Enhancing network security through machine learning: A study on intrusion detection system using supervised algorithms

Zi Huang[1,4,†], Zhenmin Li[2,†] and Jiaming Zhang[3,†]

[1]College of Engineering, College of Science, Virginia Polytechnic Institute and State University, Blacksburg, 24060, United States
[2]Computer Science Department, Campbellsville University, Campbellsville, 42718, US
[3]Faculty of Science, Hebei Zhengzhong high school, Hebei, 050031, China

[4]zih19@vt.edu
[†]All authors' contributions are consistent

Abstract. The topic of Intrusion Detection System (IDS) has become a highly debated issue in cybersecurity, generating intense discussions among experts in the field. IDS can be broadly categorized into two types: signature-based and anomaly-based. Signature-based IDS employ a collection of known network attacks to identify the precise attack the network is experiencing, while anomaly-based IDS employ machine learning models to detect anomalies present in the network traffic that could indicate a potential attack. In this study, we concentrate on anomaly-based IDS, evaluating the effectiveness of three supervised learning algorithms - Decision Tree (DT), Naive Bayes (NB), and K-Nearest Neighbor (KNN) - to determine the most suitable algorithm for each dataset based on its source. We conducted tests to evaluate each algorithm's performance and choose the best one for each dataset. Our findings show that anomaly-based IDS is highly effective in enhancing network security, providing valuable insights for organizations looking to improve their security measures.

Keywords: Network Intrusion Detection, Decision Tree, Naive Bayes, K-Nearest Neighbor.

1. Introduction

With the rapid development of the Internet, network security has become a crucial issue affecting daily life, particularly for people who lack knowledge on how to use Internet properly. However, current resolutions still have some shortcomings to be addressed. Intrusion Detection System (IDS) is one solution that can effectively secure sensitive information such as passwords and personal data. Nevertheless, IDS is often viewed as a challenge for those with expertise in networking. The process begins with data analysis and visualization, which reveals how network attacks are distributed. The article introduces two data preprocessing techniques, namely StandardScaler and LabelEncoder, where StandardScaler is used for columns of numeric types such as int and float, and LabelEncoder for columns of object type. Finally, the entire dataset is divided into four specific groups according to the "Dataset" column, utilizing three machine learning algorithms, namely Decision Tree (DT), Naive Bayes (NB), and K-Nearest Neighbor (KNN), to showcase their performance and evaluate which machine learning models are suitable for each dataset.

2. Relevant theory and work

Table 1. A 46 features in total in NF-UQ-NIDS-v2 are recorded and organized as a table.

Feature	Explanation
IPV4_SRC_ADDR	IPV4 Source Address
IPV4_SRC_PORT	IPV4 Source Port Number
IPV4_DST_ADDR	IPV4 Destination Address
IPV4_DST_PORT	IPV4 Destination Port Number
PROTOCOL	IP Protocol Identifier Type
L7_PROTO	Layer 7 Protocol (Numeric)
IN_BYTES	Incoming Number of Bytes
IN_PKTS	Incoming Number of Packets
OUT_BYTES	Outgoing Number of Bytes
OUT_PKTS	Outgoing Number of Packets
TCP_FLAGS	Cumulative of All TCP Flags
CLIENT_TCP_FLAGS	Cumulative of All Client TCP Flags
SERVER_TCP_FLAGS	Cumulative of All Server TCP Flags
FLOW_DURATION_MILLISECONDS	Flow Duration in Milliseconds
DURATION_IN	Client to Server Stream Duration(msec)
DURATION_OUT	Server to Client Stream Duration(msec)
MIN_TTL	Minimum Flow TTL
MAX_TTL	Maximum Flow TTL
LONGEST_FLOW_PKT	Longest Packet(bytes) of The Flow
SHORTEST_FLOW_PKT	Shortest Packet(bytes) of The Flow
MIN_IP_PKT_LEN	Length of The Smallest Flow IP Packet Observed
MAX_IP_PKT_LEN	Length of The Largest Flow IP Packet Observed
SRC_TO_DST_SECOND_BYTES	Source to Destination (bytes/sec)
DST_TO_SRC_SECOND_BYTES	Destination to Source(bytes/sec)
RETRANSMITTED_IN_BYTES	Number of Retransmitted TCP Flow Bytes (Src -> Dst)
RETRANSMITTED_IN_PKTS	Number of Retransmitted TCP Flow Packets (Src-> Dst)
RETRANSMITTED_OUT_BYTES	Number of Retransmitted TCP Flow Bytes (Dst -> Src)
RETRANSMITTED_OUT_PKTS	Number of Retransmitted TCP Flow Packets (Dst-> Src)
SRC_TO_DST_AVG_THROUGHPUT	Source to Destination Average Throughput
DST_TO_SRC_AVG_THROUGHPUT	Destination to Source Average Throughput
NUM_PKTS_UP_TO_128_BYTES	Packets Whose IP Size is Less Than or Equal to 128 Bytes
NUM_PKTS_128_TO_256_BYTES	Packets Whose IP Size is Greater Than 128 Bytes but Less Than or Equal to 256 Bytes
NUM_PKTS_256_TO_512_BYTES	Packets Whose IP Size is Greater Than 256 Bytes but Less Than or Equal to 512 Bytes
NUM_PKTS_512_TO_1024_BYTES	Packets Whose IP Size is Greater Than 512 Bytes but Less Than or Equal to 1024 Bytes
NUM_PKTS_1024_TO_1514_BYTES	Packets Whose IP Size is Greater Than 1024 Bytes but Less Than or Equal to 1514 Bytes

Table 1. (continued).

TCP_WIN_MAX_IN	The Maximum TCP Window (Src -> Dst)
TCP_WIN_MAX_OUT	The Maximum TCP Window (Dst -> Src)
ICMP_TYPE	ICMP Type * 256 + ICMP Code
ICMP_IPV4_TYPE	ICMP Type
DNS_QUERY_ID	DNS Query Transaction ID
DNS_QUERY_TYPE	DNS Query Type
DNS_TTL_ANSWER	TTL Of the First A Record
FTP_COMMAND_RET_CODE	FTP Client Command Return Code
LABEL	Label Determined To Be Attacked
ATTACK	Attack Types Encountered by Network Data
DATASET	The Source of Network Data

The capability of Intrusion Detection Systems (IDS) technology to discern threats in real-time is one of its main benefits taken into account [1]. This fact allows security personnel to respond quickly to potential threats and prevent them from causing significant damage [2]. In addition, the configuration of NIDS technology can generate alerts and reports that provide detailed information about network traffic and potential threats [1]. This information helps spot shifts in the network, further strengthening and consolidating its security [3].

While IDS is an effective tool for detecting and blocking network attacks, it is critical to be aware that it is not a flawless patch [2]. IDS has good potential in conjunction with other security measures, such as firewalls, antivirus software, and employee training, to provide comprehensive network security [3]. At the same time, it must be developed and maintained properly to ensure the exceptional efficacy of detecting and preventing attacks [1].

To sum up, IDS is a superb tool for enhancing network security and preventing network attacks [2]. It is widely used in industries that either deal with sensitive or confidential data or monitor a wide range of network traffic in real-time [3]. Even though IDS is generally regarded to be superior for network attacks, it must be adopted in juxtaposition with other security measures and defined properly to guarantee its reliability [1]. As the threat landscape continues to evolve, IDS will be essential for defending networks and sensitive data they hold [2].

3. Application and methodology

Soon after the brief description of IDS and its accomplishments, several methodologies that will be discussed in this paper — including data preprocessing techniques and machine learning approaches— can be analyzed thoroughly to perceive any distinctions between the content of this paper and that written by other people in this field.

3.1. Dataset

Compared to network datasets that are frequently used like KDD Cup 99, the dataset selected is called NF-UQ-NIDS-v2 [4], a comprehensive dataset unifying four particular Netflow-based datasets. These four datasets, coupled with the addition of netflows, are all derived from traditional network data that have been formatted into pcap files. While taking a look at this combined dataset, a total of 46 features should be considered, where 4 of them are continuous and the rest being discrete. As shown in table 1.

However, there are two unique circumstances that should be strongly emphasized. One of them has to do with the amount of network data to be analyzed. As suggested in Section 3.1, the dataset selected is a cumulative version consisting of multiple network sources, thus consuming as much memory as possible. Similar to the speculation above, the combined dataset occupies 12GB of computer memory that is quite hard to imagine. To alleviate this complex problem, only 20,000 data out of a 12GB data file can be sampled, allowing the datafile to be used smoothly in the ensuing experiment.

In addition to extracting a limited number of network data for detailed analysis, the network attacks in each sub-dataset segmented by the column Dataset in NF-UQ-NIDS-v2 must be organized into general categories of network attacks common to people with an area of expertise in cybersecurity. According to a series of data science libraries and packages implemented in Python [5], all network attacks are partitioned into 11 groups as follows:

Benign, Dos, DDoS, Phishing, XSS, Password, SQL Injection, Botnet, Brute Force, Zero-Day Exploit, Malware.

3.2. Data preprocessing

Once the framework of the cumulative dataset is overviewed, all features connected to it must be standardized so that the process of fitting machine learning models can be conducted successfully without encountering any bizarre syntax errors. To explain any approaches used to standardize data, both StandardScaler and LabelEncoder are two crucial tactics that should be examined in detail.

3.2.1. StandardScaler. StandardScaler is one type of standardization technique wherein the numerical data of type integers and floating numbers can be scaled by transforming the statistical distributions of data into a format with mean 0 and variance 1. At the same time, the StandardScaler under the context of machine learning can serve as a common requirement for determining the behavior of data. For instance, if any one of features within the dataset has a variance that is greater than others in terms of orders of magnitude, then this designated feature will probably dominate the objective function, consequently leading to the machine learning estimator being unable to learn from other features correctly as expected [6].

$$z = \frac{x-\mu}{\sigma^2} \sim N(0,1) \tag{1}$$

3.2.2. LabelEncoder. Contrary to StandardScaler for dealing with columns of numerical type, LabelEncoder is specialized in columns that are marked as an object type. To comprehend how it works, the columns with an object type are targeted, ranging from 0 to n.classes - 1 based on the number of groups each column encompasses [6]. For example, the column named Species in Iris Dataset [6] has three flower species that are available to be divided into three categories: Setosa, Virginicia, and Versicolor. Each flower species can be written as a number from 0 to 2, where 0 refers to Setosa, 1 to Virginicia, and 2 to Versicolor. Thus, On the basis of this explanation, three columns of object type in NF-UQ-NIDS-v2, which are IPV4_SRC_ADD, IPV4_DST_ADD, and Dataset, should be converted into categorical integers, combining columns implemented by StandardScaler to filter features that are somewhat redundant and negligible.

3.3. Feature selection

After all network features have been effectively standardized under the application of StandardScaler and LabelEncoder, each network characteristic should be strictly assessed so that those assumed to be significant are potentially considered in advance. To take the process with regard to feature selection into account, two important means named Random Forest Classification(RFC) and Recursive Feature Elimination(RFE) need to be brought up so that each feature in the network dataset is able to be observed in an incisive manner without any coincidence that may adversely affect the performances of machine learning methods.

3.3.1. Random forest classifier (RFC). The feature selection algorithm defaults to using RFC, a traditional scheme that is often used in data analysis. It is mainly made up of as many decision trees as possible to improve accuracy and control overfitting [7, 8]. To reach these two intentions, each feature in the network dataset can be processed and visualized in a descending order, and features plotted at the left side of the graph are reviewed as factors contributing to the appearance of efficacious machine learning models most. As shown in Figure 1.

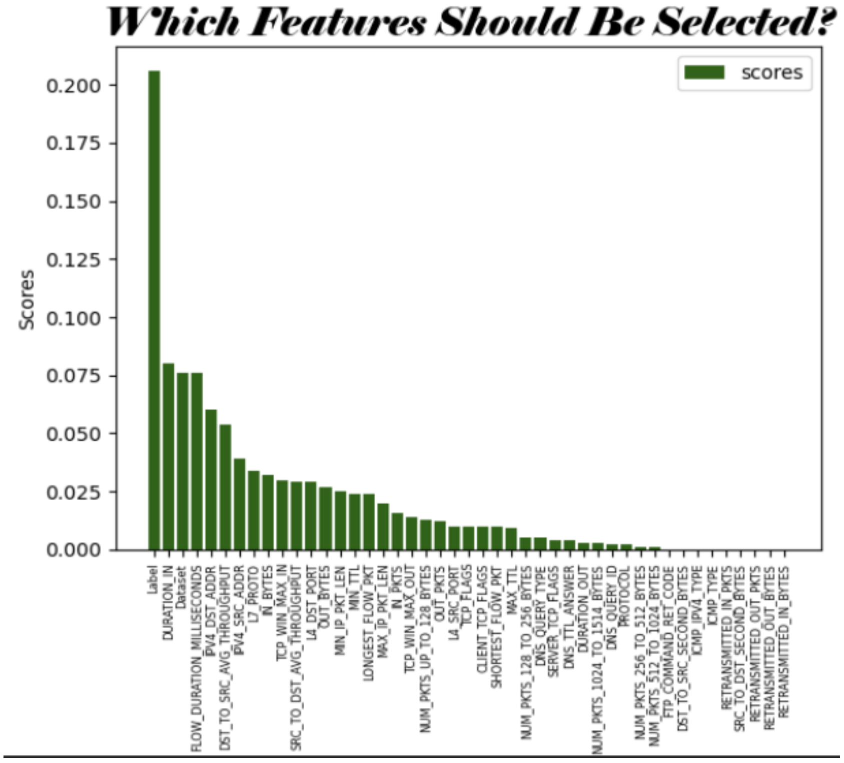


Figure 1. Describes the overall framework of RFC in NF-UQ-NIDS-v2[11].

The importance of each feature is sorted in a sequence from largest to smallest while looking at the graph from left to right.

3.3.2. Recursive feature elimination (RFE). Another algorithm whose functionality is identical to RFC is called Recursive Feature Elimination(RFE). Although RFE can be deployed as one method for selecting features that are useful for running machine learning models, it can better be generalized as an ingenious method in not only choosing features of great importance but also reevaluating how exact and accurate RFC is. To explain the logic behind RFE, every column in the dataset needs to be iterated at least one time. The feature that is proved to be the least important is deleted after each round of RFE, and the procedure should then be recurred until the features remaining are equal to the boundary case that was established by RFE before [9, 10]. After gaining a fundamental understanding of RFE and contrasting it with RFC, the final ten features that are satisfied by machine learning models, along with Figure 1, are bulleted as follows: IPV4_SRC_ADDR,IPV4_DST_ADDR, L7_PROTO, IN_BYTES, OUT_BYTES, FLOW_DURATION_MILLISECONDS, DURATION_IN, MAX_IP_PKT_LEN, DST_TO_SRC_AVG_THROUGHPUT, LABEL

3.4. Machine learning algorithms

Last but not least, it is of great importance in choosing machine learning algorithms to adequately judge network attacks. While thinking of machine learning algorithms studied extensively, two intents associated with them need to be categorized: regression and classification. Regression is a concept in Statistics supposed to explain how one or more explanatory variables are pertinent to the response variable, whereas classification, on the other hand, attempts to speculate and represent the category attached to an object or item based upon the corresponding distinct attributes the group owns. Because the final outcome of the experiment is to deduce network attacks for each network data, three widely used classification algorithms that are conducive to object detection may be deliberated.

3.4.1. Decision tree (DT). Decision Tree (DT) is a method frequently used for both regression and classification applications. It is a straightforward yet effective model that splits the data into subsets according to its specific attributes measured. Each subset is divided into smaller subsets in a recursive manner until a stopping condition is reached, creating a tree-like structure that allows for prediction. One of the benefits regarding DT is that it is simple to visualize and follow, making it a cogent tool for acquiring knowledge of how a model makes decisions. Additionally, DT can handle data that are both numerical and categorical, becoming an omnipotent option in various applications. However, one of its potential downsides is its tendency to overfit the data, which can lead to poor generalization and performance on new data.

The interconnectedness between DT and IDS is understood by taking a classification model into account to distinguish multiple groups of network attacks. The model can be trained on a dataset containing many network traffic data, where each instance corresponds to a network connection and all of its features that go along with it (such as flow duration, throughput, bytes, etc.). Network attacks like DoS/DDoS, Port Scan, or Malware are the label for this dataset.

When implementing DT, the highest priority is given to maximizing information gain, selecting the node/attribute with the highest information gain that should be splitted first.

$$\text{Information Gain} = \text{Entropy (S)} - [\text{(Weighted Avg)} * \text{Entropy (each feature)}]$$

$$\text{As } Entropy(s) = -p_1 log_2(p_1) - p_2 log_2(p_2) \tag{2}$$

Where p_1 is the proportion that the examples belong to class 1 in "S" and p_2 to class 2. The base 2 beneath the logarithmic notation can help measure the entropy.

The entropy of a set is maximum when the classes are evenly distributed and minimum when all examples are gathered at the same class. When building a tree similar to DT, the ultimate goal is to minimize the entropy of the subsets that come from a split, thus computing the difference between the entropy of the parent node and the weighted average of the entropies of the child nodes.

3.4.2. K-Nearest Neighbor (KNN). K-Nearest Neighbor (KNN) is one nonparametric supervised algorithm first proposed by Fix and Hodges [12] in 1951 to predict the group the point targeted belongs to without knowing the prior distribution of data. More specifically, given points that are mainly distributed in a fixed number of groups, the motivation behind this algorithm is to take advantage of both k, the number of neighbors observed, and the equation of Euclidean Distance to output what the data point is supposed to position within whatever group. The visualization of KNN, as well as its steps and execution, can be demonstrated below [13, 14].

Step 1: know both x and y coordinates of the point investigated

Step 2: choose k to be scrutinized

Step 3: Use Euclidean Distance to determine how far it is between every point in the xy- plane and the point under inspection

Step 4: order their distances in an ascending order and select the first fixed number of points corresponding to the value of k

Step 5: utilize the basic law of probability in Statistics to verify the group the point targeted actually belongs to.

While the KNN algorithm is outlined above, the equation to calculate Euclidean Distance mentioned in Step 3 has to be written in case of someone forgetting its mathematical mechanism.

$$Euclidean\ Distance: \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}\ , given\ A \in (x_1, y_1)\ B\ \in (x_2, y_2) \tag{3}$$

Meanwhile, the associativity between KNN and Network Intrusion Detection(NID) is such an essential topic that should be elaborated. As stated in Table 1, many features, including port numbers, average throughput, and IP addresses, play a considerable role in determining whether the network is currently experiencing an attack. When counting the overall number of features the dataset contains, a total of 44 columns excluding Dataset must be taken into consideration, where each property represents a dimension drawn on the plane. In order to display each network data in a 2D space only, Principal Component Analysis(PCA), a technique involving multiple variables in which a dataset higher than 2 dimensions is transformed into a two-dimensional dataset merely by extracting its important information and representing them as a series of directional orthogonal variables named principal components [15], is used, thus depicting any potential correlation among ten features in the network.

3.4.3. Naive bayes (NB). The final algorithm, though it is not as advanced as the two algorithms mentioned previously, is called Naive Bayes(NB). The Bayes Theorem adaptability to a wide range of classification tasks, including spam filtering and text categorization [16], makes Naive Bayes to be connected to the law of probability.

$$P(c|p) = \frac{P(p|c) * p(c)}{P(p)} \tag{4}$$

where the conditional probability P(c|p), denoted as the probability that the classifier is assigned given its properties, depends upon the classifier chosen and the properties the classifier has.

However, one scenario that should be noted carefully is that the classifier does have more than one feature. In order to confirm this statement, the simplified version of Bayes Theorem can further be derived in terms of $P(c|p_1, p_2, \ldots\ldots, p_n)$ instead of $P(c|p)$ as an individual element only [17, 18]. The process above is clarified in Equation 4.

$$P(c|p_1, p_2, \ldots, p_n) = \frac{P(p_1,p_2,\ldots,p_n|c)*P(c)}{P(p_1,p_2,\ldots,p_n)} = \frac{P(p_1|c) * P(p_2|c) * \ldots * P(p_n|c) * P(c)}{P(p_1) * P(p_2) * \ldots * P(p_n)} = \frac{P(c) * \prod_{i=1}^{n} P(p_i|c)}{P(p_1) * P(p_2) * \ldots * P(p_n)} \tag{5}$$

where $P(c|p_1, p_2, \ldots, p_n)$ is directly proportional to the specific classifier $P(c)$ and each property $\prod_{i=1}^{n} P(p_i|c)$ owned by the classifier.

To interpret how Bayes Theorem is capable of predicting network attacks in NF-UQ-NIDS-v2, both Equation 4 and Table 2 can be made use of together to quickly cope with this problem. To further delve into the content of this algorithm, the numerical analysis is corroborated in Equation 5, In this case, the network attack opted for in this case is said to be XSS.

$$P(XSS|ipv4\ src\ addr, ipv4\ dst\ addr, \ldots, label) = \frac{P(ipv4\ src\ addr|\ XSS)}{P(ipv4\ src\ addr)} * \frac{P(ipv4\ dst\ addr|XSS)}{P(ipv4\ dst\ addr)} * \ldots * \frac{P(label|XSS)}{P(label)} \tag{6}$$

4. Experiment and results

Once both data preprocessing techniques and machine learning algorithms have been examined in section III, conducting an experiment on the dataset from section 3.1 and analyzing its result are two proper maneuvers to help evaluate which machine learning algorithms achieve the highest efficiency at detecting the network abnormalities. However, instead of using *NF-UQ-NIDS-v2*, the dataset that is already generalized briefly, to try this experiment, a delimiter is instantiated with respect to four data sources on the column *Dataset*, searching for one of three machine learning algorithms that would work with each of them.

4.1. Machine learning model initialization

Before checking the performance of each machine learning classifier applied to four data sources above, each classifier's parameter value(s) may need to be identified precisely so that the test error determined later can be as low as possible to reduce the possibility of being overfitted. No parameters should be included within Naive Bayes due to the Gaussian distribution to be followed, but both KNN and DT differ in that the parameters are anticipated to be imputed to clearly improve the accuracy of both model training and testing.

4.2. Performance evaluation

To expound how the dataset is performed, four major indicators plus three machine learning algorithms in Section 3.4 are necessary to be illuminated, which are cross-validation, accuracy, confusion matrix, and classification report.

4.2.1. Cross-Validation

Cross-validation is a resampling method used to assess machine learning models given the limited amount of data [19]. When defined with the parameter k, the amount of folds for segregation [19], data that is ready to be executed in the dataset under cross-validation is divided into k segments. Under an iterative procedure ranging from 0 to k - 1, where 0 refers to the first segment and k - 1 to the last segment in the dataset, data with a segment specified under a loop are deemed as the test set, whereby data in the remaining segments are accumulated into the train set [19, 20]. Finally, the result obtained from cross-validation is fitted on the train set, making a comparison with the output of a test set and reporting its accuracy. As shown in Figure 2.

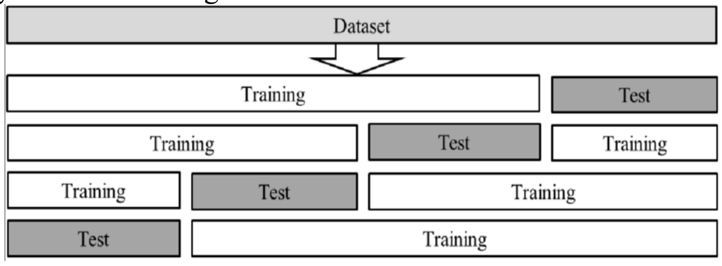


Figure 2. Explains how cross-fold validation is operated. The training set is viewed as a part for experimentation, and the test set is intended to prove how authentic the values in the train set are outputted [19].

4.2.2. Confusion matrix

On the contrary of illustrating the principle behind cross-validation individually, combining accuracy, confusion matrix, and classification report could be far more optimistic because of their shared similarities. To figure out which concept should be delivered first, the confusion matrix can be a starting point that lays a solid foundation for both accuracy and the classification report. A confusion matrix is indeed an N*N matrix to check whether the performance of a machine learning model is productive [21]. While concentrating on the letter N, it merely means the number of groups available for investigation. For instance, if there are 2 groups in the entire dataset, then the dimension of the matrix will be 2 * 2, where the diagonal line of a matrix is a category predicted correctly in response to the category that was actually observed. These entries along the diagonal line are termed as True Positive(TP) and True Negative(TN) respectively. As shown in Figure 3.

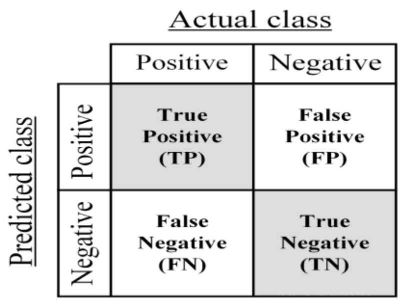


Figure 3. Exhibits the structure of a confusion matrix. A confusion matrix hasfour possibilities demonstrated above [21].

4.2.3. Accuracy

After setting up the confusion matrix, the second concept called accuracy is able to be conveyed succinctly. Accuracy is defined as a probability for which how many samples out of the total number of samples are indicated to be correct [20-22], whereby the format can be either a percent or decimal based upon the interest of mathematicians or machine learning scientists. Therefore, accuracy under the confusion matrix drawn in Figure 3 can be described as samples that are allocated in True Positive(TP) and True Negative(TN).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

Last but not least, the classification report is a key metric to extend the knowledge obtained from both the confusion matrix and accuracy. Besides the term accuracy that mainly deals with any items in TP and TN, other three criteria in this metric, which are called precision, recall, and F1-Score, are also recorded to process choices not only in TP and TN but also FP and FN in Equation 6 as well. Precision is the fraction of samples retrieved that are correctly identified among all retrieved samples [21]. Recall, on the other hand, is referred to as a circumstance in which the outcome of each label is congruent given different conditions to achieve the result [9, 21-22]. When both precision and recall are known, the final indicator F1-score is captured naturally because it is directly proportional to both precision and recall [9].

$$\begin{gathered} Precision = \frac{TP}{TP + FP} \\ Recall = \frac{TP}{TP + FN} \\ F1\,Score = \frac{2 * precision * recall}{precision + recall} \end{gathered} \tag{8}$$

4.3. Experiment setup

As long as all four indicators in section 4.2 are illustrated thoroughly, the experiment is able to be commenced so that each dataset partitioned according to its source of data can attain its own corresponding confusion matrix, classification report, and accuracy. To begin the experiment, the four sources of network data must be justified as follows.

4.3.1. NF-BoT-IoT-v2. The network data NF-BoT-IoT formatted into pcap come from a familiar dataset NF-BoT-IoT-v2 that is integrated with Internet of Things(IoT) and Robots(BoT) . However, the main difference between NF-BoT-IoT-v2 and NF-BoT-IoT is that many netflows are wrapped into NF-BoT-IoT-v2, facilitating the exploration of the amount of netflow data that are both currently under attack and remaining stable, where 37,628,460 out of 37,763,497 (99.6%) data flows are attack samples and the rest of 135,037 data flows(0.4%) are samples claimed to be benign [5].

While considering the parameters in each machine learning classifier, the number of neighbors in KNN is 7 by using an explicit algorithm that implements the logic connected with plotting each number and its corresponding error in a list with consecutive numbers. As for the maximum height, as well as the minimum number of sample splits in DT, they are 10 and 5 respectively.

4.3.2. NF-CSE-CIC-IDS2018-v2. The original version of NF-CSE-CIC-2018-v2 is the dataset that is specialized in anomaly detection to accommodate shortcomings faced by networks such as privacy protection and intensive anonymization that are unable to reflect the current trend or pattern of network behaviors [23]. There are 7 types of network attacks that are analyzed with the addition of netflows, whereby most of the network data do not encounter any fatal attacks.

Similar to the process demonstrated in the second paragraph of 4.3.1, the parameters in each machine learning classifier should be declared in a practical way. The number of neighbors in KNN, while utilizing the plotting techniques to look for the test error in each neighbor, is 5. Both the maximum depth and the minimum number of sample splits in DT are set as 10 and 5 either.

4.3.3. NF-UNSW-NB15-v2. A desirable dataset that wishes to be discussed is NF-UNSW-NB15-v2. According to the name of the dataset implied, the network data all came from NF-UNSW-NB15, a dataset created by University of New South Wales(UNSW) in Sydney to generate networks that are a hybrid of real modern normal activities and synthetic contemporary attack behaviors [24], capturing those that are in danger. The attack types within this network can be divided into 9 subcategories, and only 4% of them are attack samples.

To determine what parameter values are written for each machine learning classifier, the number of neighbors typed within this dataset is 2. On the other hand, the parameter values can still be written as 10 for the maximum depth of the tree and 5 for the minimum number of sample splits.

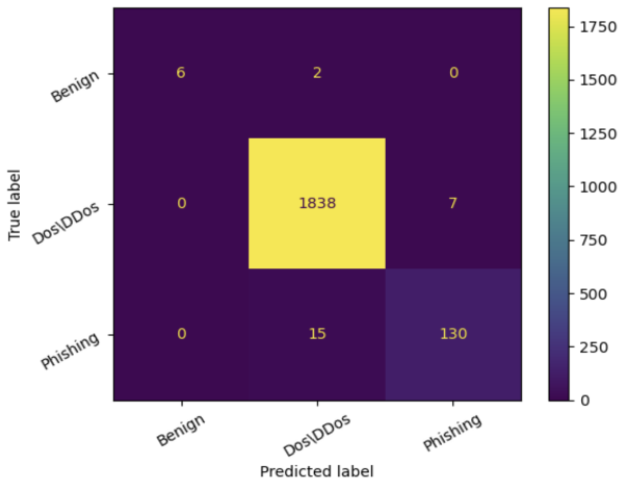
4.3.4. NF-ToN-IoT-v2. As the only dataset that is publicized in a pcap format, the dataset NF-ToN-IoT, along with netflow-based records, takes advantage of Internet of Things(IoT) and industrial technologies to test how superior the functionality of different cybersecurity applications are in three areas: Artificial Intelligence(AI), Machine Learning(ML), and Deep Learning(DL). The distribution of network data that are both benign and detrimental, compared to three datasets above, is somewhat even, where 64% are under attack and 36% are harmless in approximation.

At the same time, the parameters associated with each machine learning classifier in NF-ToN-IoT-v2 are guaranteed. The method in figuring out what parameter values are chosen is nearly identical to the process above. The number of neighbors preferred to be selected is 4. However, the maximum depth, as well as the minimum number of sample splits, in DT remained to be the same, which are 10 and 5 sequentially.

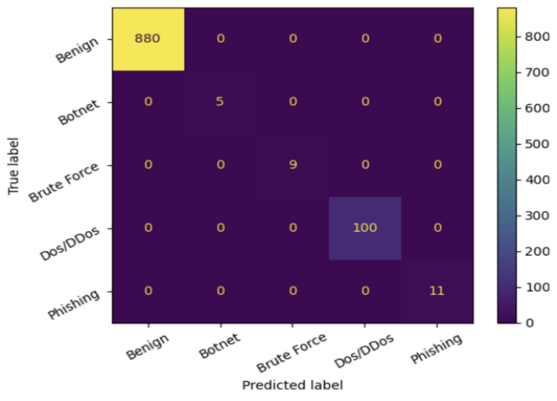
4.4. Process & results

On the basis of the implementation of three machine learning classifiers and the tools provided to evaluate their performances, four datasets containing different network data are available to be compared, looking for the most appropriate machine learning classifier that largely improves and optimizes their efficiency. To better shed light on what the objective of the experiment is intended to be communicated , the explanation outlined below can be revealed.

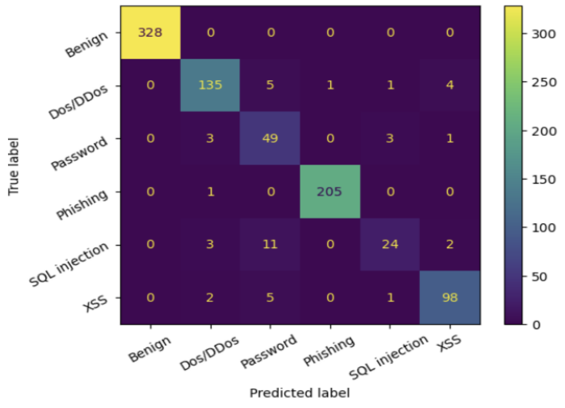
4.4.1. Decision tree result. As highlighted in Section 3.4.1, DT is one binary classification algorithm for detecting objects based on their features involved. The confusion matrix can be shown as follows when DT is incorporated into four datasets in Section 4.3. As shown in Figure 4.



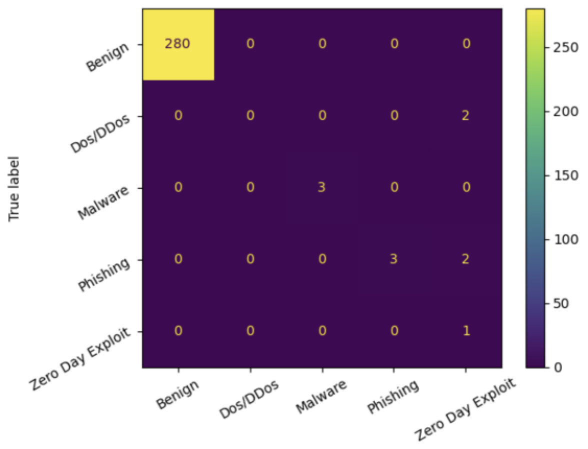
a. *NF-BoT-IoT-v2*



b. NF-CSE-CIC-IDS2018-v2



c. NF-ToN-IoT-v2



d. NF-UNSW-NB15-v2

Figure 4. Shows how the confusion matrix under the execution of Decision Tree looks like.

Therefore, it is not difficult to compute the accuracy of each confusion matrix belonging to four datasets. The following results can be finalized through the law of probability.

Decision Tree Accuracy:

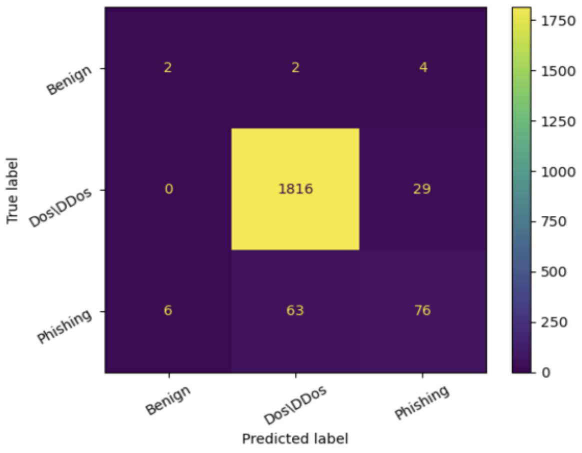
NF-BoT-IoT-v2: 0.98

NF-CSE-CIC-IDS2018-v2: 1

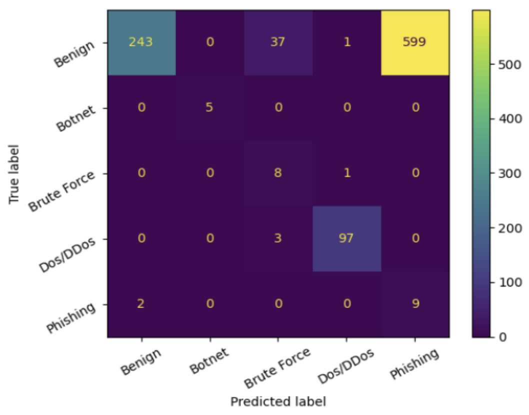
NF-ToN-IoT-v2: 0.951

NF-UNSW-NB15-v2: 0.986

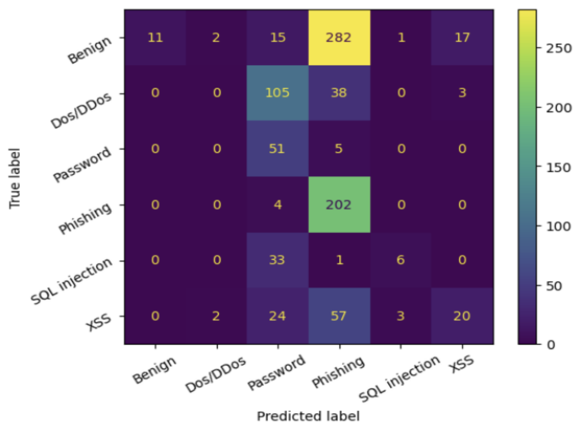
4.4.2. Naive bayes result. The confusion matrix can also be used by Naive Bayes(NB) in addition to DT to testify how the algorithmic efficiency might vary in each four dataset. Similar to the techniques illustrated above, the confusion matrices connected to four network datasets under the implementation of Naive Bayes are displayed in the following images. As shown in Figure 5.



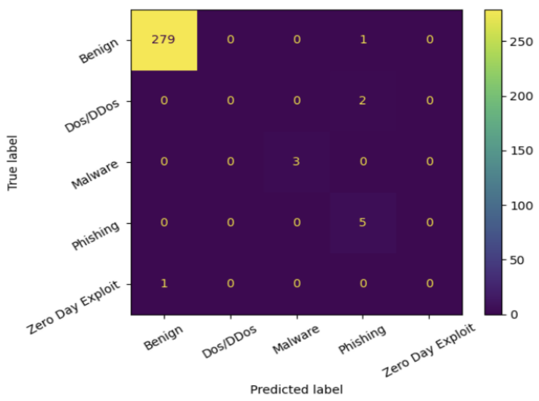
a. NF-BoT-IoT-v2



b. NF-CSE-CIC-IDS2018-v2



c. NF-ToN-IoT-v2



d. NF-UNSW-NB15-v2

Figure 5. Demonstrates the use of the confusion matrix in Naive Bayes.

At the same time, the accuracies with regard to 4 confusion matrices in each dataset should not be disregarded due to their significance. The results can be disclosed as follows.

Naive Bayes Accuracy:

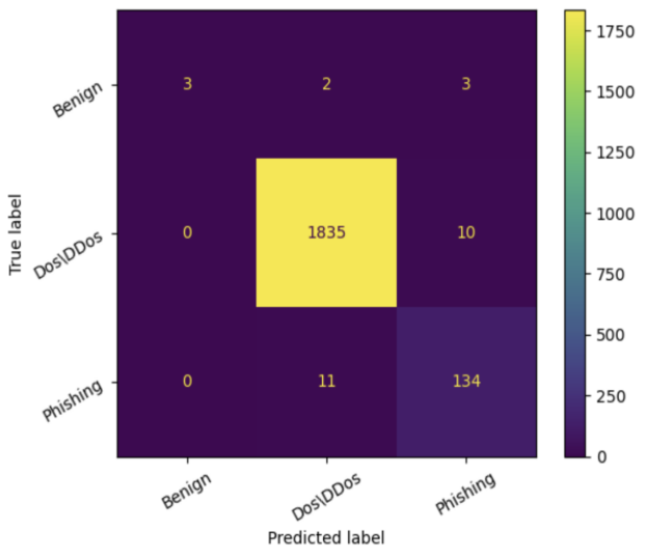
NF-BoT-IoT-v2: 0.948

NF-CSE-CIC-IDS2018-v2: 0.360

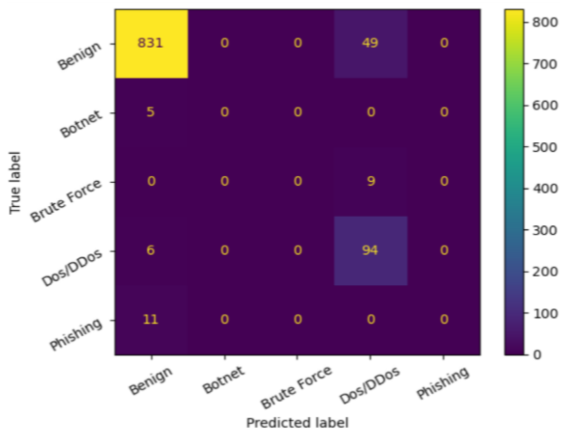
NF-ToN-IoT-v2: 0.329

NF-UNSW-NB15-v2: 0.986

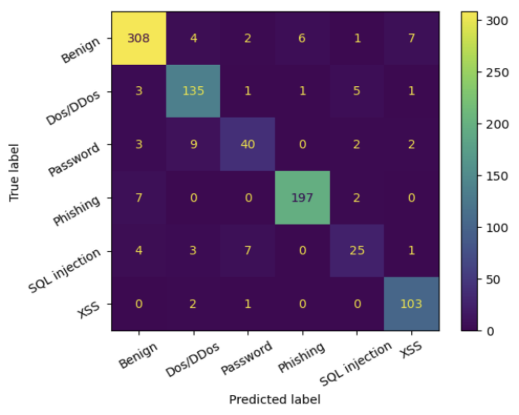
4.4.3. K-Nearest neighbor result. Last but not least, the performance metrics regarding K-Nearest Neighbor(KNN) need to be clearly emphasized. KNN, as summarized in 3.4.2, is one type of algorithm that is relevant to object classification. By placing KNN into four network datasets, the confusion matrix, as well as the corresponding accuracy obtained, can check their plausibility. The analysis associated with this algorithm is detailed as follows. As shown in Figure 6.



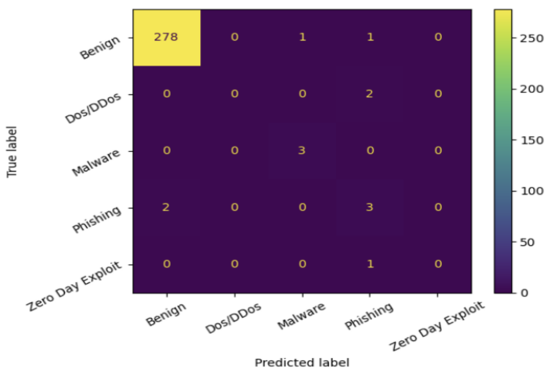
a. NF-BoT-IoT-v2



b. NF-CSE-CIC-IDS2018-v2



c. NF-ToN-IoT-v2



d. NF-UNSW-NB15-v2

Figure 6. Is used to present the result of the confusion matrix in terms of KNN, compared to DT and NB.

In order to determine the accuracy score in each four dataset, the same formula illustrated in Equation * can be enforced, attempting to prove and hypothesize whether KNN achieves a higher performance than DT and NB. The result in each dataset can be displayed below.

K-Nearest Neighbor Accuracy:
NF-BoT-IoT-v2: 0.987
NF-CSE-CIC-IDS2018-v2: 0.920
NF-ToN-IoT-v2: 0.916
NF-UNSW-NB15-v2: 0.976

4.4.4. Result. Before the final decision is made, it is better to use a table to collect information about the performance of each machine learning algorithm to stay organized. The information of the table can be structured as below. As shown in table 2 and 3.

Table 2. NF-UNSW-NB15-v2 and NF-ToN-IoT-v2.

	NF-UNSW-NB15-v2				NF-ToN-IoT-v2			
	Accuracy	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score
Decision Tree	0.986	0.640	0.720	0.678	0.951	0.899	0.887	0.893
Naïve Bayes	0.986	0.524	0.599	0.559	0.360	0.329	0.444	0.378
K-Nearest Neighbor	0.976	0.434	0.549	0.485	0.916	0.871	0.855	0.863

Table 3. NF-BoT-IoT-v2 and NF-CSE-CIC-IDS2018-v2.

	NF-BoT-IoT-v2				NF-CSE-CIC-IDS2018-v2			
	Accuracy	Precision	Recall	F1 Score	Accuracy	Precision	Recall	F1 Score
Decision Tree	0.988	0.980	0.881	0.928	1.000	1.000	1.000	1.000
Naïve Bayes	0.948	0.637	0.587	0.611	0.360	0.631	0.791	0.702
K-Nearest Neighbor	0.987	0.968	0.765	0.855	0.920	0.318	0.377	0.345

While taking a careful review of Table 2, an assumption is proposed that Decision Tree(DT) out of all three classifiers is usually preferred, but KNN and Naive Bayes should not be completely abandoned. For example, if paying attention to the performance of Naive Bayes, the datasets other than NF-CSE-CIC-IDS2018-v2 and NF-ToN-IoT-v2 can potentially be applied to detect any networks that are relatively weird. Instead, to help figure out the problem written above, several measures may be meant to be taken to try to increase the likelihood for both algorithms to be commonly used.

5. Conclusion

Based on the above discussion, it can be concluded that Decision Tree (DT) is the most accurate and efficient method for solving key issues in network information security. This study's introduction and analysis demonstrate that people already have a cognitive understanding and have developed certain means to detect and identify malicious network intrusion problems. However, more algorithms, such as KNN and NB, are needed to be improved or developed further to decrease the probability of network hacking, as they do not perform as well as DT.

References

[1] Denis Rangelov, Philipp Lämmel, Lisa Brunzel, Stephan Borgert, Paul Darius, Nikolay Tcholtchev, & Michell Boerger. (2023). Towards an Integrated Methodology and Toolchain for Machine Learning-Based Intrusion Detection in Urban IoT Networks and Platforms. Future Internet, 15(98), 98. https://doi.org/10.3390/fi15030098

[2] Rakas, S. V. B., Stojanovic, M. D., & Markovic-Petrovic, J. D. (2020). A Review of Research Work on Network-Based SCADA Intrusion Detection Systems. IEEE Access, Access, IEEE, 8, 93083–93108. https://doi.org/10.1109/ACCESS.2020.2994961

[3] Dhanya, K. A., Vajipayajula, S., Srinivasan, K., Tibrewal, A., Kumar, T. S., & Kumar, T. G. (2023). Detection of Network Attacks using Machine Learning and Deep Learning Models. Procedia Computer Science, 218, 57–66. https://doi.org/10.1016/j.procs.2022.12.401

[4] Sarhan, M., Layeghy, S., & Portmann, M. (2022). Towards a standard feature set for network intrusion detection system datasets. Mobile networks and applications, 1-14.

[5] VanderPlas, J. (2023). Python Data Science Handbook. O'Reilly Media, Inc.

[6] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (1970, January 1). Scikit-Learn: Machine learning in Python. Journal of Machine Learning Research. Retrieved May 2, 2023, from https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

[7] P. Ferreira, D. C. Le and N. Zincir-Heywood, "Exploring Feature Normalization and Temporal Information for Machine Learning Based Insider Threat Detection," 2019 15th International Conference on Network and Service Management (CNSM), Halifax, NS, Canada, 2019, pp. 1-7, doi: 10.23919/CNSM46954.2019.9012708.

[8] Kanimozhi, V., & Jacob, P. (2019). UNSW-NB15 dataset feature selection and network intrusion detection using deep learning. Int. J. Recent Technol. Eng, 7, 443-446.

[9] Li, J., Zhao, Z., & Li, R. (2017). A machine learning based intrusion detection system for software defined 5G network. arXiv preprint arXiv:1708.04571.

[10] WS, J. D. S., & Parvathavarthini, B. (2020, July). Machine learning based intrusion detection framework using recursive feature elimination method. In 2020 International Conference on System,

[11] Mati W P. Transferability of Intrusion Detection Systems Using Machine Learning between Networks[D]. University of Windsor (Canada), 2022.

[12] Nikhitha, M., & Jabbar, M. A. (2019). K nearest neighbor based model for intrusion detection system. Int. J. Recent Technol. Eng, 8(2), 2258-2262.

[13] Li, W., Yi, P., Wu, Y., Pan, L., & Li, J. (2014). A new intrusion detection system based on KNN classification algorithm in wireless sensor network. Journal of Electrical and Computer Engineering, 2014.
[14] Rao, B. B., & Swathi, K. (2017). Fast kNN classifiers for network intrusion detection systems. Indian Journal of Science and Technology, 10(14), 1-10.
[15] Mishra, S. P., Sarkar, U., Taraphder, S., Datta, S., Swain, D., Saikhom, R., ... & Laishram, M. (2017). Multivariate statistical data analysis-principal component analysis (PCA). International Journal of Livestock Research, 7(5), 60-78.
[16] Chauhan, N. S. (2022, April 8). Naïve Bayes Algorithm: Everything you need to know. KDnuggets. Retrieved April 21, 2023, from https://www.kdnuggets.com/2020/06/naive-bayes-algorithm-everything.html
[17] Wickramasinghe, I., & Kalutarage, H. (2021). Naive Bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation. Soft Computing, 25(3), 2277-2293.
[18] Yassin, W., Udzir, N. I., Muda, Z., & Sulaiman, M. N. (2013). Anomaly-based intrusion detection through k-means clustering and naives bayes classification.
[19] Krause B. Questionable Research Practices–Anmerkungen zur aktuellen Diskussion[C]//Empirische Evaluationsmethoden Band 22 Workshop 2017. 109.
[20] Bates, S., Hastie, T., & Tibshirani, R. (2023). Cross-validation: what does it estimate and how well does it do it?. Journal of the American Statistical Association, (just-accepted), 1-22.
[21] Wang Z, Liang M, Delahaye D. Data-driven conflict detection enhancement in 3d airspace with machine learning[C]//2020 International Conference on Artificial Intelligence and Data Analytics for Air Transportation (AIDA-AT). IEEE, 2020: 1-9.
[22] Perez, D., Astor, M. A., Abreu, D. P., & Scalise, E. (2017, September). Intrusion detection in computer networks using hybrid machine learning techniques. In 2017 XLIII Latin American Computer Conference (CLEI) (pp. 1-10). IEEE.
[23] Search UNB. University of New Brunswick est.1785. (2018). Retrieved April 21, 2023, from https://www.unb.ca/cic/datasets/ids-2018.html
[24] The UNSW-NB15 Dataset. The UNSW-NB15 Dataset | UNSW Research. (2021, June 2). Retrieved April 23, 2023, from https://research.unsw.edu.au/projects/unsw-nb15-dataset