SecureDNP3: A MACHINE LEARNING APPROACH TO IDENTIFY DNP3 PROTOCOL ANOMALIES AND THREATS

Shubham Pravin Patil
Master's in Data Analytics
San Jose State University
San Jose, USA
shubhamparvin.patil@sjsu.edu

Shamala Chandrappa
Master's in Data Analytics
San Jose State University
San Jose, USA
shamala.chandrappa@sjsu.edu

Shabari Vignesh
Master's in Data Analytics
San Jose State University
San Jose, USA
shabarivignesh@sjsu.edu

Shreya Chilumukuru
Master's in Data Analytics
San Jose State University
San Jose, USA
nagashreya.chilumukuru@sjsu.edu

Sowmya Neela
Master's in Data Analytics
San Jose State University
San Jose, USA
sowmya.neela@sjsu.edu

***Abstract*—As the digital world evolves, more so with the emergence of Industrial Internet of Things (IoT), network threats have become more complicated and complex necessitating improved and dynamic security measures. Traditional intrusion detection systems often fail to address these advanced and evolving threats. This document introduces SecureDNP3: an innovative intrusion detection system that employs machine learning algorithms for enhanced network security in DNP3 (Distributed Network Protocol 3) environments. SecureDNP3 is designed to detect patterns of traffic on a network and any deviations from normalcy that could imply insecurity. The system is trained using a set of historical traffic in order to identify different types of attacks such as cold_restart, disable_unsolicited, enumerate, info, init_data, mitm_dos, replay, stop_app, warm_restart among others. The current goal is to create an accurate model capable of identifying each type above while achieving high precision and recall rates but at the same time minimizing false positives. In general the aim here is to boost network security resilience through implementation a sophisticated machine-learning-based defense mechanism against novel threats that ensures pro-active dynamic protection. We used Decision Trees, Naïve Bayes, KNN, Random Forest, XGBoost, CATBoost, MLP, LSTM, SGD concluded with Random Forest as best model giving accuracy of 91%. This method seeks to enhance early warning capability in combating sophisticated intrusion into networks hence its relevance.**

I. Introduction

In the ever-changing cyber landscape and Industrial Internet of Things (IOT) that is seen today, it is important to have a strong network security. Traditional methods of intrusion detection often fail in efficient identification and mitigation of complex threats. Thus, we suggest SecureDNP3, a computerized intrusion detection system for networks that uses machinery learning. This system utilizes sophisticated machine learning models to analyze internet traffic patterns and establish any abnormal activities typical of intruders. To make predictions at the present time, Decision Trees, Naïve Bayes, K-Nearest Neighbors, Random Forest, XGBoost, CATBoost , Multilayer Perceptron , Long Short Term Memory and Stochastic Gradient Descent models were trained with historical data as part of this study. A complicated matrix evaluates our model's performance using such major indicators as model accuracy, precision, recall, F1-score, AUC curve/ROC score among others. By mining historical data it provides an advanced approach to network security by enabling proactive identification and response to emerging threats.

A. Problem Statement

Traditional intrusion detection systems are now inadequate in recognizing and ameliorating the complex and developing cyber threats posed to network security; this is happening against a backdrop of a highly sophisticated digital environment with the integration of Industrial Internet of Things (IoT). An advanced solution that can adapt to, and handle the dynamic nature of network threats is needed due to critical vulnerabilities found within the existing security systems especially in DNP3. The problem, therefore, lies in creating a more efficient intrusion detection system that uses state-of-the-art machine learning algorithms so as to detect, classify and respond accurately to a variety of network attacks with high accuracy and low rates for false positives thereby making it possible for networks to be more secure against complex threats.

B. Project Background

The creation of SecureDNP3 was necessitated by an urgent need for enhanced network intrusion detection systems that are applicable in DNP3 settings, which are largely utilized in industrial setups like utility networks. The conventional intrusion detection systems mostly lag behind the current complex and sophisticated cyber threats, thus making the vital infrastructure susceptible to vulnerabilities. To address this problem, the project employs an extensive collection of data from ieee- dataport.org that represents a wide range of network behaviors and specific attack scenarios related to DNP3. This dataset contains logs with detailed information on different types of attacks and network activities including network configuration, traffic data, interactions and metadata. Every log file concerns different attack vectors or some events in the network such as: Disable Unsolicited Messages Attack, Restart Attack, Warm Restart Attack, Enumerate, Info, MITM (Man-In-The-Middle), DoS (Denial of Service), Replay Attack, Stop Application Attack. This forms a basis for training machine learning models that can identify various intrusions thereby facilitating building an IDS system that is not only reactive but also proactive in sensing danger before any significant damage has been done.

II. Proposed Method

The following diagram shown in Figure 1 provides a visual representation of the machine learning model development lifecycle. The workflow commences with Data Collection, the

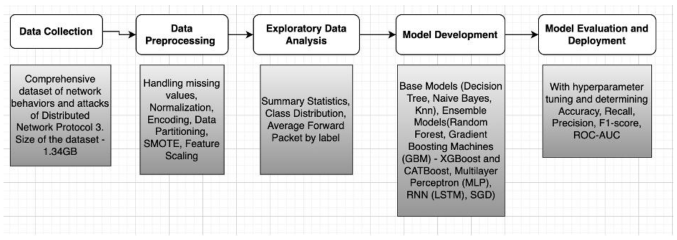

Data Collection
Data Preprocessing
Exploratory Data Analysis
Model Development
Model Evaluation and Deployment
Comprehensive dataset of network behaviors and attacks of Distributed Network Protocol 3. Size of the dataset - 1.34GB
Handling missing values, Normalization, Encoding, Data Partitioning, SMOTE, Feature Scaling
Summary Statistics, Class Distribution, Average Forward Packet by label
Base Models (Decision Tree, Naive Bayes, Knn), Ensemble Models(Random Forest, Gradient Boosting Machines (GBM) - XGBoost and CATBoost, Multilayer Perceptron (MLP), RNN (LSTM), SGD)
With hyperparameter tuning and determining Accuracy, Recall, Precision, F1-score, ROC-AUC

Fig. 1. Proposed Methology

first phase of Business Understanding, where the objectives are defined, and data needs are identified. This phase emphasizes understanding the project objectives and requirements from a business perspective, then converting this knowledge into a machine learning problem definition and a preliminary plan designed to achieve the objectives. Following collection, Data Preprocessing is the next phase, data is explored to show insights and subsequently cleansed and formatted, addressing issues such as missing values, normalizing the data, data partitioning, SMOTE to balance the data points and feature scaling is done. Exploratory Data Analysis is the next step to understand more about the data, the summary statistics, the various class distribution and various other visualizations. Model Selection and training then take place, where various algorithms are evaluated, and the most suitable model is trained using the dataset. This corresponds to the Model Development phase, where various modeling techniques are selected and applied, and their parameters are calibrated to optimal values. The subsequent step, Model Evaluation and Deployment, where hyperparameter tuning ensure that the model generalizes well to new, unseen data and performs accurately. The Model Evaluation phase assesses the model against specific success criteria before proceeding to deployment. If the model's performance is unsatisfactory, the workflow loops back for further refinement. The model performance is calculated using various evaluation metrics such as accuracy, recall, precision, F1-score, ROC-AUC curve. This methodology performs as CRISP-DM principles which provides a structured approach to developing a machine learning model, ensuring that it not only meets technical standards of accuracy and reliability but also aligns with and fulfills business objectives.

We will collect comprehensive historical datasets that includes network behaviors and attacks of Distributed Network Protocol 3. Our process will include handling missing values, and outliers, standardizing format, and encoding categorical variables (such as class names) into numerical representations for machine learning models. Our goal would be to create a well-structured and reliable ABT that forms the foundation for training and testing our intrusion detection system.

III. Data Preparation

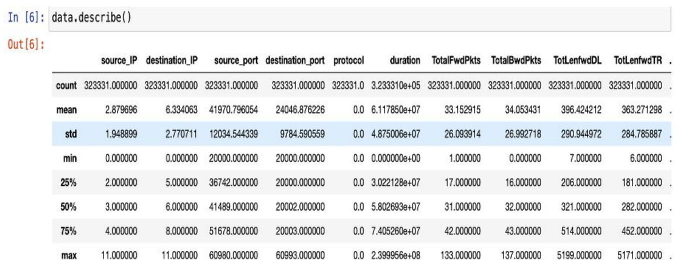
In this data preparation process, we had to get the data ourselves from available in public datasets. We got it from ieee-dataport.org which is a comprehensive dataset of all the network behaviors and all the attacks related Distributed Network Protocol 3. The dataset was checked for missing values but there were none. Duplicated instances found within the dataset were purged. In addition, irrelevant columns were dropped off. This way, we cleaned our data making it more manageable and effective for subsequent analysis.

A. Historical Dataset Description

[1] We have considered the publicly available dataset from ieeedataport.org for network behaviors and attacks of Distributed Network Protocol 3. The size of this dataset is 1.34GB with robust set of data points. Figure 2 shows the description of all the features in the dataset.

B. Exploratory Data Analytics

As part of Exploratory Data Analysis, we explored to understand our dataset and various visualizations were plotted to understand the patterns present in this dataset. Generated summary statistics, plotted various Box plots and Histograms to understand data distribution, plotted a heatmap for Correlation matrix to get crucial insights. Figure 2 below displays the summary statistics of the dataset related to DNP3 attacks which was publicly available.



```
In [6]: data.describe()
Out[6]:
```

	source_IP	destination_IP	source_port	destination_port	protocol	duration	TotalFwdPkts	TotalBwdPkts	TotLenfwdDL	TotLenfwdTR	.
count	323331.000000	323331.000000	323331.000000	323331.000000	323331.0	3.233310e+05	323331.000000	323331.000000	323331.000000	323331.000000	.
mean	2.879696	6.334063	41970.796054	24046.876226	0.0	6.117850e+07	33.152915	34.053431	396.424212	363.271298	.
std	1.948899	2.770711	12034.544339	9784.590559	0.0	4.875006e+07	26.093914	26.992718	290.944972	284.785887	.
min	0.000000	0.000000	20000.000000	20000.000000	0.0	0.000000e+00	1.000000	0.000000	7.000000	6.000000	.
25%	2.000000	5.000000	36742.000000	20000.000000	0.0	3.022128e+07	17.000000	16.000000	206.000000	181.000000	.
50%	3.000000	6.000000	41489.000000	20002.000000	0.0	5.802693e+07	31.000000	32.000000	321.000000	282.000000	.
75%	4.000000	8.000000	51678.000000	20003.000000	0.0	7.405260e+07	42.000000	43.000000	514.000000	452.000000	.
max	11.000000	11.000000	60980.000000	60993.000000	0.0	2.399956e+08	133.000000	137.000000	5199.000000	5171.000000	.

Fig. 2. Summary Statistics

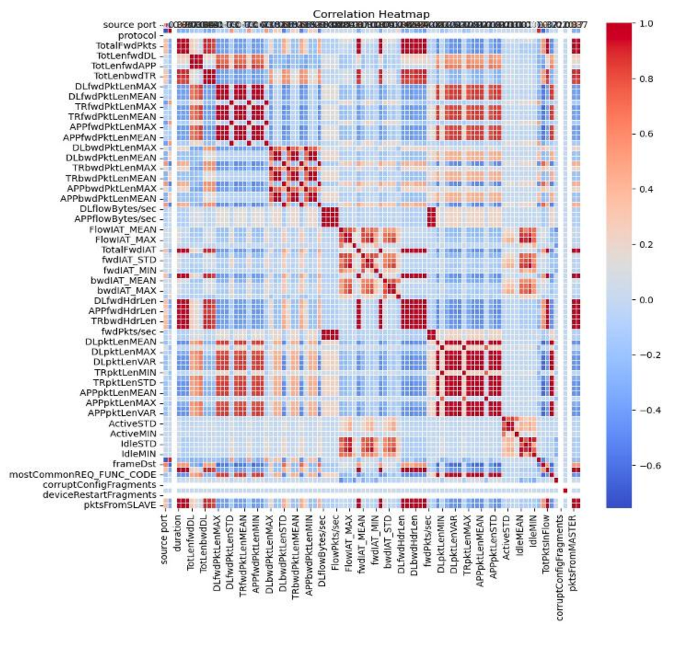


Fig. 3. Heatmap for Correlation Matrix

The figure 3 shows the correlation matrix. The highly correlated features are:
1. APPfwdPktLenMIN 2. TRfwdPktLenMIN 3. fwdPkts/sec 4. FlowPkts/sec 5. pktsFromSLAVE 6. APPfwdHdrLen 7. DLfwdPktLenMIN 8. DLfwdPktLenMAX 9. TRfwdHdrLen 10. TotalBwdIAT 11. APPfwdPktLenMEAN 12. TotLenfwdAPP 13. TRpktLenVAR 14. APPbwdPktLenMIN 15. DLbwdPktLenMEAN
Figure 4 illustrates the representation of a boxplot which can be used to examine the distribution of total forward packets through various attacks. The two events have wider ranges and higher medians compared to other activities such as Stop_App which is having range and median that are significantly lower.

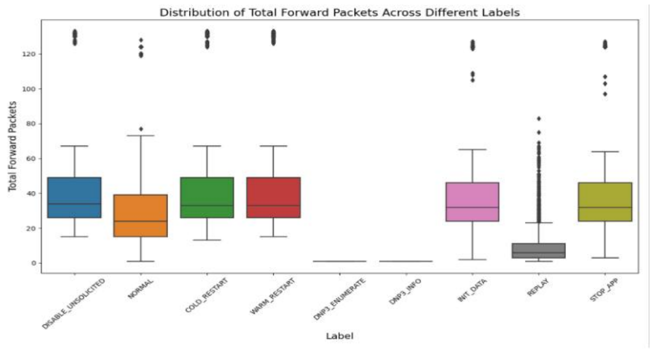


Fig. 4. Box plot for Total Forward Packets

Spotted in Figure 5, a boxplot presents the distribution of total number of packets transmitted backward across different cyber-attacks. However, 'Enumerate' and 'Replay' events show a greater variation or inconsistency in data spread as far as back ward packet flow is concerned. For instance, "Cold_Restart" and "Warm_Restart" both have median that is similar to one another implying consistent back ward packet behavior among these two attacks. It is also notable that there were reduced numbers of backward packets received during this event compared to others with lower median values which implies lesser network traffic for this event. Finally, some anomalies are seen from the plot; for example, "Replay" and "Enumerate" events contain outliers which could imply unusual or anomalous network behavior during these periods of time. Figure 6 below shows the scatterplot of total forward

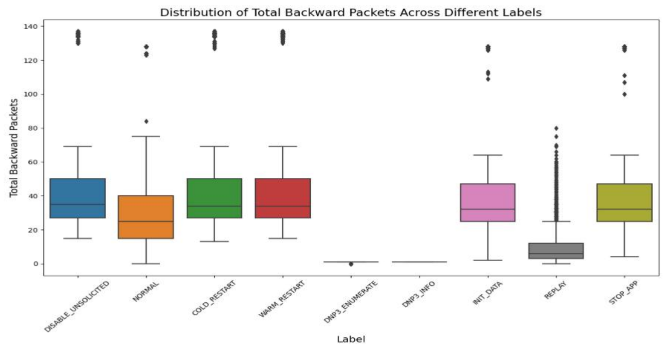


Fig. 5. Box plot for Total Forward Packets

packets against total backward packets. We can observe from the scatterplot that the majority of the data points lie close to the line, implying a very strong linear relationship. This also hints at what we know already: when there is high traffic in one direction, then there will be high traffic in reverse direction as well.There are also some other few points which are diverging from general trend mostly at lower packet counts.

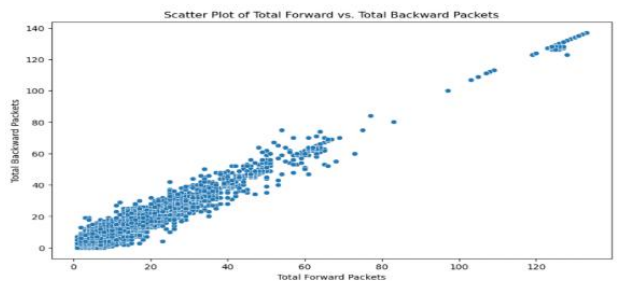


Fig. 6. Scatterplot of Total Forward Packets vs Total Backward Packets

The following bar plot in the figure 7 illustrates the frequency of each label. It means that it depicts various kinds of attacks, and how many times they occur in our data. In addition, the largest number of instances belongs to 'NORMAL' group which suggests that this dataset is predominantly made up of benign network activity records. Among other things, the reason for high occurrences of 'DISABLE_UNSOLICITED' and 'COLD_RESTART' groups could be due to their commonness in such kind of attacks. However, on the contrary, the incidences of 'REPLAY' and 'STOP_APP' are less significant compared to other classes. The average number of forward

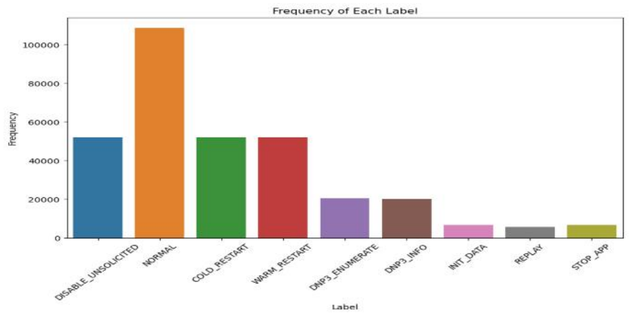


Fig. 7. Frequency of each label

packets for each of these attacks is shown in the bar chart in the figure 8. 'COLD_RESTART', 'WARM_RESTART' and 'DISABLE_UNSOLICITED" have higher distribution of averages indicating that messages are sent in forward direction. Similar observations were made for 'INIT_DATA' and 'REPLAY'. On the other hand, numbers are less for below normal, enumerate and DNP3_INFO which showed lower averages than those above thus suggesting that they have low quantity of communication moving towards a given direction. Lastly, STOP_APP has the smallest average since there is little communication during an application shutdown.

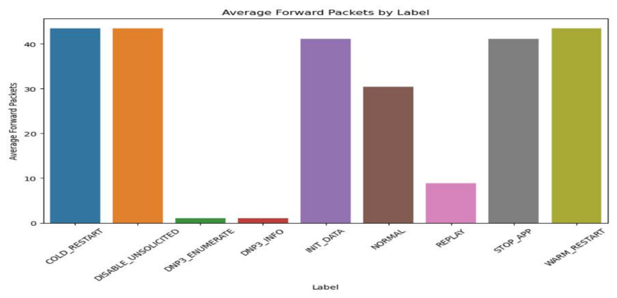
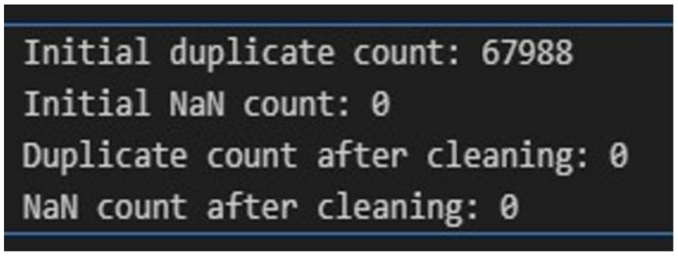


Fig. 8. Average number of Forward packets for each label

C. Data Cleaning

We found duplicates at the time of Data Cleaning. 67, 9788 duplications were observed. We removed them. After scanning our data set for null values we didn't find any of them. The dataset after cleaning is shown in the following figure 9:



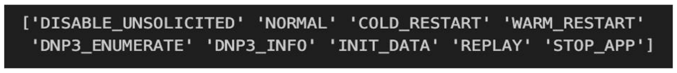
```
Initial duplicate count: 67988
Initial NaN count: 0
Duplicate count after cleaning: 0
NaN count after cleaning: 0
```

Fig. 9. Duplicates and Null value counts

Every network flow is characterized by Flow ID, Source IP, Destination IP, Source Port, Destination Port as unique features. Our analysis revealed that these features have no predictive information or attributes of network traffic. Therefore, we should not include these kind of features in our model training because there would be a problem of overfitting. So they were removed from the columns.

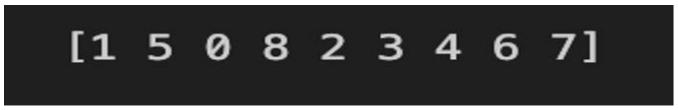
D. Data Transformation

As part of Data Transformation, Feature Scaling has been done by use of Standard Scalar. This method standardizes the features by deleting the mean and scaling to unit variance. Afterwards, Standardized classes were mapped into integer using Label Encoder as shown in Figure 10 and 11.



```
['DISABLE_UNSOLICITED' 'NORMAL' 'COLD_RESTART' 'WARM_RESTART'
 'DNP3_ENUMERATE' 'DNP3_INFO' 'INIT_DATA' 'REPLAY' 'STOP_APP']
```

Fig. 10. Label Encoding



```
[1 5 0 8 2 3 4 6 7]
```

Fig. 11. Label Encoding

SMOTE was performed for balancing our dataset. It generates synthetic samples by interpolating between the instances of the minority class and their neighbors.

IV. MODEL DEVELOPMENT

A. Model Proposals

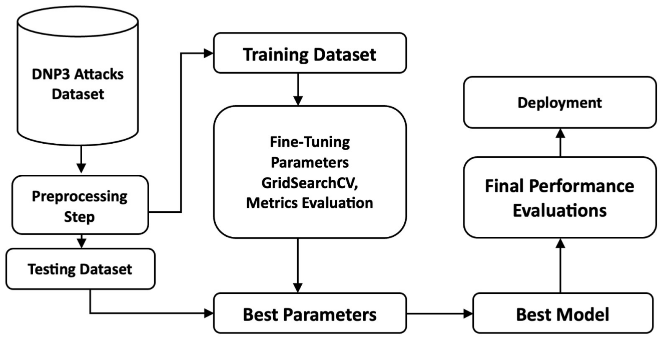


Fig. 12. An Overview of proposed framework

To achieve the final goal of SecureSNP3 we have employed various Machine Learning techniques like Decision Trees, Naïve Bayes, K-Nearest Neighbors, Random Forest, Gradient Boosting techniques like XGBoost, CATBoost and also used Deep Learning techniques such as Multilayer Perceptron, Long Short Term Memory and Stochastic Gradient Descent to come up with a new way and attain high accuracy with less false positives in detecting the DNP3 network attacks.
In the context of our project which is SecureDNP3, a machine learning based network intrusion detection system is developed to improve the security in DNP3 environments and these

models are also used to detect and correctly classify various types of network attacks.
Following are the machine learning and deep learning models used for this purpose to detect and classify the network intrusion attacks that are present in the dataset:
1) Decision Trees: These are models that use decision trees for classification which are based on non-linear splits that create tree-branched structures out of input datasets making choices upon feature values.
Breaking down complex datasets into smaller subsets while developing an associated decision tree renders them useful too. The decisions they make are clear and easy to understand making them reliable when interpretability matters a lot.
The information gain is calculated for each feature to determine how good it is at splitting the data. This helped us to prioritize features important for model development.
Below figure 12 shows top 20 features present in our dataset based on information gain.

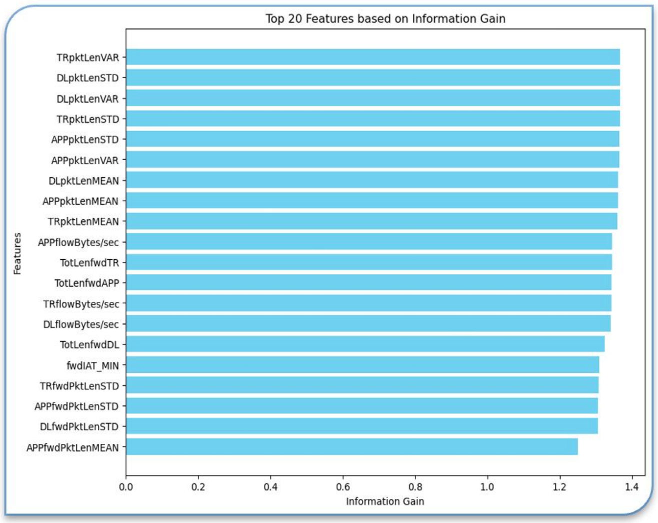


Fig. 13. Top 20 features based on Information Gain

2) Naïve Bayes: Naïve Bayes Classifier, chosen as the second algorithm for this purpose, is based on bayes theorem and assumes that data points are independent of one another. It works better in high dimensional datasets.
3) K-Nearest Neighbors (KNN): Both classifying or regressing, KNN is nonparametric. Classifications are made by taking into account the nearest neighbors through majority vote to determine the membership of a class. Since it has a simple nature, and can classify unobserved data points effectively KNN is useful in Intrusion Detection System.
4) Random Forest: In classification problems, Random Forest is an example of ensemble techniques which is built on multiple decision trees at training time whereby it outputs the class that is mode – most frequent class among individual trees. Also Random Forests work well with large datasets with many dimensions and they can evaluate feature importance for classification as well as being effective against overfitting.
5) XGBoost: Extreme Gradient Boosting or XGBoost model is designed for speed and performance. Structured data of any kind can be processed by XGBoost; moreover, it scales well. Additionally, L1 (Lasso) and L2 (Ridge) regularization techniques provided by XGBoost assist in combating overfitting problem. This suits our case since we have complex imbalanced classes dataset with high dimensions.
6) CATBoost: Another model known as CATBoost has to be mentioned besides XGBoost which handles categorical values and thus referred to as CATBoost for "Categorical Boosting". This model does not suffer from over fitting when there are numerous categorical features. Also when working with large datasets the tool demonstrates its superiority in terms of speed and efficiency than most other algorithms used today.
7) Multilayer Perceptron (MLP): The Multilayer Perceptron model is composed of three layers; an input layer, a hidden layer and an output layer. This method is effective at discovering intricate designs as well as nonlinear patterns in information which are primarily utilized for classification problems involving high- dimensional datasets. Such a model can be employed to detect network intrusions, differentiating the normal from anomalies. If MLP is made deeper by increasing the number of layers will result into better accuracy.
8) Long Short-Term Memory (LSTM): The subsequent type of model is that known as Long Short-Term Memory (LSTM), which comes under Recurrent Neural Network for classifying based on sequences whose order may matter. Similarly, it can also be applied in cases like Network Intrusion Detection systems in environments such as DNP3 where network data is sequential.
9) Stochastic Gradient Descent (SGD): Stochastic Gradient Descent is a form of simple approach implemented when fitting linear classifiers against convex loss functions such as logistic regression and support vector machines . This mainly used when dataset size exceeds the memory capacity . Once again, this technique is very quick way of achieving optimality. Furthermore, it suits large datasets since each iteration uses only part of the data for training the model update step by step so it works well on our project because we have large amount of volume of data set involved on our sample selected datasets. Additionally , this kind of model also has capability to avoid overfitting training dataset by utilizing L1 and L2 regularization options present in this case.

B. *Model Support*

The models for detecting and classifying the network event types were developed on our Jupyter Notebook and Visual Studio Code on Windows System. The environments where we developed our models also include Google Colab running on Python 3.9. For Data Cleaning and manipulation we made use of Python Pandas library, data visualizations like bar charts, box plots, histograms and heatmaps are done using Matplotlib and Seaborn libraries. Then dataset was standardized using "StandardScaler". The comprehensive model evaluation in scikit-learn was used. This library helped us in evaluating our classifiers including precision, recall and F-1 score.

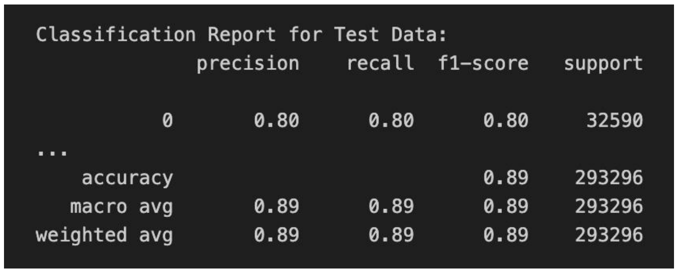
C. Model Evaluation Methods

Detection and classification of different network related attacks in DNP3 understanding, using machine learning models among the major aims of our SecureDNP3 project makes model performance evaluation a necessity.
There are several metrics we used to evaluate our model performances as follows:
a. Confusion Matrix: The confusion matrix is a table describing how well the classifier performs on our test dataset where true values are known. It summarizes correct classifications and any misclassifications made by the models. Hence for each kind of attack present in our dataset, the number of true positives (attacks correctly predicted), false negatives (undetected attacks), true negatives (predicted as non-attack) and false positives (non-attacks misclassified as attacks) have been shown in the confusion matrices.
b. Accuracy, Precision, F1-Score, Recall Accuracy: This helps us to get an idea about how accurately our models have predicted with respect to overall predictions because it is calculated on the basis of ratio between total number of predictions that were correct as against all other predictions that were made by them. Precision: It indicates how accurate or correct positive predictions were i.e., if they indicated right thing or not. As such it is defined as ratio between True Positives and Predicted Positives by our models.
Recall: It shows how good model was at finding positive instances out of entire possible instances present in data. In simple terms it can be stated as ratio between True positive ones divided by Actual Positive instances existing within dataset.
F1-Score: Harmonic mean between precision achieved and recall obtained is known as F1-Score which perfectly balances both these parameters. These also help balancing trade-off done while achieving recall and precision.
c. ROC Curve and AUC:
ROC Curve: Receiver Operating Characteristic curve plots binary classifier's ability by varying discrimination threshold.
AUC (Area Under the Curve): this measures area under whole ROC curve from (0,0) to (1,1). It provided cumulative measure of performance for all possible classification thresholds too.
Hence plotting ROC curve and calculating AUC for each of our attacks and mean ROC curve across all classes has given us a clear visual and quantitative assessment on how well our models have distinguished between classes. These metrics are important in providing deeper insights into the performances of our models in our SecureDNP3 project.

V. EVALUATION RESULTS

A. Decision Tree

1. Classification Report:
The precision, recall and F1-score show almost perfect balance across all classes as shown in figure 13, averaged at about 0.89 confirming that this model is effective at predicting and distinguishing between various types of network attacks throughout this dataset.



```
Classification Report for Test Data:
              precision    recall  f1-score   support

           0       0.80      0.80      0.80     32590
...
    accuracy                           0.89    293296
   macro avg       0.89      0.89      0.89    293296
weighted avg       0.89      0.89      0.89    293296
```

Fig. 14. Classification report for Decision Trees

2. Confusion Matrix:

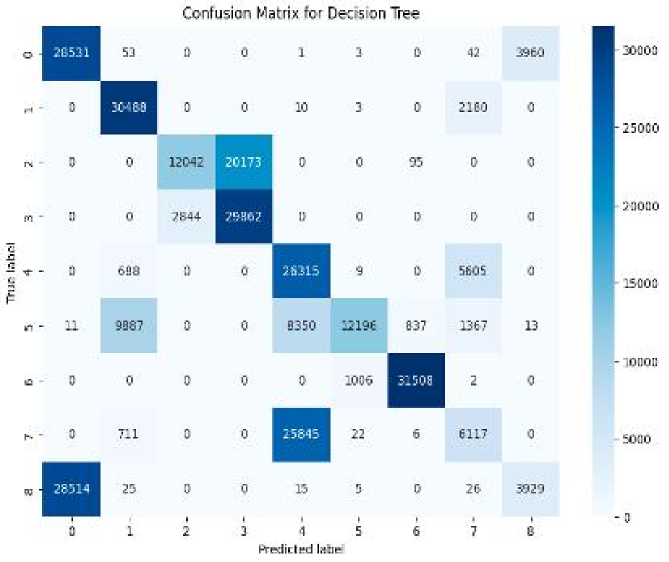


Fig. 15. Confusion Matrix for Decision Trees

The confusion matrix represented in the figure 14 indicates that 0 and 2 have high true positives hence indicating a good detection rate by our model for these attacks. It also shows misclassifications to other class affecting overall accuracy.
3. ROC Curve and AUC: Plotting ROC curve and calculating AUC for each of our attacks and mean ROC curve across all classes has given us a clear visual and quantitative assessment on how well our models have distinguished between classes.These metrics are important in providing deeper insights into the performances of our models in our SecureDNP3 project. The ROC curve represented the figure 15 shows how well the model separates positive from negative instances with an average area under curve (AUC) of 0.94 which indicates a very good performance of the classifier for differentiating between classes. There are variations in individual class AUCs because they range from perfect discrimination in class 6 (AUC = 1.00) to less discrimination in others like class 2 (AUC = 0.85).

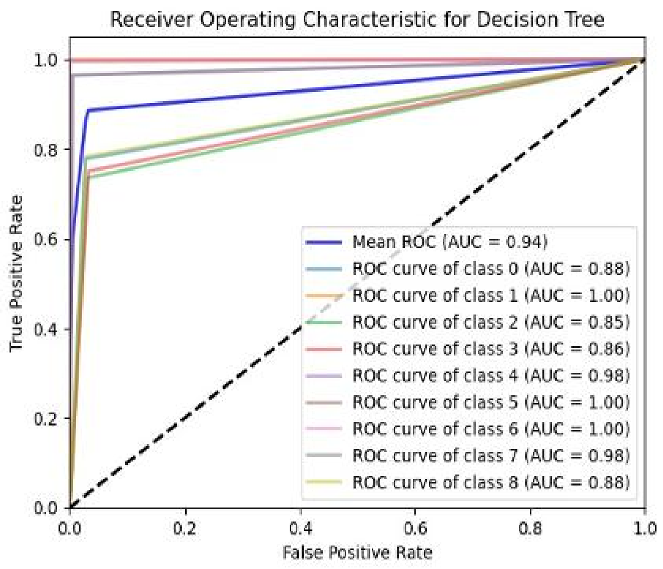


Fig. 16. ROC Curve and AUC for Decision Trees

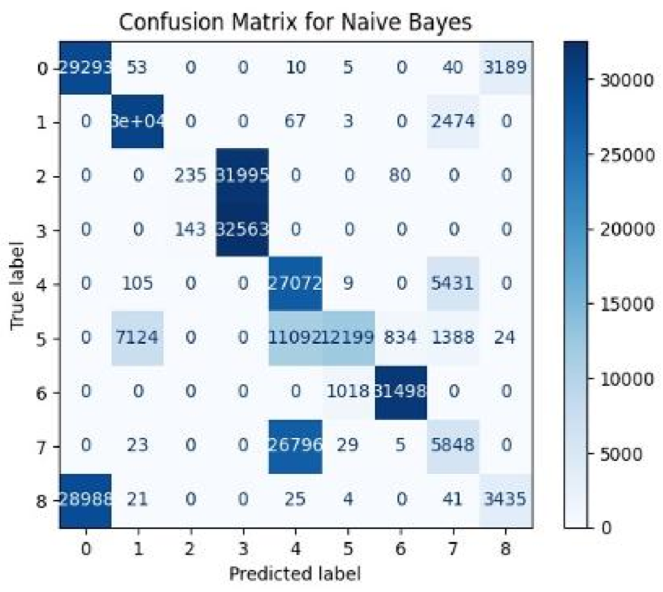
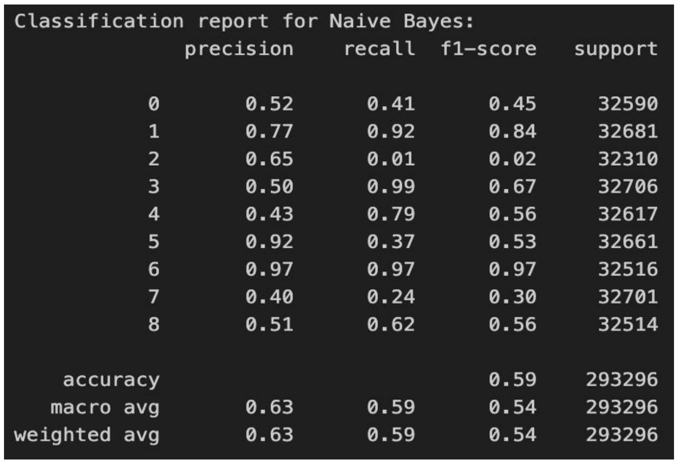


Fig. 18. Confusion Matrix for Naive Bayes

B. Naïve Bayes

The classification report, confusion matrix and ROC curves illustrate different performance levels of Naïve Bayes classifier for different classes:
1. Classification Report:



```
Classification report for Naive Bayes:
              precision    recall  f1-score   support

           0       0.52      0.41      0.45     32590
           1       0.77      0.92      0.84     32681
           2       0.65      0.01      0.02     32310
           3       0.50      0.99      0.67     32706
           4       0.43      0.79      0.56     32617
           5       0.92      0.37      0.53     32661
           6       0.97      0.97      0.97     32516
           7       0.40      0.24      0.30     32701
           8       0.51      0.62      0.56     32514

    accuracy                           0.59    293296
   macro avg       0.63      0.59      0.54    293296
weighted avg       0.63      0.59      0.54    293296
```

Fig. 17. Classification report for Naive Bayes

From figure 16, despite some imbalances such as class 6 having high precision and recall but others like class 2 exhibits very low ones resulting in average precision, recall and F1-score being approximately 0.63, 0.59 and 0.54 correspondingly. This indicates challenges of making predictions on all classes equally.
2. Confusion Matrix:

It is evident in the figure 17 that a number of classes were misclassified more compared to others such as the case with class 2 and class 3 showing inability to distinguish between certain types of network behavior or attacks.
3. ROC Curve:

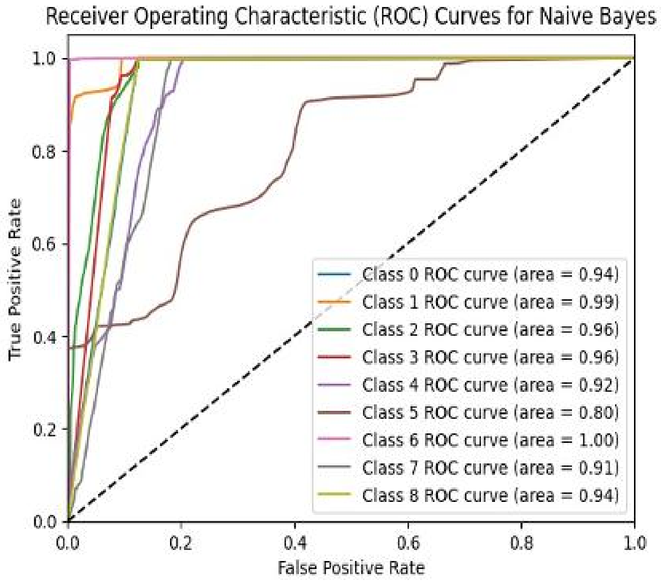
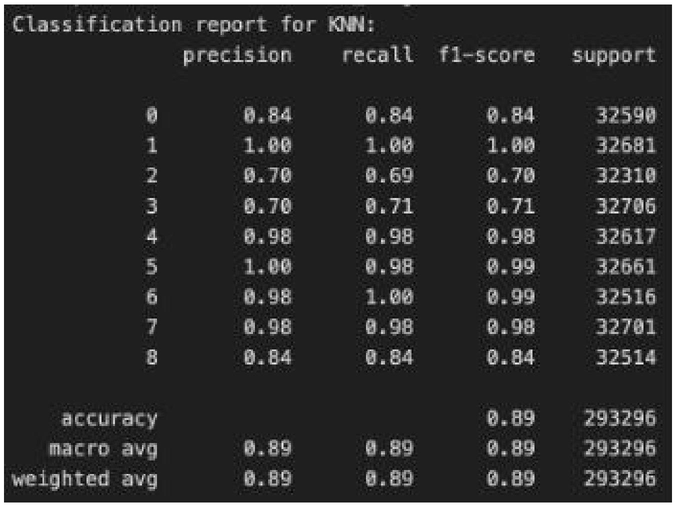


Fig. 19. ROC Curve and AUC for Naive Bayes

From figure 18, they have good discriminative ability for individual classes notwithstanding their overall low accuracy or precision in some cases due to the fact that most of them recorded high AUC values excepting class six which has AUC of one while class five has an AUC value of about 0.80 only. These visuals demonstrate how well the Naive Bayes model deals with diversity in the dataset by identifying some types of attacks effectively while struggling with others probably because the model assumes independence among features.

C. K-Nearest Neighbors

The K-Nearest Neighbors (KNN) model is well-performed in classifying different types of network attacks according to classification report, confusion matrix and ROC curves.
1. Classification Report: From the figure 19, we can say it



```
Classification report for KNN:
              precision    recall  f1-score   support

           0       0.84      0.84      0.84     32590
           1       1.00      1.00      1.00     32681
           2       0.70      0.69      0.70     32310
           3       0.70      0.71      0.71     32706
           4       0.98      0.98      0.98     32617
           5       1.00      0.98      0.99     32661
           6       0.98      1.00      0.99     32516
           7       0.98      0.98      0.98     32701
           8       0.84      0.84      0.84     32514

    accuracy                           0.89    293296
   macro avg       0.89      0.89      0.89    293296
weighted avg       0.89      0.89      0.89    293296
```

Fig. 20. Classification report for K-Nearest Neighbors

has high precision, recall and F1-scores for most classes with overall accuracy and average scores 0.89; thus, it can be concluded that the KNN effectively discerns the dissimilarities among diverse network behaviors.
2. Confusion Matrix: Figure 20 shows that the model displays

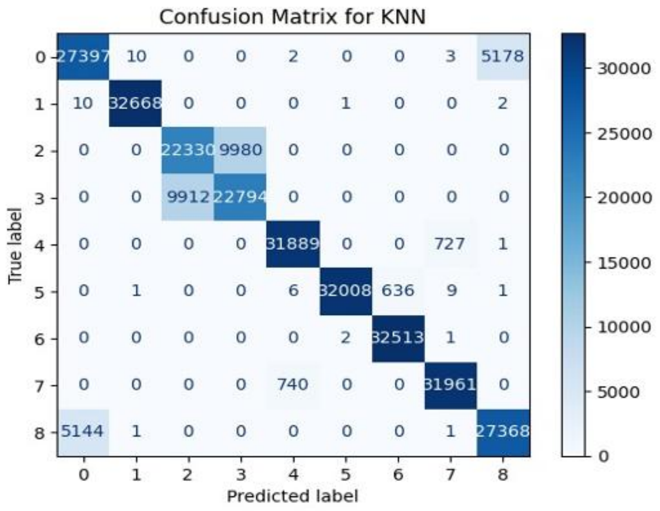


Fig. 21. Confusion Matrix for K-Nearest Neighbors

varied true positive rates for most classes including class 1 and class 4 having high rates which implies a strong classifier with minimal confusion between classes even if there are some misclassifications in this classification matrix.
3. ROC Curve:

The ROC curves in figure 21 have perfect or near perfect AUC scores of 1.00 or close to it for majority of the classes,

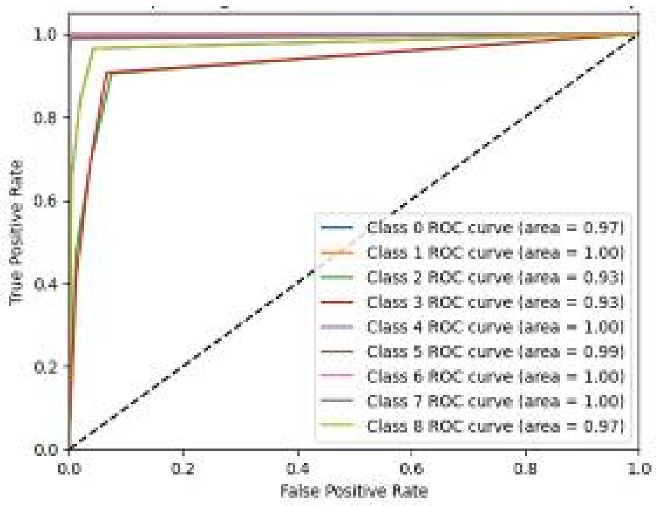
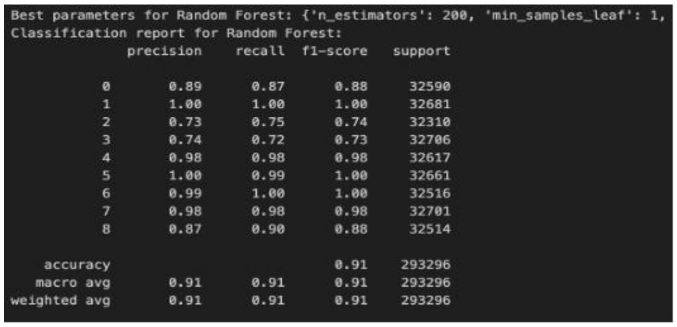


Fig. 22. ROC Curve and AUC for K-Nearest Neighbors

indicating excellent discriminative power of KNN towards normal operations relative to various types of attacks which is depicted by all AUC values being above 0.90.
In general, KNN model has established itself as an effective one within SecureDNP3 system while providing an extensive range of network events and intrusions detection mechanism that can be used.

D. Random Forest

The Random Forest model exemplifies robustness in classification of different network events and attacks:
1. Classification Report:



```
Best parameters for Random Forest: {'n_estimators': 200, 'min_samples_leaf': 1,
Classification report for Random Forest:
              precision    recall  f1-score   support

           0       0.89      0.87      0.88     32590
           1       1.00      1.00      1.00     32681
           2       0.73      0.75      0.74     32310
           3       0.74      0.72      0.73     32706
           4       0.98      0.98      0.98     32617
           5       1.00      0.99      1.00     32661
           6       0.99      1.00      1.00     32516
           7       0.98      0.98      0.98     32701
           8       0.87      0.90      0.88     32514

    accuracy                           0.91    293296
   macro avg       0.91      0.91      0.91    293296
weighted avg       0.91      0.91      0.91    293296
```

Fig. 23. Classification report for Random Forest

In figure 22, with a total accuracy of 0.91, Random Forest has performed very well on this data set with high precision, recall, and F1-scores for most classes, implying that it can efficiently distinguish between different types of network behaviors.
2. Confusion Matrix:

The Confusion Matrix in figure 23 clearly portrays the ability of the model to correctly classify a large number of instances from all classes with few misclassifications particularly important in class such as 5 and 6 where there is high

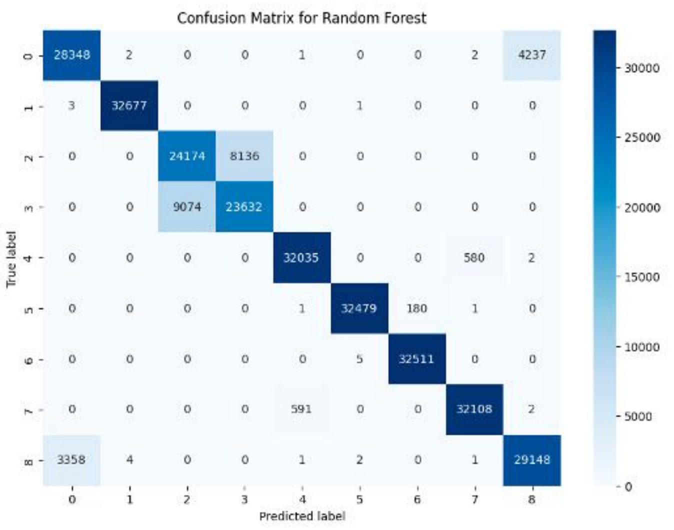


Fig. 24. Confusion Matrix for Random Forest

support.
3. ROC Curves:

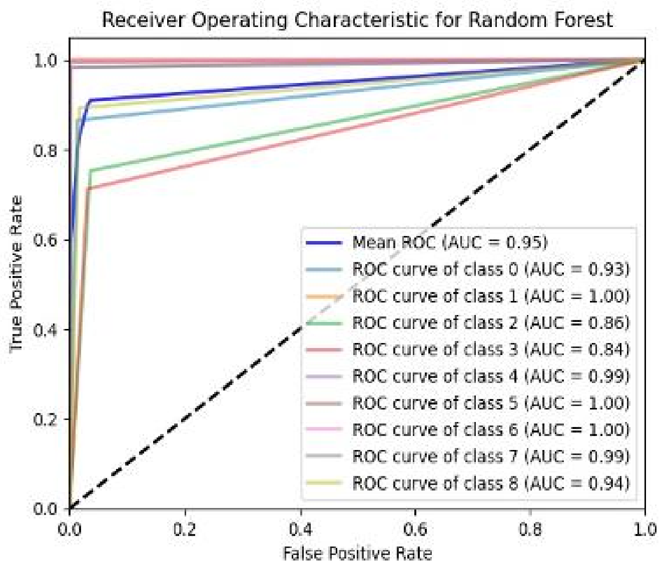


Fig. 25. ROC Curve and AUC for Random Forest

The ROC curves in figure 24 have remarkable AUC scores for each class averaging 0.95, indicating how well the model discriminates between classes; some reach perfect or near perfect AUC values.
Overall, the Random Forest model with optimal parameters (‘n_estimators‘: 200, ‘min_samples_leaf‘: 1) becomes highly efficient and reliable in its performance making it ideal for detecting network intrusions in the SecureDNP3 system as this is always expected by general publics.

E. Multilayer Perceptron

The performance of the classifier is not consistent across the classes:

1. Classification Report:

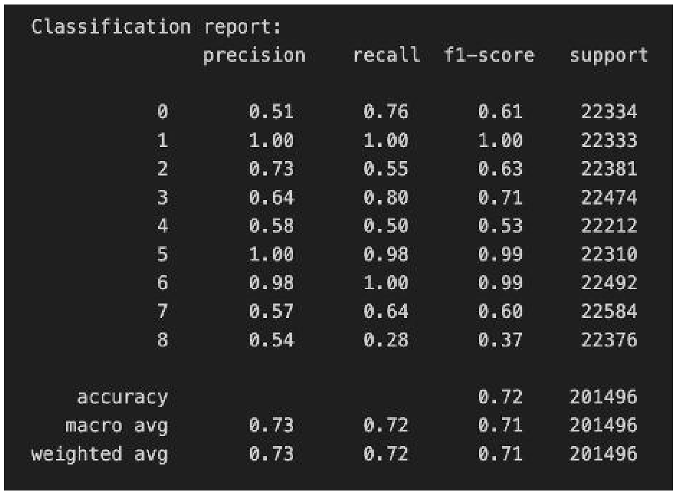

```
Classification report:
              precision    recall  f1-score   support

           0       0.51      0.76      0.61     22334
           1       1.00      1.00      1.00     22333
           2       0.73      0.55      0.63     22381
           3       0.64      0.80      0.71     22474
           4       0.58      0.50      0.53     22212
           5       1.00      0.98      0.99     22310
           6       0.98      1.00      0.99     22492
           7       0.57      0.64      0.60     22584
           8       0.54      0.28      0.37     22376

    accuracy                           0.72    201496
   macro avg       0.73      0.72      0.71    201496
weighted avg       0.73      0.72      0.71    201496
```

Fig. 26. Classification report for Multilayer Perceptron

The figure 25 presents different precision, recall and F1-score values by class indicating that it performs well in identifying some events (e.g. high scores for class 1 and class 6) while others are difficult to identify (for instance, low scores for class 8).
2. Confusion Matrix:

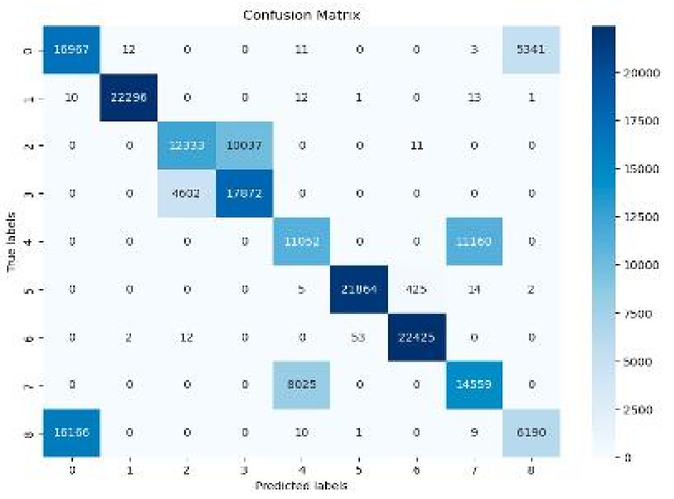


Fig. 27. Confusion Matrix for Multilayer Perceptron

The confusion matrix from figure 26 shows big errors in classification for a number of classes, especially for class 0 and class 8 which have many instances being misclassified in them thus affecting overall accuracy of this model.
3. ROC Curves:
The ROC curves in figure 27 exhibit good to excellent ability of between-class discrimination at most times though there may be some classes whose performance on precision and recall is lower; most classes have AUC above 0.90.
From this performance data, it can be seen that the model is capable of distinguishing most classes with further tuning

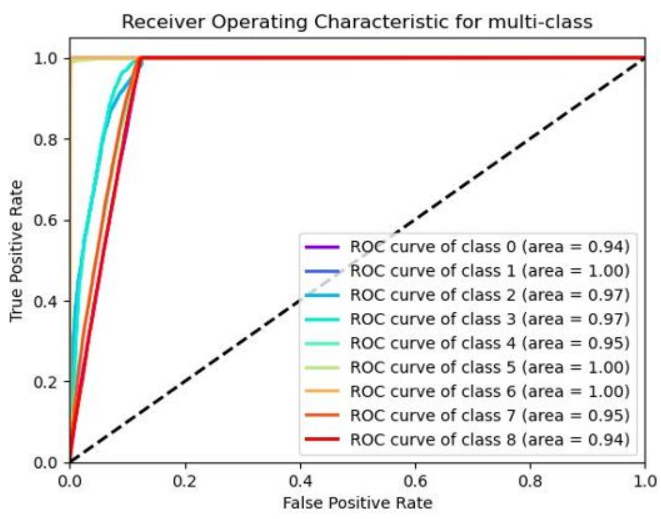
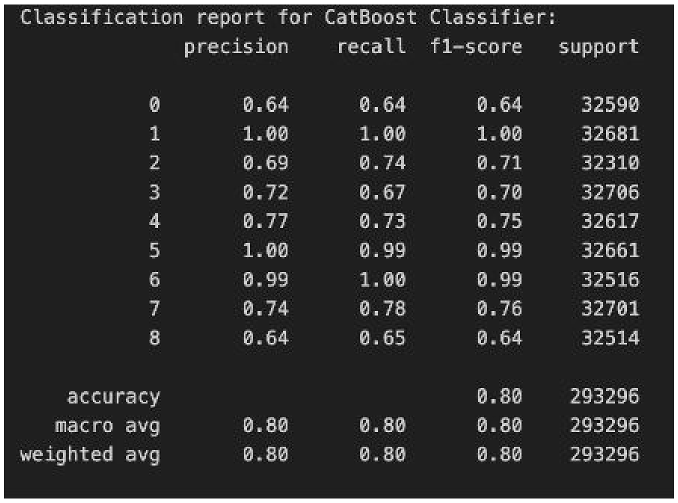


Fig. 28. ROC Curve and AUC for Multilayer Perceptron

needed or more training data on weak performing ones to improve its overall predictive accuracy and dependability.

F. CATBoost

The CatBoost classifier is very good for network event classification overall:
1. Classification Report:

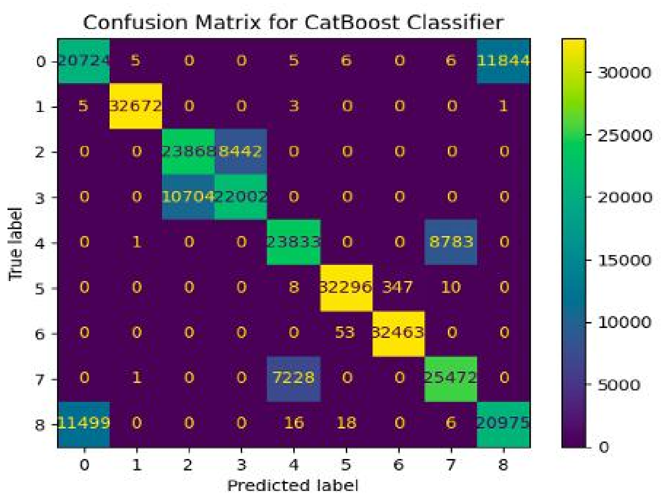


```
Classification report for CatBoost Classifier:
                      precision    recall  f1-score   support

                 0       0.64      0.64      0.64     32590
                 1       1.00      1.00      1.00     32681
                 2       0.69      0.74      0.71     32310
                 3       0.72      0.67      0.70     32706
                 4       0.77      0.73      0.75     32617
                 5       1.00      0.99      0.99     32661
                 6       0.99      1.00      0.99     32516
                 7       0.74      0.78      0.76     32701
                 8       0.64      0.65      0.64     32514

          accuracy                           0.80    293296
         macro avg       0.80      0.80      0.80    293296
      weighted avg       0.80      0.80      0.80    293296
```

Fig. 29. Classification report for CatBoost

From figure 28, we can say it has high accuracy and average scores of 0.80, as well as a great level of precision, recall and F1-scores for many classes, especially class 1, 5 and 6, that can successfully identify particular network behaviors accurately.
2. Confusion Matrix:

Figure 29 illustrates most of the correct classifications for several classes though there are some significant misclassifications especially in classes 0 and 8 which could be improved by model tuning.



	0	1	2	3	4	5	6	7	8
0	20724	5	0	0	5	6	0	6	11844
1	5	32672	0	0	3	0	0	0	1
2	0	0	23868	8442	0	0	0	0	0
3	0	0	10704	22002	0	0	0	0	0
4	0	1	0	0	23833	0	0	8783	0
5	0	0	0	0	8	32296	347	10	0
6	0	0	0	0	0	53	32463	0	0
7	0	1	0	0	7228	0	0	25472	0
8	11499	0	0	0	16	18	0	6	20975

Fig. 30. Confusion Matrix for CatBoost

3. ROC Curves:

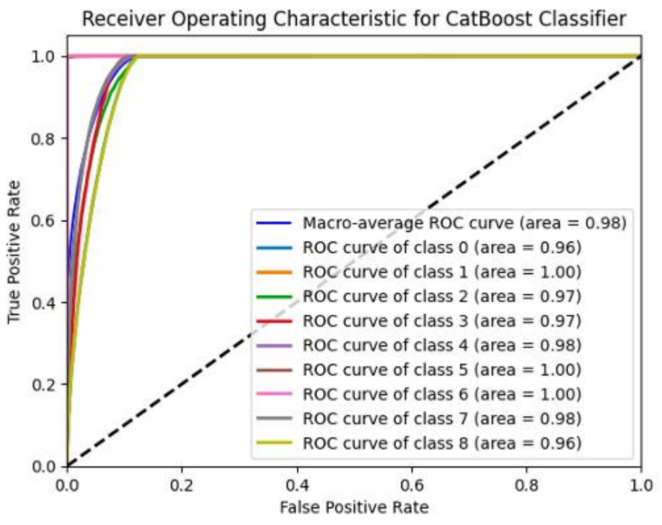


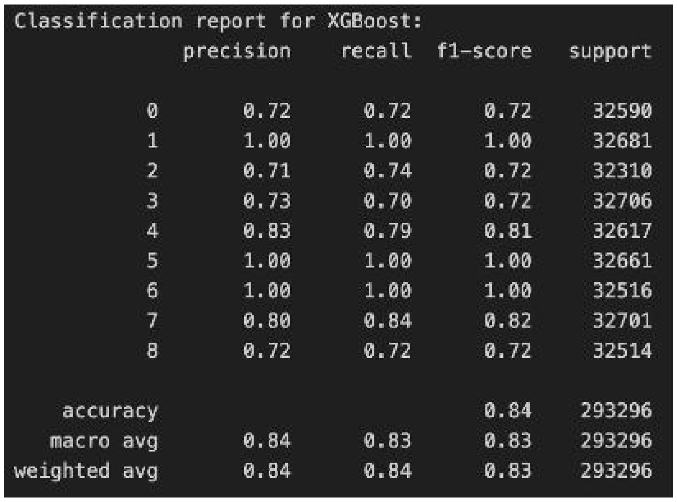
Fig. 31. ROC Curve and AUC for CatBoost

The macro-average ROC curve area from figure 30 shows 0.98, means that the CatBoost classifier is excellent at differentiating between various types of network events within all class labels according to its ROC curves.
According to these metrics, it can be stated that the CatBoost classifier is a solid tool inside SecureDNP3 system which handles a variety of difficult classification tasks with high levels of accuracy and consistency.

G. XGBoost

The model XGBoost demonstrates strong performance in classifying network events:
1. Classification Report:

Figure 31 shows that the exhibits a good precision, recall and F1-scores especially on classes 1, 5 and 6 where it has

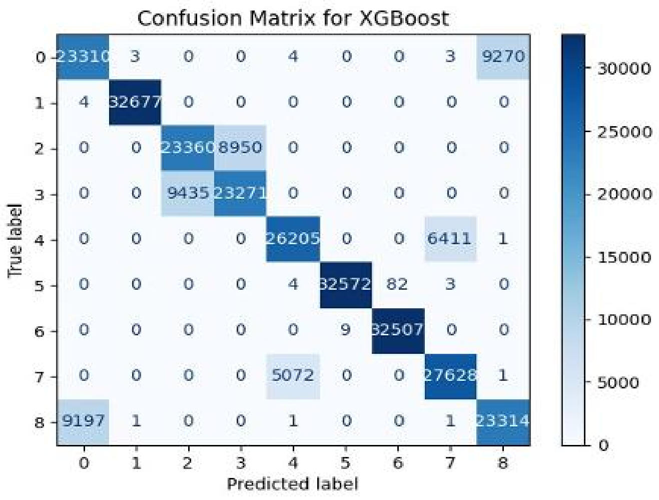


```
Classification report for XGBoost:
              precision    recall  f1-score   support

           0       0.72      0.72      0.72     32590
           1       1.00      1.00      1.00     32681
           2       0.71      0.74      0.72     32310
           3       0.73      0.70      0.72     32706
           4       0.83      0.79      0.81     32617
           5       1.00      1.00      1.00     32661
           6       1.00      1.00      1.00     32516
           7       0.80      0.84      0.82     32701
           8       0.72      0.72      0.72     32514

    accuracy                           0.84    293296
   macro avg       0.84      0.83      0.83    293296
weighted avg       0.84      0.84      0.83    293296
```

Fig. 32. Classification report for XGBoost

perfect scores showing that the model is able to effectively differentiate these specific events from the other events. It can be seen that its overall accuracy and average scores are approximately 0.84.

2. Confusion Matrix: Figure 32 generally indicates that the



	0	1	2	3	4	5	6	7	8
0	23310	3	0	0	4	0	0	3	9270
1	4	32677	0	0	0	0	0	0	0
2	0	0	23360	8950	0	0	0	0	0
3	0	0	9435	23271	0	0	0	0	0
4	0	0	0	0	26205	0	0	6411	1
5	0	0	0	0	4	32572	82	3	0
6	0	0	0	0	0	9	32507	0	0
7	0	0	0	0	5072	0	0	27628	1
8	9197	1	0	0	1	0	0	1	23314

Fig. 33. Confusion Matrix for XGBoost

model does well across most classes with high true positives mostly for classes number 1, 5 and 6 although there are some misclassifications in classes 0 and 8.

3. ROC Curves: For all classes, this has very good AUC values particularly as high as 1 for class 1, 5 and 6, indicating the model's ability to make distinctions among various kinds of network operations.

In general terms from Figure 33, XGBoost's excellent class-specific accuracy rates coupled with its capability to distinguish between different categories of network events make it an invaluable tool in systems like SecureDNP3 designed to enhance network security through accurate threat identification

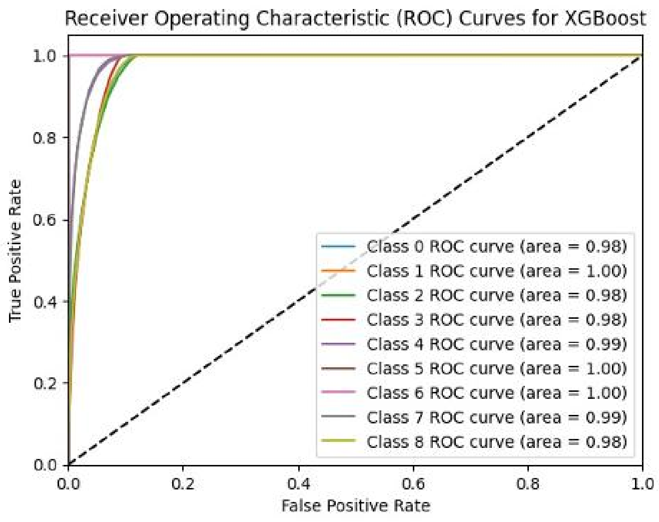


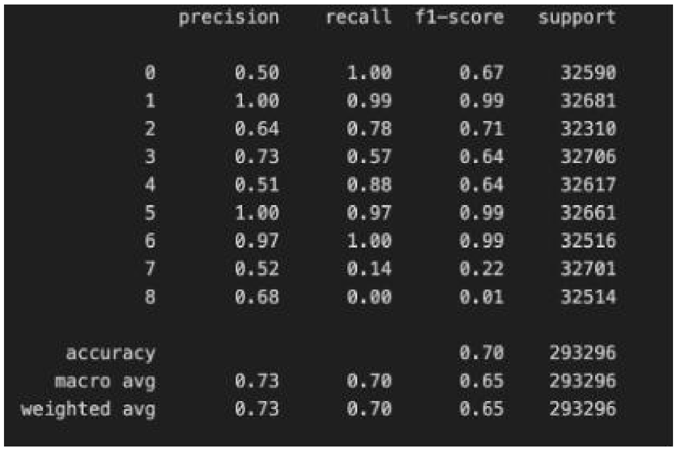
Fig. 34. DROC Curve and AUC for XGBoost

and categorization.

H. Long Short-Term Memory:

The model LSTM demonstrates strong performance in classifying network events:

1. Classification Report and Confusion Matrix:



```
              precision    recall  f1-score   support

           0       0.50      1.00      0.67     32590
           1       1.00      0.99      0.99     32681
           2       0.64      0.78      0.71     32310
           3       0.73      0.57      0.64     32706
           4       0.51      0.88      0.64     32617
           5       1.00      0.97      0.99     32661
           6       0.97      1.00      0.99     32516
           7       0.52      0.14      0.22     32701
           8       0.68      0.00      0.01     32514

    accuracy                           0.70    293296
   macro avg       0.73      0.70      0.65    293296
weighted avg       0.73      0.70      0.65    293296
```

Fig. 35. Classification report for LSTM

Figure 34 and 35 indicate the high precision and recall for effective detection of some classes such as 'DISABLE_UNSOLICTED' and 'REPLAY', but indicate very low F1-scores and precision in classifying events like 'Stop App' and 'WARM_RESTART'.

2. ROC Curve: For most classes in figure 36, the ROC curves are characterized by very good AUC values, despite the fact that some categories have lower performance according to classification report; indicating that the overall model has a good ability to differentiate between network behavior types. This means that while there is strong indication of its capability in certain scenarios, other cases clearly show limitations

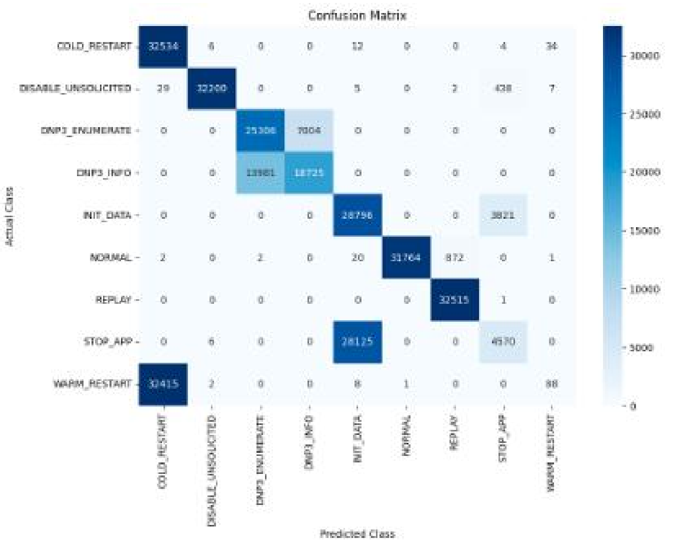


Fig. 36. Confusion matrix for LSTM

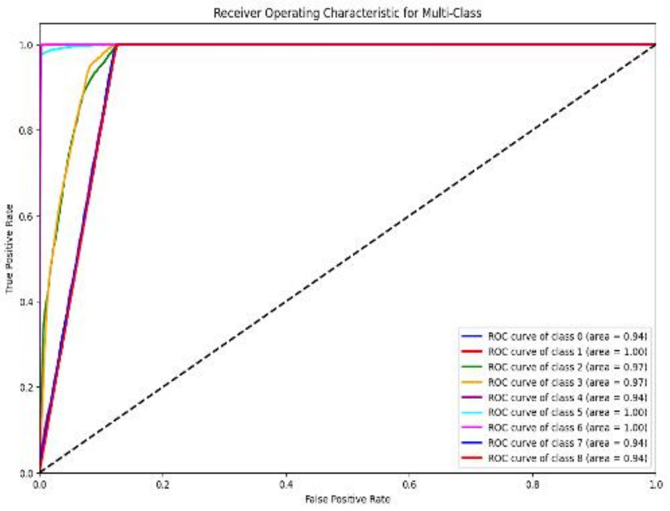


Fig. 37. ROC curve for LSTM

for which it may need further refinement or more training data so that it can improve on its accuracy levels generally across various sorts of network threats.

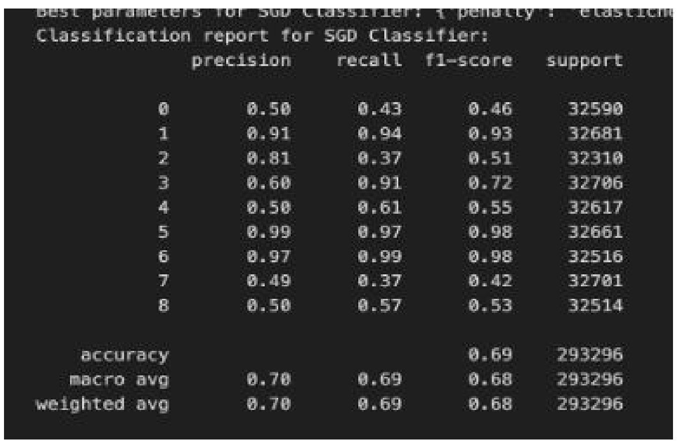
I. Stochastic Gradient Descent

The performance of the SGD Classifier varies across classes:
1. Classification Report and Confusion Matrix:

From figure 37 and 38, the model shows excellent precision and recall for some classes (e.g., class 1 and class 5) but struggles significantly with others (e.g., class 2 and class 7), as indicated by lower F1-scores. This suggests inconsistent performance in accurately classifying certain types of network events.

2. ROC Curves:

Despite some inconsistencies in precision and recall, the ROC curves in figure 39 illustrate high AUC scores for all classes, indicating the classifier's strong ability to distinguish



```
Best parameters for SGD Classifier: {'penalty': 'elasticn
Classification report for SGD Classifier:
              precision    recall  f1-score   support

           0       0.50      0.43      0.46     32590
           1       0.91      0.94      0.93     32681
           2       0.81      0.37      0.51     32310
           3       0.60      0.91      0.72     32706
           4       0.50      0.61      0.55     32617
           5       0.99      0.97      0.98     32661
           6       0.97      0.99      0.98     32516
           7       0.49      0.37      0.42     32701
           8       0.50      0.57      0.53     32514

    accuracy                           0.69    293296
   macro avg       0.70      0.69      0.68    293296
weighted avg       0.70      0.69      0.68    293296
```

Fig. 38. Classification report for Stochastic Gradient Descent

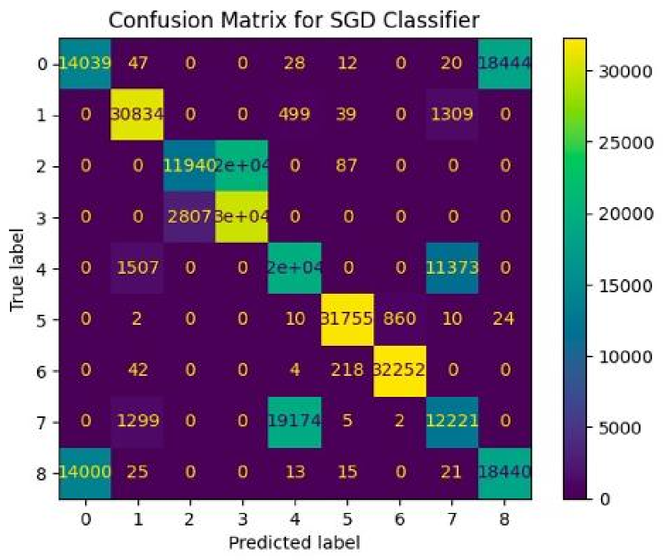


Fig. 39. Confusion Matrix for Stochastic Gradient Descent

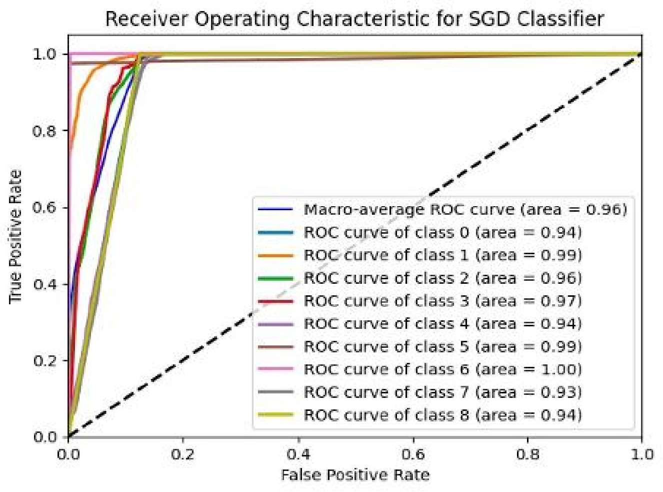


Fig. 40. ROC curve for Stochastic Gradient Descent

between classes at different decision thresholds. This suggests good overall model discriminative power.
Overall, while the SGD Classifier is effective in distinguishing between classes, variability in performance across different classes implies that further tuning or additional feature engineering may be necessary to improve consistency and reliability in predictions.

VI. Model Comparisons

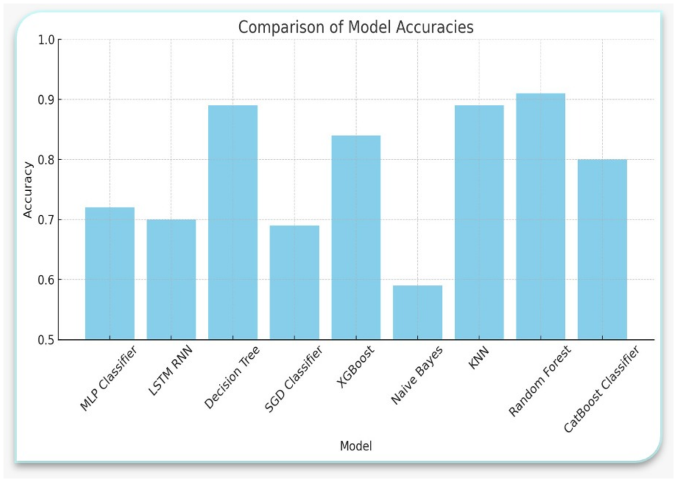


Fig. 41. Comparison of model accuracies

Random Forest, KNN, Decision Tree and XGBoost classifiers have the highest accuracies on the graph in figure 40 meaning that they could potentially be among the best models for this given dataset out of those tested. These models are popular for handling multiple types and complexities of data; therefore, this may contribute to their excellent performance. MLP Classifier, LSTM RNN and CatBoost also perform good but slightly lower than the leading performers. This implies that these models are quite efficient but may need more tuning or might perhaps not be as appropriate to the details of the data as Random Forest and CatBoost. The SGD Classifier, and Naive Bayes show low accuracies.

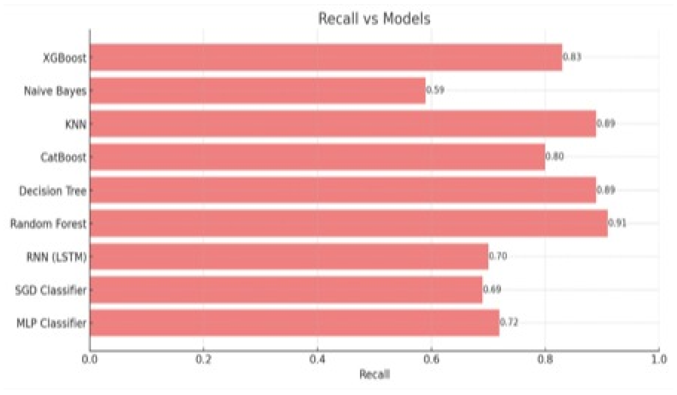


Fig. 42. Recall for all the models

1. Recall vs Models Chart: The chart in figure 41 shows the values of recalls for different models. Recall measures a model's ability to retrieve all relevant instances for a dataset.Random Forest is the best performing model with a recall of 0.91 followed by XGBoost and MLP Classifier.

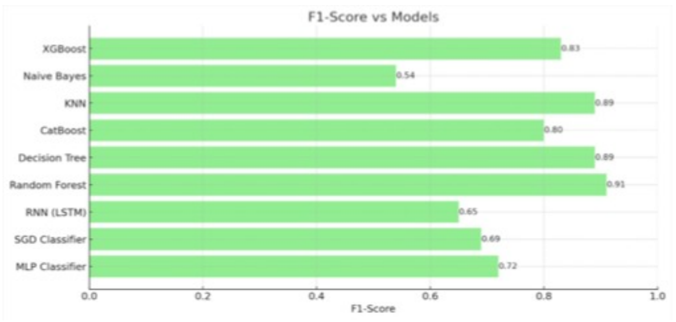


Fig. 43. F1-score for all the models

2. F1-Score vs Models Chart: F1-score is a combination of precision and recall that measures the accuracy of a model. It provides an equilibrium between false positives and negatives hence Random Forest and Decision Tree take the lead with an F1-score of 0.91 showing strong overall performance as shown in figure 42.

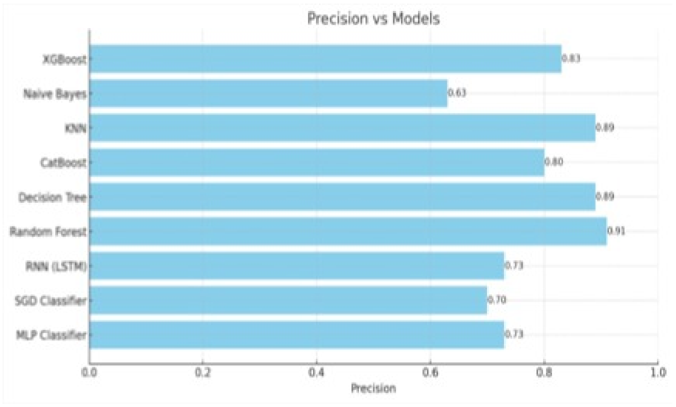


Fig. 44. Precision for all the models

3. Precision vs Models Chart: Precision measures how accurate are positive predictions made by the model.In this figure, both Random forest and decision tree perform best with a precision level at 0.91 which implies that they can predict true positives better than others do without many false positives as shown in figure 43.

VII. Deployment

In terms of the infrastructure for the deployment of our machine learning project, we resorted to Google Colab due to its simplicity of use, accessibility, and the possibility of using a large computational potential. Google Colab also allowed us to easily share code and collaborate and therefore streamline the development and testing processes for our models. In turn, we also took into account the possible problems of scalability and dynamics – conditions of use of free users of servers with limited availability of resources for calculation, unstable computational power, session expiration, etc. To mitigate these issues, we suggest that continuous access to resources be provided by promoting Colab Pro subscriptions; model checkpoints can be arranged and Colab's logging system can be used to regulate dynamic resource allocation; Also, the session management features built into Colab can be leveraged to minimize disruptions. We managed to create and test a Network Intrusion Detection System for the DNP3 Protocol using the Google Colab platform based on the analysis of its strengths and weaknesses.

VIII. Conclusion

The outcome of this effort was a fully functional Intrusion Detection System (IDS) that uses machine learning to identify different DNP3 attacks. For example, the system guarantees high accuracy and low false positives when it comes to detecting various forms of network intrusions. It easily becomes a part of existing network infrastructure thus making it possible to monitor traffic as it happens in real time. Moreover, IDS can improve the security posture of a network by always adjusting itself in response to new threats that are being generated all over. In conclusion, Random Forests, KNN, Decision Tree and XGBoost classifiers are highly documented as the best performing algorithms for DNP3 intrusion detection systems with Random Forest giving an accuracy of 91% .

IX. Future Scope

The future scope of the machine learning project on network intrusion detection in the DNP3 protocol is very vast which entails improvement in model performance through optimization techniques and ensemble learning, working with real-time intrusion detection for quick response, and large-scale deployment of the system using distributed computing platforms. The application of CNNs, GRUs, Transformers, and other advanced deep learning models can be considered, aiming to enhance the detection accuracy; the use of explainable AI techniques (e.g., SHAP, LIME) will also be useful for obtaining insights into the model's decisions. Machine learning constructions would help the system to learn and teach new inputs and new threats while universality across other industrial protocols will increase its potential use. Alignment with cybersecurity frameworks, interface and visualization tool improvement, and industry partner engagement for standardization will enable a comprehensive, technically, and commercially scalable intrusion detection solution for industrial control systems.

Acknowledgment

The authors would like to thank Dr. Shih Yu Chang and Sourab Saklecha at San Jose State University for guiding us through the project and helping us in successfully implementing the intrusion detection system.

References

[1] Panagiotis Radoglou-Grammatikis, Vasiliki Kelli, Thomas Lagkas, Vasileios Argyriou, Panagiotis Sarigiannidis. (2022). DNP3 Intrusion Detection Dataset. IEEE Dataport. https://dx.doi.org/10.21227/s7h0-b081

[2] V. Kelli et al., "Attacking and Defending DNP3 ICS/SCADA Systems", 2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2022, pp. 183-190, doi: 10.1109/DCOSS54816.2022.00041.

[3] V. Kelli, P. Radoglou-Grammatikis, T. Lagkas, E. K. Markakis and P. Sarigiannidis, "Risk Analysis of DNP3 Attacks", 2022 IEEE International Conference on Cyber Security and Resilience (CSR), 2022, pp. 351-356, doi: 10.1109/CSR54599.2022.9850291.

[4] P. Radoglou-Grammatikis, P. Sarigiannidis, G. Efstathopoulos, P.-A.Karypidis, and A. Sarigiannidis, "Diderot: An intrusion detection and prevention system for dnp3-based scada systems", in Proceedings of the15th International Conference on Availability, Reliability and Security, ser. ARES '20.New York, NY, USA: Association for Computing Machinery, 2020, doi: 10.1145/3407023.3409314.

[5] J. Shah, "Understanding and study of intrusion detection systems for various networks and domains," 2017 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2017, pp. 1-6, doi: 10.1109/ICCCI.2017.8117726.

[6] 1815-2012 - IEEE Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3). (2012). IEEE Standard — IEEE Xplore. https://ieeexplore.ieee.org/document/6327578

[7] 1815.1-2015 - IEEE Standard for exchanging information between networks implementing IEC 61850 and IEEE STD 1815(TM) [Distributed Network Protocol (DNP3)]. (2016, December 16). IEEE Standard — IEEE Xplore. https://ieeexplore.ieee.org/document/7786998

[8] G. Clarke and D. Reynolds, Practical Modern SCADA Protocols: DNP3, IEC 60870.5 and Related Systems, Newnes, Oxford, United Kingdom, 2004.

[9] K. Curtis, A DNP3 Protocol Primer (Revision A), DNP3 Users Group, Calgary, Canada (www.dnp.org/About/DNP3%20Primer%20Rev%20A .pdf), 2005.