 November 2015; pp. 539–547.

44. 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]
45. Kuang, F.; Xu, W.; Zhang, S. A novel hybrid KPCA and SVM with GA model for intrusion detection. *Appl. Soft Comput.* **2014**, *18*, 178–184. [CrossRef]
46. Singh, R.; Kumar, H.; Singla, R. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Syst. Appl.* **2015**, *42*, 8609–8624. [CrossRef]
47. De La Hoz, E.; Ortiz, A.; Ortega, J.; De la Hoz, E. Network anomaly classification by support vector classifiers ensemble and non-linear projection techniques. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, Salamanca, Spain, 11–13 September 2013; pp. 103–111.
48. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
49. Tsang, C.H.; Kwong, S.; Wang, H. Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. *Pattern Recognit.* **2007**, *40*, 2373–2391. [CrossRef]
50. Kayacik, H.G.; Zincir-Heywood, A.N.; Heywood, M.I. A hierarchical SOM-based intrusion detection system. *Eng. Appl. Artif. Intell.* **2007**, *20*, 439–451. [CrossRef]
51. Raman, M.G.; Somu, N.; Kirthivasan, K.; Sriram, V.S. A hypergraph and arithmetic residue-based probabilistic neural network for classification in intrusion detection systems. *Neural Netw.* **2017**, *92*, 89–97. [CrossRef]