*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

# AN EFFICIENT INTRUSION DETECTION TECHNIQUE FOR TRAFFIC PATTERN LEARNING

**Umukoro, I. I.[1], Eke, B.O.[2] and Edward, O.[3]**
[1,2,3]Department of Computer Science, University of PortHarcourt

*Received: 29-01-2024*
*Accepted: 26-02-2024*

https://dx.doi.org/10.4314/sa.v23i2.3

Journal Homepage: http://www.scientia-african.uniportjournal.info
Publisher: *Faculty of Science, University of Port Harcourt.*

## ABSTRACT

*Efficient intrusion detection algorithms are required for network traffic learning patterns in order to protect advanced network communication channels. These systems can be used to detect normal and unusual patterns, signatures, and rule violations. In recent years, conventional and deep machine learning algorithms have been utilized in the field of network intrusion detection for network traffic learning systems. The use of machine learning opens up new attack surfaces that are very intriguing to investigate. Attackers can introduce noisy data into training data to influence testing patterns in computer networks. The goal of this work is to create an efficient intrusion detection solution for network traffic learning patterns using a supervised and unsupervised technique. We developed an effective intrusion detection system (IDs) using an appropriate NSL-KDD dataset for network traffic patterns. The model was trained and evaluated using the Genetic Optimization Algorithm (GOA) and the Niave Bayesian technique to recognize usual and unexpected network traffic patterns. We created a strategy that begins with a random population and subsequent iterates through the fitness function, returning the best parents with high detection accuracy. The best parents were determined using the n-parameters iterated by the crossover and mutation procedures. A cross over function was created to combine genes from two fitness parents by randomly selecting portions from each parent. The individual components of the crossover off-springs are randomly flipped to achieve the mutation. The fitness of the previous generation was obtained to generate a new generation, and this process was repeated n times. This was created to detect network intrusions using Nave Bayes' binary categorization problem and evolutionary algorithms. We accomplished this task by aggregating noise into training set before broadcasting the average number, and it is critical not to have that public average too frequently. The experimental results reveal that our proposed GA fared better than the NB technique, with a detection accuracy of 95.0% versus a recommendable detection accuracy of 53.0%.*

**Keywords:** Feature extraction, genetic algorithm, Naïve Bayesian, Intrusion detection

## INTRODUCTION

Intrusion detection system (IDS) is a system designed to detect, to notice, or to spot an intruder, trying to gain an unauthorized access into a system. In information technology, it is a system to notice, or spot an unauthorized access into a networking system or environment (Yasmeen et al.,2022). Wireless Sensor Networks (WSNs) are one of the most intriguing and demanding engineering study topics. Zhou, (2005) observed issues in the networking

environment in which certain network users launched assaults on cooperative and individual networks using malicious software and codes. The goal is to steal data, destroy and hijack networking systems, or even upload material on the network in order to manipulate network operations. One of the most concerning networking security concerns today is the man-in-the-middle threat, in which the attacker builds a fake network and entices the victim into connecting to the fake network. Existing methodologies in use are untrustworthy, resulting in low rates of false positive and false negative classification reports(Abayomi-Alli et al.,2015, Ghosal & Halder 2013).IDS are commonly utilized in conventional systems of operation, but they confront a variety of challenges in the cloud context(Chandre, et al.,2020). One issue is the division of duty between the corporation and the consumer, as well as the reality of who and how the IDS must be controlled (Govindarajan and Chandrasekaran 2012). Traditional IDS are inefficient and incompatible with cloud structures due to the features of cloud computing structures (elasticity, reliability, QoS, agility, adaptability, and availability). We propose various needs for a IDSs, including the capacity to deal with massive-scale, dynamic multi-tiered self-sufficient computing and data processing settings, as well as to quickly adapt to new attacks, configurations, and deployments, and to be scalable (Jain and Abouzakhar 2013). However, subscribers can use IDS to identify attacks on their services, and providers can use IDS to detect attacks on their infrastructure. It is perplexing that the IDPS must be segregated from the network that it is monitoring (Kumar 2014, Rakshe and Gonjari 2017). NIDS attempt to detect cyber-attacks, malware, Denial of Service (DoS) attacks or port scans on a computer network or a computer itself. NIDS monitor network traffic and detect malicious intent by detecting strange patterns in incoming packets (Surantha and Wicaksono 2019). Any harmful behavior or violation is often reported to an administrator or amassed centrally using a security statistics and event management (SIEM) device.

These approaches intelligently work in the background of networks to quietly observe site visitors as they navigate the devices on which they operate (Usha and Suganthi 2018). It may behardware or software-primarily based totallystructures and, relyingat theproducer of the system, can connect to numerousnetwork mediums which include Ethernet, FDDI, and others. Oftentimes, NIDS have network interfaces. One is used for taking note ofnetwork conversations in promiscuous mode and the alternative is used for manage and reporting. With the arrival of switching, which isolates unicast conversations to ingress and egress transfer ports, network infrastructure companies have devised port-mirroring strategiesto duplicate all networksite visitors to the NIDS. There are differentapproaches of impartingsite visitors to the IDS which includesnetwork taps. There are numerous NIDS companies, all structuresgenerally tend to feature in certainly considered one among ways; NIDS are both signature-primarily based totally or anomaly-primarily based totallystructures. Both are mechanisms that separate benign site visitors from its malicious brethren. Potential troubles with NIDS consist of high-pacenetworkrecords overload, tuning problems, encryption, and signature improvement lag time (Wazid 2017). The goal of this study is to create an efficient intrusion detection approach for traffic pattern learning using the Naive Bayesian (NB) and Genetic Algorithm (GA). We intend to develop an intrusion detection model (IDM) for network traffic pattern learning using NSL-KDD system dataset, train and evaluate machine learning models to learn and recognize normal and anomalous network traffic patterns.

We are developed an initial population of intrusions for Machine learning-based virtual sensor network traffic patterns, and a function will be defined to identify them as normal or

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

abnormal. If a particular chromosome has more normal patches, then it will be considered to be more fit; consequently. Several data points with normal status will be selected for mating to create children, and the offspring will be replaced with those from the population having malicious content (full dataset), and this procedure will be repeated as long as the fitness of the fitted population is insufficient. We shall generate off-springs by sifting through the population for the best-fitting chromosomes. The Initialization process will primarily aim to establish the population's randomness with a dataset built in Python utilizing the numpy pool of libraries and containing genetic chromosomes.

This research will aid in improving network traffic monitoring in the domain of virtual sensor networking, identifying suspicious network traffic behaviors, improving network security, and detecting potential network threats.

The paper is organized as follows: Section 1 provided an introduction; Section 2 offers a brief assessment of prior approaches related to the topic and the gap in studying the proposed model; Section 3 introduces the model's materials and methods; Section 4 covers the results and a thorough discussion of the results; and Section 5 provides the paper's conclusion.

## RELATED WORKS

Borgohain(2019) provides an overview of the significance of the combination of Genetic Algorithm and Fuzzy Logic System for identifying a net intruder in a wireless connecting environ in his article FuGeIDS: Fuzzy Genetic paradigms in Intrusion Detection Systems. Ambusaidi et al. (2014) used set of algorithms based on mutual records that chooses the best characteristics for each category in an analytical manner. They used three different intrusion detection datasets, including KDD-Cup99, KDD-NSL, and Kyoto 2006+. Mohammad et al. (2011) developed a Novel Local Net Intrusion Detection System Based on Support Vector Machine. Numerous methods and strategies were used to simulate the IDS, but some of them contributed little or nothing toward its task. The intrusion methods and intrusion links of nearby location nets were established prior to forwarding user requests to the LAN's intrusion detection machine. Solanki et al.(2020), examined variety of intrusion detection systems and methods based on SVM, Fuzzy Logic Systems, and Supervised Learning. Furthermore, they have compared a variety of methodologies based on the notion of their correctness using KDD-NSL data sets. They also suggested that the method may be enhanced by combining SVM with machine-learning knowledge accuracy. Parag (2012) suggested that cloud IDS can help manage large volumes of data packets, and created reports by merging technologies with behaviour assessment to identify intrusions. They stated that data transfer secrecy are needed to provide safety in a distributed CPU. A new multi-threaded distributed cloud IDS version was adopted to handle huge scale net access request and administrative manipulation of data and applications in the cloud. Learning about intruder operations reduces the possibility of infiltration on machine learning model because cloud computing is a network of networks on the internet, Gyanchandani et al. (2012) propose a taxonomy of ultra-modern anomaly based fully intrusion detection set-ups that classify all workable of intruders. The aim is to find attacks on data-driven systems and cutting-edge data systems. They designed and implemented a novel way for discovering these vulnerabilities, which coincided with the development of cutting-edge vulnerability techniques. It described the categories of modern anomaly-based intrusion detection systems in terms of their properties, as well as their advantages and disadvantages. Their research includes several examples of both historical and present-day jobs. Yao (2020), built an intrusion detection system using a pattern recognition tool like Support Vector Machines (SVM) to detect attacks on network traffic packets, however. They employed a weighted kernel function with an enhanced

SVM for intrusion detection that is purely dependent on a given functions. Hu et al. (2018) investigated an intruder detection system in an industrial setting, as well as the distinctive characteristics of new industrial control system standards. They developed a new taxonomy of intruder detection systems for use in industrial control environments. Ahmed (2013) proposed a temporal logic and network data streaming technology-based online intrusion detection system. The temporal data were transformed into a connected stream of queries which was subsequently utilized by the Stream Data Process (SDP) to monitor the network and detect attackers. Aljebreen(2018) developed an intelligent safety network intrusion detection system by enhancing the cloud container security as it is currently configured. He suggested a data representation tactic in order to expedite the modeling and classification process. Pelechrinis and Lappas (2019) focused on various data mining approaches and their use in detecting network attacks. They examined the significance of such techniques as well as the advantages and negative aspects of such procedures. Sheikh et al. (2013) proposed a cluster mobile sensor system with a hybrid intrusion detection system. It was an empirical investigation that used a hybrid approach to identify intrusions in a wireless sensor network. They discussed the use of WSN and detection methods, as well as comparing their security measures. Rastegari (2015) offered a systematic approach to cellular network security, which he divided into three parts: attacker preparation, system detection of attack, and system alarm mechanism. The researcher used multiple ML algorithms, such as DT, KNN, and others, to analyze the behavior of wireless networks and the various attacks carried out by hackers. Qurashi (2020) created a Random Geometric graph to capture specific network aspects, such as communication difficulties and the energy overhead of an IDS in a networking system. The nodes were utilized to identify network layer intrusions through observing local networking information.

Chua and Salam (2020) examined the long-term performance of various ML-based intrusion detection systems. They analyzed the ML-based IDS using a data set created after the training data stage. However, it could no longer improve the IDS's long-term performance because it better reveals changes in the nature of attacks. The proposed method improved the long-term overall performance of ML-based IDS because the testing data set shows the adjustments in the form of attacks and changes in network infrastructure over time. They used six well-known ML models for the IDS: Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Nave Bayes (NB), Artificial Neural Network (ANN), and Deep Neural Network (DNN). The CIC-IDS2017 and CSE-CIC-IDS2018 data sets were employed in their experiment, with SVM and ANN as the most resistant to over-fitting. The experimental results also revealed that DT and RF suffered the most from overfitting, despite working effectively with the training data set. They also developed a new way to measure the long-term effectiveness of an Intrusion Detection System (IDS) based on machine learning (ML). According to the results, DT was better suited for small datasets with attacks.

## MATERIALS AND METHODS

It is critical to emphasize that efforts have been made to network intrusion detection system and we were able to leverage ML experience and focus our research on the areas of detecting network traffic intrusion. This study employs the genetic and naïve bayesian techniques to acquire knowledge from NSL-KDD training and testing datasets with fine-tuned hyper-parameter values in order to make better decisions. This requires machine learning architecture trained using an experimental dataset acquired from school libraries in order to boost accuracy, precision, and temporal complexity. The suggested architecture has been revised to aid in the detection of intrusions and reported attack

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

types, and the measures undertaken for carrying out the proposed design are outlined below:

**Dataset**

The experimental data was sourced from kaggle online website, which can be reached via the UCI machine-learning repository website at https://www.kaggle.com/datasets/hassan06/nslkdd. It contains attributes like protocol type, flag, src bytes, dst bytes, wrong fragment, number of failed logins, number of compromised accounts, number of su attempts, and others. The proposed dataset has 47738 items in total, divided into 22545 testing sets and 22545 tutoring sets. The KDD-NSL is a revised and enhanced version of the KDD Cup'99 dataset that fixes several purport issues.

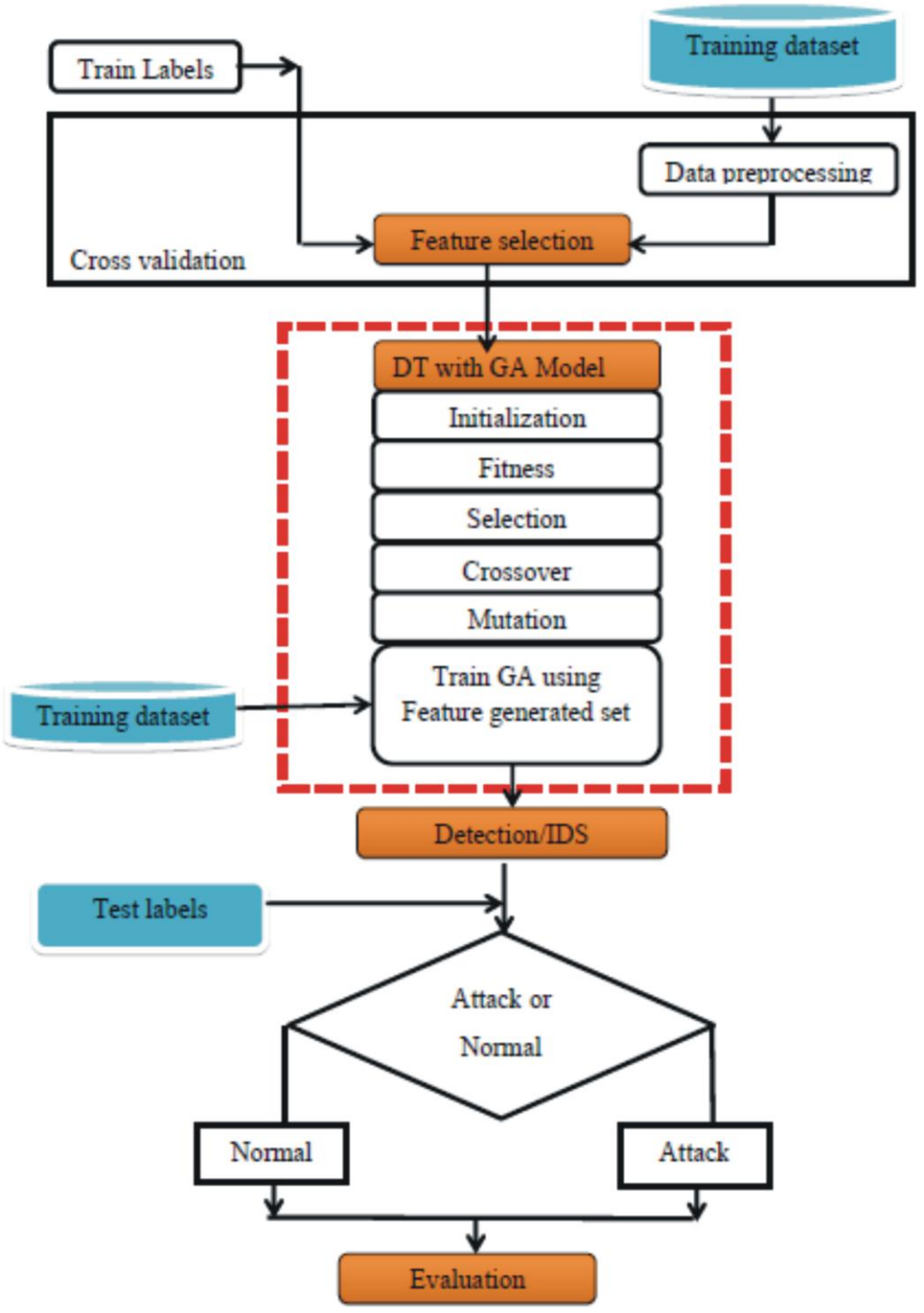


**Figure 1:** Study Strategy of IDS models

Figure1 depicts the suggested system strategy, which includes IDS dataset (testing and training set), preprocessing, feature extraction, model (NB, GA), and detection.

**The training-testing input dataset**: The input dataset was uploaded and split into 2-sets (80% training and 20% testing set). The input dataset has been loaded, and the preprocessing component has been structured in a bottom-up approach by some preparatory and classification operations, namely: preprocessing, model building, model training and evaluation.

**Pre-processing step**: Preprocessing step is a preliminary process that involves different classification processes with the aim of formatting data obtained from the kaggle official site. The classified data patterns are carried out effectively in the analysis phase of preprocessing. The preprocessing component was employed to reduce threshold value and data fashioning for training and testing purpose of the GA.

**Feature Extraction**: feature extraction technique is mandatory for any application and it trains the model with the features to assist the task of object classification. The feature extraction phase is very import because it has the capability of influencing the classification task in adopting the proposes model

**Naïve Bayesian algorithm**

Tthe Nave Bayesian ML class library was imported from the Sklearn package, with X and y serving as input and output variables for the NB-classifier. After assigning a set of testing data from the test split to a predictor variable, the predictor's mean and standard deviation were determined. Calculate the class probabilities for existing employees and the posterior probability for everyone who is staying in the organization using Bayesian theory until the complete test dataset is exhausted. Calculate the odds of each class for normal and unusual network patterns that arrive at the highest probabilities and return value.

**Genetic Algorithm (GA)**

We are employing a method that start with random generation of n-population and then iterates via the fitness function, which returns the best parents with excellent accuracy. The best parents were chosen based on the n-parameters iterated by the crossover and mutation functions, respectively. A cross over function was developed to mix genes from two fitness parents by randomly selecting parts from each parent. The crossover child's bits are randomly flipped to achieve the mutation. The fitness parent from the previous generation was chosen to create a new generation, and this process was repeated for n generations.

The genetic algorithm (GA) is a cutting-edge tactic inspired by human genetic cycles of transmitting genes from to the next generation. It is utilized as an optimization. This will help to improve the intrusion detection accuracy of learning-based virtual sensor net traffic patterns by improving the NB with GA for greater feature selection. The design of GA contains the following steps: population initiation, fitness, selection, crossover, and mutation.

**Initialization of population**: For learning-based virtual sensor net traffic patterns, an initial populace of incursions was created, and a function was established to determine whether they were normal or pathological. Chromosome-1 is said to be of more fit than chromosome-2 since it contains more normal patches than the latter. For mating and producing children, we picked the points with normal status. We then substituted the offspring with those from the population with malicious content (the whole dataset), and we repeated the procedure as long as the fitted populace's fitness was insufficient. We created children using the populace's best-fitting chromosomes. The Python-developed numpy library pool, which consists of genetic chromosomes,

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

was utilized to generate the population randomly supplied dataset and was first initialized in the Initialization phase, as indicated in the code snippet in below.

```
147
148     #defining various steps required for the genetic algorithm
149     def initilization_of_population(size,n_feat):
150         population = []
151         for i in range(size):
152             chromosome = np.ones(n_feat,dtype=np.bool)
153             chromosome[:int(0.3*n_feat)]=False
154             np.random.shuffle(chromosome)
155             population.append(chromosome)
156         return population
157
```

**Figure 2:** Initialization

**Fitness**: As demonstrated in the screen shot below, the function calculates fitness as derived from the training sample for each chromosome using parameters that specify a suggested task and a solution that the genetic algorithm is seeking to solve.

```
158
159     def fitness_score(population):
160         scores = []
161         for chromosome in population:
162             Svm_clf.fit(X_train.iloc[:,chromosome],y_train)
163             predictions = Svm_clf.predict(X_test.iloc[:,chromosome])
164             scores.append(accuracy_score(y_test,predictions))
165         scores, population = np.array(scores), np.array(population)
166         inds = np.argsort(scores)
167         return list(scores[inds][::-1]), list(population[inds,:][::-1])
168
169
```

**Figure 3: Fitness**:

**Selection**: We construct a new population to pass on the genes to future or next generation, while the Selection stage progressively selects the best-fitting chromosomes as parents, thereby returning the variable n parents as shown below in the Python code segment/

```
169
170
171     def selection(pop_after_fit,n_parents):
172         population_nextgen = []
173         for i in range(n_parents):
174             population_nextgen.append(pop_after_fit[i])
175         return population_nextgen
176
```

**Figure 4: Selection**

**Crossover phase**: The Cross-over module combines parents with newly developed chromosomes to produce a new population group as shown in statement line 179 to 185.

```
178
179     def crossover(pop_after_sel):
180         population_nextgen=pop_after_sel
181         for i in range(len(pop_after_sel)):
182             child=pop_after_sel[i]
183             child[3:7]=pop_after_sel[(i+1)%len(pop_after_sel)][3:7]
184             population_nextgen.append(child)
185         return population_nextgen
186
187
```

**Figure 5: Crossover phase**

**Mutation**: Mutation will aid in the flipping of certain genes in order to form a new or future population, and that was carried out following each crossing over cycle as shown below in the code snippet. The objective is to inject random samples (or genes) to the offspring while preserving variation within the feature population so as to prevent convergence.

```
187
188     def mutation(pop_after_cross,mutation_rate):
189         population_nextgen = []
190         for i in range(0,len(pop_after_cross)):
191             chromosome = pop_after_cross[i]
192             for j in range(len(chromosome)):
193                 if random.random() < mutation_rate:
194                     chromosome[j]= not chromosome[j]
195             population_nextgen.append(chromosome)
196         #print(population_nextgen)
197         return population_nextgen
198
```

**Figure 6: Mutation**

With the best chromosomes and score values, GA was trained and evaluated utilising feature initialization, fitness value, selection, cross-over, and mutation steps.

**Detection System:** The detection system is a module that keeps track of normal and malicious attacks on network. This is the model's projected values following the training step, utilizing testing (unseen) data to evaluate its performance.

**Performance evaluation**

The performance evaluation stage is the final stage in properly measuring the success rate of GA and existing NB classifier. The detection accuracy, root-mean-square-error (RMSE), confusion matrix, precision, ROC, and recall diagnostic tools were used in this study to assess the accuracy metrics of both strategies. The reported results are also discussed. The precision: is a metric used for positive classifications, and it is represented as follows:

$$\text{Precision}=\frac{TP}{TP+FP} \quad (1)$$

Recall employed to measure the false negative classifications represented as:

$$\text{Recall} =\frac{TP}{TP+FN} \quad (2)$$

**F1-score**: takes into consideration the true-positive (TP) and false-positive (FP) regardless of false negative and false positive classifications. The F1-score represents the sensitive-level to which class is positive/negative shown in equation 3.

$$\text{F1-score}=\frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} = \frac{2*TP}{2*TP+FP+FN} \quad (3)$$

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

The proposed ANN accuracy metrics is measured with the classifications (TP+TN) from the overall cases shown below:

$$\text{Accuracy}=\frac{\text{The Total}-\text{no. of correct classifications}}{\text{The overall no. of cases}}=\frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \quad 4$$

Where TN represents true negative, FP is false positive, TP is true positive and FN is false negative cases.

**Algorithm 1**: Naive Bayes(NB)

| Step | **Processes involved** |
|---|---|
| 1 | **Input:**Training_Dataset(T) |
| 2 | **Output**: Class of testing items |
| 3 | Create NB classifier |
| 4 | F= ($f_1$, $f_2$, $f_3$,....$f_n$) // the predictor variables for testing items |
| 5 | **Compute** mean and standard deviation of predictor variables in each class |
| 6 | **Repeat** this step<br>(a). Compute probabilities required for the Bayesian theorem for Exiting employees<br>(b). Compute posterior probability of all those that are not leaving the organization |
| 7 | **Compute** the likelihood of each class(first and second class) |
| 8 | **Get** the greatest likelihood |
| 9 | Return |

**Algorithm 2**: Genetic Algorithm(GA)

| Step | **Processes involved** |
|---|---|
| **1** | Starting |
| **2** | Iniatialize(K=0) and make a populace($P_k$): of n randomly-created data points |
| **3** | Comput fitness(i) for each i$\epsilon P_k$ |
| **4** | **Selection**, Make choosing with members of $P_k$ and inset into $P_{k+1}$ |
| **5** | **Crossover**: choose and pair members of Pk; create offspring, inset the offspring into $P_{k+1}$ |
| **6** | **Mutation**: pick μ x n members of Pk+1; upturn a randomly-picked bit for each |
| **7** | **Evaluate** Pk+1; Calculate fitness(i) for each i$\epsilon P_k$ |
| **8** | K=k+1 // increment k by 1 |
| **9** | Repeat if the suited individual's fitness level in Pk is insufficient. |
| **10** | **Return** fitness individuals from $P_k$ |

## RESULTS AND DISCUSSION

The results are presented, and discussed in this section. We presented and discussed about results of the NB and GA with some evaluation tools. The accuracy metrics, MSE, RMSE, classification report, Confusion matrix, area under the curve (AUC), receiver operating system curve (ROC) and heatmap graphs are fully used and discussed to ascertain its efficiency. The design and implementations were done using variety of fine-tuned hyper-parameter values to get a superior classification report,. The prediction and classification accuracy of both models are visualized and discussed using the confusion matrix, ROC, and classification report provided below:

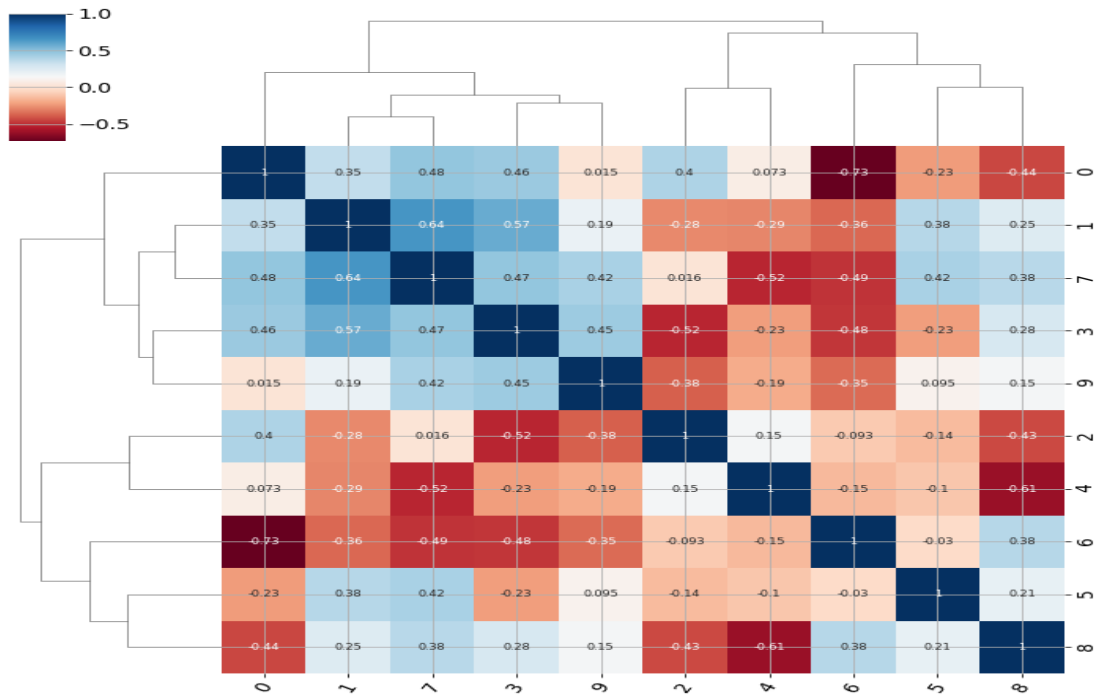


**Figure 7**: The correlation matrix.

Figure 1 displays the feature correlation matrix of our proposed IDS dataset, highlighting the groupings of highly correlated features that make significant contributions to model predictions. It displays a graphical representation of the linked features in three groups.

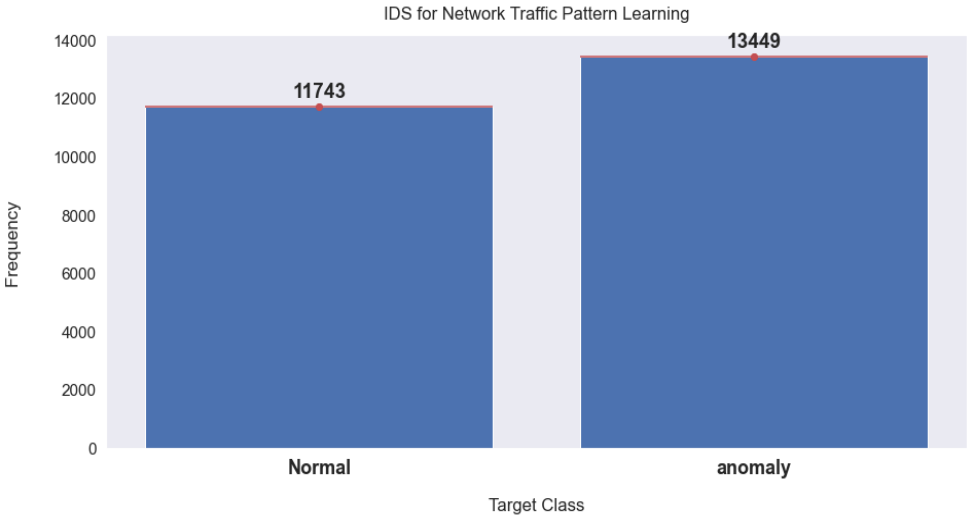


**Figure 8**: target class variable class ration

Figure 1 uses bar charts with values plotted on top bars to represent the unequal ratios of minority and majority target data as a record. The anomaly target class produced 13449 items, which were distributed almost evenly, while the normal class produced 11743 items.

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

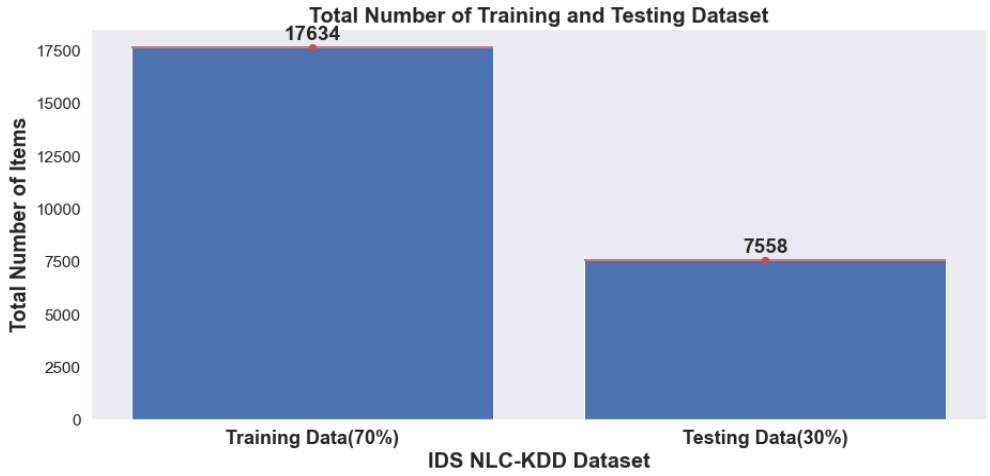


**Figure 9:** Training/testing data split

Figure 2 depicts the percentage of NLC-KDD datasets used for tutoring and testing, which produced a total of 25182 items. The test split command in Python ANACONDA was used to test the model using the remaining 30% of the total set after 70% of it had been used for tutoring.

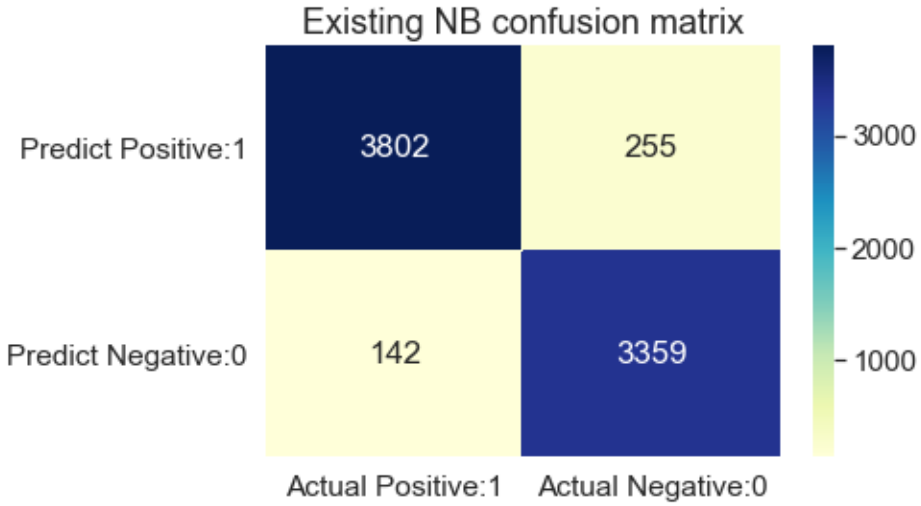


**Figure 10**: The confusion matrix of Naïve bayes

The proposed NB classifier's confusion matrix is shown in Figure 10, with the leading main diagonal or values (or off-diagonal elements) showing the total number of correctly predicted values that are equal to the actual or true values above and below the main diagonal cell values. The prediction is more accurate the higher the diagonal values. The confusion matrix shows that the total number of correct predictions =FP+FN =142+255=.397 and the total number of accurate predictions =TP+TN =3802+3359=7161.

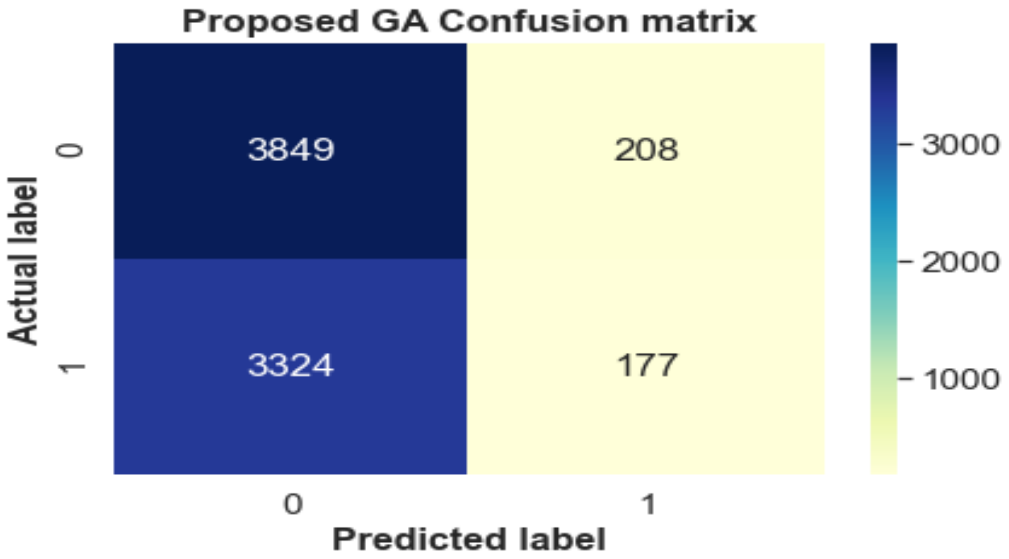


**Figure 11**: The confusion matrix of GA algorithm

Figure 4 shows the confusion matrix of GA at testing stage, with correct predictions shown at the secondary diagonal and incorrect predictions recorded above and below the main diagonal, or off-diagonal elements, respectively. Figure 7 displays the total number of correct predictions, where TP stands for true positive, FP for false-positive, FN for false-negative, and TN for true negative, likewise the number of incorrect predictions, where FP and FN together total 208 and 3324, respectively.

**Table 2:** Classification report of NB

| | **precision** | **recall** | **F1-score** | **support** |
|---|---|---|---|---|
| 0 | 0.54 | 0.95 | 0.69 | 4057 |
| 1 | 0.46 | 0.05 | 0.09 | 3501 |
| | | | | |
| accuracy | | | 0.53 | 7558 |
| Macro avg | 0.50 | 0.50 | 0.39 | 7558 |
| Weighted avg | 0.50 | 0.53 | 0.41 | 7558 |

Table 1 displays the NB classification report, which includes info on departing and remaining staff members' precession, recall, and f1-score accuracy. The scores for precision, accuracy, recall, and f1-score for normal data classes are 0.54, 0.95, 0.69, and for anomaly data classes, they are 0.46, 0.85, and 0.09, respectively.

**Table 2:** Classification report of genetic algorithm

| | **precision** | **recall** | **F1-score** | **support** |
|---|---|---|---|---|
| 0 | 0.96 | 0.94 | 0.95 | 4057 |
| 1 | 0.93 | 0.96 | 0.94 | 3501 |
| | | | | |
| accuracy | | | 0.95 | 7558 |
| Macro avg | 0.95 | 0.95 | 0.95 | 7558 |
| Weighted avg | 0.95 | 0.95 | 0.95 | 7558 |

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

Table 2 displays the classification report of the genetic optimization algorithm, which has classification accuracy of 96% for both normal and anomalous data classes in terms of precession, recall, and f1-score. The f1-score was 95%, the recall was 94%, and the precision accuracy score was 96%. When comparing the GA classification report data to the old system classification report data, there is a purport improvement as demonstrated in the precision, recall, and f1-score values from the classification report, as documented.

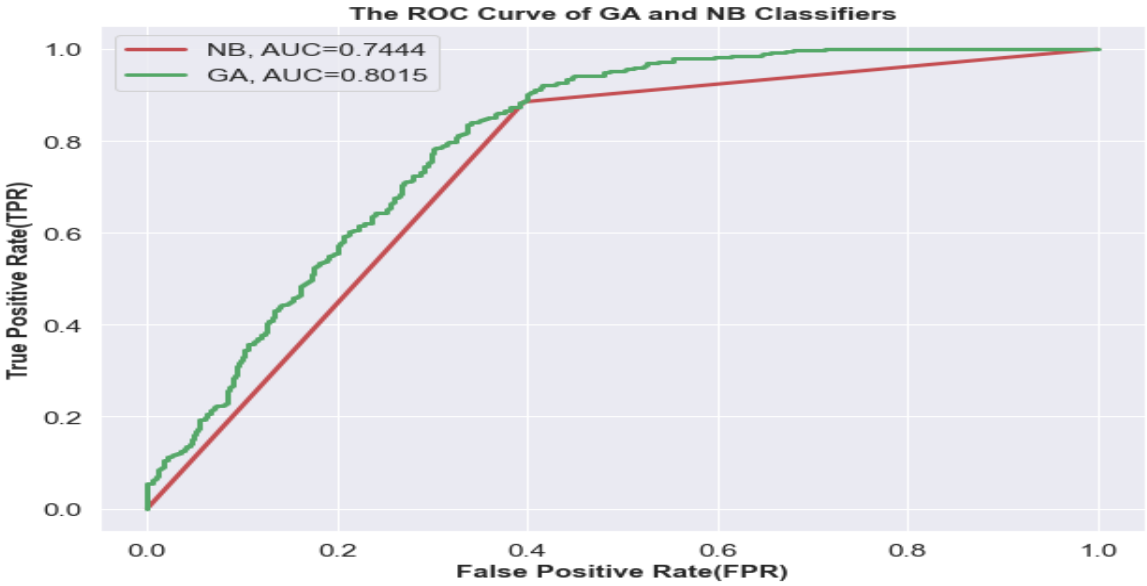


**Figure 12**: The ROC graph of GA and NB algorithm

Figure 5 is the ROC curve of GA and NB showing the trade-off between the sensitivity, or true positive rate, and specificity(1-FPR). The GA ROC curve surpasses the NB model, is perfect, and is located towards the top-left corner of the graph. In the ROC graph, the SVM curve's number of false-positive rates (FPR) and true positive rates (TPR) are a little off the top x and y axes (TPR). The proposal produced the predicted diagonal points (True Positive Rate=False Positive Rate).

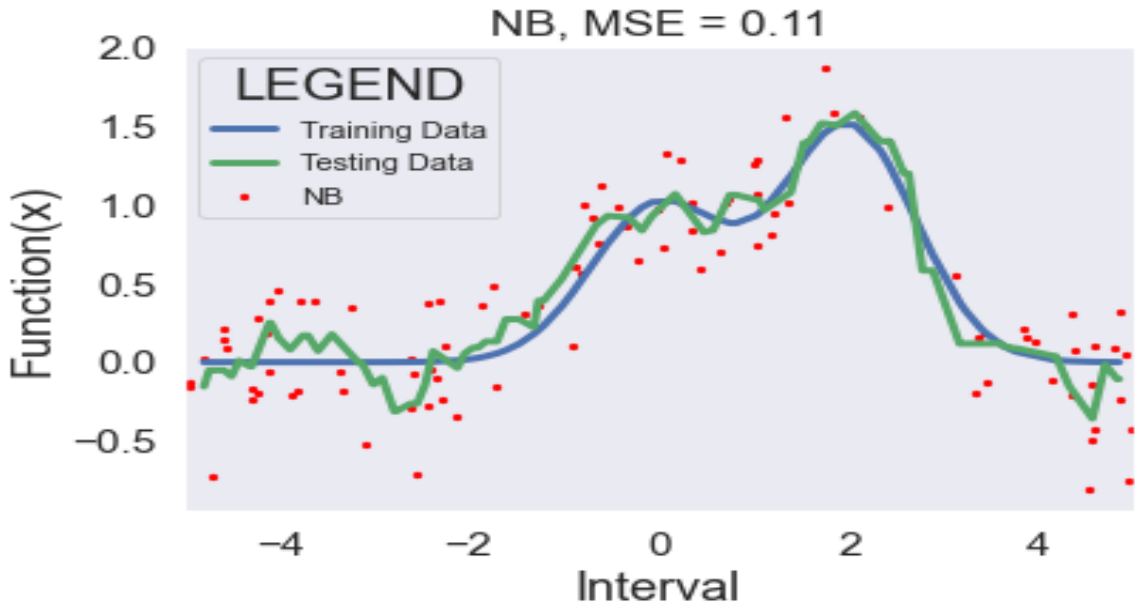


**Figure 13**: The training and test data patterns of NB Classifier

Figure 6 depicts the training and testing patterns of NB classifier IDS. In this case, the testing data patterns were altered as it varied across the time span. The intrusions had a negative impact on the NB classifier's accuracy, which had a mean square error (MSE) of 0.11. In this case, by comprehending how the model transforms data into learning parameters, intruders can assess model parameters and get data patterns from the testing set.

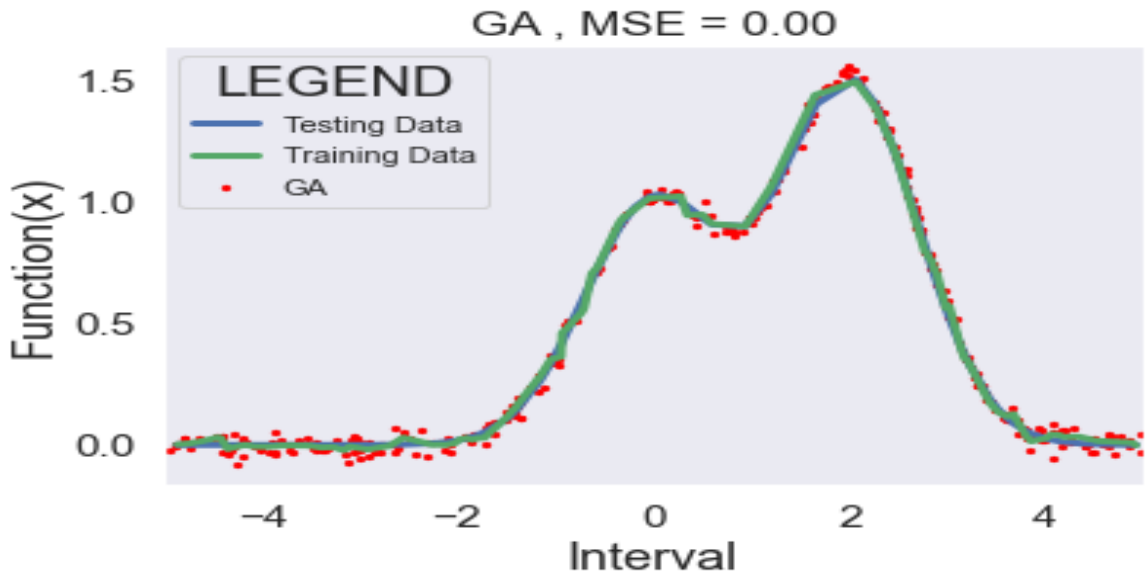


Figure 14: Training and testing data patterns of GA

Figure 7 depicts the training and testing traffic data patterns of GA at its learning stage. The data injected by the intruder had no impact on the testing set, as shown in the testing data pattern by the learning model in network traffic intrusion detection. The GA offered a verifiable assurance that intruders would never again have access to training data and had a low error rate and high accuracy rate. As shown, there will be fewer invasions, negative actions, model probing, and unauthorised access to testing datasets.

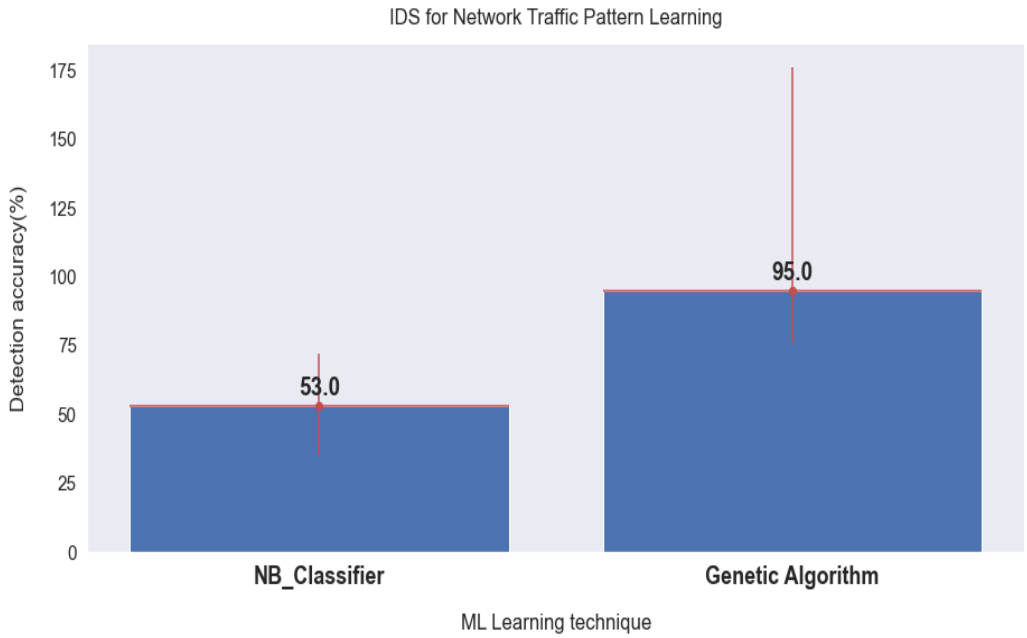


**Figure 15**: The performance accuracy of GA and NB classifier

*Scientia Africana, Vol. 23 (No. 2), April, 2024. Pp 25-40*
*© Faculty of Science, University of Port Harcourt, Printed in Nigeria* *ISSN 1118 – 1931*

Figure 8 shows how accurate the GA and NB models' performance is. The GA generated 95.0% of IDS whereas the NB only produced 53%, which was much less than anticipated. The GA surpassed the NB classifier in terms of network traffic intrusion detection accuracy,

## CONCLUSION

The suggested tactic maintains secure, effective, trustworthy, and accurate upshots while outperforming the current model of operation (Naive Bayesian) in practise when it comes to IDS for virtual net pattern learning. The new system has addressed the majority of the current system's flaws, including the unbalanced classes and poor detection rate (genetic algorithm of IDS). We draw the conclusion from the suggested system optimization strategy which was promising in identifying back door assaults in NIDS for net traffic learning patterns, which is much more accurate and precise than the current method (system).

## REFERENCE

Abayomi-Alli, A. A., Ikuomola, A. I., Robert, I. S. and Abayomi-Alli, O. O. (2015) An Enterprise Cloud-Based Electronic Health Records System, Journal of Computer Science and Information Technology, 2(2), 21-36.

Ahmed, S. and Yadav, M. (2018), Anomaly Detection in Wireless Sensor Networks - Critical

Survey, Second International Conference on Advancement in Computer Engineering and Information Technology, 42, 1-14.

Aljebreen, M. (2018), Towards Intelligent Intrusion Detection Systems for Cloud Computing, A dissertation submitted to Florida Institute of Technology, 1-147

Ambusaidi, M. et al (2012), Building an intrusion detection system using a filter-based feature selection algorithm, International Journal of Innovative Research & Studies,112(23), 25-27.

Borgohain, R. (2019). Fuzzy Genetic paradigms in Intrusion Detection Systems. International Journal of Advanced Networks and Applications, Cryptography and Security, 3(6), 1409 - 1415.

Chandre, P., Mahalle, P. N. and Shinde, G.(2020) Deep Learning and Machine Learning Techniques for Intrusion Detection and Prevention in Wireless Sensor Networks: Comparative Study and Performance Analysis, Design Frameworks for Wireless Networks, 51(7), 95-120.

Chua T. and Salam I. (2020), "Evaluation of Machine Learning Algorithms in Network-Based Intrusion Detection System, Cryptology ePrint Archive, 32-41, https://eprint.iacr.org/2022/335.pdf

Ghosal A. and Halder S. (2013), Intrusion Detection in Wireless Sensor Networks: Issues, Challenges and Approaches, Research Gate: 1(12), 2-23.

Govindarajan, M. and Chandrasekaran, R. (2012), Intrusion Detection using an Ensemble of Classification Methods, Proceedings of the World Congress on Engineering and sComputer Science, 1(1), 8-23.

Gyanchandani M., Rana, J. L. and Yadav, R.N (2012), Taxonomy of Anomaly Based Intrusion Detection System: A Review, International Journal of Scientific and Research Publications, 2(12), 1-13.

Hu Y. Li, H., Sun, Y. and Yang, A. (2018), A survey of intrusion detection on industrial control systems, International Journal of Distributed Sensor Network, 14(8), 1-14

Jain, R. and Abouzakhar, N. (2013), A Comparative Study of Hidden Markov Model and Support Vector Machine in Anomaly Intrusion Detection, Journal of Internet Technology and Secured Transactions (JITST), 2(3/4), 12-39.

Kumar K. (2014), Support Vector Machine for Network Intrusion and Cyber-Attack Detection, International Journal of Advanced Research in Computer and Communication Engineering, 3(6), 23-33.

Mohammad, N. M., Ulaiman, N. and Khalaf, E. T. (2011), A Novel Local Network Intrusion Detection System Based on Support Vector Machine, Journal of Computer Science, 7(10), 1560-1568.

Parag K. (2012), Intrusion Detection System for Cloud Computing, International Journal of Scientific & Technology Research,1(4), 12-16.

Qurashi M (2020), Intrusion Detection in IPv6-enabled Sensor Networks, PhD thesis submitted to the faculty of science and technology computing and informatics,1-120, https://eprints.bournemouth.ac.uk/34641/1/AL%20QURASHI%2C%20Mohammed_Ph.D._2020.pdf

Rakshe, T. and Gonjari, V. (2017), Anomaly based Network Intrusion Detection using Machine Learning Techniques, International Journal of Engineering Research & Technology (IJERT), 6(5), 12-22.

Rastegari S. (2015), Intelligent network intrusion detection using an evolutionary computation approach computation approach, Samaneh Rastegari Edith Cowan University, 23-30.

Pelechrinis K. and Lappas T (2019). , Data Mining Techniques for (Network) Intrusion Detection Systems, Department of Computer Science and Engineering UC Riverside, Riverside CA, 925(21), 1-13

Sheikh A.I., Kewadkar, P. and Gupta, H. (2013), Analytical Study on Hybrid Approach towards Intrusion Detection System for Wireless Sensor Network, International Journal of Advanced Research in Computer and Communication Engineering, 2(10), 3863- 3868

Solanki, S., Gupta, C. and Rai, K.(2020), A Survey on Machine Learning based Intrusion Detection System on NSL-KDD Dataset, International Journal of Computer Applications, 176(20), 121-128.

Surantha N. and Wicaksono W. (2019), An IoT based House Intruder Detection and Alert System using Histogram of Oriented Gradients", Journal of Computer Science, 31(41), 1101-1121.

Usha D. And Suganthi S. (2018), A Survey of Intrusion Detection System in IoT Devices, International Journal of Advanced Research (IJAR): 23, 12-23.

Wazid M. (2017), Design and Analysis of Intrusion Detection Protocols for Hierarchical Wireless Sensor Networks, Design and Analysis of Intrusion Detection Protocols for Hierarchical Wireless Sensor Networks, 1(23), 23-28.

Yao J., Zhao, S. and Fan, L.(2020), An Enhanced Support Vector Machine Model for Intrusion Detection, Lecture Notes in Computer Science, 1-7, file:///C:/Users/ZIWERITIN/Downloads/An_Enhanced_Support_Vector_Machine_Model_for_Intru.pdf

Yasmeen S. Almutairi, Y. S., Alhazmi, B. and Munshi, A. A.(2022), Network Intrusion Detection Using Machine Learning Techniques, Advances in Science and Technology Research Journal, 16(3), 193–206