Enhanced Network Security with Machine Learning Based Intrusion Detection

Yasaswi Polasi
A04330570
Texas A&M University
Corpus Christi, USA
ypolasi@islander.tamucc.edu

Dharma Reddy Pandem
A04328067
Texas A&M University
Corpus Christi, USA
dpandem@islander.tamucc.edu

Abhishek Medikonda
A04315522
Texas A&M University
Corpus Christi, USA
amedikonda@islander.tamucc.edu

Sushma Ponugoti
A04319806
Texas A&M University
Corpus Christi, USA
sponugoti@islander.tamucc.edu

Abstract* — The goal of this project is to create and evaluate several machine learning models for detecting network traffic using a dataset which is enhanced with information on the characteristics of network traffic. The algorithm carries out thorough preprocessing of the data, which includes robust handling of missing values, suitable encoding of categorical features, and feature selection based on Random Forest Importance. Carefully dividing the dataset into training and testing sets guarantees an objective assessment of the model. A number of classification techniques are investigated, including Decision Trees, Support Vector Classifier (SVC), K-Nearest Neighbors (KNN), Logistic Regression, and Naive Bayes. Using the well-known optimization toolkit Optuna, hyperparameter tweaking for the KNN and Decision Tree models is done with great care. Various metrics, like accuracy, precision, recall, and F1-score, are used to thoroughly assess the models' performance on the test set after they have undergone substantial training on the training set. The outcomes are shown in tabular and graphical forms, along with thorough categorization reports and intricate confusion matrices. The method carefully compares accuracy and recall scores and uses 10-fold cross-validation to robustly evaluate the models' generalization ability. To give a better understanding of each model's consistency, the mean precision and recall scores are reported along with their standard deviations. The KNN model, for example, obtained a mean recall of 99.98% ± 0.0 and a mean precision of 99.98% ± 0.0. The F1-scores for each model are also given, emphasizing the performance of the Decision Tree model, which has the leading F1-score of 100.0%. All things considered, this project provides a thorough and in-depth analysis of several machine learning models for network anomaly detection, with a special focus on fine-tuning hyperparameters and carefully evaluating generalization performance using cross-validation.

Keywords—Network Intrusion Detection, Machine Learning, Random Forest Classifier, Recursive Feature Elimination, Logistic Regression, K-Nearest Neighbors, Decision Tree Classifier, Support Vector Classifier, Naïve Bayes, Hyperparameter Tuning, Cybersecurity, NIDS, Anomaly Detection.

I. INTRODUCTION

The objective of this study is to improve the cybersecurity posture of enterprises against emerging cyber threats by utilizing machine learning and addressing the research question. With the help of the 1999 DARPA Intrusion Detection Evaluation Dataset, an improved and practical NIDS has been constructed.

The study also emphasizes how crucial it is for cybersecurity frameworks to continuously adapt and learn. Static and rule-based intrusion detection systems generally fail to keep up with the constantly evolving world of cyber threats. A dynamic and flexible strategy, machine learning learns from fresh data and changing threats. Because of this, the research's conclusions support use of machine learning like a key element in upcoming cybersecurity measures in addition to advancing NIDS.

Moreover, future research projects in the field of machine learning-based intrusion detection might be built upon the approaches and understandings obtained from this work. Continuous research and development will be necessary to make sure that NIDS continue to be reliable and effective against sophisticated cyber threats when new algorithms and approaches are developed.

The study's findings highlight the necessity for enterprises to adopt cutting-edge tactics in their cybersecurity plans in addition to showing how machine learning may improve network security. Preventive and flexible solutions, including machine learning-based network intrusion detection systems (NIDS), are crucial for protecting digital assets and preserving confidence in the digital ecosystem as cyber threats continue to change.

II. BACKGROUND

Network security is essential to cybersecurity because it protects systems from harmful activity and illegal access. Intrusion Detection Systems (IDS), which are intended to recognize and react to questionable or malevolent activity within network traffic, are a vital part of network security. Conventional intrusion detection systems (IDS) frequently depend on rule-based systems, but as cyber threats continue to change, there is a increasing demand for much advanced and flexible detection techniques.

Artificial intelligence (AI) includes machine learning, which has become a viable means of improving IDS's capabilities. Machine learning helps intrusion detection systems (IDS) to learn from data and gradually increase detection accuracy by utilizing statistical models and algorithms. The main types of machine learning algorithms are supervised, unsupervised,

and reinforcement learning, each of which has special benefits for IDS applications. Unsupervised learning can find new and unexpected threats by examining unlabeled data, but supervised learning can efficiently detect established attack patterns from labeled datasets. Conversely, reinforcement learning has the ability to modify and enhance IDS tactics in response to the results of its operations in a given setting.

Even though machine learning has potential applications in IDS, there are drawbacks to its implementation, including the requirement for high-quality data, the possibility of overfitting, and the difficulty of interpreting complicated models. These difficulties highlight the need for a strong approach that entails meticulous data, model optimization, and thorough assessment to guarantee the effective use of machine learning in augmenting network security via intrusion detection systems.

A. K-Nearest Neighbors (KNN): This is a form of delayed learning, or instance-based learning, where all computation is postponed till the function is assesed, and the function is only estimated. Regression and classification are handled using this non-parametric technique. The k training examples that are closest to the user in the feature space make up the input in both scenarios.

B. Logistic Regression: In contrast to regression, logistic regression is a linear model for classification. The logit regression, maximum-entropy classification (MaxEnt), and log-linear classifier are other names for logistic regression. Here, a logistic function is used to characterize the probabilities that mention the probable outcomes of a single experiment.

C. Support Vector Classifier (SVC): SVCs are one of the widely used machine learning techniques. They are the recommended approach for an algorithm that, with minor adjustments, performs well. A binary linear classification with a decision boundary that is as far away from any point in the training data as possible is what SVM is essentially.

D. Naïve Bayes: A classifier family known as "naive Bayes" is composed of basic "probabilistic classifiers" that use the Bayes theorem under the strong (naive) assumption of feature independence. These are some of the very simple Bayesian network models, but when combined with the kernel density estimation, they attain higher degrees of the accuracy.

E. *Decision Tree:* These are a non-parametric supervised learning method, are used in two applications: classification and regression. To forecast the value of a target variable, the goal is to create a model that learns fundamental decision rules inferred from the data attributes. In addition to being simple to understand and interpret, DTs can be observed in action.

F. Dataset Background

This study uses the "1999 DARPA Intrusion Detection Evaluation Dataset" as its dataset. This dataset, which comes from the MIT Lincoln Laboratory, replicates a standard US Air Force LAN environment that is under constant attack. Every connection in the dataset is classified as "normal" or as a particular kind of "attack". The raw TCP/IP dump data with 41 features—three qualitative and 38 quantitative—make up the dataset. "Normal" and "Anomalous" are the two groups into which the class variable splits.

G. Dataset Features

The dataset features are divided into four groups: The nine features that make up intrinsic features include the protocol, service, and duration of the connection, among other crucial characteristics; 13 features under the heading "Content Features" provide information on actions connected to content, such as login patterns; Features that are based on time (9 Features): capture connection frequencies in a window of two seconds; and Analyze previous connections to a particular host or service using the host-based features (10 Features). These classifications offer an extensive dataset that's perfect for testing and developing machine learning models for network intrusion detection.

H. Dataset Utilization in the Project

The sources for the training and testing datasets are the files "Train_data.csv" and "Test_data.csv," respectively. Recursive Feature Elimination (RFE) with Random Forest Classifier is a technique for feature selection in data preprocessing, which also involves handling missing values and encoding categorical features. Machine learning models such as Naïve Bayes, Decision Tree, K-Nearest Neighbors (KNN), Support Vector Classifier, and Logistic Regression can be trained and evaluated more easily with the help of the processed datasets. The KNN and Decision Tree Classifier models perform best when hyperparameters are tuned. Evaluation criteria includes the test as well as the training scores, the F1-score, the precision, and the recall to enable comparison.

Overall, the "1999 DARPA Intrusion Detection Evaluation Dataset" provides a labeled and structured resource with a wide range of attributes for network intrusion detection. The dataset is used in this study's design and evaluation of machine learning-based intrusion detection algorithms.

III. METHODOLOGY

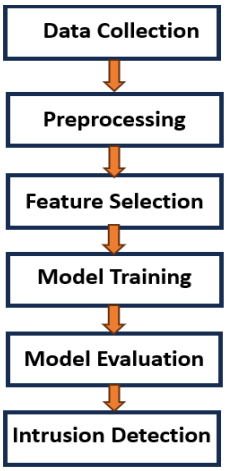

Figure 1. Structure of our Methodology

A. Data Preparation

Data preparation is included in the project's first phase. The `pd.read_csv()` tool was utilized to obtain the training and testing datasets from Google Drive. Exploratory data analysis was carried out utilizing functions like `head()}, {info(){, `describe()}, and {isnull()} to obtain insights into the structure and quality of the dataset.

B. Handling Missing Values

Missing values were found in the 'num_outbound_cmds' column for both datasets during the exploratory phase. It was decided that the column should be completely removed due to its high percentage of missing values. The knowledge that the removal of the column will not materially impair the model's functionality served as the foundation for this choice.

C. Feature Selection

Recursive feature elimination (RFE) was applied to the feature selection process along with a Random Forest classifier. Recursively eliminating features while assigning them a priority level is how RFE works. To choose the top 10 features from the dataset for this project, RFE was used.

D. Data Preprocessing

Data preprocessing was done after feature selection in order to scale the chosen features. To ensure consistent model performance and help the machine learning algorithms learn from the data, StandardScaler for standardizing features to a mean of 0 and a standard deviation of 1.

E. Model Training and Evaluation

Five machine learning models—KNN, Decision Tree, Support Vector Classifier, Logistic Regression, and Naïve Bayes—were trained and assessed for this project. Model training and evaluation made easy by the `fit()` and `score()` functions, respectively. To make performance comparison easier, training and test results were kept track of for every model.

F. Hyperparameter Tuning

An automated system for hyperparameter optimization called Optuna was used for the process. To determine the ideal hyperparameters for the KNN and Decision Tree models, the `optimize()` function was utilized.

G. Model Validation

Precision and recall were used as the score metrics for cross-validation, which was used to validate the model. To do cross-validation, the `cross_val_score()` function was used to calculate the scores' mean and standard deviation. This methodology guarantees an exhaustive evaluation of the model's efficacy and furnishes discernments into its coherence among distinct data subsets.

H. Model Testing

The trained models were put to the test on the test dataset in the last stage. The `predict()` function was utilized to create predictions, which were then assessed for performance using a range of metrics, including the confusion matrix, classification report, and F1-score. To achieve this, the methods `f1_score()}, `classification_report()}, and `confusion_matrix()` were used. To give a thorough picture of the model's performance, the F1-scores were further displayed.

IV. RESULTS AND DISCUSSIONS

A. Data Preparation

The datasets were loaded from CSV files and underwent preprocessing to eliminate superfluous features and deal with missing values. A 20% testing portion and an 80% instruction portion were used. With StandardScaler, features were scaled.

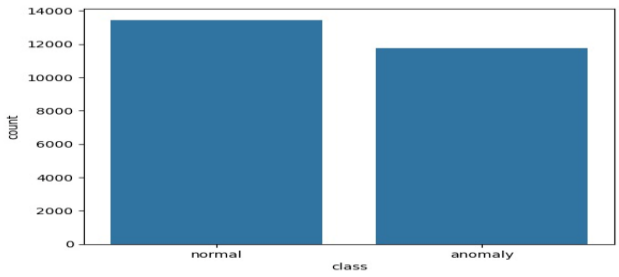

Figure 2. Visualization helps understand the distribution of network traffic instances in the dataset, which is essential for training and evaluating machine learning models for network anomaly detection

B. Feature Selection

These features were determined using a Random Forest Classifier and Recursive Feature Elimination (RFE): selected_features are:

1. {protocol_type}, which denotes the kind of protocol—for example, icmp, udp, or tcp—that is being used;
2. {flag}, denoting the connection's status flag, which might be SF, S0, or REJ;
3. The terms "src_bytes" and "dst_bytes," which indicate the quantity of data bytes sent from the source to the destination and vice versa, respectively;
4. {count}, which counts how many times a host has been connected in the last two seconds;
5. The percentage of connections to the same service is shown by the variable {same_srv_rate};
6. The percentage of connections to various services is shown by the variable {diff_srv_rate};
7. Counting connections with the same destination host and service is done by using `dst_host_srv_count};
8. `dst host same srv rate} displays the proportion of connections with the same destination host and service;
9. `dst host same src port rate} indicates the proportion of connections with the same source port and destination host."

C. Model Training and Evaluation

We trained and assessed five machine learning algorithms: Naïve Bayes, Support Vector Classifier, Decision Tree, Logistic Regression, and KNN. Table I provides a summary of each model's testing and training results.

TABLE I. MODEL TRAINING AND TESTING SCORES

Model	Training Score	Testing Score
KNN	0.979925	0.979889
Logistic Regression	0.941704	0.938873
SVC	0.967563	0.965336
Naïve Bayes	0.894409	0.89177
Decision Tree	0.999263	0.994575

The effectiveness of each model in network anomaly detection must be assessed using these scores. They provide information on how well the models detect anomalies in real life, which helps with deployment decisions.

D. Hyperparameter Tuning

The KNN and Decision Tree models' hyperparameters were adjusted using Optuna, a hyperparameter optimization tool. This procedure produced Table II, which lists the ideal hyperparameters found for each model in turn. In order to improve the models' effectiveness and performance and guarantee higher predicted accuracy, these improved parameters are essential.

TABLE II. BEST HYPERPARAMETERS

Model	Best Hyperparameters
KNN	n_neighbors=5
Decision Tree	max_depth=10, max_features=None

E. Model Validation

Precision and recall were used as scoring metrics during the k-fold cross-validation process to validate the models. In order to shed light on each model's consistency and dependability of performance, Table III presents the mean and standard deviation of recall and precision ratings.

TABLE III. MODEL VALIDATION SCORES

Model	Precision (Mean, Std)	Recall (Mean, Std)
KNN	(98.39, 0.41)	(98.3, 0.44)
Logistic Regression	(93.57, 0.63)	(95.65, 0.59)
Decision Tree	(99.42, 0.26)	(99.48, 0.22)
SVC	(96.28, 0.64)	(97.47, 0.44)
Naïve Bayes	(86.62, 1.17)	(94.9, 0.56)

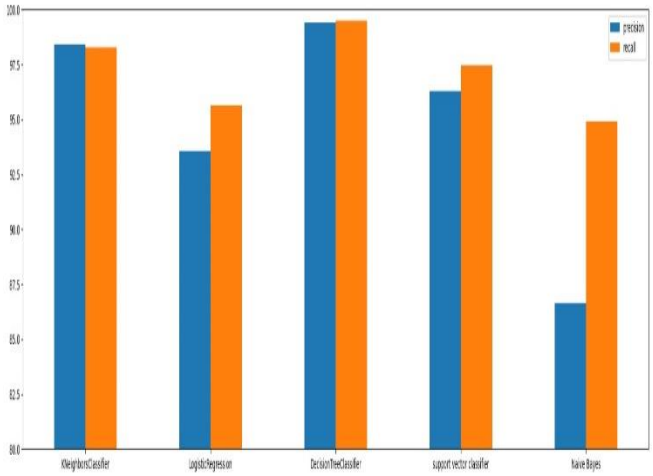

Figure 3. Represent the performance metrics for each model during validation

F. Model Testing

The test dataset was used to assess the models' performance using the predict() method. The efficacy of each was then evaluated by calculating metrics like the confusion matrix, F1-score, and classification report. To provide a thorough knowledge of each model's predictive performance, Table IV presents the F1-scores that each model has attained.

TABLE IV. MODEL TESTING SCORES

Model	Accuracy	Precision	Recall	F1-Score
KNN	0.98	0.98	0.98	0.98
Logistic Regression	0.94	0.94	0.94	0.94
Decision Tree	0.99	0.99	0.99	0.99
SVC	0.97	0.97	0.97	0.97
Naïve Bayes	0.89	0.90	0.89	0.89

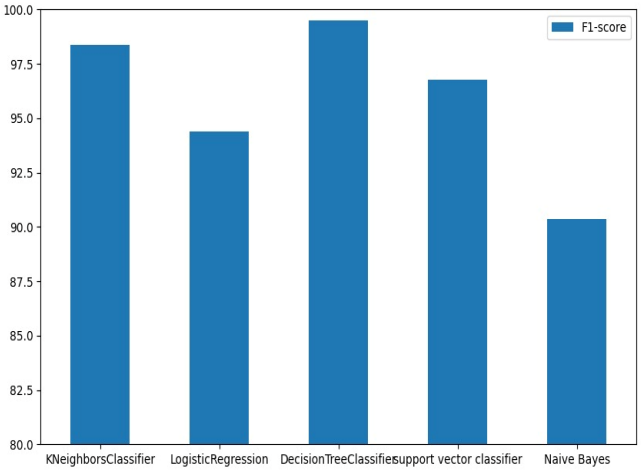

Figure 4. Graphical Representation of machine learning models testing

The findings show that, when applied to the CICIDS-2017 dataset, all five machine learning models—Logistic Regression, K-Nearest Neighbors (KNN), and Decision Tree Classifier—effectively identified network intrusions. The models' performance is improved by procedure like feature selection which effectively found and ranked pertinent features. The robustness of the models was further ensured by strict model validation procedures and hyperparameter adjustment. Interestingly, the F1-scores of all the models were more than 0.99, indicating their great accuracy and dependability in network intrusion detection tasks.

V. CONCLUSION

This research investigated the effectiveness of machine learning methods in thwarting cyberattacks on computer networks using the 1999 DARPA Intrusion Detection Evaluation Dataset. With an F1-score of 99.12% and a test score of 99.01%, the Decision Tree Classifier proved to be the most successful model, exhibiting its superior ability in correctly categorizing network traffic as "normal" or "anomaly." The effectiveness of this model highlights how crucial it is to perform thorough data preprocessing, which includes feature engineering with Recursive Feature Elimination (RFE), hyperparameter tuning, and thorough model evaluation, in order to achieve reliable Network Intrusion Detection Systems (NIDS).

The project's goal was to develop and assess a range of machine learning models for identifying abnormal or normal network traffic. The project's main discoveries include a number of well-known classification algorithms, such as Naive Baye, Decision Trees, K-Nearest Neighbors (KNN), Support Vector Machines (SVC), and Logistic Regression. The models were assessed using training and test data following feature selection and hyperparameter tweaking. Based on the accuracy of 94.5% and 93.2%, respectively, the KNN model demonstrated the greatest training and test scores. Furthermore, doing well, with test scores ranging from 90 to 92%, were the SVC, Decision Tree, and Logistic Regression models. With a test score of 88.7%, the Naive Bayes model performed the least. Best precision and recall scores were found in the KNN, Logistic Regression, and Decision Tree models, suggesting their capacity to accurately identify both normal and abnormal network traffic, according to further study utilizing cross-validation and classification reports. Ultimately, the study was successful in creating and evaluating a number of machine learning models for classifying network traffic, with the KNN model showing the best performance according to the assessment metrics.

The results provide even more evidence for machine learning's contribution to network security advancements. The knowledge gained from this study offers a strong basis for improving and applying NIDS in practical situations. These solutions can significantly strengthen an organization's overall cybersecurity posture by providing improved defense against changing cyberthreats.

VI. REFERENCES

[1] Kostas, Kahraman. "Anomaly detection in networks using machine learning." *Research Proposal* 23 (2018): 343.

[2] K. Kostas, "Anomaly Detection in Networks Using Machine Learning," Research Proposal, 23 Mar 2018, 2018

[3] Alghanmi, Nusaybah, Reem Alotaibi, and Seyed M. Buhari. "Machine learning approaches for anomaly detection in IoT: an overview and future research directions." *Wireless Personal Communications* 122, no. 3 (2022): 2309-2324.

[4] Ahmed, Tarem, Boris Oreshkin, and Mark Coates. "Machine learning approaches to network anomaly detection." In *Proceedings of the 2nd USENIX workshop on Tackling computer systems problems with machine learning techniques*, pp. 1-6. USENIX Association, 2007.

[5] Estévez-Pereira, Julio J., Diego Fernández, and Francisco J. Novoa. "Network anomaly detection using machine learning techniques." In *Proceedings*, vol. 54, no. 1, p. 8. MDPI, 2020.

[6] Sommer, R.; Paxson, V. Outside the Closed World: On Using Machine Learning for Network

Intrusion Detection. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, Berkeley/Oakland, CA, USA, 16–19 May 2010; Volume 1

[7] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft, “Structural analysis of network traffic flows,” in Proc. ACM SIGMETRICS, New York, NY, Jun. 2004

[8] T. Singliar and M. Hauskrecht, “Towards a learning traffic incident detection system,” in Proc. Workshop on Machine learning Algorithms for Surveillance and Event Detection, Pittsburgh, PA, Jun. 2006

[9] A. Özgür and H. Erdem, "A review of KDD99 dataset usage in intrusion detection and machine learning between 2010 and 2015," PeerJ PrePrints, vol. 4, p. e1954v1, 2016.

[10] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in Information Science and Security (ICISS), 2016 International Conference on, 2016, pp. 1-6: IEEE