A Comparative Analysis of Adaboost and XGBoost Meta-Algorthms for Improving Network Security

eISSN: 2408-1906© 2020 AU eJIR.
http://www.assumptionjournal.au.edu/index.php/eJIR/index

A Comparative Analysis of Adaboost and XGBoost Meta-Algorthms for Improving Network Security

Akinrotimi Akinyemi Omololu*, Ogundokun Roseline Oluwaseun, Awotunde Joseph Bamidele, Mabayoje Modinat Abolore, Oladele Tinuke Omolewa,Ameen Ahmed Oloduowo, Akintola Ganiyat Abimbola

Received: June 1, 2024. Revised: December 10, 2024. Accepted: December 24, 2024.

Abstract

Purpose: This study explores network security, focusing on the application of advanced machine learning algorithms to enhance Intrusion Detection Systems. The increasing frequency of network attacks necessitates robust defense mechanisms and sophisticated methodologies.
Research design, data and methodology: The CRISP-DM framework and a multi-class dataset from the NSL KDD Cup Dataset were used for feature selection and boosting. Adaboost outperformed XGBoost in accuracy, error rate, precision and sensitivity, highlighting the relationship between algorithmic selection and high detection rates in intrusion detection systems. The study emphasizes the importance of considering diverse machine learning models and datasets to refine and advance intrusion detection techniques. The research highlights the evolving landscape of network security and encourages further exploration and integration of various machine learning models and datasets into intrusion detection methodologies.
Conclusions: The results of this study shows that the XGboost algorithm outperforms the Adaboost algorithm achieving an accuracy and sensitivity of 99.93% and 89.02% respectively compared to the Adaboost algorithm with an accuracy and sensitivity of 99.86% and 67.45% respectively, in detecting real-time network intrusions. Real-world implementations are encouraged, focusing on scalability and adaptability. The synergy of advanced machine learning algorithms, meticulous feature selection, and robust methodologies is a promising avenue for fortifying network defenses and ensuring critical system security. This study contributes to the ongoing dialogue on network security, advocating for a proactive approach in refining and implementing intrusion detection systems.

Keywords: Keywords Algorithm, Dataset, Intrusion, Machine Learning, Models, Network Security.

1* Akinrotimi Akinyemi Omololu, Dept. of Computer Science, College of Pure and Applied Sciences, Caleb University, Lagos State, Nigeria. Email: akinrotimi.akinyemi@calebuni versity.edu.ng
2 Ogundokun Roseline Oluwaseun, Dept. of Computer Science, College of Pure and Applied Sciences, Landmark University, Kwara State, Nigeria. Email: ogundokun.roseline@lmu.edu.ng
3 Awotunde Joseph Bamidele, Dept. of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Kwara State, Nigeria. Email: awotunde.jb@unilorin.edu.ng
4 Mabayoje Modinat Abolore, Dept. of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Kwara State, Nigeria. Email: mabayoje.ma@unilorin.edu.ng
5 Oladele Tinuke Omolewa, Dept. of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Kwara State, Nigeria. Email: oladele.to@unilorin.edu.ng
6 Ameen Ahmed Oloduowo, Dept. of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Kwara State, Nigeria. Email: aminamed@unilorin.edu.ng
7 Akintola Ganiyat Abimbola, Dept. of Computer Science, Faculty of Communication and Information Sciences, University of Ilorin, Kwara State, Nigeria. Email: akintola.ag@unilorin.edu.ng

1. Introduction

In Network traffic security is swiftly becoming one of the primary concerns for computer networks due to the internet's rapid development. Every day there are more attacks on networks. An incursion is the network traffic attack that has gotten the most attention. To locate intrusions and protect information security aims against hazards, an intrusion detection system has been applied. Traditional intrusion detection systems have shortcomings and are unable to handle the problem altogether (Youssef et al., 2019). They search network data for potentially dangerous activity, occasionally identifying anomalies and genuine security threats. False positives, however, arise routinely, and they generate alerts when there is nothing wrong with the network or fail to detect malicious activities.

Intrusion detection systems (IDS) are commonly utilized as an additional layer of safety for information systems since they function as instruments for monitoring and detecting computer breaches. One of the many reasons why intrusion detection is such an important aspect of the overall defensive system is the limits of many traditional systems and applications that are created without taking into account the security concerns of the environment where the systems are deployed. A secure isolated system that is connected to the Internet stands the danger of losing its security. IDS track efforts to take advantage of these security flaws in program architecture. The limitations of information security and software engineering methodologies are another challenge, since they enable cyber attackers to undermine the security of the systems or applications by leveraging faults or vulnerabilities that result from early system design blunders. Based on their techniques of detection, IDSs can commonly be categorized into two categories: anomaly detection and abuse detection (Joldzic et al., 2016). Anomaly detection is designed primarily to seek for variations from a profile's typical behavior in order to find risky activities. Usually, anomaly-based Intrusion Detection System (IDS) uses known baseline to detect patterns which have deviated from normal behavior (Hatungimana, 2018). These IDSs are better at identifying novel attack types, but they were unable to reduce the quantity of false positives (FP) that they created (Papamartzivanos et al., 2018). Abuse detection, on the other hand, can successfully identify hazardous from valid occurrences based on known patterns (Kim et al., 2016). Although these IDSs may reliably identify known attacks, they cannot distinguish between unknown assaults and known attack variations.

Unfortunately, as attackers improve, new threats and vulnerabilities arise. There is an immediate and large rise in critical infrastructure damage risk. However, intrusion detection systems (IDS) have been upgraded to detect and respond to new assaults. Therefore, numerous strategies have been investigated and developed to boost IDS efficacy and detection rate. This list includes ML technology. This works with abuse and anomaly detection models (Mishra et al., 2018). An IDS must analyze network traffic going through critical network nodes to distinguish malicious from benign traffic and determine the type of attack in the protected system.

The data training method affects machine learning model quality. Data preprocessing like feature selection and normalization produces good training data. Feature selection is based on how important each feature's estimated value is to the data label (Jovic et al., 2015) data minimization removes outliers' data (Saini & Sharma, 2018).

Network security relies on intrusion detection systems (IDSs). Due to network data's high dimensionality, current security systems often get false alarms and have poor intrusion detection accuracy. IDSs struggle to evaluate and detect actual intrusion events due to the volume of data moving over networks, resulting in false positives and negatives. The issue is twofold. IDSs struggle to analyze and identify intrusion datasets due to their high dimensionality, which is defined by many attributes. Too many features can cause noise, redundancy, and unnecessary information, slowing detection and increasing false alarms. Second, excessive network data worsens the issue. The constant intake of data overwhelms IDSs with vast amounts of data to process and evaluate in real time. Performance, efficiency, and real intrusion event detection suffer. As a result, the study used a thorough test environment to discover methods for improving intrusion detection performance while also shortening intrusion detection length and employing the use of Meta-huristic algorithms such as XGboost and Adaboost. This research presents an effective intrusion detection system that identifies cyber-attacks with high accuracy and efficiency by utilizing K-Best for feature selection and performing comparative boosting techniques on intrusion detection network datasets.

2. Literature Review

Research on network intrusion detection systems is not new and quite a number of algorithms and combination of algorithms have been used in developing network intrusion detection models and systems. Table 1.0 showcases different related work on the subject matter, identifying the authors of the various studies, the algorithms used by the authors, the major contribution(s) of their work as well as the limitations.

Table 1: Related Work on Network Intrusion Detection Systems

Authors/Year	Algorithms Used	Major Contribution(s)	Limitation(s)
Yu et al. (2021)	The authors developed a deep learning model using a convolutional neural network (CNN) and long short-term memory (LSTM) for anomaly detection. Evaluated on the CICIDS2017 dataset.	Achieved accuracy of 98.5% and F1-score of 98.1%, demonstrating high effectiveness in identifying anomalies.	Requires significant computational resources and may be vulnerable to adversarial attacks.
Al-Hajri et al. (2021)	This study proposes an ensemble method combining three machine learning algorithms: logisticregression, Naïve Bayes, and decision trees. The individual models learn from the training data, and their predictions are combined to improve the overall accuracy and robustness of the intrusion detection system.	The authors achieved an accuracy of 97.2% using their ensemble model, demonstrating improved performance compared to individual algorithms like Naïve Bayes (94.1%).	As a limitation noted, the authors of the study mentioned the need for further evaluation on larger and more diverse datasets. However, the effectiveness against zero-day attacks (previously unknown attacks) might require further investigation.
Mohammed and Hussein (2022).	The authors proposed an ensemble learning model that combines Decision Trees and Logistic Regression classifiers in developing an Intrusion Detection System.	The proposed ensemble model achieved an accuracy of 99.25% on the NSL-KDD dataset, a benchmark dataset for intrusion detection research. The ensemble model outperformed individual models like Decision Trees (98.75% accuracy) and Logistic Regression (98.50% accuracy), demonstrating the benefit of combining these techniques.	The study doesn't explicitly report the FPR. A high FPR can lead to labeling normal traffic as intrusions, causing unnecessary system disruptions. Further research is needed to optimize the model's ability to minimize false positives while maintaining high accuracy.
Hidayat et al. (2023)	The authors applied the following Machine Learning algorithms in developing an intrusion detection system: Decision Tree, AdaBoost, K-Nearest Neighbors algorithm (KNN) and Multilayer Perceptron (MLP).	The summary of the performance of the algorithms used (in terms of accuracy) are as follows: decision tree (99.6%), AdaBoost (99.98%), KNN (99%), and MLP (99.2%).	There was no implementation of Intrusion detection for unknown attacks in the network in real time.
Almutairi et al. (2022)	Four Machine Learning Algorithms: RF, NB, (J48), and SVM were used by the authors in this study in detecting network intrusion.	Performance accuracy for the Algorithms RF, NB, J48, and SVM are: 97.9%, 87.4%, 97.4%, and 96.4% respectively.	Although the presented satisfactory results and are capable of Intrusion detection, however, the NSL-KDD benchmark data set used suffers from several issues, such as imbalanced classes and the recorded malicious traffic and as previously noted above, they are synthetic, do not reflect real-world attacks.
Yadav and Ningshen (2023)	The authors compared the performance of various machine learning algorithms for intrusion detection using the NSL-KDD dataset. Algorithms used: Support Vector Machine (SVM), I Bayes (NB), Logistic Regression (LR), and the proposed Ensemble Model (combining multiple learning algorithms). Feature Selection: Chi-squared feature selection method was also used. With the target variable (i.e., attack type). The researchers used Chi-square due to its strength in multi-class classification problems.	Chi-squared feature selection method used, helped identify and prioritize the most important features in the dataset that correlate strongly. All algorithms achieved higher accuracy for DoS (Denial-of-Service) attacks, ranging from 82.7% to 94.54%. Performance for other attack types (Probe, R2L, U2R) varied significantly, with lower accuracy for R2L and U2R attacks. Performance: The proposed Ensemble Model achieved the highest overall accuracy: DoS: 99.91% Probe: 99.60% R2L: 99.90%	The research uses the NSL-KDD dataset, which is a well-known benchmark but is relatively old (1999) and might not reflect modern attack patterns. This could limit the model’s generalizability to real-world scenarios. Lack of real-world data evaluation and Limited scope of attack types are also limitations discovered in the study.

Authors/Year	Algorithms Used	Major Contribution(s)	Limitation(s)
		U2R: 98.15% Other supervised learning algorithms also exhibited good performance, with NB and LR achieving higher accuracy than SVM for most attack types.	
Awajan (2023)	Authors proposed a novel deep learning architecture called DeepIoT-IDS is proposed for intrusion detection in IoT networks. In this study, Network traffic is preprocessed through normalization and feature extraction. The UNSW-NB15 dataset is used for training and testing the model and the metrics: Accuracy, Precision, Recall and F1-score are used to evaluate the model's performance.	DeepIoT-IDS achieved an accuracy of 99.15% in identifying various attack types, including DoS, DDoS and user-to-root attacks. Compared to previous deep learning models, Deep-IoT IDS showed better performance in terms of accuracy, especially for low-rate attacks.	The study only evaluates the model on a single dataset, limiting generalizability to other datasets and real-world scenarios.
Wang et al. (2023)	The study proposes a lightweight intrusion detection system for IoT devices by combining Deep learning, Dynamic Quantization and Feature Reduction. Three benchmark datasets (CIC IDS2017, N-BaIoT and CICIoT2023) are used for training and evaluation.	Dynamic quantization significantly reduced model size and computational requirements, making it suitable for resource-constrained IOT devices. The combination of deep learning, dynamic quantization and IPCA offered a good balance between accuracy and resource efficiency.	The research focuses on specific deep learning architecture, limiting direct application to other models. While multiple datasets were used, further exploration on diverse real-world scenarios with different attacks and traffic patterns is needed.
Lee and Samantha (2024)	The authors applied k-means clustering to network flows to segment traffic into normal and anomalous groups. Flows identified as anomalous indicated potential intrusions.	On the CICIDS2017 dataset, their approach achieved over 85% accuracy in segregating intrusive traffic from normal flows with a low false positive rate of 5%.	NIL

3. Research Methods and Instruments

The developed system leverages a CRIPS-DM technique which represents an idealized chronology of occurrences. In practice, many of the tasks can be accomplished in any order, and it is usually essential to return to prior jobs and repeat key steps. The approach does not aim to capture all feasible data mining routes. The following data mining process were engaged.

1. Data Collection
2. Data Pre-processing
3. Data Normalization
4. Feature Engineering Selection
5. Data Classification (Classification of selected features using the boosting algorithms).
6. Performance Evaluation

The stepwise approach in achieving the stated objectives of this research is highlighted below:

3.1 Data Collection

This project used 25,192 instances from the NSL KDD Cup Dataset with four primary attack classes and the regular non-attack class for its system experimental set up, along with 41 attributes. NSL-KDD is a standard dataset in network intrusion detection research because it resolves some limitations of the original KDD'99 dataset, such as reducing redundancy and balancing records of various classes, with a variety of attacks. Therefore, this dataset is appropriate for evaluating the efficiency of machine learning models in detecting network intrusions.

3.1.1 Dataset Attacks (Class label)

The dataset is grouped under the following non-attack and sub-attacks category:

Table 2: Attack Labeling

ATTACKS	DATA LABELLING
Normal	1
DOS (Denial of Service)	2
Probe	3
U2R (User to Route)	4
R2l(Remote to Local)	5

3.2 Data Pre-processing

Pre-processing data for modeling includes removing mistakes and outliers, inconsistent data, and category variables to numeric representation.

3.3 Data Normalization

Normalization is used to prepare machine learning data. Normalization converts numeric columns in the dataset to a similar scale without distorting ranges or losing information. Min-Max will be used for this investigation. Every feature is linearly rescaled to [0, 1] via the min-max normalizer. Min-max scaling employs a formula to replace every column value with a new value, like z-score normalization. Here, the formula is:

m = (x - xmin) / (xmax - xmin) (3.1)

Where:

-m is our new value

-xmin is the minimum value of the column

-xmax is the maximum value of the column

3.4 Feature Engineering Selection

The KBest technique was used to select optimal features from the normalized dataset.

3.4.1 Kbest Method (Chi-Square)

The KThe chi-square test determines two variables' reliance using statistical independence. It is similar to the coefficient of determination, R². Chi-square test only works with categorical or nominal data. Every feature variable and the target variable were compared using chi-square statistics, revealing a link. If the target variable is independent of the feature variable, discard it. If dependent, feature variable was chosen.

Using the formula of Chi Square test:

$$x^2 = \frac{1}{d} \sum_{k=} \frac{(o_{k-} E_k)}{E_k} \quad (3.2)$$

3.5 Data Classification

The Classifying features like buildings is the goal of feature classification. It could be used to assess natural catastrophe damage to a building. The classification stage was implemented using Adaboost and XGboost machine learning algorithms in a comparative manner. The decreased characteristics were fed into the boosting approach at 75% for training and 25% for testing. 75% of the dataset lets each boosting algorithm develop a knowledge retention pattern based on the NSL KDD CUP tuned, while 25% evaluates the predictive effectiveness of the two boosting methods using machine learning measures.

3.5.1 Adaboost Algorithm

AdaBoost adapts weak learners' settings to perform better on misclassified data. It may overfit less than other learning algorithms in some cases. To prove convergence to a strong learner, weak learners must do marginally better than random guessing. Below is Adaboost pseudocode:

```
Given: (x1; y1),…..,(xm;ym), xi ϵ X, yi ϵ Y ={-1,1}. (3.3)
Initialize D1 (i) = 1/m.
For t = 1…….. T:
    1. Train weak classifier using distribution Dt
    2. Get weak hypothesis ht:X        {-1,1} with error
       $\sum_{1:h_t}^{n} \neq Dt(x_i)$        (3.4)
    3. Choose αt = (1/2) log [(1-εt)/(εt)]
    4. Update: If instance I is correctly classified
Dt + 1(i) = [Dt (i)/ Zt (i)] = $\begin{cases} e^{-\propto_t \omega} \\ hixt \\ e^{\propto_t} \end{cases}$        (3.5)
    Where Zt is a normalization factor chosen so that
$\sum_{-}(i = 1)^{\wedge} D_{t+1} = 1$        (3.6)
    Output the final hypothesis:
H(x) = $(\sum_{-}(t = 1)^{\wedge} T_{\propto tht}(x)) = 1$        (3.7)
```

3.5.2 Xgboost Algorithm

The XGBoost is the greatest option for event accuracy due to its strong predictive power, integration of a linear model and a tree learning method, and it's roughly ten times faster speed than gradient booster techniques. Objective functions like regression, classification, and ranking are supported. The fact that XGBoost is a regularized boosting approach is intriguing. This reduces data overfitting. We will train and predict data using this cutting-edge machine learning technique.

The Xgboost algorithm is shown here:

```
Input: training set{(xi,yi)} a differentiable loss function L(y, F(x)), number of iteration M.
Algorithm:
    1. Initialize model with a constant value:
       F0(x) = arg min$\sum_{i=1}^{n} L(yi, \gamma)$        (3.8)
    2. For   m = 1 to M:
       2.1 Compute so-called pseudocode-residuals:
            ϒim  = - $\frac{\partial L(yi, F(xi))}{\partial F(xi)}$-
            For I = 1,……….,n.        (3.9)
```

F(x) = F m-1(x)

2.2 Fit a base learner (e.g tree) hm (x) to pseudo-residual, i.e train it using the training set

{(xi,γ_{im})} Υim)} [-(i=1)^n] (3.10)

2.3 Compute multiplier ϒim by solving the following one-dimension optimization problem:

ϒim= arg min $\sum_{i=1}^{n} L(yi, Fm - 1(x1) + \gamma hm)(xi)$ (3.11)

2.4 Update the model:

Fm(x) = Fm-1 (x) + ϒmhm(x) (3.12)

3. Output FM (x) (3.13)

3.6 Performance Evaluation

An in-depth comparison of Adaboost and XGboost Algorithms to ascertain which one is the best and most effective boosting algorithms are shown below. The evaluation parameters reveal classification results. Testing was done utilizing True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN), Error rate, Classification Accuracy, Sensitivity, Specificity, and Error Rate.

3.6.1 Classification Accuracy

Accuracy is the number of correct predictions divided by the total number of predictions.

Classification Accuracy (%)

$$=\frac{TP+TN}{TP+TN+FP+FN} X\,100 \quad (3.14)$$

3.6.2 Error Rate (EER)

Error rate can be calculated as the total number of incorrect predictions made on the test set divided by all predictions made on the test set.

$$\text{Error Rate} = \frac{Incorrect\ Predictions}{Total\ Predictions} \quad (3.15)$$

3.6.3 Sensitivity

Sensitivity is the Ratio of true positives to total (actual) positives in the data.

$$\text{Sensitivity} = \frac{TP}{TP+FN} \quad (3.16)$$

3.6.4 Specificity

Specificity is the Ratio of true negatives to total negatives in the data.

$$\text{Specificity} = \frac{TN}{TN+FP} \quad (3.17)$$

3.6.5 Precision

Precision is defined as the number of true positives divided by the total number of positive predictions (i.e., true positives + false positives).

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3.18)$$

3.6.6 Data Feature Engineering Selection

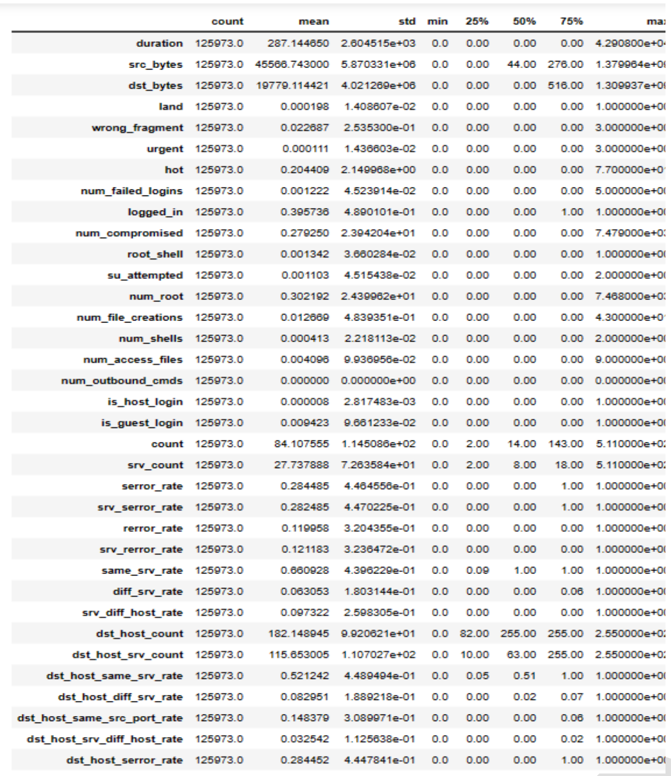
The feature engineering helps to pick significant features that have high predictive power to the response variable. Feature engineering was performed for both the multi class dataset. The selected features index shows the index location of the 16 selected features as well the corresponding feature name and score by the chi-square filter-based selector.

4. Results and Discussion

This section presents the results obtained from each section accordingly.

4.1 Explorative Data Analysis

The explorative data analysis helps to gain more insight into the dataset and also perform all necessary normalization and filtering technique to enhance the state of the dataset for better performance during the classification stage. The Data explorative processes is shown in the screen shot below: A summary statistic of each factor was obtained showing the count, mean, standard deviation, minimum and maximum values, 25% inter quarter ranges up till the 75% interquartile ranges. The result of the descriptive or summary statistics is shown in figure 1.



	count	mean	std	min	25%	50%	75%	ma
duration	125973.0	287.144650	2.604515e+03	0.0	0.00	0.00	0.00	4.290800e+0
src_bytes	125973.0	45566.743000	5.870331e+06	0.0	0.00	44.00	276.00	1.379964e+0
dst_bytes	125973.0	19779.114421	4.021269e+06	0.0	0.00	0.00	516.00	1.309937e+0
land	125973.0	0.000198	1.408607e-02	0.0	0.00	0.00	0.00	1.000000e+0
wrong_fragment	125973.0	0.022687	2.535300e-01	0.0	0.00	0.00	0.00	3.000000e+0
urgent	125973.0	0.000111	1.436603e-02	0.0	0.00	0.00	0.00	3.000000e+0
hot	125973.0	0.204409	2.149968e+00	0.0	0.00	0.00	0.00	7.700000e+0
num_failed_logins	125973.0	0.001222	4.523914e-02	0.0	0.00	0.00	0.00	5.000000e+0
logged_in	125973.0	0.395736	4.890101e-01	0.0	0.00	0.00	1.00	1.000000e+0
num_compromised	125973.0	0.279250	2.394204e+01	0.0	0.00	0.00	0.00	7.479000e+0
root_shell	125973.0	0.001342	3.660284e-02	0.0	0.00	0.00	0.00	1.000000e+0
su_attempted	125973.0	0.001103	4.515438e-02	0.0	0.00	0.00	0.00	2.000000e+0
num_root	125973.0	0.302192	2.439962e+01	0.0	0.00	0.00	0.00	7.468000e+0
num_file_creations	125973.0	0.012669	4.839351e-01	0.0	0.00	0.00	0.00	4.300000e+0
num_shells	125973.0	0.000413	2.218113e-02	0.0	0.00	0.00	0.00	2.000000e+0
num_access_files	125973.0	0.004096	9.936956e-02	0.0	0.00	0.00	0.00	9.000000e+0
num_outbound_cmds	125973.0	0.000000	0.000000e+00	0.0	0.00	0.00	0.00	0.000000e+0
is_host_login	125973.0	0.000008	2.817483e-03	0.0	0.00	0.00	0.00	1.000000e+0
is_guest_login	125973.0	0.009423	9.661233e-02	0.0	0.00	0.00	0.00	1.000000e+0
count	125973.0	84.107555	1.145086e+02	0.0	2.00	14.00	143.00	5.110000e+0
srv_count	125973.0	27.737888	7.263584e+01	0.0	2.00	8.00	18.00	5.110000e+0
serror_rate	125973.0	0.284485	4.464556e-01	0.0	0.00	0.00	1.00	1.000000e+0
srv_serror_rate	125973.0	0.282485	4.470225e-01	0.0	0.00	0.00	1.00	1.000000e+0
rerror_rate	125973.0	0.119958	3.204355e-01	0.0	0.00	0.00	0.00	1.000000e+0
srv_rerror_rate	125973.0	0.121183	3.236472e-01	0.0	0.00	0.00	0.00	1.000000e+0
same_srv_rate	125973.0	0.660928	4.396229e-01	0.0	0.09	1.00	1.00	1.000000e+0
diff_srv_rate	125973.0	0.063053	1.803144e-01	0.0	0.00	0.00	0.06	1.000000e+0
srv_diff_host_rate	125973.0	0.097322	2.598305e-01	0.0	0.00	0.00	0.00	1.000000e+0
dst_host_count	125973.0	182.148945	9.920621e+01	0.0	82.00	255.00	255.00	2.550000e+0
dst_host_srv_count	125973.0	115.653005	1.107027e+02	0.0	10.00	63.00	255.00	2.550000e+0
dst_host_same_srv_rate	125973.0	0.521242	4.489494e-01	0.0	0.05	0.51	1.00	1.000000e+0
dst_host_diff_srv_rate	125973.0	0.082951	1.889218e-01	0.0	0.00	0.02	0.07	1.000000e+0
dst_host_same_src_port_rate	125973.0	0.148379	3.089971e-01	0.0	0.00	0.00	0.06	1.000000e+0
dst_host_srv_diff_host_rate	125973.0	0.032542	1.125638e-01	0.0	0.00	0.00	0.02	1.000000e+0
dst_host_serror_rate	125973.0	0.284452	4.447841e-01	0.0	0.00	0.00	1.00	1.000000e+0

Figure 1: Screen-Shot of Summary Statistic

4.2 Data Preparation for the Multi Class Dataset

In this section the dataset for grouped into the multi-class within the range of 0 and 4 is shown. All the non- attack (Normal) class were grouped and formatted under the class group of 0 while the attack class were labeled to be within the range of 1-4 based on the four categories of attack types in NSL KDD Cup dataset. The figure 2 below shows the pie chart for the normal and abnormal class.

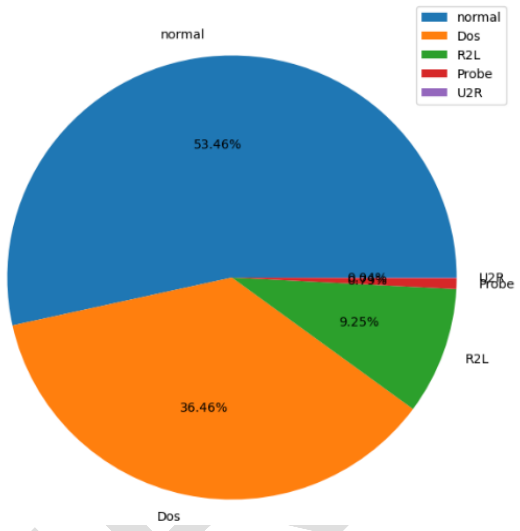


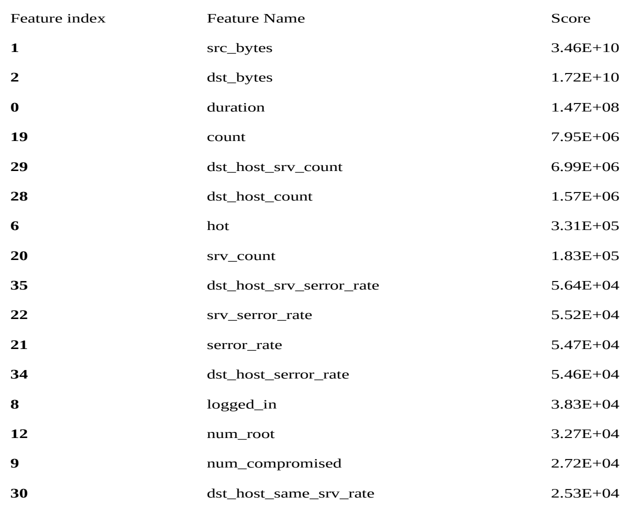
Figure 2: Pie Chart Distribution of Normal and Abnormal data

4.3 Data Feature Engineering Selection

The feature engineering helps to pick significant features that have high predictive power to the response variable. Feature engineering was performed for both the multi class dataset. The selected features index shows the index location of the 16 selected features as well the corresponding feature name and score by the chi-square filter-based selector.

Table 3: Comparative Evaluation Parameters for the XGBoost and Adaboost Algorithm (Multi Class).

Algorithm	**F-score**	**Recall**	**Precision**	**Specificity**	**Sensitivity**	**Accuracy (%)**	**Error Rate**
XGBoost (Multi Class)	1	0.9986	1	0.9995	0.9986	99.93	0.0007
AdaBoost (Multi Class)	1	0.9986	1	0.9995	0.9986	99.93	0.0007

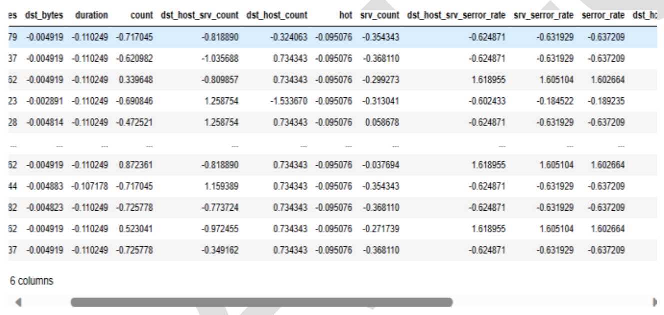


Feature index	Feature Name	Score
1	src_bytes	3.46E+10
2	dst_bytes	1.72E+10
0	duration	1.47E+08
19	count	7.95E+06
29	dst_host_srv_count	6.99E+06
28	dst_host_count	1.57E+06
6	hot	3.31E+05
20	srv_count	1.83E+05
35	dst_host_srv_serror_rate	5.64E+04
22	srv_serror_rate	5.52E+04
21	serror_rate	5.47E+04
34	dst_host_serror_rate	5.46E+04
8	logged_in	3.83E+04
12	num_root	3.27E+04
9	num_compromised	2.72E+04
30	dst_host_same_srv_rate	2.53E+04

Figure 3: Screen-Shot of Multi Class Selected Features by Chi-Square

4.4 Data Scaling

The Standard Scalar function is employed to transform the distribution of values in a dataset, such that the resulting distribution has a mean of 0and a standard deviation of 1. Sample of scaled intrusion dataset is shown below.



es	dst_bytes	duration	count	dst_host_srv_count	dst_host_count	hot	srv_count	dst_host_srv_serror_rate	srv_serror_rate	serror_rate	dst_h:
79	-0.004919	-0.110249	-0.717045	-0.818890	-0.324063	-0.095076	-0.354343	-0.624871	-0.631929	-0.637209	
37	-0.004919	-0.110249	-0.620982	-1.035688	0.734343	-0.095076	-0.368110	-0.624871	-0.631929	-0.637209	
62	-0.004919	-0.110249	0.339648	-0.809857	0.734343	-0.095076	-0.299273	1.618955	1.605104	1.602664	
23	-0.002891	-0.110249	-0.690846	1.258754	-1.533670	-0.095076	-0.313041	-0.602433	-0.184522	-0.189235	
28	-0.004814	-0.110249	-0.472521	1.258754	0.734343	-0.095076	0.058678	-0.624871	-0.631929	-0.637209	
...	...	...	...	...	...	...	...	...	...	...	
62	-0.004919	-0.110249	0.872361	-0.818890	0.734343	-0.095076	-0.037694	1.618955	1.605104	1.602664	
44	-0.004883	-0.107178	-0.717045	1.159389	0.734343	-0.095076	-0.354343	-0.624871	-0.631929	-0.637209	
82	-0.004823	-0.110249	-0.725778	-0.773724	0.734343	-0.095076	-0.368110	-0.624871	-0.631929	-0.637209	
62	-0.004919	-0.110249	0.523041	-0.972455	0.734343	-0.095076	-0.271739	1.618955	1.605104	1.602664	
37	-0.004919	-0.110249	-0.725778	-0.349162	0.734343	-0.095076	-0.368110	-0.624871	-0.631929	-0.637209	

6 columns

Figure 4: Screen-Shot of Scaled Dataset

4.5 Model Creation

The model was created by splitting the data into training set and testing set at a ratio of 75% training data and 25% testing data. The test size set at 0.25 denotes the partitioning range.

4.6 Experimental Results Evaluation

This section examines performance evaluation of the two boosting algorithms used, the table 3shows the comparative evaluation parameters of the XGBoost and AdaBoost for the multi-class dataset. The evaluation parameters for classification rate wereachieved using the Classification Accuracy, Sensitivity, Specificity and Error Rate.

4.6.1 Classification Accuracy

The classification accuracy shows the correct prediction rate attained by the boosting algorithm, the XGBoost is seen to outperform the Adaboost for multi class instances.

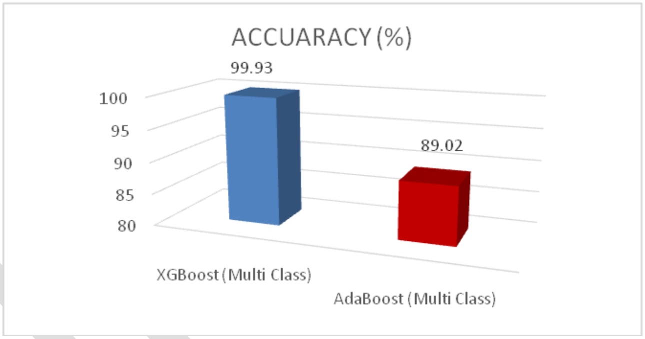


Figure 5: Comparative Classification Accuracy

4.6.2 Sensitivity and Specificity

The Sensitivity (SN) is calculated as the number of correct positive predictions divided by the total number of positives, Specificity (SP) is calculated as the number of correct negative predictions divided by the total number of negatives. The best sensitivity and specificity fall at 1. From the obtained results the best positive rate was attained by XGBoost Algorithm with a value of 0.9986 and negative rate of 0.9995 as compared to Adaboost which attained a rate of 0.9448 and 0.6745 respectively for multi class dataset.

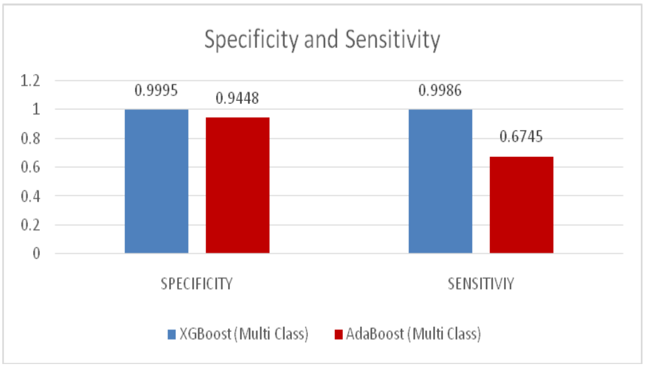


Figure 6: Comparative Sensitivity and Specificity Chart

4.6.3 Error Rate

The error rate shows the lowest possible error rate for any classifier in a random outcome during the classification.

The XGBoost algorithm shows the tremendously low error rate for the multi class dataset with an error rate of 0.0007 and 0.1098 attained by the XGBoost and the Adaboost algorithms respectively.

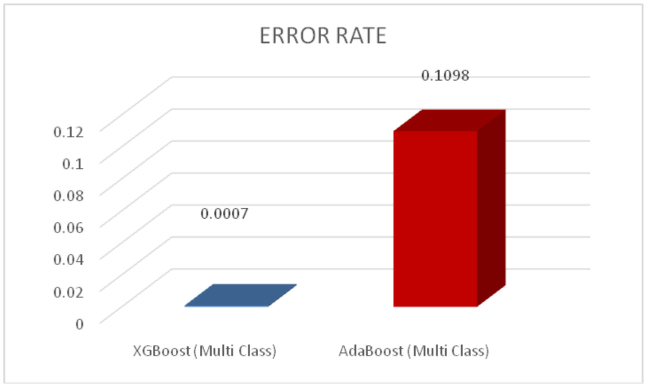


Figure 7: Comparative Error Rate

4.6.4 Precision and F1 Score

The F1 Score is calculated as the harmonic mean of precision and recall, serving as a balanced measure of both measurements. It is particularly advantageous when seeking to achieve an equilibrium between incorrect positive results and incorrect negative results. Precision is a metric that evaluates the accuracy of a model in correctly identifying positive instances while minimizing the occurrence of incorrectly categorizing negative examples as positive. Apparently from the same F1-Score was obtained for the boosting algorithms while the Xgboost shows the better performance in terms correctly identifying positive instances and minimizing the occurrence of incorrectly categorizing negative examples as positive.

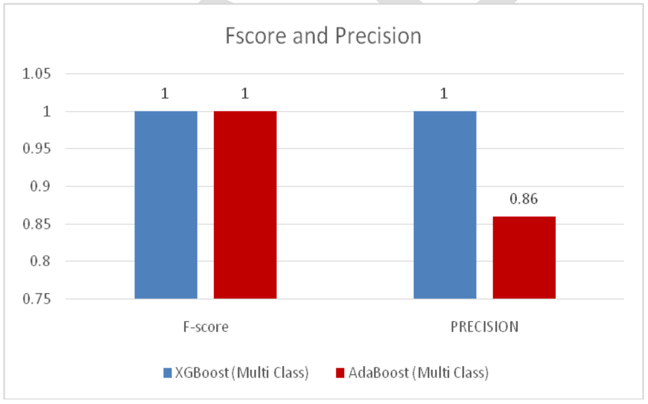


Figure 8: Comparative Error Rate

5. Conclusion

This study explores network security, focusing on improving intrusion detection systems (IDS) using machine learning algorithms. The increasing frequency of network attacks calls for robust defenses and the need for advanced methodologies. The CRISP-DM framework is used, with a multi-class dataset from the NSL KDD Cup Dataset as a basis. The past few years has witnessed the dense application of Artificial Neural Networks (ANN) to various areas in the field of science: such as, system identification and control function approximation, time series prediction and damage processing (Akinrotimi & Mabayoje, 2019). As such, the use of ANN would have also been an option for this study however we considered the use of Machine Learning techniques instead because they are simpler, faster, and more interpretable than artificial neural networks, rendering them suitable for small datasets (like the one used in this study). The K-Best method is used to select features and Adaboost and XGBoost are used for boosting. The results shows that the XGboost algorithm outperforms the Adaboost algorithm achieving an accuracy and sensitivity of 99.93% and 89.02% respectively compared to the Adaboost algorithm with an accuracy and sensitivity of 99.86% and 67.45% respectively, in detecting real-time network intrusions, thereby highlighting the importance of algorithmic choice in achieving high detection rates. The study suggests further exploration and refinement of intrusion detection techniques, incorporating diverse machine learning models and datasets. Real-world implementations and considerations of scalability and adaptability are crucial for translating findings into practical solutions. The combination of advanced machine learning algorithms, thoughtful feature selection, and robust methodologies offers a promising avenue for enhancing network defenses and ensuring the integrity and security of critical systems.

6. Future Research Considerations

For scaling these findings to modern large-scale networks, further studies could focus on optimizing XGBoost for distributed computing environments such as Apache Spark that handle high data volumes, adapt these models for real-time intrusion detection by integrating them with streaming platforms such as Apache Kafka and improve their robustness by incorporating adaptive retraining mechanisms to handle evolving threats and network dynamics.

References

Akinrotimi, A. O., & Mabayoje, M. A. (2019). A Neural Network Approach in Performing Non-Invasive Analysis of Muscle Impulses For Hand Gesture Recognition. Annals. *Computer Science Series*, *17*(2), 104-108.

Al-Hajri, A., Zidan, M. A., Al-Nashash, A. H., & Habib, M. H. (2021). A new ensemble-based intrusion detection system for Internet of Things. *Arabian Journal for Science and Engineering*, *46*(12), 13241-13252. https://link.springer.com/article/10.1007/s13369-021-06086-5

Almutairi, Y. S., Alhazmi, B., & Munshi, A. A. (2022). Network intrusion detection using machine learning techniques. *Advances in Science and Technology Research Journal*, *16*(3), 193-206.

Awajan, A. (2023). A Novel Deep Learning-Based Intrusion Detection System for IOT Networks. *Computers, 12*(2), 34.

Hatungimana, G. (2018). Pairwise clusters optimization and cluster most significant feature methods for anomaly-based network intrusion detection system (POC2MSF). *Malaysian Journal of Computing (MJoC), 3*(2), 93-107.

Hidayat, I., Ali, M. Z., & Arshad, A. (2023). Machine Learning-Based Intrusion Detection System: An Experimental Comparison. *Journal of Computational and Cognitive Engineering*, *2*(2), 88-97. https://doi.org/10.47852/bonviewJCCE2202270

Joldzic, O., Djuric, Z., & Vuletic, P. (2016). A transparent and scalable anomaly-based dos detection method. *Computer Networks, 104*, 27-42. doi:10.1016/j.comnet.2016.05.004.

Jovic, A., Brkic, K., & Bogunovic, N. (2015). A review of feature selection methods with applications. In 38th Int. Conv. Inf. Commun. Technol. Electron. *Microelectron, 112*(5), 25-9.

Kim, J., Kim, J., Thu, H. L. T., & Kim, H. (2016). Long short term memory recurrent neural network classifier for intrusion detection. *International Conference on Platform Technology and Service (PlatCon)*, 1-5.

Lee, J., & Samantha, H. (2024). K-Means Clustering for Network Segmentation and Intrusion Prevention. *ACM Transactions on Intelligent Systems and Security, 6*(1), 2024.

Mishra, P., Varadharajan, V., Tupakula, U., & Pilli, E. S. (2018). A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE Communications Surveys & Tutorials*, *21*(1), 686-728.

Mohammed, S. Q., & Hussein, M. A. (2022). Performance Analysis of different Machine Learning Models for Intrusion Detection Systems. *Journal of Engineering, 28*(5), 61-91.

Papamartzivanos, D., Mármol, F. G., & Kambourakis, G. (2018). Dendron: Genetic trees driven rule induction for network intrusion detection systems. *Future Generation Computer Systems*, *79*, 558-574. doi:10.1016/j.future.2017.09.056.

Saini, O., & Sharma, S. (2018). A review on dimension reduction techniques in data mining. *Comput Eng Intell Syst*, *9*(1), 7-14.

Wang, Z., Chen, H., Yang, S., Luo, X., Li, D., & Wang, J. (2023). A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. *PeerJ Computer Science*, *9*, e1569.

Yadav, M. K., & Ningshen, M. (2023). *Enhancement of Intrusion Detection System using Machine Learning.* Central Research Laboratory Bharat Electronics Limited Ghaziabad.

Youssef, G., Li, K., Guo, Z., & Emam, A. (2019). Semi-Supervised K-Means DDoS Detection Method Using Hybrid Feature Selection Algorithm. *IEEE Access*, *7*(1), 64351-64365. doi:10.1109/ACCESS.2019.2917532.

Yu, S., Wang, L., Liu, Y., Wang, Y., Zhou, Y., & Chen, Z. (2021). Deep Learning Based Anomaly Detection for Network Intrusion Prevention Systems. *IEEE Access, 9*, 147439-147451.