Article

Version August 31, 2024 submitted to Journal Not Specified

Abstract: Here is a classification of more nuanced cyber attacks. All these steps bridge this gap in the present study. In this case, machine learning-based classification can help thanks to automated analytical skills of ML and fierce competition at the same time. It involves handling of null values, working around float numbers and smart feature engineering to improve the model functionality. It gets sillier: the project even offers up an automatic user categorization toolset, for web-based or mobile applications to separate attackers from real users. The study creates an ensemble model by assessing individual models, cross-validation and hyperparameter optimization. Through intrusion detection techniques and automating user classification, the research proactively advances cyber security while simultaneously contemplating its own potential for auto-improvement.

Keywords: Feature engineering, Cross Validation, Ensemble Model, Hyperparameter Tuning

1. Introduction

Given the ever-evolving cyber security scenario, now more than before Enterprises need stronger Intrusion Detection Systems (IDSs). This is the problem statement of this study applied to common approaches using machine learning for the purpose increasing precision and effectiveness of intrusion detection systems. Cyberattacks are becoming more and more sophisticated; the traditional, rule-based ones that we focus on detecting in networks are largely insufficient to discover these novel threats. Machine learning algorithms when trained with high quality datasets like UNSW-NB15 dataset can enhance detection rates and reduce false positive rates of intrusion detection systems.

An ensemble-based classifier is a wise choice as the intrusion detection technology has matured and multiple algorithm types have to be incorporated into one system. In order to provide more accurate results, these classifiers aggregate predictions from a set of base models. The data has to be processed well, you need to select the right features and tune your model parameters correctly. Ensemble-based classifiers development looks more promising in enhancing the performance of IDS solutions, These classifiers aggregate the predictions of multiple base models and approach from different angles to make more accurate decisions as a whole. The performance of an ensemble-based IDS, nevertheless is influenced by number of important factors. Raw data has to be cleaned up and transform into an efficient pre-processing technique using data pre-processing techniques, for machine learning models process. Feature selection is all about selecting the most relevant and discriminative features as input of models to perform better.

Optimizing parameters for ensemble-based classifiers can improve the accuracy and performance. Parameter optimization helps the model in recognizing threats more effectively, and by optimizing its parameters it makes intrusion detection even faster.

[1] Corresponding Author: Name, e-mail:@gmail.com

The main significance of the task may be enhancing Mobile Application Security. As mobile applications become more important in our lives, maintaining their security is critical. The research contributes to the protection of mobile applications against a wide range of risks by establishing a model capable of accurately identifying cyber-attacks. It also will be helpful for Sensitive Data Protection. Mobile applications frequently handle sensitive user data, ranging from personal to financial information. Our project protects this important data from fraudsters and malicious assaults by introducing robust attack classification algorithms. Another benefit of the strategy is that it helps to avoid financial losses. Individuals and companies can suffer large financial losses due to cyber assaults. By preventing data breaches, fraud, and other financially motivated attacks, this project has the potential to save millions of dollars. Future-proofing against emerging threats will reap advantages from the model. Cyber threats are constantly changing, necessitating adaptable and wise remedies. Our project enables the creation of proactive cybersecurity solutions that can keep up with new and emerging threats by building the framework for attack classification and notification. The worldwide cybersecurity community can grow more resistant to cyberattacks by distributing the information and tools created in this initiative. The entire stability of the digital ecosystem may be benefited as a result. The project's success might act as a spark for additional cybersecurity innovation. It motivates scientists and programmers to investigate novel approaches and technologies to improve attack detection and prevention.

Our study is driven by two key motivations. Firstly, in a constantly evolving cybersecurity landscape, the imperative for effective intrusion detection has never been greater to safeguard networked systems. Leveraging the UNSW-NB15 dataset and employing ensemble classifiers, we want to bolster ongoing efforts to develop intrusion detection systems that are not only more reliable but also remarkably accurate, offering a robust defense against emerging threats. Secondly, as interest and innovation in machine learning continue to surge, there is an exciting opportunity to enhance the performance of Intrusion Detection Systems (IDS). We believe that we can push the boundaries of IDS capabilities by harnessing the power of ensemble approaches, coupled with meticulous data preprocessing and hyperparameter optimization. This study seeks to delve into the vast potential of these cutting-edge technologies, paving the way for novel solutions to address the ever-evolving challenges in cybersecurity.

The significance of this study rests in its potential to advance intrusion detection technology. We intend to enhance the dataset quality and subsequently the performance of the model by carefully handling null values, processing floating-point integers, and selecting essential features through Feature engineering. Moreover, we provide a point of reference for intrusion detection accuracy using separate models Creation values highlight model performance at different thresholds. The use of cross-validation makes our models robust and have a better power to be generalized in novel data. A more accurate model to generate an effective intrusion detection system is created by combining models and afterward, the hyperparameters are optimized.

The following is a list of our paper's key contributions:

- **Feature engineering:** an essential data science and machine learning preprocessing step that significantly increases the prediction accuracy by carefully choosing, transforming or creating input variables to extract meaningful information while chipping off noise.
- **Cross Validation:** It is a technique to assess the model's performance of machine learning with testing and training sub-datasets that help generalize the data.
- **Ensemble Model:** Combining the three independent models to create an ensemble-based classifier yielded, demonstrated the value of employing multiple models together for enhanced intrusion detection.
- **Hyperparameter Tuning:** Merging the three individual models into a combined ensemble-based classifier demonstrated that using multiple models together improved Intrusion detection.

The research manuscript is structured to explore its content systematically. Chapter 2 conducts a comprehensive literature review, providing insights from existing research. In Section 3, the research methodology of our framework is unveiled, including dataset description, preprocessing, model evaluation, parameter tuning, and ensemble classifier evaluation. The next chapter, Chapter 4, presents a Result analysis and discusses key findings, leading to Chapter 5, which concludes the study, summarizes the findings, and provides implications for future research.

2. Literature Review

Yanping Shen et al. [1] developed a Chaos Bat Algorithm-Based Class-Level Soft-Voting Ensemble (CBA-CLSVE) for intrusion detection, proving best on the UNSW-NB15 dataset, but lacks detailed accuracy and limitations explanations.

Maya Hilda Lestari Louk et al. [2] developed a hybrid ensemble and PSO-driven feature selection for anomaly detection in intrusion detection systems, despite a lack of accurate accuracy estimates, enhancing performance characteristics.

Eva Maia et al. [3]developed an intelligent system for network intrusion detection using RF, MLP, and LSTM models, achieving 99.94% accuracy on the CIDDS-001 dataset, but faced drawbacks like small datasets and lack of evaluation metrics.

Maryam Yousefnezhadi et al. [4] developed an IDS ensemble using deep learning and kNN/SVM ensemble models for intrusion detection, assessing datasets from UNSW-NB15, CICIDS2017, NSL-KDD, and KDD99.

Ebrima Jaw et al. [5] proposed an efficient intrusion detection system using a KODE ensemble classifier with hybrid feature selection, achieving 99.99% accuracy on CIC-IDS2017, NSL-KDD, and UNSW-NB15 IDS datasets, but did not address limitations or data limitations.

Muataz Salam Al-Daweri et al. [6] developed a dynamic artificial neural network (HOE-DANN) for intrusion detection using a homogenous ensemble of DANNs from a single class, achieving high accuracy rates without any difficulties or methodological constraints.

Prabhat Kumar et al. [7] proposed a Fog computing and distributed ensemble IDS for IoT network security, using XGBoost, K-Nearest Neighbors, and Gaussian Naive Bayes. However, the accuracy rate and efficacy of the method remain unknown.

KAIYUAN JIANG et al. [8] proposed a Network Intrusion Detection Combined Hybrid Sampling with Deep Hierarchical Network, achieving 83.58% and 77.16% accuracy on the NSL-KDD and UNSW-NB15 datasets, but may have high false detection rates.

Muhammad Zeeshan et al. [9] developed a robust intrusion detection system for DoS and DDoS attacks using deep learning and unsupervised LSTM models, achieving 96.3% classification accuracy, despite dataset limitations.

Xavier Larriva-Novo et al. [10] proposed an IoT-focused intrusion detection system using preprocessing techniques for cybersecurity datasets. However, limitations include not considering redundant records, outdated datasets, direction-based traffic characteristics, and scalability for large-scale datasets.

Bamidele Adebisi, et al. [11] developed a Deep Learning Algorithm for Botnet Detection in Internet-of-Things Networks, outperforming existing models in precision, recall, F1 score, AUC, geometric mean, and Matthews correlation coefficient, but with higher computation costs and longer training time.

Cite	Author Name	Method/Algo	Dataset	Limitation
[1]	Yanping Shen	The Chaos Bat Algorithm (CBA) was employed in the paper's Class-Level Soft-Voting Ensemble (CLSVE), which included SVM, KNN, and DT base learners	For testing and comparison, the article employed the NSL-KDD, UNSW-NB15, and CICIDS2017 datasets. On the UNSW-NB15 dataset, CBA-CLSVE fared better than other models	The suggested CBA-CLSVE scheme's shortcomings are not discussed in the study, demanding more research or sources to fully understand its specifics
[2]	Maya Hilda Lestari LoukA	To improve the performance of the intrusion detection system (IDS), the article employed a hybrid ensemble model made up of the Gradient Boosting Machine (GBM) and Bootstrap Aggregation (bagging)	Three datasets were evaluated in the paper: NSL-KDD, UNSW-NB15, and CICIDS-2017	The studies and proposed approach's shortcomings in enhancing intrusion detection system performance utilizing a hybrid ensemble and PSO-driven feature selection strategy are not addressed in the article
[3]	Eva Maia	To detect anomalies in network traffic data, the article used Random Forest (RF), Multi-Layer Perceptron (MLP), and Long-Short Term Memory (LSTM) models	CIDDS-001 dataset	The work admits limitations: small datasets, no Random Forest/MLP models, potential anomaly ID constraints, and lacking assessment metrics or experimental settings
[4]	Maryam Yousef Nezhad	The article used ensemble models (kNN and SVM) for intrusion detection	The article utilized network traffic datasets: UNSW-NB15, CICIDS2017, NSL-KDD, Cyber Systems/MIT Lincoln Lab, and KDD99 for assessment	The publication doesn't discuss computation, scalability, data quality changes, or generalizability to diverse networks
[5]	Ebrima Jaw	For intrusion detection, they employed an ensemble classifier that integrated K-means, One-Class SVM, DBSCAN, and Expectation-Maximization (KODE) in a hybrid feature selection (HFS) method	For evaluation, the paper employed the CIC-IDS2017, NSL-KDD, and UNSW-NB15 standard IDS datasets	The study's limits, difficulties, and areas for development are not fully discussed in the report, nor are the authors' flaws or data limitations addresse
[6]	Muataz Salam Al-Dawer	For intrusion detection, the article used a homogeneous ensemble based on single-class dynamic Artificial Neural Networks (DANNs)	The datasets utilized for evaluation in the paper were KDD99, UNSW-NB15, and gas pipeline data logs (GPDL)	The article makes no mention of the limitations of the study or the suggested methodology, and the sources cited make no reference of any particular problems experienced when conducting the study

Table 1. Literature Review

Vikash Kumar et al. [12] developed an integrated rule-based intrusion detection system using UNSW-NB15 and RTNITP18 datasets. The system achieved high performance with 83.8% accuracy and 88.29% attack detection rate, but struggled to identify zero-day threats for large-scale networks.

Sydney M. Kasongo et al. [13] analyzed an intrusion detection system using a feature selection method on the UNSW NB15 dataset, achieving 90% accuracy. However, the system's limitations, including dataset description, model capabilities, and computational resources, hinder evaluation.

Jielun Zhang et al. [14] developed an ensemble network intrusion detection model using Bayesian deep learning and Bayesian Convolutional Neural Network (CNN). The model outperformed baseline models, demonstrating robustness against adversarial attacks.

Hadeel Alazzam et al. [15] developed a feature selection algorithm for an intrusion detection system using pigeon-inspired optimizers, outperforming other algorithms in terms of TPR, FPR, accuracy, and F-score.

Jianlei Gao et al.[16] research on network intrusion detection uses incremental Extreme Learning Machine and Adaptive Principal Component Analysis, offering stronger learning and faster convergence, compared to other algorithms.

Ahmad S. Almogren et al. [17] proposed a study on intrusion detection in Edge-of-Things computing and found a DBN-based approach with higher accuracy, but its performance depends on training dataset quality and computational requirements.

Andrew Churcher et al. [18] conducted an experimental analysis on machine learning for attack classification in IoT networks using the Bot-IoT dataset, achieving 99% accuracy but facing limitations like class imbalance and feature size variations.

Omar Almomani et al. [19] developed a feature selection model for Network Intrusion Detection Systems using PSO, GWO, FFA, and GA algorithms, achieving the best accuracy and sensitivity using Rules 13 and 12.

3. Research Methodology of Our Framework

We used the UNSW-NB15 dataset[21], which has 225,744 rows and 79 feature columns, for this investigation. We then carried out preprocessing operations on the dataset including normalizing all data 0 to 1s, replacing infinite values by the biggest finite number and null values with mean calculation. We used recursive feature elimination (RFE) to identify the most relevant features. The split of the dataset into testing and training datasets was based on an 80:20 ratio. After which the train data set is splitted in 80:20 ratio to Train and validation. Make an Ensemble Model for Decision Tree, Logistic Regression and Naive Bayes model. It was employed to train individual categorization models: Decision Tree, Logistic Regression, Naive Bayes and Ensemble model. A 5-fold cross validation is performed to evaluate model performance. In this comprehensive study, the ensemble model was also improved by high-parameter tuning to optimize higher predicted accuracy using a grid-search strategy for extracting the best group of parameters.

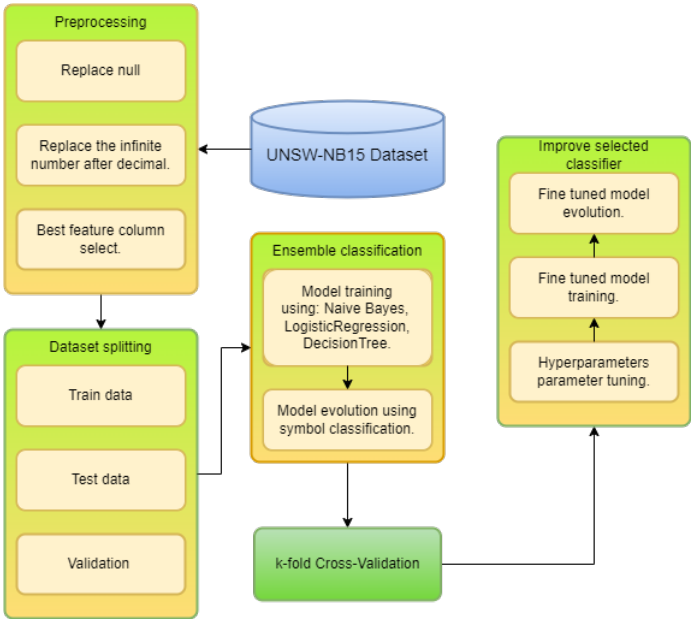

Figure 1. Workflows for the proposed methodology.

Each model is trained using a patient five-fold cross-validation technique on the four available folds of training data and tested carefully with the one reserved fold for testing after we split UNSW-NB15 [18] dataset into his partitions. The objective of this procedure is to analyze how well the model performs, allowing us to identify optimal parameters through high-parameter tuning. During the complex optimization, every model to maximize its prediction power is further tuned closely. The models are then expertly 'stacked' through ensemble orchestration to their positive potential. Then, the reserved data set is used to test the cumulative predictive ability of this ensemble. Having these processes synchronized inside the framework, ensures a thorough inspection of data details and hence helping us come up with an ensemble model which by far epitomizes accuracy and robustness in predictive analytics.

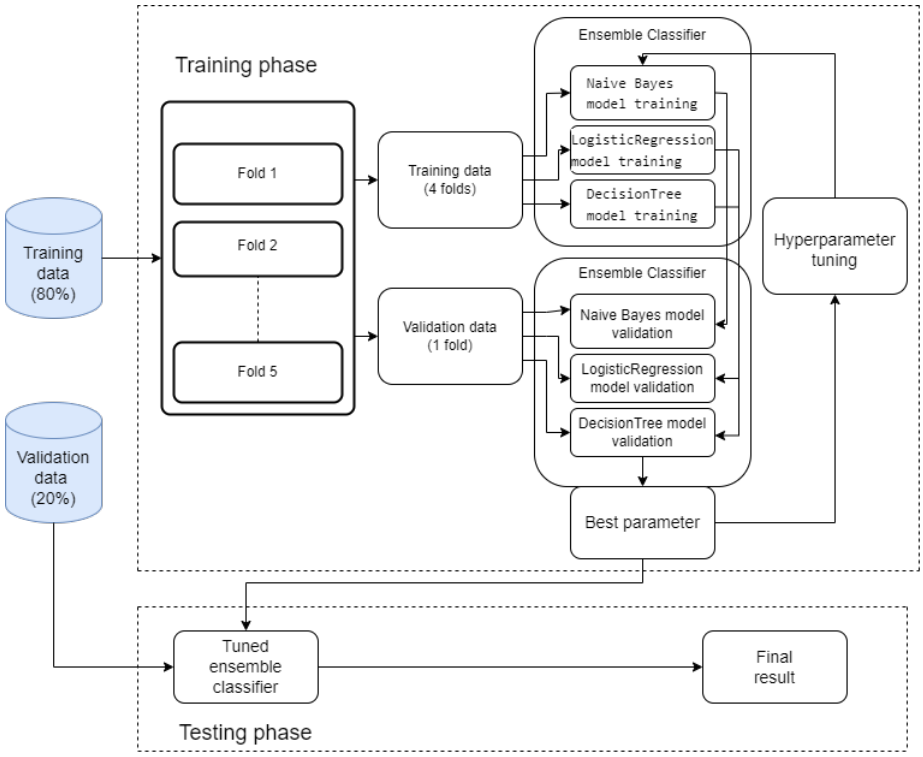

Figure 2. Cross validation along with hyperparameter tuning.

3.1. Dataset Description

The UNSW-NB15 dataset [18] is a good example of network traffic data used in cybersecurity research. Created with the IXIA PerfectStorm tool part of the UNSW Canberra Cyber Range in a Box TextureHandlerContext Plugin Mapping Test Cases. The dataset is full of some raw network packets from tool which can be used to observe many networking activities. With focus on network security investigation, the data points are structured in 225,744 rows with at most maximum specific details totaling to 80 columns making it a theoretical manifesto for research. The dataset will be Study and Test for the machine learning model, 80% of which is used for training, and 20% are going to test and validation. The most important 25 associated feature columns from the dataset to increase model performance-remove these unnecessary features. Its dataset includes abundant feature selection and curation, as a result of which it is capable for developing machine learning models tailored to network security applications. The UNSW-NB15 dataset [18] is a rich and interesting set of network traffic data used in cybersecurity research.

3.2. Preprocessing

The first step in our data preparation workflow was to put this dataset into a Pandas DataFrame. Missing values in the dataset were managed by imputing means of corresponding columns, resulting in a complete dataset. However, we needed to handle infinite values by replacing them with the largest possible integer that the program could process. This is basically a data cleaning step that tries to validate the validity of our study by ensuring that there are no outliers in our dataset or other stray large values. The full dataset was then normalized to a range [0, 1] by min-max normalization of all data points between zero and one. We use Recursive Feature Elimination (RFE) with the goal to select features and we set it up in such a way that after this process has been carried out, 25 of the most correlated attributes for the dependent variable list will be selected. We have splitted the total processed data into train and test (80%-20%) using 80:20 Split ratio. These extensive preparation steps

are needed in order to guarantee data quality, increase our models predictive power and pass-through more analysis and modeling on our data.

3.3. Model Evaluation:

3.3.1. Cross-Validation:

Cross-validation is an essential technique in machine learning, used to evaluate the generalization and predictive capabilities of classification models.

- **Stratified K-Fold Cross-Validation:** Stratified K-Fold Cross-Validation: For model evaluation in classification tasks, Stratified k-fold cross-validation is probably the most commonly used technique. Stratified K-Fold technique was capable of keeping the distribution over class labels on each fold used in this investigation, ensuring representative samples for both training and testing as well. In our case we set the value of k as 5, meaning that the dataset was split into k=5 even folds or subsets. We selected $k = 5$ such that we obtain a trade-off in the time it takes for computation, and still get a rigorous evaluation. Important to note that as k gets larger, it is often more computationally expensive even with large datasets; and conversely when k becomes small data results contain significant variation. The Stratified K-Fold cross-validation technique splits the dataset into 'k' sets to determine performance variation, trains it on 'k-1' groups and tests this published model over a final fold while also providing an accuracy score for each set.

3.3.2. Ensemble Classification:

Data instances are considered as sequences of symbols, such as letters or other discrete units, in the context of ensemble classification algorithms. These sequences are categorized into predetermined classes or categories using machine learning methods. In situations where data naturally presents itself in a sequential pattern, such as text categorization, genomics, and, in our particular context, the classification of cyber assaults, this technique proved particularly helpful.

Each data instance is converted into an ensemble sequence inside the ensemble classification framework, and several attack classification techniques are used to classify these sequences into multiple classes. We use an ensemble learning technique called "Hard Voting," which makes use of the well-known Python package sci-kit-learn. With this approach, the results of many separate machine-learning models are combined to provide a single forecast.

In this type of time, all the constituent models in ensemble are allowed to vote for an Input regarding which class label they think would be best suitable. It creates a robust and democratic categorization by selecting the class label with the highest votes as a final prediction. Ensemble classification allows us to utilize the best of three unique machine learning approaches = decision trees, logistic regression and Gaussian Naive Bayes. Combined, these algorithms help to enhance the performance of our categorization Engine in terms of consistency and precision. Logistic Regression is a linear classification method, Decision Trees use hierarchical rule-based decisions and Gaussian Naive Bayes leveraging probabilistic modeling. Combining these several strategies, we construct a more deep and strong ensemble classification system to deal with our particular classification issues.

3.3.3. GaussianNB (Gaussian Naive Bayes):

The Gaussian Naive bayes is a probabilistic classification method that assumes the Gaussian probability distribution for estimation of likelihood to represent continuous features. The 'naive' assumption of independence between features is made to simplify calculations in calculating the posterior probability distribution over class labels. This mathematical technique basically uses the prior probabilities, class-conditional probabilities and Bayes' theorem to determine which of mutually exclusive classes a collection of characteristics most probably belong. You can also rely on Laplace smoothing to deal

with zero probability and make the models more robust against unseen words. Particularly effective in tasks like spam detection and sentiment analysis, this method's equation is:

$$P(y|x) = \frac{P(y)\pi_{i=1}^{n} P(x_i|y)}{P(x)} \tag{1}$$

Here, P(y|x) denotes the posterior probability of class y given sequence x, P(y) signifies the prior probability of class y, $P(x_i \mid y)$ represents the likelihood of observing symbol x_i given class y and P(x) stands for the marginal likelihood of the sequence.

3.3.4. Decision Trees (DecisionTree):

A nonparametric algorithm for classification known as decision trees discriminatively splits the data into subgroups based on which features are most beneficial. A Decision Tree classifier is a hierarchical structure in which each internal node represents judgements based on the attributes and each leaf node contains class labels. We have seen how with the help of features Gini impurity or entropy, we decide when to split and in whole learning process it keeps recursively splitting on datasets based on best feature splits. Model performance is assessed using evaluation metrics such as accuracy, recall, precision and F1 score. In order to understand how a Decision Tree model makes its decisions and its uses in Machine Learning, it is important that you know the mathematics behind this concept. Decision Tree Equation-

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2(p_i) \tag{2}$$

$$Gain(S, A) = Entropy(S) - \sum_{v \epsilon Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \tag{3}$$

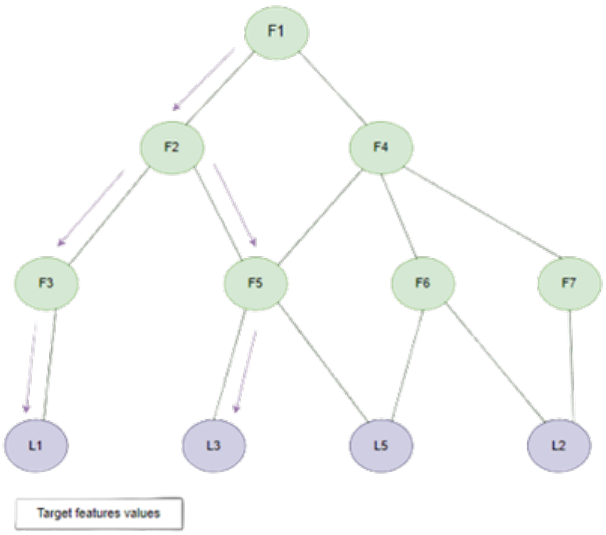

Figure 3. Decision Tree[22]

3.3.5. Logistic Regression:

Logistic regression, which is a very important basic method for binary classification belongs to the category of generalized linear models in statistics. Its building block is the logistic transform - a sigmoid that maps real-valued inputs to numeric predictions in (0, 1). The quality of the combined

logistic function is defined by linear sets of input characteristics that are predicted as a result through model parameters offering to Introduction functions about. Here likelihood of seeing data given model parameters (which are estimated by Maximum Likelihood Estimation, MLE) defined the maximum a posteriori estimate is converted to convex optimization problem. Cost function: let generated probability distribution be P, it is a measure of how far off anticipated vs real labels are in terms of the label space; essentially - negative log-likelihood. This cost function is minimized using numerical optimization techniques, most notably gradient descent. Then please thoroughly understand these mathematical basis that must be comprehended in order to use the parameter tuning and model interpretability features for binary classification effectively. Logistic Regression Equation

$$P(y = 1|X) = \frac{1}{1 + e^{(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_N x_N)}} \tag{4}$$

Here, $P(y = 1|X)$ denotes the probability that the sequence X belongs to class 1. The coefficients $\beta_0, \beta_1, \beta_2, ...\beta_N$ are learned during the training process, and $x_1, x_2, ...x_N$ represent the features extracted from the sequence.

In the context of cyber attack classification, three algorithms within Ensemble Classification rely on these fundamental equations.

3.4. Parameter Tuning:

Hyperparameter tuning would definitely boost the performance of ensemble classification models. We also tuned the hyperparameters for each of the three classifiers used in this work; Gaussian Naive Bayes, Decision Tree and Logistic Regression. We also tuned the hyperparameters of ensemble classifier built using a voting technique.

- **Individual Classifier Hyperparameters:** In the domain of individual classifier hyperparameters, each classifier features a distinct array of parameters crucial in molding its behavior and predictive accuracy. Of particular significance is the *var_smoothing* hyperparameter in the Gaussian Naive Bayes classifier. This parameter plays a pivotal role by introducing a minute value to feature variances during computations, thus ensuring model stability, especially in scenarios where feature variances approach zero.

  Within the realm of Decision Tree classifiers, the *max_depth* hyperparameter stands out for its role in governing the depth or intricacy of the decision tree structure. This hyperparameter spans a spectrum of values, including *None*, allowing for unrestricted tree growth, and specific depth levels like 5 and 10, which impose constraints on the tree's complexity.

  Meanwhile, in the context of Logistic Regression classifiers, the hyperparameter *C* emerges as a cornerstone, representing the reciprocal of regularization strength. Modulating this parameter across various values, ranging from 0.1 to 10.0, exerts a discernible impact on the model's bias-variance trade-off, thereby influencing its overall predictive performance.
- **Hyperparameter Tuning via Grid Search:** The grid search method and cross-validation were both used throughout the hyperparameter tuning phase. In order to find the ideal collection of hyperparameters that optimizes model performance, grid search methodically investigates various combinations of hyperparameters. Cross-validation is crucial to this procedure because it enables us to evaluate each combination's performance over many data divisions, hence lowering the danger of overfitting.

  To improve hyperparameters for classifiers and ensembles, such as Logistic Regression, Decision Tree, Gaussian Naive Bayes, and voting techniques, a parameter grid called "param_grid" was developed. Ranges of values were investigated.

3.5. Evaluation of Ensemble Classifier:

With the training datasets ('x_train' and 'y_train'), the ensemble classifier 'eclf1' is trained to learn from the patterns and relationships in the dataset as a whole. A separate test dataset ('x_test') is used to evaluate the ensemble's performance after training. In this study, the following assessment measures were used:

Accuracy: A measure of overall predicted accuracy derived from the percentage of properly identified occurrences in the total. Recall: Recall is a statistic (derived from macro-averaging) that evaluates the ensemble's accuracy in identifying genuine positive events. Precision: Another macro-averaging metric that expresses how accurately the ensemble classified positive cases. F1 Score: A fair assessment of the ensemble's performance derived from the harmonic mean of recall and accuracy.

The study explores the effectiveness of an ensemble classifier, focusing on its accuracy in prediction and identification of positive examples. It highlights the importance of grid search and cross-validation optimization for enhanced performance.

4. Experimental Analysis:

We are seeing an ever-growing number of cyber-attacks and information systems getting more vulnerable. Researchers and cybersecurity people require robust threat detection and classification systems. This work explores an experimental evaluation of cyberattack taxonomy, performance metrics and data mean for algorithms. The results help to class cyberattacks and give tips for prevention/ mitigation, enhancement cybersecurity implementation as well as resilient assurance in the safety of digital environments.

For Cyber Attack Classification, the first step is renaming all column name of data in dataset. As a crucial step in this qutrition database venture, it aims to standardize the identification of every feature more intelligible by industry standards. This initial renaming procedure allows our experiments to be solid grounds so that the evaluation of classification algorithms becomes more digestible and effective. It sets the ground for a deep delve into cyberattack classification.

Row Index	Columns with Null Values
6796	Flow Bytes/s
14739	Flow Bytes/s
15047	Flow Bytes/s
209728	Flow Bytes/s

Table 2. Identifying Missing Values.

After naming every column in the dataset to the right name, we will check any missing or null values. During that, we managed to notice four null values (to be more specific in column 'Flow Bytes/s'). It was absent in four different rows, and their row indices were 14738, 6795,15046,and209727. But these null values in the 'Flow Bytes/s' column makes us aware that we have to deal with them during our data preprocessing and cleaning steps. After identifying the presence of null values in "Flow Bytes/s" column we used mean imputation to replace them with Column.mean. This ensures the accuracy and data completeness for further analytical processes. The problem with infinite numbers is that they can wreak havoc on data analysis, causing computation hiccups and numerical instability (and thus machine learning mistakes). This is how solution to this kind of problem is stylised in standard terms with replacing whatever infinite values (positive and negative infinity) we can come up within finate large value of the same number. This way, we make sure our data is always between 2 defined and appropriate values. It promotes consistency of data, integrity and stability to facilitate reliability and verification by impairing other jobs in the range from our studies Socionoemetrics for a variety of errors (especially when processing various types algorithms dataset).

We used mean imputation to replace any null values in the 'Flow Bytes/s' column with the column mean after finding them. With this method, the accuracy and completeness of the data are maintained for subsequent analysis.

Our data had infinite numbers, which can cause several problems for data analysis, such as computation hiccups, numerical instability, and machine learning mistakes. We took a standard approach to this problem, substituting the greatest finite value that could be found for infinite values—both positive and negative infinity. This method guarantees that our data stays inside a defined and suitable range. It makes our dataset more dependable and appropriate for a range of data processing jobs and algorithms by fostering data consistency, numerical stability, and the avoidance of mistakes in our studies.

The intermediate step in our data preparation process is known as Data Scaling. This is a common method used called min-max scaling which rescale our data to have values between 0 and1. The data does remain in the same proportion to each other but is just scaled so that these values are within a desired range We use min-max scaling to make data of our future studies more robust against different modeling and analysis methods allowing us the better interpretability and effectiveness out of those later. We need to preprocess our data in order for classification algorithms and other pattern recognition methodologies to work, but this is an essential process.

Feature	Probability (%)
Flow IAT Mean	49.9998
Flow IAT Std	50.0002
Flow IAT Max	50.0001
Flow IAT Min	49.9996
Fwd IAT Total	49.9994
Fwd IAT Mean	49.9997
Fwd IAT Std	50.0003
Fwd IAT Max	50.0004
Fwd IAT Min	50.0008
Bwd IAT Total	49.9989
Bwd IAT Max	50.0009
Bwd IAT Min	49.9996
Fwd Packets/s	49.9976
Bwd Packets/s	49.9992
Avg Bwd Segment Size	50.0004
Subflow Fwd Bytes	49.9997
Subflow Bwd Bytes	50.002
Init_Win_bytes_forward	50.0006
Init_Win_bytes_backward	49.9997
Active Mean	50.0003
Active Max	50.0002
Active Min	50.0003
Idle Std	50.0002
Idle Max	50.0001
Idle Min	49.9999

Table 3. Feature Probability Table

In the data science bubble, we select a list to consider important and prioritize feature selection Milica OrlovicUse Recursive Feature Elimination (RFE) function to get columns that are most significant,in this case in UNSW-NB15 dataset. How RFE applies a systematic evaluation to rank features After obtaining the best possible feature set, we calculate how probable each of chosen columns are. These probabilities provide great insights to important components by exposing the importance of different features in solving our problem into categorizing cyber attacks. Also, The UNSW-NB15 Dataset possesses inherent richness for researchers based on packet flow dynamics and this rich behavior

was captured in detailed features of the dataset that is vital to study Security. This includes the more fine-grained information at temporality, communication patterns as well payload characteristics such segments sizes and packet rates which can be exploited for better anomaly detection (e.g., Flow IAT Mean, Fwd/Bwd IAT)

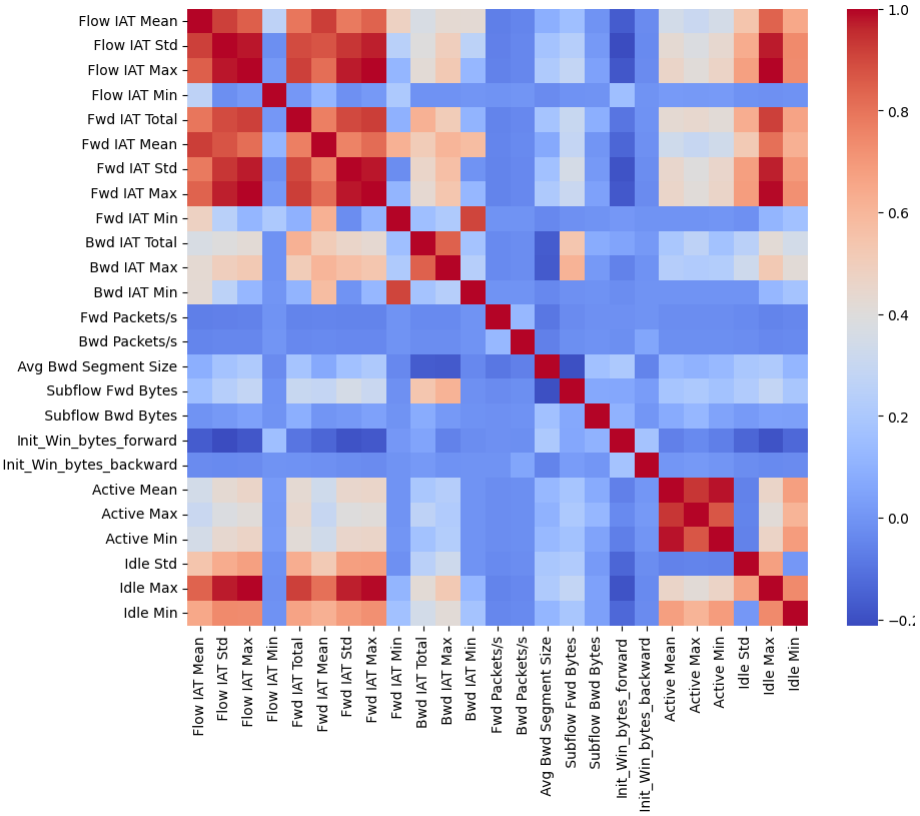

Figure 4. Heatmap.

After taking probablility analysis and feature selection, we move on to data visualization. We create a heatmap of the links and correlations between different characteristics that we have selected. The Heatmap allows us to focus on patterns, dependencies, and potential multicollinearity which may give a good sense of how features are interacting with each other. This plot helps in understanding the load on each dimensions, importance of feature clusters and where we need to dig more into. The heatmap improves the readability and understanding of our whole dataset, so it tells us how different characteristics relate to one another.

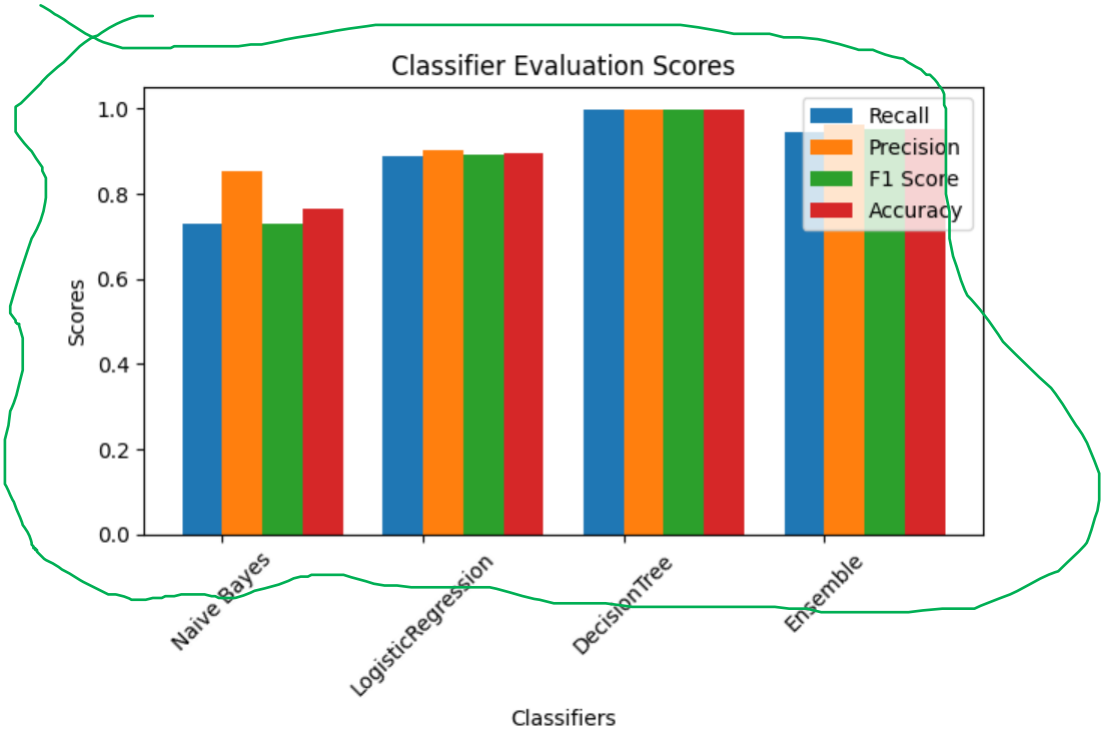

Figure 5. Individual Model Score.

As part of the current phase of our investigation, we have thoroughly tested three different categorization models: Decision Tree, Logistic Regression, Naive Bayes, and Ensemble model. The accuracy, recall, precision, and F1 score—four key performance metrics—have all been carefully evaluated concerning these models. With an accuracy of 0.7728, a recall of 0.7374, a precision of 0.8550, and an F1 score of 0.7388, Naive Bayes performed rather well. An additional increase in performance was shown by Logistic Regression, which had an F1 score of 0.8870, an accuracy of 0.8914, a recall of 0.8810, and a precision of 0.9003. Decision Tree performs remarkably well compared to the other models, showing near-perfect accuracy, recall, precision, and F1 score, all of which register at 0.9997. Ensemble model showing accuracy 0.9552, recall 0.9484, precision 0.9631, F1 score 0.9537. These measurements act as an essential reference point and offer priceless information about how well our models categorize cyberattacks. To assure the stability and dependability of our cyber attack categorization system, more studies and improvements are now underway, but it is crucial to stress that these results represent a major milestone in our research.

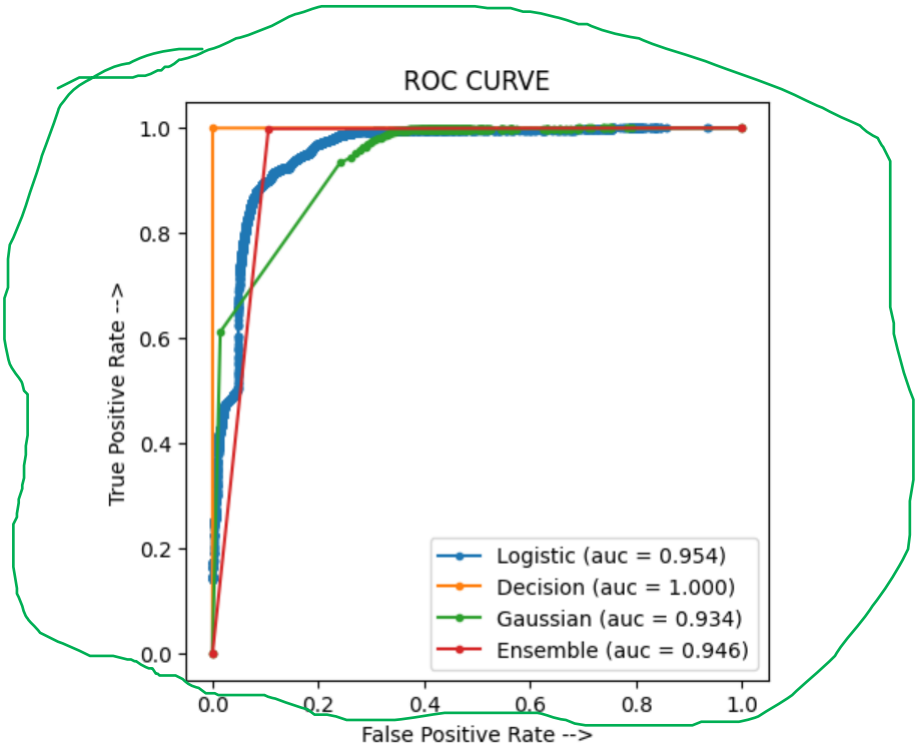

Figure 6. ROC Curve.

Using ROC (Receiver Operating Characteristic) curves, a fundamental tool for measuring the ability of a model to discriminate across classes, we examined whether or not our classification models have been effective. What Type or Kind of Information I Can Obtain from These Curves about Trade Off? The Logistic Regression model achieved a near perfect discriminating power based on the very high AUC score of 0.954 The Decision Tree model scores 1.000 AUC, no more to say! The performance of the Gaussian model on AUC dropped to an acceptable but low discriminatory ability (AUC: 0.934), however, satisfactory discrimination was retained for Ensemble having same score as ROC-AUC 0.946. The ROC curve results are an essential part in our evaluation because they tell us how good the above models will be able to identify cyberattacks and allow us know which model we should use for our classification task.

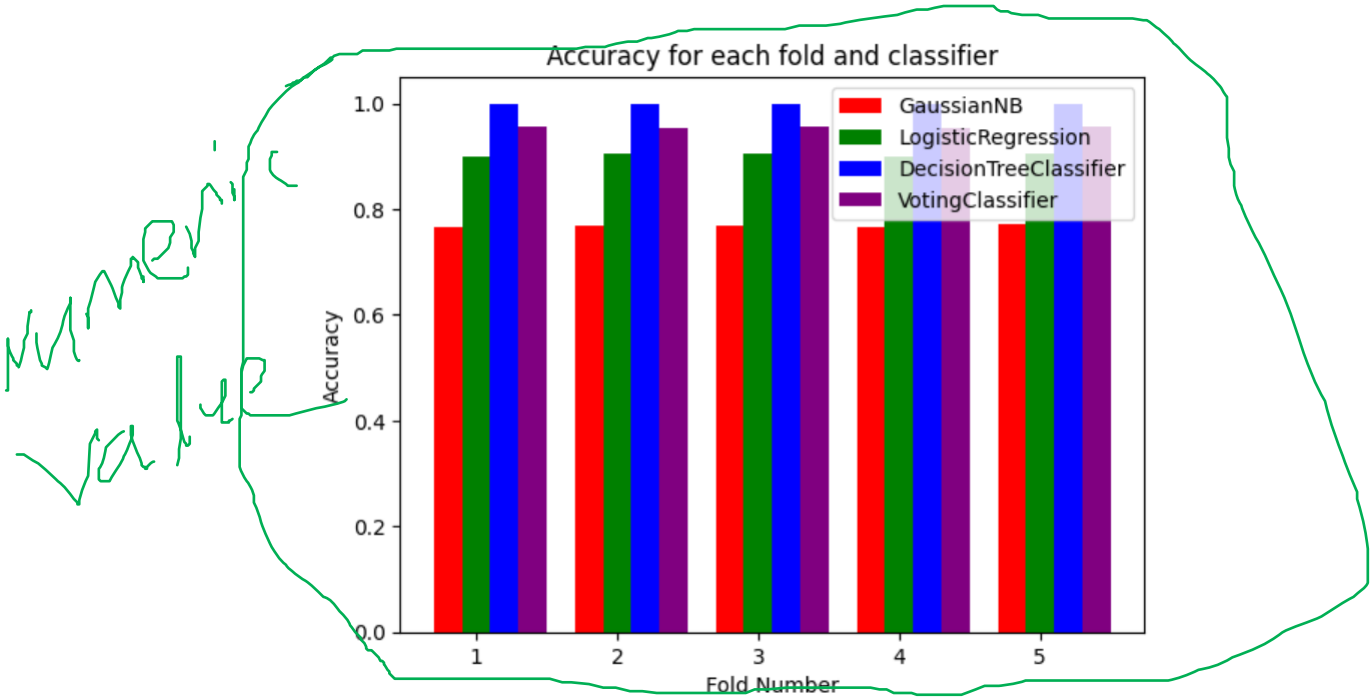

Figure 7. Cross-validation accuracy.

Robustness of classification models was evaluated by five-fold cross-validation, which consists on Gaussian Naive Bayes (GNB), Decision Tree Classifier and Logistic Regression as well as Ensemble model. The results deliver valuable insights into model performance under differing data subpopulations It performed the best and achieved a perfect accuracy score of 1.00 all folds for detecting cyber-attacks using Decision Tree Classifier. The accuracy of Gaussian Naive Bayes remained the same in every fold, 0.77 is an indication that it performing consistently across folds. On the other hand, The Logistic Regression rather performed well by giving accurate result in threshold over 0.86 to 0.91 which also underline respects its utility on this kind of assaults identification task In contrast, the Ensemble model demonstrated a spread of 0.95 to 0.96 Which is probably well for non-simulated data due it's above range which is closer in certainty than the M3s dealing with simulated natural language called NatLang (natural languages). These cross-validation results are an essential step in our model evaluation process, that gives us some assurance and provision approving the robustness of models.

4.1. Key Findings

The key findings of the proposed methodology are:

- According to how well they perform, the models are placed in the following order Decision Tree>Logistic Regression>Naive Bayes.
- The Decision Tree achieves superior accuracy at 0.9997, surpassing both Logistic Regression (0.8914) and Naive Bayes (0.7728).
- Subsequently, an ensemble model exhibits 0.9552 accuracy.
- In cross-validation, logistic regression varied from 0.86 to 0.91, Gaussian Naive Bayes stayed at a constant 0.77 in every fold, while decision trees continuously achieved an impeccable 1.00 accuracy in every fold.
- Achieving an accuracy range of 0.95 to 0.96, the ensemble model is applied in the final step.

5. Conclusion

Machine learning algorithms demonstrate the potential to strengthen cybersecurity defenses, especially when applied to the UNSW-NB15 datasets. With 0.9997 accuracy, Decision Trees lead and demonstrate excellent pattern recognition. At 0.8914, logistic regression comes next and provides trustworthy information on cyber threats. At 0.7728, Gaussian Naive Bayes shows usefulness despite its simplicity. Key variables are found via Recursive Feature Elimination, which improves model performance. With an accuracy rate of 0.9552, the ensemble classifier highlights the importance of integrating several approaches. Proactive cybersecurity is encouraged by techniques like hard voting and cross-validation, which improve system accuracy and efficiency even more. These findings represent a major advancement in the development of robust systems that can respond to changing cyber threats in the complex digital environment of today.

References

1. Shen, Y., Zheng, K., Yang, Y., Liu, S. and Huang, M., 2022. CBA-CLSVE: A Class-Level Soft-Voting Ensemble Based on the Chaos Bat Algorithm for Intrusion Detection. Applied Sciences, 12(21), p.11298.
2. Louk, M.H.L. and Tama, B.A., 2022. PSO-Driven Feature Selection and Hybrid Ensemble for Network Anomaly Detection. Big Data and Cognitive Computing, 6(4), p.137.
3. Oliveira, N., Praça, I., Maia, E. and Sousa, O., 2021. Intelligent cyber attack detection and classification for network-based intrusion detection systems. Applied Sciences, 11(4), p.1674.
4. Yousefnezhad, M., Hamidzadeh, J. and Aliannejadi, M., 2021. Ensemble classification for intrusion detection via feature extraction based on deep Learning. Soft Computing, 25(20), pp.12667-12683.
5. Jaw, E. and Wang, X., 2022. A novel hybrid-based approach of snort automatic rule generator and security event correlation (SARG-SEC). PeerJ Computer Science, 8, p.e900.
6. Al-Daweri, M.S., Abdullah, S. and Ariffin, K.A.Z., 2021. A homogeneous ensemble-based dynamic artificial neural network for solving the intrusion detection problem. International Journal of Critical Infrastructure Protection, 34, p.100449.
7. Kumar, P., Gupta, G.P. and Tripathi, R., 2021. A distributed ensemble design-based intrusion detection system using fog computing to protect the Internet of Things networks.Journal of ambient intelligence and humanized Computing, 12, pp.9555-9572.
8. Almogren, A.S., 2020. Intrusion detection in Edge-of-Things computing. Journal of Parallel and Distributed Computing, 137, pp.259-265.
9. Mohmand, M.I., Hussain, H., Khan, A.A., Ullah, U., Zakarya, M., Ahmed, A., Raza, M., Rahman, I.U. and Haleem, M., 2022. A machine learning-based classification and prediction technique for DDoS attacks. IEEE Access, 10, pp.21443-21454.
10. Jiang, K., Wang, W., Wang, A. and Wu, H., 2020. Network intrusion detection combined hybrid sampling with the deep hierarchical network.IEEE Access, 8, pp.32464-32476.
11. Larriva-Novo, X., Villagrá, V.A., Vega-Barbas, M., Rivera, D. and Sanz Rodrigo, M., 2021. An IoT-focused intrusion detection system is an approach based on preprocessing characterization for cybersecurity datasets. Sensors, 21(2), p.656.
12. Kumar, V., Sinha, D., Das, A.K., Pandey, S.C. and Goswami, R.T., 2020. An integrated rule-based intrusion detection system: analysis on UNSW-NB15 data set and the real-time online dataset. Cluster Computing, 23, pp.1397-1418.
13. Kasongo, S.M. and Sun, Y., 2020. Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. Journal of Big Data, 7, pp.1-20.
14. Zhang, J., Li, F. and Ye, F., 2020, June. An ensemble-based network intrusion detection scheme with Bayesian deep learning. In ICC 2020-2020 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.
15. Alazzam, H., Sharieh, A. and Sabri, K.E., 2020. A feature selection algorithm for intrusion detection system based on pigeon-inspired optimizer. Expert systems with applications, 148, p.113249.
16. Louk, M.H.L. and Tama, B.A., 2023. Dual-IDS: A bagging-based gradient boosting decision tree model for network anomaly intrusion detection system. Expert Systems with Applications, 213, p.119030.

17. Gao, J., Chai, S., Zhang, B. and Xia, Y., 2019. Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis. Energies, 12(7), p.1223.
18. Popoola, S.I., Adebisi, B., Ande, R., Hammoudeh, M., Anoh, K. and Atayero, A.A., 2021. smote-drnn: A deep learning algorithm for botnet detection in the internet-of-things networks. Sensors, 21(9), p.2985.
19. Alazzam, H., Sharieh, A. and Sabri, K.E., 2020. A feature selection algorithm for intrusion detection system based on pigeon-inspired optimizer. Expert systems with applications, 148, p.113249.
20. Almomani, O., 2020. A feature selection model for network intrusion detection system based on PSO, GWO, FFA, and GA algorithms. Symmetry, 12(6), p.1046.
21. Moustafa, N. and Slay, J., 2015, November. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In 2015 military communications and information systems conference (MilCIS) (pp. 1-6). IEEE.
22. Kumar, V. et al. 2020. An integrated rule-based intrusion detection system: analysis on UNSW-NB15 data set and the real-time online dataset. Cluster Computing, 23, pp.1397-1418.

© 2024 by the author. Submitted to *Journal Not Specified* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).