International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

NETWORK-BASED INTRUSION DETECTION SYSTEMS USING MACHINE LEARNING ALGORITHMS

Amin Lama
Student
Department of MCA
JAIN (Deemed-to-be University)
Bangalore, India

Dr. Preeti Savant (Project Guide)
Professor
Department of MCA
JAIN (Deemed-to-be University)
Bangalore, India

***Abstract*—The importance of network security in today's information era cannot be underestimated. There is a more significant risk of network penetration than ever before due to the rapid increase of network-enabled devices. Machine learning (ML) algorithms have lately sparked a lot of interest in network security because of their rapid growth and noteworthy outcomes in a variety of fields. The network-based intrusion detection system (NIDS) has many potentials to be the final line of defense against intrusions in today's information communication technology (ICT) age, and it's an essential aspect of network security. Intrusion detection datasets are publicly available due to the dynamic nature of attacks. The CSE-CIC-IDS2018 on AWS dataset, which contains real-life modern network traffic, was used for this research. This paper employed different machine learning techniques to conduct binary classification on this dataset, including Logistic Regression, Random Forest, and Gradient Boosting. I have gone through the dataset in detail, including data cleaning, pre-processing, feature engineering, and feature selection. The features selected after feature engineering are sent for training, and four different classifiers are generated; baseline, logistic regression, random forest, and gradient boosting. Comprehensive comparisons were made among generated models utilizing evaluation metrics. To evaluate the performance of my research, I used evaluation metrics such as recall (weighted average) and precision (weighted average). Gradient boosting surpassed all the measures, indicating superior to other models with precision, recall and f1-score of 0.98. The model was also tested on the test dataset, and it achieved similar values, which proves the model performs well on data that hasn't been seen before.**

***Keywords*—intrusion detection system (ids), machine learning, network security, network-based intrusion detection system, logistic regression, random forest, gradient boosting, recall, precision, precision-recall curve.**

I. INTRODUCTION

An Intrusion Detection System (IDS) is used to identify network intrusion. It's a network security appliance that looks at all inbound and outgoing network traffic for unusual patterns that might signal a network or system security breach. When an IDS finds signatures in network traffic that match known intrusion patterns, it raises the alarm. Unlike a firewall, which only looks for intrusions from the outside, an IDS monitors the network from within. An IDS is deployed behind the firewall to detect intrusions, scanning every traffic for heuristics and pattern matching. The vast majority of today's IDSs fall into one of two groups. They are intrusion detection systems that are based on signatures and anomalies. An IDS based on signature monitors patterns of data packets in the network and compares them to pre-configured network attacks patterns. This method uses string comparison operations to compare ongoing activity, such as a packet or a log entry, against a list of signatures. Existing attacks are well identified by this IDS, while new (unseen) attacks are frequently missed. The next category is IDS based on the anomaly, which detects malicious traffic by looking for variations from conventional network traffic patterns. Next, it identifies anomalous system activity. In this approach, alarms for abnormal activities are generated by evaluating network patterns. For example, anomaly-based IDS monitors' normal internet bandwidth usage, failed logon attempts, processor utilization levels, etc.
Because of substantial advancements in information technology, the number of network-enabled devices linked to

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

the internet is expanding in size and accessibility. As a result, Network Security is extremely important in today's world. The possibility of a more significant surface assault increases as the number of networked devices grows. As a result, worldwide cybercrime costs are predicted to rise 15% yearly over the next five years, reaching $10.5 trillion in 2025, up from $3 trillion in 2015 [1]. It is vital to develop an effective intrusion detection system (IDS) that combats unauthorized access to network resources in order to protect information [2]. Therefore, IT administrators' primary responsibility has been to build secure networks.

However, many intrusion detection systems in use today have significant flaws. Signature-based detection systems can identify the most prevalent types of attacks, but they have the drawback of failing to detect new attack types. To overcome this constraint, a lot of research is being done on network-based intrusion detection systems that employ ML and deep learning to identify network anomalies in a more dynamic manner. For example, [2][3][4][5][6][7]proposed IDS using different machine learning algorithms on different datasets to detect an anomaly, [8][9][10]proposed IDS using neural network and deep learning. It's impossible to pick a single excellent work because there are many diverse experimental configurations. Furthermore, it is hard to compare those works because they all work on different datasets and follow different validation setups.

The main goal of this research is to create IDS classifiers based on machine learning techniques. In this research, three ML algorithms are used for training in train dataset, the performance is evaluated using validation dataset, and the best one is selected as a final estimator. The following are the work's features and contributions:

- The CSE-CIC-IDS2018 on AWS dataset was thoroughly explored, including data cleaning, pre-processing, feature engineering, and feature selection.
- The dataset was divided into train/eval/test with 80/10/10 percentage.
- The training dataset was used for training three different ML models.
- A comparison of performance was made using the validation dataset.
- The final estimator was selected, and testing was done on unseen data using the test dataset.

The remainder of the paper has been organized as follows. The related work on NIDS utilizing various machine learning techniques, including deep learning, is presented in Section II. In Section III, the proposed methodology is outlined. The work's implementation is described in Section IV. Section V contains the comparison, evaluation, and result. Finally, Section VI brings this paper to a close with a conclusion.

II. RELATED WORK

Many research has been done previously in this field. For my research, the idea evolved from gleaning information about the past and current research being carried out on IDS using different ML techniques.I spend a significant amount of time reading different research papers, consulting with peers and holding discussions with my project guide. Furthermore, I have published a Survey Paper[11]titled "A Survey On Network-Based Intrusion Detection Systems Using Machine Learning Algorithms". Abrief description of related work is shown in Table 1.

Table 1. Summary Table of the Reviewed Papers

Paper	Authors	Year	Dataset	Algorithm Used	Accuracy (%)
Machine Learning Based Intrusion Detection System	Anish and Sundarakantham [3]	2019	NSL-KDD	SVM and Naïve Bayes	SVM- 93.95 Naïve Bayes- 71.001
A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks	Chuanlong et al. [8]	2017	NSL-KDD	RNN	97.09
Building an Effective Intrusion Detection System by Using Hybrid Data Optimization Based on Machine Learning Algorithms	Jiadong et al. [6]	2019	UNSW-NB15	Random Forest	92.8
A Machine Learning Approach for Intrusion Detection System on NSLKDD Dataset	Iram et al. [2]	2020	NSL-KDD	SVM, RF, ETC, DT, KNN, MLP, LR and NB	Above 99 on RF, ETC, and DT
Enhanced Network Anomaly Detection Based on Deep Neural Networks	Naseer et al. [5]	2015	NSL-KDD	ELM, KNN, DT, RF, SVM, NB, QDA	DCNN and LSTM models showed exceptional performance with 85% and 89%

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

					Accuracy
Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks	Abdulsalam et al. [7]	2021	NSL-KDD	XGBoost	95.55
Artificial Intelligence based Network Intrusion Detection with Hyper-Parameter Optimization Tuning on the Realistic Cyber Dataset CSE-CICIDS2018 using Cloud Computing	Kanimozhi and Jacob [10]	2019	CSE-CIC-IDS2018	ANN	99.97

III. PROPOSED METHODOLOGY

This section describes the proposed system architecture and workflow of the machine learning model for developing NIDS. The basic methodology is shown in Figure 1.

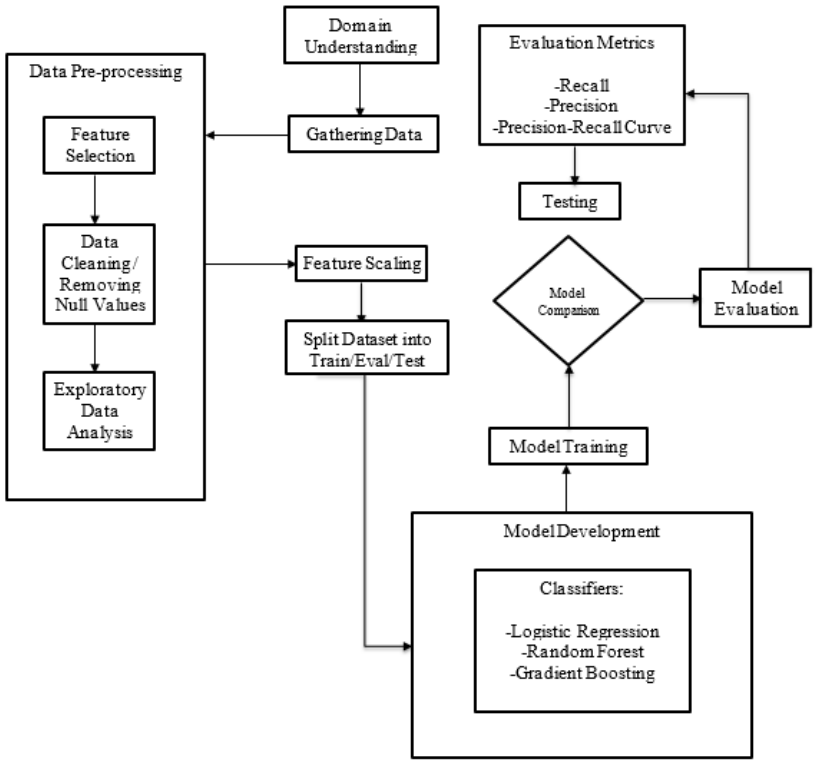

Figure 1. Block Diagram of Proposed NIDS

The workflow followed in this research is divided into different phases:

A. Data Gathering

For my research, I chose the "CSE-CIC-IDS2018 on AWS" dataset. The Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC) collaborated on the "CSE-CIC-IDS2018 on AWS" Dataset[12]. It is one of the latest publicly available datasets. It was built by recording all network traffic over the course of 10 days in a controlled network environment on AWS, where realistic background traffic and several attack scenarios were tested. As a result, the dataset includes both benign network traffic and samples of popular network assaults.

B. Data Cleaning

Before starting the analysis and training, the CSE-CIC-IDS2018 on AWS dataset has to be cleaned. The dataset is

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

composed of ten CSV files, each of which contains the recorded network traffic for a single day of operation and is named after the day on which the traffic was recorded. Data cleaning plays a significant role in the field of machine learning. It is the process of discovering the incomplete, inaccurate or missing data and then updating or deleting them as needed.

C. Data Pre-processing and Exploratory Data Analysis (EDA)

After the data cleanup, the next phase is data pre-processing and exploratory data analysis. For any machine learning project, data pre-processing and exploratory data analysis are crucial tasks. One of the first jobs is finding the missing values in a dataset. The features with too many missing values are removed. The infinity values of features are also dropped in this phase. New labels are also created in this phase if required for analyzing the dataset. After all these steps, we can visualize the dataset for getting more in-depth knowledge of the dataset and draw correlation heatmap.

D. Feature Scaling and Model Selection

Our dataset will often comprise features with a wide range of magnitudes, units, and degrees. However, the majority of machine learning algorithms compute the Euclidian distance between two data points. All features must be brought to the same magnitude level. This can be accomplished by feature scaling.

The use of labelled datasets to train algorithms to reliably identify data or predict outcomes is referred to as supervised learning.Since our dataset is labelled and we are making the model based on a classifier, different supervised algorithms are selected.

❖ Training and Testing the Models on data

In order to train a model, we first split it into three sections which are "Training", "Validation", and "Testing". We train the classifier using "training dataset" and then evaluate the model using validation dataset, and after that, a classifier is tested on an unseen "test dataset".

❖ Evaluation

A Recall, Precision, Precision-Recall curve and F1-Score is used to define the performance of a classification model. The confusion matrix is also used to show the comprehensive overview by summarizing the results. The precision-recall curve summarizes the tradeoff between precision and recall.

- Precision is a metric for the positive predictive value, and that is calculated using the following formula:

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

- Recall is a measure for the true positive rate, and that is calculated using the following formula:

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

- F-score is a harmonic mean for both recall and precision,and that is calculated using the following formula:

$$\text{F} - \text{score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

E. Model Comparision and Testing

This is the last phase in which different generated models are compared on the basis of the validation dataset, and the model which performs best of all models will be selected as the final estimator. Now, to predict real-world performance, the final estimator is evaluated on the test dataset.

IV. IMPLEMENTATION

I just wanted to do binary classification on the dataset. For running my models, I have used Google Colab Pro, where I got access to the fastest GPUs like Tesla T4 and P100 GPU. Along with that, I got ~32 GB system Ram and ~210 GB disk storage. The performance indicators used for evaluation and how the dataset was prepared for experimentation are discussed in the subsections below.

A. Data Gathering

CSE-CIC-IDS2018 on AWS dataset is chosen for this project. CSE-CIC-IDS2018 on AWS [13] is the most recent intrusion detection dataset, which is big data, publicly available, and covers a broad spectrum of attack types [14]. The Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC) collaborated on this dataset. There are seven attack scenarios included in the dataset. It consists of raw PCAP network captures and processed CSV files made using CICFlowMeter-V3 that provide 80 statistical aspects of individual network flows together with their labels.

Table 2. CSE-CIC-IDS2018 on AWS Dataset Description

Type	Train	Validation	Test
Normal	4861716	607715	607714
Anomaly	1736594	217074	217075
Total	**6598310**	**824789**	**824789**

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

The dataset was divided into train, validation, and test sets to use in this work. The same is shown in Table 2. Since there is an imbalance of data in several attack types, the machine learning model will be difficult to train for multiclass classification. So, I have considered binary classification for this study.

Only 29 features were selected out of 80 features from CSE-CIC-IDS2018 on AWS dataset. Table 3 provide the description of the selected 29 features used for the model generation.

Table 3. List of Selected Features and Its Description

Feature Name	Description
Protocol	Transport protocol
Flow Duration	Flow duration
Tot Fwd Pkts	Total packets in the forward direction
Tot Bwd Pkts	Total packets in the backward direction
TotLen Fwd Pkts	Total size of packet in forward direction
Fwd Pkt Len Mean	Mean size of packet in forward direction
Fwd Pkt Len Std	Standard deviation size of packet in forward direction
Flow Byts/s	flow byte rate that is number of packets transferred per second
Flow Pkts/s	flow packets rate that is number of packets transferred per second
Flow IAT Std	Standard deviation time two flows
Flow IAT Min	Minimum time between two flows
Fwd IAT Tot	Total time between two packets sent in the forward direction
Fwd IAT Min	Minimum time between two packets sent in the forward direction
Bwd IAT Tot	Total time between two packets sent in the backward direction
Bwd IAT Min	Minimum time between two packets sent in the backward direction
Fwd PSH Flags	Number of times the PSH flag was set in packets travelling in the forward direction (0 for UDP)
Bwd PSH Flags	Number of times the PSH flag was set in packets travelling in the backward direction (0 for UDP)
Fwd URG Flags	Number of times the URG flag was set in packets travelling in the forward direction (0 for UDP)
Bwd Pkts/s	Number of backward packets per second
RST Flag Cnt	Number of packets with RST
PSH Flag Cnt	Number of packets with PUSH
ACK Flag Cnt	Number of packets with ACK
URG Flag Cnt	Number of packets with URG
Down/Up Ratio	Download and upload ratio
Init Fwd Win Byts	Number of bytes sent in initial window in the forward direction
Init Bwd Win Byts	Number of bytes sent in initial

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

	window in the backward direction
Fwd Seg Size Min	Minimum segment size observed in the forward direction
Active Mean	Mean time a flow was active before becoming idle
Idle Mean	Mean time a flow was idle before becoming active

B. Data Cleaning

I used the following procedures to do data cleaning on the dataset.

❖ **Change The Data Type:**

While running the info() method, the dataframe showed that pandas inferred all columns as object data type rather than numeric data type, which would be more appropriate. So, all the columns data types are changed to suitable numeric data types.

❖ **Removal Of Duplicate Headers:**

The headers are present numerous times inside the input file, interwoven with the raw data rows, according to the analysis of the file. This is caused by creating a single CSV file by concatenating several CSV files while duplicating the headers. To address the problem, all headers having duplicate values are removed from the dataframe.

❖ **Substitution Of Occurance Of Infinity With Inf:**

Pandas only support infinity values in the "inf" format, as Pandas read_csv() method cannot interpret "infinity" values correctly. As a result, all instances that contain "infinity" are replaced with the string "inf".

❖ **Renaming The Column Names:**

In the next step, the column names are converted to lowercase with non-word characters removed for easier access to the columns. Here, I've imported the re module. The re module defines a collection of strings that match it; the methods in this module let us verify whether a particular string matches a regular expression [15].

C. Data Pre-Processing and EDA

This is the next phase of the project. After the data cleaning phase, the obtained data need to be pre-processed [16]. The steps performed in data pre-processing and exploratory data analysis are:

❖ **Dealing With Missing And Infinity Values:**

The dataset has inf values which will produce an error in computation. So, all columns containing inf values are identified first. There are two columns containing inf values, which are "flow_byts_s" and "flow_pkts_s". The columns containing inf values are transformed to be treated as null values by Pandas since there are other missing values in the dataset too. Finally, all null values are dropped.

❖ **Label Creation:**

New labels are created as a categorical data type to analyze the dataset in terms of binary classification (benign/attack)andmulti-classification (benign/attack-type).

- label_is_attack specifies if a network flow represents a benign or a malicious flow.
- label_is_attack_[attack_type] specifies if a network flow represents a certain type of attack.

❖ **Exploratory Data Analysis**

In EDA, different graphs are plotted for analysis using data visualization. Following sub-steps are performed under the EDA process:

- **Ratio Of Benign Network Flows And Malicious Flows:**

A simple bar graph is plotted showing the ratios between benign and malicious flows available in our dataset using matplotlib and seaborn library. This will help in the analysis of the binary classification. In Figure 2, 0 represents the benign network flows, and 1 represent malicious network flows. The dataset is substantially biased in favor of benign network flows, which make up for ~73.6% of total data.

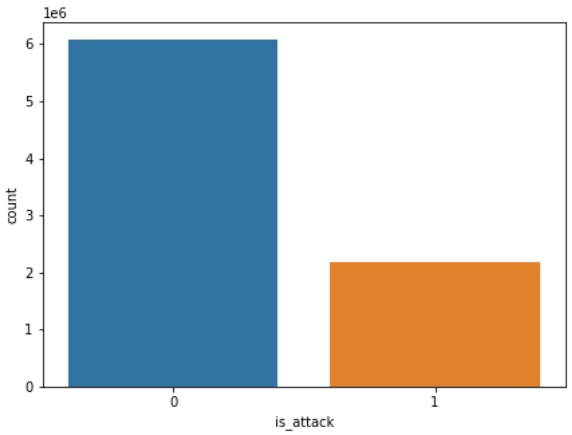

Figure 2. Ratio of Benign and Malicious Network Flows

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

- **Number Of Network Flows Per Attack Type:**

The following graph in Figure 3 shows the number of flows accounting for the different attack types. The chart illustrates an imbalance of attack types, so it will be difficult to train multiclass classification.

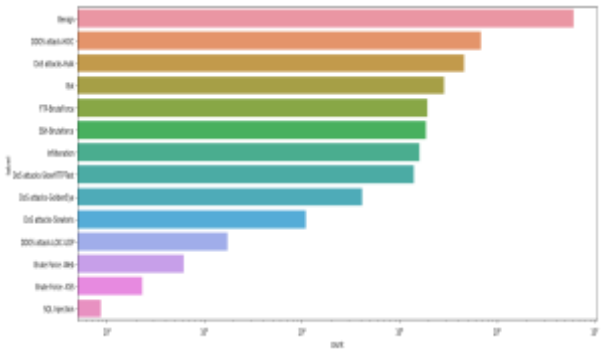

Figure 3. Number of Network Flows per Attack Type

- **Check For Feature Correlations:**

To analyze the correlation between different features, a heatmap is created on the overall correlation of the dataset. The heatmap shows a lot of strong correlations for a number of feature pairs.

- **Check For Features Having A Correlation With The Binary Class Of Network Flows:**

In the next step, features correlating to the binary attack label are identified. The features with the highest correlations are analyzed via their basic statistics and distribution to evaluate if those features might be relevant in predicting the label of the network flow.

D. Feature Engineering

After the data is pre-processed, we need to perform feature engineering to choose and transform the features which can be used in a model generation [17]. The dataframe is split into the predictor variables X and the target variables y. The steps performed in feature engineering are:

❖ **Removing Zero Variance Features:**

All features with zero variance are detected first and then removed as the initial step in feature engineering since they will have no effect on the model's prediction. Pandas is used to construct descriptive statistics [18] in order to locate such features. All features with a standard deviation of zero are chosen and will be removed later.

❖ **Dropping Undesired Features:**

To detect attacks regardless of time or the destination port they are conducted against, a classifier's predictions should be agnostic to the features timestamp and dst port. Thus, both features are removed from the dataset.

❖ **Features Clustering**

Spearman rank-order correlation [19] is performed to identify features of hierarchical clustering. After selecting a threshold, each cluster's single feature is maintained in the dataset.Heatmap is created visualizing the correlation between different pairs of features, and a dendrogram [20] is created illustrating the clustering of the features.

In the following stage, features with a high correlation between them are deleted. These features will not contribute any predictability, but they may generate noise.The heatmap demonstrates that the dataset has a fair amount of features with high correlation amongst each other. The cluster distance threshold of 1 is used to choose features from their respective clusters in order to eliminate duplicated features.

After removing all highly correlated features, the remaining features are selected for further analysis. There are only 29 features remaining in a pandas library.

E. Model Selection And Training Preparation

The main objective of this project is to build a machine learning model that can automatically classify and predict benign and malicious network flows. In this phase, the machine learning algorithms are selected, and then training and evaluation are performed in these algorithms. Three supervised machine learning algorithms are selected for this project which is: Logistic Regression, Random Forest and Gradient Boosting. The steps performed in this phase are:

❖ **Train/Test Split:**

A train/evaluation/test split is constructed to train and analyze different models with ratios of 80/10/10. The attack category is used to stratify the split, ensuring that all attacks are represented in the training and test sets according to their frequency in the dataset. To use algorithms expecting categorical values being one-hot-encoded, copies of the training and test data are created (X_train_oh, X_test_oh) with the feature protocol being one-hot-encoded. Table 2 shows the total value count of each type of network flows on every set.

❖ **Selecting Evaluation Metrics:**

The distribution of classes shows that the dataset is highly imbalanced, with class 0 - Benign contributing to ~73% of all the samples. For this reason, the metric accuracy is not suitable to measure the performance of a classifier based on this dataset as the accuracy of a dummy classifier reporting the class 0 - Benign would already be 0.74

Two metrics will be used to evaluate a classifier's performance. They are:

- The primary metric is Recall (weighted average), as the goal of the classifier should be to detect as many attacks as possible. This is the measure for which classifiers will be tuned.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

- Precision (weighted average) will be utilized as a secondary classifier to reduce the number of false positives. To have a maximum of 5% false positives, this measure should have a value greater than 0.95.

❖ Creating Baseline Classifiers:

A dummy classifier is developed[21]to set a baseline for all classifiers, using the class with the most frequent occurrences to make predictions. Because this is the overwhelming class of the dataset, all samples are classed as 0, i.e. Benign.Around 73 percent of all training samples fall into class 0, i.e. Benign, according to this classifier.

❖ Creating Logistic Regression Classifier:

The Logistic Regression implementation from scikit-learn will be used to fit a linear model to the data. The predictor variables are scaled using a StandardScaler in order to apply Logistic Regression. It is fitted on the train data, i.e. X_train_oh. StandardScaler converts values using the following equation[22]:

$$x = \frac{x - \mu}{\sigma}$$

Where μ and σ is the mean and standard deviation of the training samples respectively.

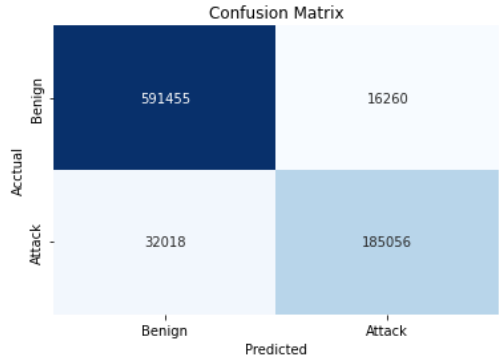

Figure 4. Confusion Matrix of Logistic Regression Classifier

Using Logistic Regression, a weighted recall of 0.94, a precision of 0.94, and an average precision of 0.822 is obtained, which is better than the baseline classifier but did not meet our requirement of atleast 0.95.

❖ Creating Random Forest Classifier

The Random Forest Classifier implementation from scikit-learn is the next algorithm to be evaluated. The estimator is trained using the default values.

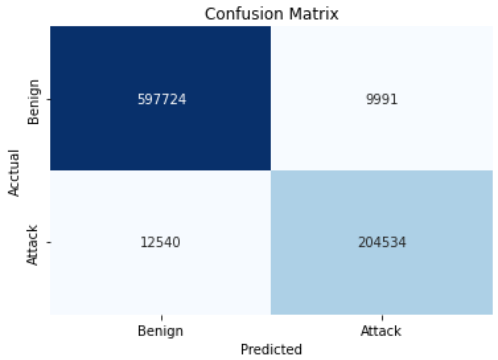

Figure 5. Confusion Matrix of Random Forest Classifier

With a recall of 0.97, an accuracy of 0.97, and an average precision of 0.913, the Random Forest Classifier performs well.

❖ Creating Gradient Boosting Classifier

Gradient Boosting, which is facilitated by the library CatBoost, is the final approach to be evaluated. In order to find the best parameters, a grid search with cross-validation over a variety of hyper-parameters is used.

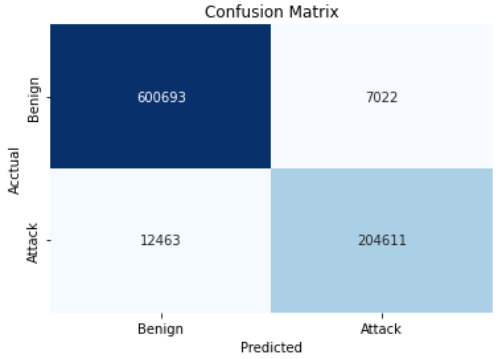

Figure 6. Confusion Matrix of Gradient Boosting Classifier

With a recall of 0.98, an accuracy of 0.98, and an average precision of 0.926, the Gradient Boosting Classifier performs excellently.

V. MODEL COMPARISON & RESULT

In this phase, all the build classifiers are compared and selected the best one. From Table 4, we can see that Gradient Boosting Classifier performed best of all models with a recall of 0.98, precision of 0.98, and average precision of 0.926, and it will be used as the final estimator.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 6, Issue 11, ISSN No. 2455-2143, Pages 145-155
Published Online March 2022 in IJEAST (http://www.ijeast.com)

Table 4. Model Comparison

Model	Recall	Precision	F1-score	Accuracy
Baseline	0.74	0.54	0.63	0.74
Logistic Regression	0.84	0.84	0.94	0.94
Random Forest	0.97	0.97	0.97	0.97
Gradient Boosting	**0.98**	**0.98**	**0.98**	**0.98**

To predict real-world performance, the final estimator is evaluated on the test dataset. The final estimator shows very good performance with a recall of 0.98 and a precision of 0.98.

Figure 7, Figure 8, and Figure 9 show the result of the final estimator. The models were mostly run on default parameters except for logistic regression, where StandardScaler is used.

```
[ ] print_report('Test', estimator, test_pool, y_test.label_is_attack, estimator.predict(test_pool), plot_pr=True, plot_cm=True)

Classification Report (Test):
              precision    recall  f1-score   support

           0       0.98      0.99      0.98    607714
           1       0.97      0.94      0.95    217075

    accuracy                           0.98    824789
   macro avg       0.97      0.97      0.97    824789
weighted avg       0.98      0.98      0.98    824789

Avg Precision Score: 0.9264819704559812
```

Figure 7. Performance of Final Estimator

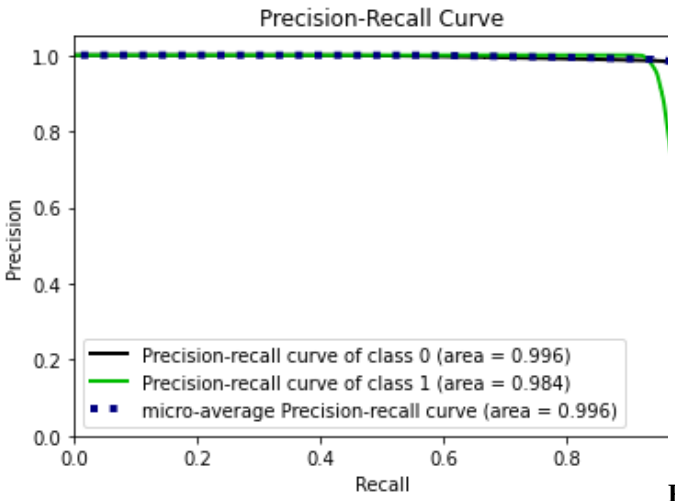

Figure 8. Final Estimator's P-R Curve

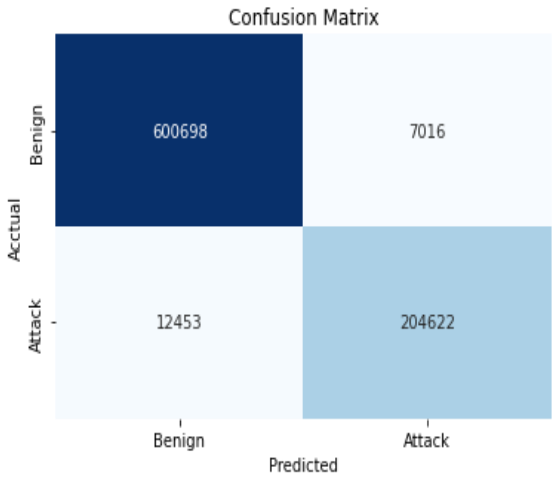

Figure 9. Final Estimator's Confusion Matrix

VI. CONCLUSION

In this research, I have presented a machine learning-based NIDS model for performing binary classification of the CSE-CIC-IDS2018 on AWS dataset. I have implemented logistic regression, random forest, and gradient boosting algorithms for misuse detection and found that gradient boosting performed better among others during evaluation. The model is built using gradient boosting and is tested with test data set and found to perform well on test data as well, validating the generalization of my model. The estimator shows a recall and precision of 0.98, which is acceptable for real-world usage.

I believe that my technique can be improved in the future. In this research, minority classes of attacks are often misclassified. Synthetic minority oversampling can be applied to the training dataset for these classes to gain better performance in the future. Additional tests can be done on novel data from different network environments in order to ensure that the estimator has as equally good performance as exhibited in the test dataset. CIC-IDS-2017 dataset will be used to perform additional tests, which comprises identical attack scenarios but are recorded in a different network environment.

VII. REFERENCES

[1] Steve Morgan, "Cybercrime To Cost The World $10.5 Trillion Annually By 2025," Cybercrime Magazine, Nov. 10, 2020. https://cybersecurityventures.com/cybercrime-damage-costs-10-trillion-by-2025/ (accessed Feb. 03, 2022).

[2] I. Abrar, Z. Ayub, F. Masoodi, and A. M. Bamhdi, "A Machine Learning Approach for Intrusion Detection System on NSL-KDD Dataset," in 2020 International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, Sep. 2020, pp. 919–924. doi: 10.1109/ICOSEC49089.2020.9215232.

[3] A. Halimaa A. and K. Sundarakantham, "Machine Learning Based Intrusion Detection System," in 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, Apr. 2019, pp. 916–920. doi: 10.1109/ICOEI.2019.8862784.

[4] Z. K. Ibrahim and M. Y. Thanon, "Performance Comparison of Intrusion Detection System Using Three Different Machine Learning Algorithms," in 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, Jan. 2021, pp. 1116–1124. doi: 10.1109/ICICT50816.2021.9358775.

[5] S. Naseer et al., "Enhanced Network Anomaly Detection Based on Deep Neural Networks," IEEE Access, vol. 6, pp. 48231–48246, 2018, doi: 10.1109/ACCESS.2018.2863036.

[6] J. Ren, J. Guo, W. Qian, H. Yuan, X. Hao, and H. Jingjing, "Building an Effective Intrusion Detection System by Using Hybrid Data Optimization Based on Machine Learning Algorithms," Security and Communication Networks, vol. 2019, pp. 1–11, Jun. 2019, doi: 10.1155/2019/7130868.

[7] A. O. Alzahrani and M. J. F. Alenazi, "Designing a Network Intrusion Detection System Based on Machine Learning for Software Defined Networks," Future Internet, vol. 13, no. 5, p. 111, Apr. 2021, doi: 10.3390/fi13050111.

[8] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," IEEE Access, vol. 5, pp. 21954–21961, 2017, doi: 10.1109/ACCESS.2017.2762418.

[9] M. Monshizadeh, V. Khatri, B. G. Atli, R. Kantola, and Z. Yan, "Performance Evaluation of a Combined Anomaly Detection Platform," IEEE Access, vol. 7, pp. 100964–100978, 2019, doi: 10.1109/ACCESS.2019.2930832.

[10] V. Kanimozhi and T. P. Jacob, "Artificial Intelligence based Network Intrusion Detection with Hyper-Parameter Optimization Tuning on the Realistic Cyber Dataset CSE-CIC-IDS2018 using Cloud Computing," in 2019 International Conference on Communication and Signal Processing (ICCSP), Apr. 2019, pp. 0033–0036. doi: 10.1109/ICCSP.2019.8698029.

[11] A. Lama, "A SURVEY ON NETWORK-BASED INTRUSION DETECTION SYSTEMS USING MACHINE LEARNING ALGORITHMS," vol. 6, no. 9, pp. 225–230, 2022.

[12] "IDS 2018 | Datasets | Research | Canadian Institute for Cybersecurity | UNB." https://www.unb.ca/cic/datasets/ids-2018.html (accessed Feb. 03, 2022).

[13] "A Realistic Cyber Defense Dataset (CSE-CIC-IDS2018) - Registry of Open Data on AWS." https://registry.opendata.aws/cse-cic-ids2018/ (accessed Mar. 16, 2022).

[14] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data," J Big Data, vol. 7, no. 1, p. 104, Dec. 2020, doi: 10.1186/s40537-020-00382-x.

[15] "re — Regular expression operations — Python 3.10.2 documentation." https://docs.python.org/3/library/re.html (accessed Mar. 17, 2022).

[16] A. Stark, "Data Preprocessing & Exploratory Data Analysis (EDA) for Data Science," Medium, Dec. 28, 2019. https://towardsdatascience.com/data-preprocessing-and-eda-for-data-science-50ba6ea65c0a (accessed Mar. 13, 2022).

[17] V. R, "Feature selection — Correlation and P-value," Medium, Feb. 23, 2020. https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf (accessed Mar. 05, 2022).

[18] "scipy.stats.ks_2samp — SciPy v1.8.0 Manual." https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ks_2samp.html (accessed Feb. 11, 2022).

[19] "Spearman's Rank-Order Correlation - A guide to when to use it, what it does and what the assumptions are." https://statistics.laerd.com/statistical-guides/spearmans-rank-order-correlation-statistical-guide.php (accessed Mar. 05, 2022).

[20] "scipy.cluster.hierarchy.dendrogram — SciPy v1.8.0 Manual." https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.dendrogram.html (accessed Feb. 15, 2022).

[21] B. Tezcan, "Why Using a Dummy Classifier is a Smart Move," Medium, Jun. 14, 2021. https://towardsdatascience.com/why-using-a-dummy-classifier-is-a-smart-move-4a55080e3549 (accessed Feb. 15, 2022).

[22] "sklearn.preprocessing.StandardScaler,"scikit-learn.https://scikit-learn/stable/modules/generated/sklearn.preprocessing.StandardScaler.html (accessed Feb. 15, 2022).