electronics

Article

Numerical Feature Selection and Hyperbolic Tangent Feature Scaling in Machine Learning-Based Detection of Anomalies in the Computer Network Behavior

Danijela Protić [1], Miomir Stanković [2], Radomir Prodanović [1,*], Ivan Vulić [3], Goran M. Stojanović [4], Mitar Simić [4], Gordana Ostojić [4] and Stevan Stankovski [4]

1 Center for Applied Mathematics and Electronics, 11000 Belgrade, Serbia; danijelaprotic318@gmail.com
2 Mathematical Institute of SASA, 11000 Belgrade, Serbia
3 Military Academy, University of Defense, 11042 Belgrade, Serbia
4 Faculty of Technical Science, University of Novi Sad, 21000 Novi Sad, Serbia; goca@uns.ac.rs (G.O.)
* Correspondence: radomir.prodanovic@vs.rs

Abstract: Anomaly-based intrusion detection systems identify the computer network behavior which deviates from the statistical model of typical network behavior. Binary classifiers based on supervised machine learning are very accurate at classifying network data into two categories: normal traffic and anomalous activity. Most problems with supervised learning are related to the large amount of data required to train the classifiers. Feature selection can be used to reduce datasets. The goal of feature selection is to select a subset of relevant input features to optimize the evaluation and improve performance of a given classifier. Feature scaling normalizes all features to the same range, preventing the large size of features from affecting classification models or other features. The most commonly used supervised machine learning models, including decision trees, support vector machine, k-nearest neighbors, weighted k-nearest neighbors and feedforward neural network, can all be improved by using feature selection and feature scaling. This paper introduces a new feature scaling technique based on a hyperbolic tangent function and damping strategy of the Levenberg–Marquardt algorithm.

Keywords: machine learning; binary classification; intrusion detection; feature scaling; feature selection

check for updates

Citation: Protić, D.; Stanković, M.; Prodanović, R.; Vulić, I.; Stojanović, G.M.; Simić, M.; Ostojić, G.; Stankovski, S. Numerical Feature Selection and Hyperbolic Tangent Feature Scaling in Machine Learning-Based Detection of Anomalies in the Computer Network Behavior. *Electronics* **2023**, *12*, 4158. https://doi.org/10.3390/electronics12194158

Academic Editors: Jheng-Jia Huang, Ray-Lin Tso and Po-Wen Chi

Received: 15 August 2023
Revised: 8 September 2023
Accepted: 11 September 2023
Published: 7 October 2023

1. Introduction

In recent years, the need for data protection has grown as computer applications process a large amount of data across the globe. Intrusion detection systems (IDSs) serve as the primary line of defense against malicious attacks to the computer networks. Signature-based intrusion detection systems protect computer networks by proactively detecting the presence of known attacks. Anomaly-based intrusion detection systems identify abnormal network behavior by detecting deviations from what is considered "normal" behavior [1]. The concept of normality is generally introduced through a formal model that describes the relationships between the variables involved in a system's dynamic, and events are identified as abnormal when the deviation from the normality model is large enough. Anomaly detection in supervised machine learning (ML) can be thought of as a binary classification problem because the datasets used to train and test the models contain binary labels.

Research efforts have been extensively dedicated to utilizing ML algorithms for anomaly detection [2–5]. In order to determine the statistical model of an anomaly-based IDS, various approaches used a number of assumptions about the training data and validation methods. To classify network traffic, the authors of [6–9] present a comparative analyses and performance evaluation of multiple supervised ML algorithms. The most widely used supervised ML models including SVM [3–5], Naïve Bayes (NB) [6,7], DT [7,8], nearest neighbors [9–11], random forest (RF) [6,7,12], artificial neural networks

(ANN) [7,10,13], logistic regression (LR) [6,7], discriminant analysis (DA) [9], and ensemble methods (EM) [14–16] were analyzed.

The main problem with supervised learning in the context of anomaly-based intrusion detection is that a large amount of data can affect classifier decisions. Dimensionality reduction is the most common method for eliminating irrelevant or duplicate features related to the target concept. Without compromising classification performance, this technique can significantly reduce processing time and improve the overall effectiveness of an IDS [17]. By identifying the features that are most important in deciding about anomalies, feature selection can improve classifier performance. The selected features must be able to distinguish between samples from different classes. If they are not, a model is more likely to overfit, resulting in poor classifier performance. It is important to note that a classifier built from the reduced feature set must be equally effective as one built from the full feature set [18].

Feature scaling is a popular method used to enhance classifier performance. Along with feature selection, many ML-based algorithms require feature scaling to ensure that all relevant features are on the same scale. This is performed to prevent a single feature from having a negative impact on the model or other features due to its magnitude. Since ML-based algorithms primary focus on numbers, different ranges can result in the more significant number becoming more important during the model training. This paper introduces a novel feature scaling technique that speeds up processing while improving model accuracy, based on hyperbolic tangent transformation and the Levenberg–Marquardt strategy.

Over the past few decades, researchers have examined a variety of anomaly-based intrusion detection systems. These systems have been tested using different datasets, and this study aims to provide analysis and comparison of 14 such datasets. Each dataset is unique in terms of the amount of data it contains, the nature of attacks, and its intended use [19–26]. These datasets are primarily designed for signature-based intrusion detection and mainly simulate real network traffic. The Kyoto 2006+ dataset, which was recorded directly from the actual network traffic, is the only dataset designed specifically for anomaly-based detection. As a result, we used the Kyoto 2006+ dataset as a baseline for tests on feedforward neural network (FNN), DT, SVM, k-nearest neighbors (k-NN), and weighted k-NN (wk-NN) [10–13,27–33]. We compared classification performance of the model using accuracy, processing time, and F1-score.

2. Related Work

Numerous research challenges in data analysis and comprehension are influenced by the availability and usability of multi-dimensional data. For that reason, feature selection is frequently used to reduce the dimensionality of the datasets [34]. Data preprocessing techniques can also include feature scaling. When feature scaling is not performed, ML algorithms tend to weigh higher values and treat smaller values as lower regardless of their units. In order to compare performance of the ML-based models, they must be trained on the same data. The confusion matrix is a well-known and widely used predictive analytics tool in machine learning that serves as the foundation for the performance metrics which are usually used in binary classification [35,36]. The following text discusses research on four critical topics in anomaly-based intrusion detection: collecting data, feature selection, feature scaling, and binary classification process.

2.1. Data Collection

The quality of data collected from network traffic is an important aspect of intrusion detection. It depends on whether the dataset is simulated or not, network traffic, sources of the attacks, the number of features and duplicates, redundancy, public availability, and more. Table 1 contains a brief summary of the most commonly used datasets.

Table 1. Datasets.

Dataset	Year	Attacks	Features	Traffic	Description
CSE-CIC-2018	2018	Potator (FTP/SSH), DoS, DDoS, Botnet, infiltration, Web attacks	80+	Emulated	The dataset includes 10 days of network traffic and log files collected from 50 machines on the attacker side and 420 PCs and 30 servers on the victim side.
CIC-IDS-2017	2017	Botnet, DoS, DDoS, Goldeneye, Hulk, HTTP, RUDY, Slowhttptest	80+	Emulated	The dataset was captured over a 5-day period; small network traffic in bidirectional flow-based and packet-based format.
Kyoto 2006+	2006–2015	Port scan, malware, shellcode, DoS	24	Real	The dataset collected from 10 years of actual network traffic.
AWID	2015	Attacks on Wi-fi 802.11	156	Emulated	Wireless LAN traffic; captured 37 million packet/h; 11 clients; small network.
UNSW-NB15	2015	Attacks behavior	49	Hybrid	Includes 31 h attacks against multiple servers; automated attack generator tool IXIA Perfect Storm.
ADFA	2014	Brute force, Java/Linux interpreter, C100 webshell	26	Hybrid	Linux/Windows OS system call.
ISCX 2012	2012	Infiltrating, DDoS, HTTP, SSH	20	Emulated	This dataset includes 7 weeks of small network traffic in packet-based format.
CAIDA	2007	DDoS	Traffic traces	Hybrid	The dataset was collected via a commercial backbone link; does not contain a diversity of attacks.
DARPA	1998–1999	DoS, R2L, U2R, probe	41	Emulated	This dataset includes 7 weeks of small network traffic in packet-based format.
KDD Cup '99	1998	DoS, R2L, U2R, probe	42	Emulated	This dataset includes 5 weeks of small network traffic in packet-based format.
NSL-KDD	1998	DoS, R2L, U2R, probe	42	Emulated	The NSL-KDD dataset is derived from the KDD-Cup'99 dataset and does not contain redundant records in the training set nor duplicates in the test set.

In 2013, the Australian Defense Force Academy (AD-FA) produced two datasets containing records from Linux/Unix (ADFA-LD) and Windows (ADFA-WD) systems. The ADFA-LD consists of system call traces collected from a temporary local Linux server, with six current types of cyber attacks [37]. The ADFA-WF is a set of DLL access requests and system calls that are used in various hacking attacks [38]. Both ADFA datasets are benchmarks for evaluating IDS based on system calls.

The Aegean Wi-Fi Intrusion Detection (AWID) dataset is a publicly available labeled dataset based on actual traces of normal and intrusion activity from an IEEE 802.11 Wi-Fi network, developed in 2016. The dataset consists of 14 simulated existing attacks and includes 155 different attributes [39].

The CAIDA dataset was produced by the Center of Applied Internet Data Analysis between 2002 and 2016. Three datasets comprise the full CAIDA dataset: CAIDA OC48, which contains various types of data observed on an OC48 link in San Hose; CAIDA DDoS, which contains one hour's worth of DDoS attack traffic broken down into 5 min PCAP files; and CAIDA Internet Traces 2016, which contains passive traffic traces from the Equinix-Chicago High-speed Internet backbone. The CAIDA datasets gather different types of data in topologically and geographically diverse locations. Due to numerous shortcomings, these benchmarking datasets lack effectiveness [40].

The Canadian Institute for Cybersecurity (CIC) created the CIC-IDS-2017 dataset in 2018. It contains harmless and most recent common attacks that cover 11 criteria. These updated attacks include DoS, DDoS, brute force, port scans, and many others [41].

The Communication Security Establishment (CSE) and the CIC jointly created CSE-CIC-2018 dataset. Seven different attack scenarios were included in the final dataset: brute-force, hearth bleed, botnet, DoS, DDoS, web attacks, and infiltration. As roughly 17% of the instances in the dataset have abnormal traffic, this indicates a class imbalance. Instead of serving as a repository for signature-based IDS, the dataset promotes anomaly-based intrusion detection [42].

The Lincoln Laboratory of the Massachusetts Institute of Technology (MIT) developed the DARPA'98 dataset. The dataset is divided into two parts: offline and online. In order to evaluate IDS offline, network traffic and audit logs collected in a simulated network are used [43].

Two profiles can be found in the ISCX 2012 dataset. While Alpha runs a variety of multi-stage attacks, Betha generates real network traffic and background noise. The network traffic for the HTTP, SMP, SSH, IMAP, POP3, and FTP protocols is included in the dataset. The dataset does not include HTTPS traces. Furthermore, the distribution of simulated attacks is based on hypothetical data rather than actual global statistics [41].

The DARPA'98 dataset is the foundation for the KDD Cup'99 dataset. The KDD Cup'99 dataset comprises approximately 5 million single connection vectors, each of which has 41 features [44]. From the KDD Cup'99 dataset, the NSL-KDD dataset was created. It fixes deficiencies in the KDD Cup'99 dataset brought on by redundant records in the training set and duplicate records in the test set. Additionally, the training and test set both have a reasonable number of records [19].

Moustafa et al. (2015) first presented the UNSW-NB 15 dataset, which includes real data and contemporary network traffic attack activities. Raw network packets from nine different attacks are included in the dataset [45].

The main reason why the Kyoto 2006+ dataset is used in the experiments is that it contains records of actual network data. Data collected from ~350 honeypots at five different computer networks inside and outside Kyoto University between 2006 and 2015 are included in the first part of the dataset [18,46–49]. The authors also developed a server that was installed in the same network as the honeypots in order to produce the data traffic. The dataset consists of 24 features. Fourteen statistical features were derived from the KDD-Cup'99 dataset, and an additional 10 features were solely used for anomaly detection. The dataset omits information on different attack types. Instead, the feature *Label* determines whether or not the session was normal [8,50,51]. The IDS Bro software 2.4 is used to convert real network traffic into sessions [52,53]. The IDS Bro is suitable for powerful network monitoring and real-time application-layer status reporting. The Internet Protocol (IP) packets are received and transformed into events by the Bro event engine.

2.2. Feature Selection

Feature selection is used to find the feature subset that reduces the model complexity, minimizes the generalization error, improves prediction power, and facilitates quick model evaluation. The main purpose of feature selection is to prevent classifier performance degradation due to redundant information in network data [12,54,55]. Feature selection algorithms are designed to reduce the dataset and speed up the classifier without significantly affecting its performance. Depending on whether or not the training set is labeled, the feature selection algorithm can be supervised [56,57], unsupervised [58,59], or semi-supervised [60,61]. The unsupervised feature selection works with data without a feature relevance that is difficult to measure. It is common for a high-dimensional datasets. The relevance of the features in semi-supervised learning is evaluated using both labeled and unlabeled data [60]. In supervised feature selection, the induction algorithm is represented with a set of training instances, which are described by a feature vector and a class label. The supervised feature selection evaluates features' relevance based on the information on the label. It should be noted that an accurate classifier requires a lot of labeled data for training, which affects processing time. In addition, supervised feature selection can be divided into three categories: filter, wrapper, and embedded [62]. The filter method selects features independently, based on classification, regardless of the classifiers used. Prior to classification, it selects a subset of features in a preprocessing stage. The filter method evaluates the features without including a learning algorithm. The method is based on the general properties of the data [63,64]. The chi-square test, variance threshold, and information gain (IG) are the examples of the most popular filter feature selection methods [17]. The wrapper method applies a black box to several variables based on their predictive power to the learning machine of interest [65–67]. The wrapper method uses a predefined training algorithm's predictive accuracy to determine the quality of the selected features. For that reason, it is expensive to be implemented for feature-rich data. The best-known wrapper algorithms are the genetic algorithm, recursive feature elimination, and sequential feature selection. The authors of [68] also present a hybrid version of feature selection method that uses both filter and wrapper techniques. The embedded method combines the efficiency of unsupervised and supervised methods. It gains knowledge of which features have the highest classification accuracy, incorporates the selection of variables into the training process, and determines the relevance of a feature analytically based on the objective of the learning model [69,70]. First, the model integrates statistical criteria to select several features from candidates with certain cardinality and then chooses the subset with the highest classification accuracy. The most well-known examples of the embedded methods are L1 and L2 regularizations and the elastic net [71]. The advantages and disadvantages of the filter, wrapper, and embedded methods are enlisted in Table 2 [72–74].

In our work, we have expected that, based on our knowledge about the dataset, we can choose features meeting specific criteria as a result of our experience. Here we used the filter-like feature selection method because of its generalization ability, scalability, and low risk of overfitting. Moreover, this feature selection does not incorporate specific processing time. In [75], the authors also examine techniques to shorten the processing time and propose feature selection for reducing memory storage space and speeding up the classification algorithm. The authors compared six types of classification based on neural networks, linear discriminant analysis, k-NN, DT, and SVM. The results show that many classifiers have a reduction of more than 50% in the processing time and about 90% of accuracy when only 40% of the features are used. In [76], the authors concentrated on SVM, DT, BN, k-means, artificial immune system (AIS), and genetic algorithm (GA) and came to the conclusion that feature selection algorithms that used a filter approach had a lower time complexity than those based on distance, consistency, information, and dependencies.

In [77], the authors advised that the correct application of the feature selection could significantly improve the processing time and overall performance of the classification.

Table 2. A taxonomic summary of filter, wrapper, and embedded techniques.

Feature Selection	Advantages	Disadvantages
Filter	Does not incorporate specific processing time. Low risk of overfitting. Excellent generalization ability. Scalable.	Ignores the dependency of features. Does not interact with the classifier.
Wrapper	Uses the performance of the learning algorithm to determine the feature subset with the best performance. Models feature dependency. Superior in performance to the filter in terms accuracy. High computational cost.	The risk of overfitting is higher. Selection is classifier dependent.
Embedded	Interacts with a classification model. Reduces the risk of overfitting. Models feature dependencies.	Problematic identification of a small set of features. Classifier-dependent selection.

Various feature selection tactics can also be used to narrow down the dataset size, and to reduce the influence of the selection of features, which depend heavily on the type of classification. In [53], the authors propose an RPFMI algorithm that offers the possibility of selecting features with concerning redundancy between features and the relationship between candidate features and classes. The mutual information, which is calculated from the joint probabilities of two random variables estimated from the frequency of the observed samples in each combination of the variable categories, represents a measure of the relationship between two random variables [78]. If and only if the two variables are statistically independent, the mutual information between them is zero [27]. The authors used the SVM to provide a Radial Basis Function (RBF) kernel to access the accuracy of detection. In the training phase, there were 10,000 normal and attack instances, while there were only 2000 in the test phase. They compared RPFMI with the MIFS [79], mRMR [80], MIFS-U [81], CIMI [82], MMIFS [83], and FMIFS [17] models. The authors showed that the accuracy of the model was 97.74% using six features for classification. In [84], the authors combined the FIMFS with the LSSVM to build IDS. The evaluation results showed that the algorithm assigned more critical features or achieved better accuracy (99.79%) than MIFS and MMIFS with six selected features. The authors also reported better results on F-measure, DR, and FPR. Anomaly-based detection, proposed by the authors of [85], is divided into preprocessing (Min-Max normalization), feature selection, and modeling phases. The features candidates are calculated using mRMR and Neighbor Component Analysis (NCA) during the feature selection phase. In [27], the authors recommended the method of filter-based feature selection in a multi-objective optimization framework to extract relevant features from an unlabeled dataset. The authors obtained a subset of optimal features using the solid multi-objective NGSA-II and six proposed models that used (1) mutual information and standard deviation, (2) mutual information and entropy, (3) IG and standard deviation, (4) IG and entropy, (5) Pearson correlation coefficient and standard deviation, and (6) Pearson correlation coefficient and entropy objective functions. The entropy was metric of information theory that described the uncertainty in a group of observations, i.e., the impurity of a feature [27]. The IG provided information about the target feature that was the most informative (meaningful) [86]. The Pearson correlation coefficient was used for jointly normally distributed data that followed a bivariate normal distribution. The correlation coefficient is scaled such that it ranged from -1 to 1; the relationship became stronger and eventually approached a straight line as the coefficient approached the absolute value of 1 [87]. In [6], the authors applied the subset of the optimal features in six different ML classifiers: DT, SVM, RF, k-NN, Adaboost, and MLP. After the subsets were found, 10-fold cross-validation was carried out on one of the subsets. The authors discovered that for the KDD Cup'99 dataset and the NSL-KDD dataset, the

DT had the best results. The authors used the DT to conduct the experiments with the Kyoto 2006+ dataset. The results are as follows: Weighted average accuracy = 99.6%, the F-measure = 99.65%, and the FAR = 0.3%. In [88], the authors proposed a model that combined the GA to select features and C4.5 for generating signatures. Afterward, the frequent element set with the association rule (for building normal traffic rules) was used for anomaly detection. With a high decision rate and a low FPR, the model Accuracy of 99.94% was achieved. In [89], the authors propose the PCA to reduce the dimensionality of the dataset. A Min-Max scaling is used to normalize features in the range [0, 1]. The experiments are carried out in k-NN, SVM, an ensemble of trees and RC algorithms to estimate the covariance matrix in multivariate data. The results show average Accuracy (60%), high Precision (0.99), low Recall (0.36) and low F1-score (0.53). The results also demonstrate that there are no gains from the PCA transformation. In [90], the authors present the general high-quality performance of the SVM, IBK, MLP, and Ensemble models based on Accuracy, DR, FAR, F-measure, and Precision. The results are presented for the PCA, IG and PCA + IG methods, with PCA + IG achieving the best results due to the ensemble classification. Additionally, the authors of [11] summarize the research on the effects of the feature selection and instance normalization and demonstrate that using nine selected numerical features, scaled between [−1, 1] result in accuracy of over 99% for k-NN, wk-NN, DT, and SVM models, with the wk-NN achieving the highest accuracy and the DT achieving the shortest processing time. In [8], the authors also examine the influence of preprocessing on the FNN used for binary classification. Compared to the other models, the FNN proved to be very precise and has the shortest processing time (PT). In [18], the authors exclude features related to the security analysis and chose three classification models: k-NN, NB, and DT. They use the Chi-squared (CS), Area Under Curve (AUC) and Signal to Noise Ratio (S2N) feature rankings. The authors of [91] test Correlation-based Feature Selection (CFS) and Particle Swarm Optimization (PSO) scenarios. When compared results of the experiments conducted to the KDD Cup'99 and the UNSWNB15 datasets, it was demonstrated that the feature selection method improved accuracy for NB, SVM and k-NN classifiers by up to 17%. The authors compared the performance of the models before and after performing preprocessing. Table 3 depicts the literature review of the presented feature selection methodology and the ML-based classification algorithms.

Table 3. Research articles on features selection used for ML-based classification.

Authors	Year	Classifiers	Features	Metrics
Alanazi and Aljuhani [85]	2022	k-NN, DT, LDA, SVM	mRMR, NCA	F1, Accuracy
Protic and Stankovic [8]	2020	k-NN, DT, FNN, SVM, wk-NN	9 numeric, Min-Max	Accuracy, t_p
Ahmad and Aziz [91]	2019	SVM, k-NN	CFS, PSO	Accuracy, TPR, FPR
Suman et al. [27]	2019	DT, SVM, k-NN, MLP	NSGA-II	F1, Accuracy
Kalavadekar and Sane [88]	2019	DT	GA	F1, Accuracy
Perez et al. [89]	2019	k-NN, DT, SVM	PCA, Min-Max	F1, Accuracy
Salo [90]	2019	MLP, SVM	PCA, IG	Accuracy, F-measure,
Protic and Stankovic [11]	2018	SVM, DT, k-NN, wk-NN	9 numeric, Min-Max	Accuracy, t_p
Zhao et al. [60]	2018	SVM	RPFMI	Accuracy
Najafabadiet et al. [18]	2016	k-NN, DT	AUC, S2N, ROC	Accuracy, TPR, FPR
Ambusaidiet et al. [17]	2014	ANN, DT, SVM, k-NN	FIMFS	F-measure, Accuracy, t_p

The proposed work and the results of the experiments given in the references [6,8,11,18,87–91] support selection of the dataset, classifiers and measurement metrics used in our work.

2.3. Feature Scaling

When the features are on significantly different scales, feature scaling can be used to reduce their mutual impact and negative effects on model evaluation. Two most-known feature scaling methods are normalization and standardization. Normalization adjusts and rescales features within the same specified range in order to make them equally important. It is appropriate when the dataset distribution is known to be non-Gaussian, which may be helpful in nearest neighbor algorithms or neural networks. Standardization does not scale features in the same range as normalization. The best-known scaling techniques are Min-Max, robust and standard scalar, Z-score, Box–Cox, Yeo–Johnson and L2 normalization [10,11,92,93].

In some cases, the selection of features does not affect the classification, but a suitable scale can affect the classification positively [89]. Feature scaling enhances ML-based classifier performance and prevents a single feature from having a negative impact on both the model and other features because of its size.

In [27], the authors propose the Redundant Penalty Between Features (RPFMI) algorithm for feature selection in terms of redundancy, influence between features and classes, and the relationship between the candidate feature and classes. The authors used the SVM to access the accuracy detection and compared the RPFMI algorithm to the Mutual Information Feature Selection (MIFS) [79], the Maximum-Relevance-Minimum-Redundancy (mRMR) [80], MIFS Uniform Information Distribution (MIFS-U) [81], Classify and Itemize Medical Image (CIMI) [82], Modified MIFS (MMIFS) [83], and Flexible MIFS (FMIFS) [17] models. The accuracy of the models served as a comparison criterion for classifiers. The authors also developed IDS by fusing the FMIFS with the Least-Squares SVM (LS-SVM) [84]. The F-measure, Detection Rate (DR), and False Positive Rate (FPR) were used to present the results. The reliable multi-objective Non-dominated Sorting Generic Algorithm II (NSGA-II) [94] for filter feature selection is presented in [81]. The relevant features are selected from an unlabeled data record. The authors used the optimal subset of features to train six different machine learning classifiers: DT, SVM, RF, k-NN, Adaboost, and Multi-Layer Perceptron (MLP). The metrics used to present the results were weighted average accuracy, decision rate, precision, F-measure, and the False Alarm Rate (FAR). Reference [89] shows how Principal Component Analysis (PCA) is used to reduce the dimensionality of the dataset. The experiments are conducted using the k-NN, ensemble of trees, SVM, and Robust Covariance (RC) algorithms, while the Min-Max normalization in range [0, 1] is used for feature scaling. The findings are presented as accuracy, precision, recall, and F1_score. It is shown that PCA transformation did not affect the models. In [90], the authors present the best classification performance of SVM, MLP, and ensemble models based on accuracy, DT, precision, F-measure, and FAR, for the PCA, IG, and PCA + IG methods. The best results are achieved for PCA + IG feature selection and ensemble model classification. In [8], the authors summarize the research on DT, SVM, FNN, k-NN, and weighted k-NN (wk-NN) models and describe feature selection and instance normalization. The instances of each feature were normalized in the range [-1, 1], and the results show the high accuracy of all the models. In [12], the authors first removed unknown data samples, infinite data values, and duplicates from the dataset. Then, they removed negative time-based samples and reduced the normal traffic samples to avoid bias. The feature were scaled with the Yeo–Johnson normalization. The highest accuracy was achieved with the deep neural network (DNN)-based IDS.

This paper refers to a feature scaling based on S-shaped hyperbolic tangent function that is centered around zero, has a very sharp gradient near zero and is limited in range ± 1. The use of the hyperbolic tangent function to Min-Max normalized instances is essentially what hyperbolic tangent scaling is.

2.4. Binary Classification

In general, the classification process can be divided into data collection, preprocessing, classification, and post-processing (optional) (see Figure 1). The term “classification” in machine learning refers to a predictive modeling problem where a class label is predicted for a specific instance of input data. In order to predict the results of future input observations, it is necessary to fit a statistical model that relates a set of features to their corresponding goals, and then use the model. Binary classification is used in anomaly detection to divide network data into normal and abnormal classes. The most popular supervised ML-based binary classification methods are DT, SVM, k-NN, ANN, and NB [27,28].

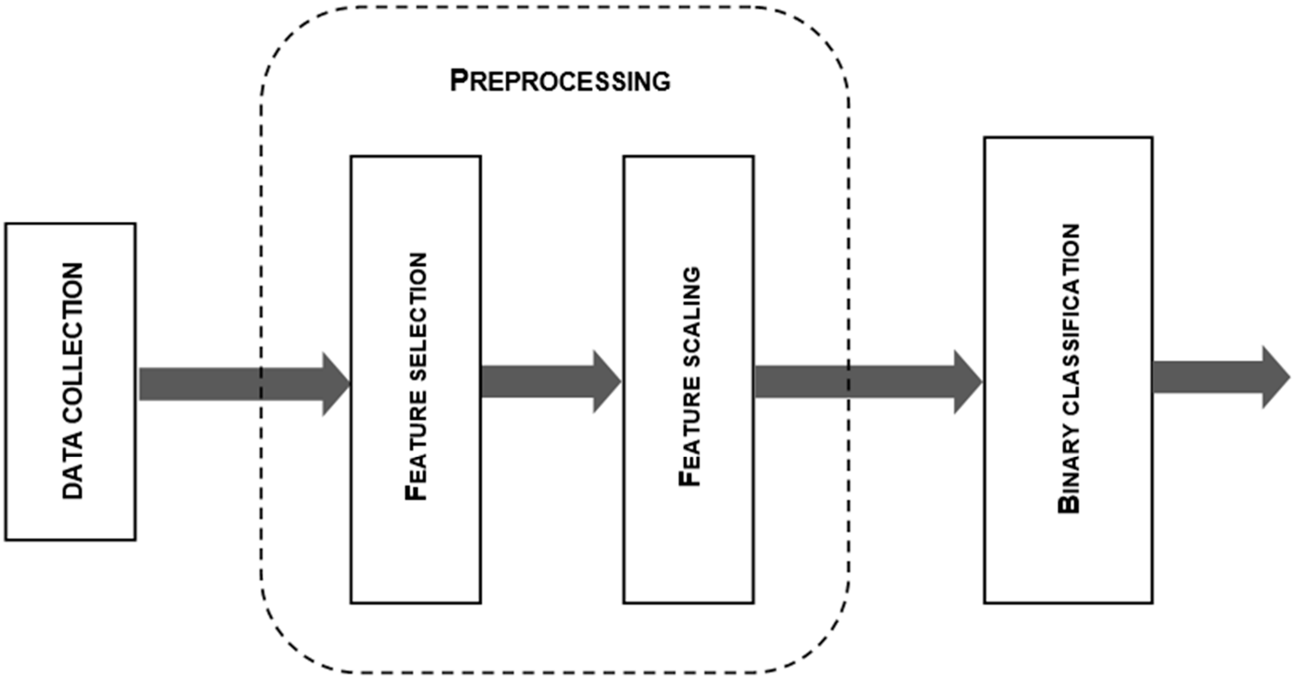

Figure 1. The binary classification process: data collection, preprocessing, and binary classification.

In a DT model, each node represents a feature, each branch represents a decision, and each leaf represents a class label. To create a tree, the model selects the features automatically and prunes them to remove unnecessary branches and prevent overfitting [28]. CART, C4.5, and ID3 are the most frequently used DT models [95]. The idea of maximum margin separation of hyperplanes in n-dimensional feature space using the kernel function is the foundation of SVM model [29–32]. The k-NN algorithm identifies a sample given on k neighbors and computes the distances to the neighbors. The performance of the classifier are affected by the number of nearest neighbors. If k is small, the model tends to overfit, while the misclassification can be a result of a large value of k [10,17]. The ANN consists of neurons, which are processing elements organized in layers, and the connections between them. Backpropagation (BP) is used to train the network [13,33]. The Naïve Bayes algorithm adopts the principle of class independence from Bayes' theorem. This indicates that each predictor has the same effect on the outcome and that presence of one feature has no impact on the presence of another feature when determining the probability of a particular outcome [27].

In [24], the authors propose a hybrid system for anomaly detection that combines the Least-Squares SVM (LS-SVM) as a cutting-edge learning system with a filter-based DT as a feature selection algorithm. The effects of various feature selection methods using SVM, NB, DT, RF, k-NN, LR, and ANN, performed with the NSL-KDD dataset have been examined in [91]. The results of the experiments also conducted to the NSL-KDD dataset as a benchmark, but using NB, SVM, and DT classifiers are presented in [96]. The authors implemented recursive feature elimination (RFE) (13 features) and PCA (8 features) as feature selection methods. The LS-SVM algorithm is given in [97]. Experiments were performed on the KDD Cup'99 dataset. When the linear correlation coefficients algorithm (LCA) and cuttlefish algorithm were used for feature selection, the accuracy was around

95%. The accuracy obtained by the authors of [91], who also used the KDD Cup'99 dataset and performed experiments with k-NN, NB and SVM with correlation, and PSO, was approximately 99.9%. To identify exploit, probe, DoS, generic and normal categories in the network, the authors of [98] proposed an integrated IDS. The 98.11% accuracy is achieved with the UNSW-NB-15 dataset using GA feature selection and the DNN classification algorithm. The supervised classification model that uses PCA for dimensionality reduction combined to the SVM model is proposed in [99]. The results of the experiments conducted to the UNSW-NB15 dataset show that the proposed model improves the performance of the model and achieves an overall accuracy of 99.99%. In [100], the authors performed classification algorithms for the NSL-KDD and UNSW-NB-15 datasets and use the PCA to select the relevant features. The authors of [101] conducted the experiments on the KDD Cup'99 dataset and used entropy-based feature selection (PCA, C5.0) and DT model. The overall accuracy was 97.7 percent. Using the NSL-KDD and UNSW-NB-15 datasets, the authors of [100] also used PCA for feature selection and presented the findings of classification based on ANN, RF, RL, and SVM. In [102], GI and IG are used for feature selection to determine an average accuracy for DT, k-NN, LR, ANN, stochastic gradient descent (SGD), and RF classifiers. The UNSW-NB-15 dataset is used. An analysis of the DT-based classification of the instances from the ISCX dataset can be found in [103]. The accuracy of k-NN models was evaluated using the ADFA-WF and CAIDA datasets as benchmarks, and the results were presented in [104]. According to the authors of [105], the accuracy of the FNN and wk-NN models is over 99.0% and 99.1%, respectively. The authors of [85] used SVM, DT, k-NN, and linear discriminant analysis (LDA) to perform classification based on the novel X-IIoTID dataset. The accuracy of SVM and LDA models was 85.8% and 85.6%, respectively. The k-NN classifier performed well in terms of accuracy rate (98.6%). Both the DT and k-NN model achieved similar performance result in terms of the F1-score (99.6% and 98.7%, respectively). The LDA classifier had an F1-score and of 83.7% and SVM classifier had an F1-score of 83.97%. In [106], the authors present the accuracy of DT (99.6%), k-NN (99.9%), and ANN (98.32%) when the NSL-KDD is used as a benchmark for the experiments.

In [8,11,18,27,60,84,88–91], the authors combine various feature selection techniques to reduce the dimensionality of the dataset and present different scaling techniques to reduce the impact of one feature on the others. The authors also make extensive use of SVM, DT, k-NN, and ANN models (see Figure 2), with accuracy being the most commonly used measure of comparison between models (see Figure 3). Table 4 shows a review of the research topics from 2018 to 2022 [12,24,85,91,96–102,106]. According to the findings in [107,108], users who gain knowledge about the datasets can choose features that meet specific criteria as a result of their experience.

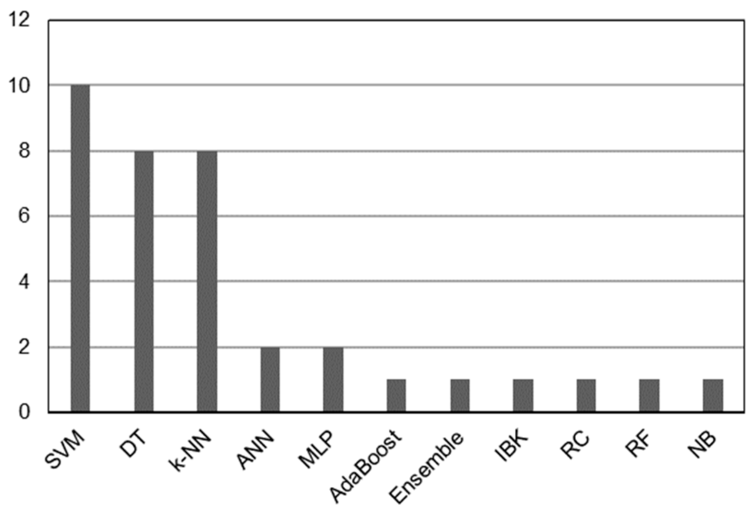

Figure 2. The most often used classifiers.

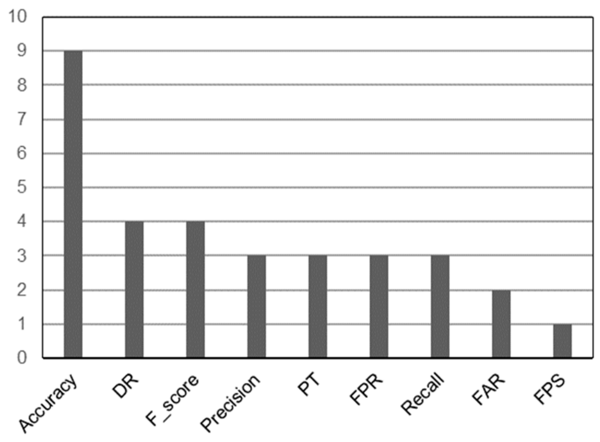

Figure 3. Metric score based on confusion matrix.

Table 4. Studies on accuracy of binary classifiers.

Authors	Year	Dataset	Classifiers	Accuracy [%]
Protic et al. [105]	2022	Kyoto 2006+	Feedforward neural network	99.0
			Weighted k-nearest neighbors	99.1
Alanazi and Aljuhani [85]	2022	X-IIoTID	Decision tree	99.36
			k-nearest neighbors	97.91
			Linear discriminant analysis	78.86
			Support vector machine	78.82
Vallejo-Huanga [100]	2021	UNSW-NB-15	Support vector machine	81.5
			Neural network	74.36
		NSL-KDD	Support vector machine	76.03
			Neural network	74.36
Siddiquiand Pak [12]	2020	ISCX-IDS-2012	Neural network	99.73
		CIC-IDS-2017	Neural network	95.2
Thakkar and Lohya [106]	2020	NSL-KDD	Support vector machine	99.6
			Decision tree	99.6
			k-nearest neighbors	99.9
			Neural network	98.32
Singh and Banerjee [96]	2020	UNSW-NB-15	Decision tree	99.46
		KDD Cup'99	Decision tree	99.88
Keserwani et al. [98]	2020	UNSW-NB-15	Neural network	98.11
Mishra et al. [99]	2020	UNSW-NB-15	Support vector machine	99.88
Wang et al. [102]	2020	SVM	Neural network	81.15
			Decision tree	83.96
Serkani et al. [24]	2019	UNSW-NB-15	Decision tree	99.46
		KDD Cup'99	Decision tree	99.88
Mohammadi et al. [97]	2019	KDD Cup'99	Support vector machine	95.03
Ahmad an Aziz [91]	2019	KDD Cup'99	Decision tree	99.88
			Support vector machine	95.03
Hamid et al. [104]	2018	ADFA-WF	k-nearest neighbors	71.43
		CAIDA	k-nearest neighbors	64.28
Idhammad et al. [103]	2018	ISCX-IDS-2012	Decision tree	99.88
Seelammal and Devi [101]	2018	KDD Cup'99	Decision tree	97.7

Likewise, the feature selection is performed here to eliminate statistical, connection duration and categorical features from the Kyoto 2006+ dataset, along with the features intended to be used in further experiments. The purpose of this step is to reduce the processing time by using only numerical features for classifier training. It is expected that accuracy and F1-score may degrade as a result.

Pruning routines, filtering, rule combination, and other similar techniques are commonly used in post-processing. These techniques are often used in machine learning because the results of ML algorithms may not be suitable for use by software applications or may not be appropriate from user's perspective. Preprocessing and post-processing tools are almost always useful when reviewing databases or optimizing acquired knowledge. These tools often use methods that are neither really symbolic nor logical.

3. Proposed Work

The main purpose of feature scaling to prevent the models from giving more weight to one feature than the others. For classifiers that make the decision based on the Euclidean distance between two points, different values can affect the classification and lead to biased and incorrect results when features are in different units of measure. If the GD is an optimization algorithm and the features are on a different scale, the classifiers can update a given weight faster than others. Feature scaling can help the GD converge faster. Algorithms used in SVM, k-NN, wk-NN, and FNN classification converge faster and give better results with scaled features.

In this paper, we describe a novel hyperbolic tangent feature scaling technique. The technique is inspired by the S-shaped, zero-centered hyperbolic tangent function (*tanh*), its sharp gradient (*tanh′*), and the damping strategy of the Levenberg–Marquardt (LM) algorithm applied to the quasi-linear part of the *tanh* function. The main idea is to find near-optimal solution of the nonlinear objective function using the properties of the very sharp gradient of the *tanh* function and the approximation of the Hessian matrix of truncated Taylor formula (second-order derivatives) with the vector-product of Jacobians (first-order derivatives) through an iterative procedure based on step length and search direction. Figure 4 shows the *tanh* function and its derivative. Because the *tanh* function is centered on 0 and ranges from -1 to 1, the first derivative of the function is very sharp near zero.

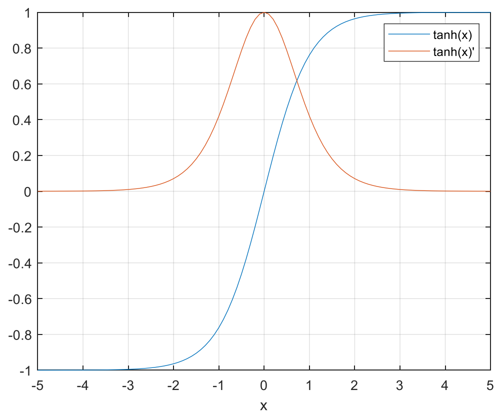

Figure 4. Hyperbolic tangent function (blue) and its first derivative (red).

The magnitudes of the features are limited to the range $[tanh(-1), tanh(1)] - [-0.7616, +0.7616]$ while changing signs, which ensures that the directions of the gradient-based updates are independent. The part of the hyperbolic tangent function corresponding to the range $[-0.7616, +0.7616]$ can be assumed to be quasi linear (see Figure 5). The quasi-linearity used in preprocessing can be considered fine-tuning before the training step. It is important to understand that due to the limitation of instances in range $[-0.7616, +0.7616]$, the product of these values and the maximum values of the weights cannot reach ± 1, which will speed up any gradient-based algorithm. Also, saturation of the FNN at the beginning of the training process can be avoided in this way.

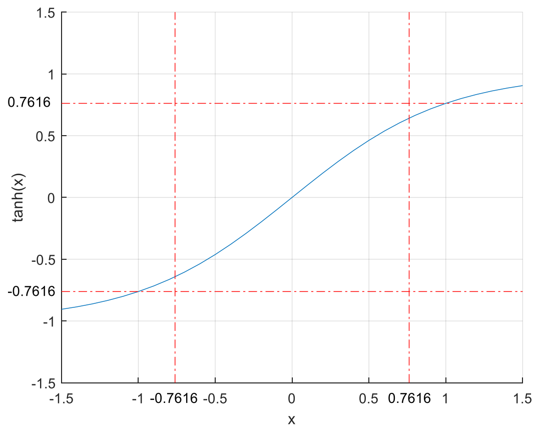

Figure 5. Quasi-linear part of hyperbolic tangent function.

The Min-Max normalization is given with the Formula (1).

$$x(k)_{normalized} = \frac{x(k) - \frac{x_{max}+x_{min}}{2}}{\frac{x_{max}-x_{min}}{2}}, \tag{1}$$

where $x(k)_{normalized} \in [-1, 1]$, x_{max} and x_{min} represent the normalized instance, maximum and minimum values, respectively. The hyperbolic tangent normalization (*NTH*) scales feature as follows (2):

$$x(k)_{NTH} = tanh(x(k)_{normalized}) \tag{2}$$

where $x(k)_{NTH}$ represents the *NTH*-normalized instance.

The main characteristic of *NTH* normalization is that the product of instance and weight chosen to range from -1 to 1 can never reach values ± 1. These characteristic further speeds up the training of the classifiers. This is due to the properties of the non-linear Levenberg–Marquardt algorithm, which combines the GD algorithm and the Gauss–Newton (GN) algorithm. The GD algorithm uses an iterative process based on the step length and a search direction determined by the negative of the gradient to minimize objective function. The Jacobian matrix (**J**) is used in the GN algorithm to simplify the calculation of the Hessian (**H**) matrix, because it is assumed that the error function is approximately quadratic close to the optimal solution [109]. The idea of LM algorithm is to transform nonlinear function $f : R \to R$ close to the current point m [110] and approximate the non-linear optimization problem to the extent that $\hat{f}(x) = f(x)$ and $x \approx m$. Consider the iterates $x^{(1)}, x^{(2)}, \ldots\ x^{(l)}$. The iterate $x^{(i+1)}$ represents the solution of the problem when $x^{(i+1)} \approx x^{(i)}$. It can be derived as in (3).

$$\begin{aligned} x^{(i+1)} = x^{(i)} - \left(\mathbf{H} + \beta^{(i)}\mathbf{I}\right)^{-1}\mathbf{J}^{\mathrm{T}} f\left(x^{(i)}\right) = \\ = x^{(i)} - \left(\mathbf{J}^{\mathrm{T}}\mathbf{J} + \beta^{(i)}\mathbf{I}\right)^{-1}\mathbf{J}^{\mathrm{T}} f\left(x^{(i)}\right),\ \beta^{(i)} > 0, \end{aligned} \tag{3}$$

where parameter $\beta^{(i)}$ represents an adjustable variable called the damping factor, **I** represents the identity matrix and Hessian matrix can be approximated so that $\mathbf{H} \approx \mathbf{J}^{\mathrm{T}}\mathbf{J}$. The parameter $\beta^{(i)}$ can be changed as follows: If $\beta^{(i)}$ is small, then $x^{(i+1)}$ is far from $x^{(i)}$, and the approximation is poor; i.e., $\beta^{(i)}$ ought to be truncated (typically $\beta^{(i+1)} = 0.5\beta^{(i)}$). Otherwise, $x^{(i+1)}$ is too close to $x^{(i)}$, which slows progress down and $\beta^{(i)}$ should be increased (typically, $\beta^{(i+1)} = 2\,\beta^{(i)}$). Additionally, when $\beta^{(i)} \to \infty$, then $\mathbf{H} + \beta^{(i)}\mathbf{I} \approx \mathbf{I}$, and the LM algorithm behaves like the GD algorithm. While $\beta^{(i)} \to 0$, the iterate $x^{(i)}$ is close to the optimal solution, and the LM algorithm behaves like the GN algorithm [61]. The change between the GD and GN algorithms is called the damping strategy [111]. The LM approximation and the damping strategy are described more detailed in our previous work [105,112]. The damping strategy is used here to speed up processing and prevent problems due to a very large or very small gradients. The main goal of the *NTH* normalization and dumping strategy approach is not only to speed up model evaluation but also to accelerate weight adjustment and avoid zigzag movements.

4. Experimental Results

To train, test, and compare the models in terms of feature selection and feature scaling, a MATLAB Classification learner is used. About 60 thousand instances from the Kyoto 2006+ dataset are used as a benchmark for the experiments. The fundamentals of the experiments are fully explained in our previous work [112]. In brief, the characteristics of the classifiers are as follows: (1) the maximum number of splits in DT classification is 20; (2) for k-NN and wk-NN models, $k = 10$, the distance measure is Euclidean, and weights are random numbers limited in range $[-1, +1]$; (3) Gaussian SVM with the One-vs-One multiclass method is applied, and data standardization is used to differentiate true and false; (4) the FNN with one hidden layer is used (nine inputs, nine hidden-layer neurons and one output neuron—all activation functions were *tanh*), the BP algorithm is based on mean squared errors (maximum number of iterations = 1000, gradient magnitude < 10^{-5}, validation checks = 6). The Kyoto 2006+ dataset is preprocessed to remove Not-a-Number (NaN) values from the features, and then ~57 thousand instances are used to train and test the models. Second, all irrelevant features—categorical, statistical, and features for additional analyses—are eliminated. The evaluation of the model was then left to nine numerical features. Anomalies are determined via the feature *Label*; if *Label* = 1, the network traffic is considered normal, if not (*Label* = 0), an anomaly is detected. Both *Min-Max* normalization and *NTH* normalization were used to classify the data, with similar results. The classifier comparison is carried out using the binary confusion matrix as a tool. The confusion matrix contains True Positives (*TP*), False Positives (*FP*), True Negatives (*TN*), and False Negatives (*FN*) [113]. Binary zero describes the negatives, while binary one describes the positives. These numbers are frequently used to describe the measurement performance of classification problems in the datasets where the true values are known [114]. Metrics calculated from the confusion matrix are enlisted in Table 5.

Table 5. Metrics based on confusion matrix.

Metric	Description
$Recall = \frac{TP}{TP+FN}$	Ratio of correctly detected positive class; useful when *FN* dominates *FP*.
$Precision = \frac{FP}{FP+TN}$	Measure of correctly predicted the target class [115]; useful when *FP* is of greater concern then *FN* [36,116].
$Acuracy = \frac{TN+TP}{TN+TP+FN+FP}$	Shows overall correctness of the prediction.
$F1 - score = \frac{2*Recall*Precision}{Recall+Precision}$	Harmonic mean of *Recall* and *Precision*.

The findings demonstrate that all models, with the exception of the DT classifier, have very accurate decision making. The DT model is one of the most efficient classifiers, but it is almost non-affected by feature scaling methodologies. Both kinds of nearest neighbor models take much longer to process than the other models. The *Min-Max* scaling prevents problems with very high or very low derivatives. This did not, however, have an impact on F1-score or accuracy. High-precision models are evident in the results, but the processing time problems remained. In the experiments based on *NTH* normalization, processing time for all classifiers was found to be significantly reduced. The results support the assumption that it is reasonable to use *NTH* normalization to speed up training. All models have a very high F1-score and accuracy (SVM is the exception). Table 6 displays the results.

Table 6. Accuracy, processing time, and F1-score of the classifiers.

	NTH			**Min-Max**		
Model	**Accuracy [%]**	**PT [s]**	**F1 [%]**	**Accuracy [%]**	**PT [s]**	**F1 [%]**
k-NN	99.3	56.1	99.01	99.41	103.2	99.2
wk-NN	99.4	56.3	99.16	99.58	105.3	99.29
SVM	99.1	26.9	98.69	99.17	43.4	98.89
DT	99.4	2.5	99.18	99.41	6.3	99.14
FNN	99.36	5	99.04	99.31	12	99.04

Min-Max normalization helps solve one of the problems with extremely small and very large derivatives that make training difficult. Except for the FNN, this normalization increases model properties in terms of processing time and F1-score. The results suggest that the issue of differing scales has no effect on model evaluation. The range of activation functions of input nodes is the key reason why this is not true for the FNN. If the number of inputs is greater than 3, non-*tanh* function in the hidden layer can become essentially saturated if weight updates are all positive or negative. Also, because of the very sharp gradient, zigzagging is prevented when the *tanh* activation function is used. When the range of $[-0.7616, +0.7616]$ is chosen, the quasi-linear part of the *tanh* accelerates the training. However, this method slightly reduces the accuracy of all models.

5. Observations, Trends, and Research Challenges

In general, research findings can be used to develop more accurate and efficient IDSs, which can help companies to protect their computer networks from malicious attacks. This article provides a comprehensive analysis of various ML-based models and their classification properties. Feature selection and feature scaling methods can help researchers choose the ML algorithms based on their available resources and application scenarios. This study reveals interesting trends in anomaly-based intrusion detection. First, the researchers develop and implement new filter methods to improve the classification process. At the same time, this study shows that the metric used to measure quality of the models is still based on the confusion matrix; there are very few new methods to determine the precision of the classification and the stability of the classifiers. Second, researchers worldwide use a series of (mostly simulated) datasets of network traffic. The influence of redundant or duplicated records on model training and testing results in low model accuracy and long processing time. Third, there are hardly any studies with practical implementation. In general, simulations do not provide a full insight into the practical results of detecting anomalies in real network traffic. An interesting aspect of the research would be the discovery of features that indicate which preprocessing and classification methods give the best results in detecting anomalies. Another aspect of the research would be to investigate the predictability and stability of the newly developed classification models. In addition, new strategies for feature selection and scaling should be proposed. Furthermore, the classification needs to be improved in terms of the classification predictability. Ideally, the features should be inexpensive to calculate to reduce processing time compared to trying different classification methods. In the future, the results of the analysis presented in this article will be used to identify new, improved models. The study is also extended to include other preprocessing methods related to both instance scaling and feature selection that were not within the scope of this research. Also, new research will look at hybrid classification to develop binary classifiers that enable more accurate decision making.

6. Conclusions

Classifying network behavior as normal or abnormal can be achieved relatively easy by identifying anomalies. The Kyoto 2006+ dataset, which includes approximately 90 million instances of real network data, is used in this study to evaluate five binary classifiers for anomaly detection. The dataset size is reduced via feature selection techniques used as a preprocessing stage in the classification process. Furthermore, the features are scaled with tangent-hyperbolic function. To speed up training, the dumping strategy is used. According to the results, the processing time is significantly shorter when compared to Min-Max normalization. Future research will use the findings of this study to identify a new search space to improve classification models. In addition, both feature selection and feature scaling will be further explored. Hybrid classifiers, which are expected to be more accurate with faster processing, will also be the subject of further research.

Author Contributions: Conceptualization, D.P.; methodology, D.P.; software, D.P.; validation, M.S. (Miomir Stanković), R.P. and I.V.; formal analysis, D.P. and M.S. (Miomir Stanković); investigation, D.P.; resources, M.S. (Miomir Stanković), G.M.S. and M.S. (Mitar Simić); data curation, D.P.; writing—original draft preparation, D.P.; writing—review and editing, M.S. (Miomir Stanković), R.P., I.V., G.M.S., G.O. and S.S.; visualization, D.P.; supervision, M.S. (Miomir Stanković), G.M.S., M.S. (Mitar Simić), G.O. and S.S.; project administration, G.M.S.; funding acquisition, M.S. (Miomir Stanković), G.M.S. and M.S. (Mitar Simić) All authors have read and agreed to the published version of the manuscript.

Funding: This work has partly received funding from Horizon Europe Framework Programme of European Commission, under grant agreement No. 101086348.

Data Availability Statement: New data were not created.

Acknowledgments: This research is a result of partly supported within the GaitREHub project which has received funding from the European Union's Horizon research and innovation programme.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Syrris, V.; Geneiatakis, D. On machine learning effectiveness for malware detection in Android OS. *J. Inf. Secur. Appl.* **2021**, *59*, 102794. [CrossRef]
2. Viegas, E.K.; Santin, A.O.; Oliviera, L.S. Toward a reliable anomaly-based intrusion detection in real-world environments. *Comput. Netw.* **2017**, *127*, 200–216. [CrossRef]
3. Aljawarneh, S.; Aldwairi, M.; Yassein, M.B. Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *J. Comput. Sci.* **2018**, *25*, 152–160. [CrossRef]
4. Ullah, I.; Mahmoud, Q.M. A filter-based feature selection model for anomaly-based intrusion detection systems. In Proceedings of the 2017 IEEE International Conference on Big Data, Boston, MA, USA, 11–14 December 2017; pp. 2151–2159. [CrossRef]
5. Vengatesan, K.; Kumar, A.; Naik, R.; Verma, D.K. Anomaly Based Novel Intrusion Detection System for Network Traffic Reduction. In Proceedings of the 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 30–31 August 2018; pp. 688–690. [CrossRef]
6. Aloraini, F.; Javed, A.; Rana, O.; Burnap, P. Adversarial machine learning in IoT from an insider point of view. *J. Inf. Secur. Appl.* **2022**, *70*, 103341. [CrossRef]
7. Pai, V.; Devidas, B.; Adesh, N.D. Comparative analysis of machine learning algorithms for intrusion detection. *IOP Conf. Ser. Mater. Sci. Eng.* **2021**, *1013*, 012038. [CrossRef]
8. Protic, D.; Stankovic, M. Detection of Anomalies in the Computer Network Behaviour. *Eur. J. Eng. Form. Sci.* **2020**, *4*, 7–13. [CrossRef]
9. Saranya, T.; Sridevi, S.; Deisy, C.; Chung, T.D.; Khan, M.K.A. Performance analysis of Machine Learning Algorithms in Intrusion Detection Systems: A review. *Procedia Comput. Sci.* **2020**, *171*, 1251–1260. [CrossRef]
10. Protic, D.; Stankovic, M. A Hybrid Model for Anomaly-Based Intrusion Detection in Complex Computer Networks. In Proceedings of the 2020 21st International Arab Conference on Information Technology (ACIT), Giza, Egypt, 28–30 November 2020; pp. 1–8. [CrossRef]
11. Protic, D.; Stankovic, M. Anomaly-Based Intrusion Detection: Feature Selection and Normalization Instance to the Machine Learning Model Accuracy. *Eur. J. Eng. Form. Sci.* **2018**, *1*, 43–48. [CrossRef]
12. Siddiqui, M.A.; Pak, W. Optimizing Filter-Based Feature Selection Method Flow for Intrusion Detection System. *Electronics* **2020**, *9*, 2114. [CrossRef]
13. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [CrossRef] [PubMed]
14. Ganapathy, S.; Kulothungan, K.; Muthurajkumar, S.; Vijayalakshimi, M.; Yogesh, P.; Kannan, A. Intelligent feature selection and classification techniques for intrusion detection in networks: A survey. *EURASIP J. Wirel. Commun. Netw.* **2013**, *2013*, 271. [CrossRef]

15. Gautam, R.K.S.; Doegar, E.A. An ensemble approach for intrusion detection system using machine learning algorithms. In Proceedings of the 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 11–12 January 2018; pp. 14–15.
16. Khonde, R.S.; Ulagamuthalvi, V. Ensemble-based semi-supervised learning approach for a distributed intrusion detection system. *J. Cyber Secur. Technol.* **2019**, *3*, 163–188. [CrossRef]
17. Ambusaidi, M.A.; He, X.; Nanda, P.; Tan, Y. Building an Intrusion Detection System Using a Filter-Based Feature Selection Algorithm. *IEEE Trans. Comput.* **2016**, *65*, 2986–2998. [CrossRef]
18. Najafabadi, M.M.; Khoshgoftaar, T.; Selyia, N. Evaluating Feature Selection Methods for Network Intrusion Detection with Kyoto Data. *Int. J. Reliab. Qual. Saf. Eng.* **2016**, *23*, 1650001. [CrossRef]
19. Protic, D. Review of KDD CUP '99, NSL-KDD and Kyoto 2006+ Datasets. *Vojnoteh. Glas. Mil. Tech. Cour.* **2018**, *66*, 580–595. [CrossRef]
20. Bohara, B.; Bhuyan, J.; Wu, F.; Ding, J. A Survey on the Use of Data Clustering for Intrusion Detection System in Cybersecurity. *Int. J. Netw. Secur. Appl.* **2020**, *12*, 1–18. [CrossRef]
21. Thakkar, A.; Lohiya, R. A Review of the Advancement in the Intrusion Detection Datasets. *Procedia Comput. Sci.* **2020**, *167*, 636–645. [CrossRef]
22. Khraisat, A.; Gondal, L.; Vamplew, P.; Kamruzzaman, J. Survey of intrusion detection systems: Techniques, datasets and challenges. *Cybersecurity* **2019**, *2*, 20. [CrossRef]
23. Ferryian, A.; Thamrin, A.H.; Takeda, K.; Murai, J. Generating Network Intrusion Detection Dataset Based on Real and Encrypted Synthetic Attack Traffic. *Appl. Sci.* **2021**, *11*, 7868. [CrossRef]
24. Serkani, E.; Gharaee, H.; Mohammadzadeh, N. Anomaly detection using SVM as classifier and DT for optimizing feature vectors. *ISeCure* **2019**, *11*, 159–171.
25. Mighan, S.N.; Kahani, M.S. A novel scalable intrusion detection system based on deep learning. *Int. J. Inf. Secur.* **2021**, *20*, 387–403. [CrossRef]
26. Soltani, M.; Siavoshani, M.J.; Jahangir, A.H. A content-based deep intrusion detection system. *Int. J. Inf. Secur.* **2021**, *21*, 547–562. [CrossRef]
27. Suman, C.; Tripathy, S.; Saha, S. Building an Effective Intrusion Detection Systems using Unsupervised Feature Selection in Multi-objective Optimization Framework. *arXiv* **2019**, arXiv:1905.06562.
28. Ruggieri, S. Complete Search for Feature Selection Decision Trees. *J. Mach. Learn. Res.* **2019**, *20*, 1–34.
29. Shmiilovici, A. Support vector machines. In *Data Mining and Knowledge Discovery Handbook*; Springer: Boston, MA, USA, 2005. [CrossRef]
30. Deris, A.M.; Zain, A.M.; Sallehudin, R. Overview of Support Vector Machine in Modeling Machining Performances. *Procedia Eng.* **2011**, *24*, 301–312. [CrossRef]
31. Halimaa, A.; Sundarakantham, K. Machine Learning Based Intrusion Detection System. In Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 23–25 April 2019; pp. 916–920. [CrossRef]
32. Bhati, B.S.; Rai, C.S. Analysis of Support Vector Machine-based Intrusion Detection Techniques. *Arab. J. Sci. Eng.* **2020**, *45*, 2371–2383. [CrossRef]
33. Nawi, N.M.; Atomi, W.H.; Rehman, M.Z. The Effect of Data Pre-processing on Optimizing Training on Artificial Neural Network. *Procedia Technol.* **2013**, *11*, 23–39. [CrossRef]
34. Jie, C.; Jiawei, L.; Shulin, W.; Sheng, Y. Feature Selection in Machine Learning: A New Perspective. *Neurocomputing* **2018**, *300*, 70–79. [CrossRef]
35. Ullah, F.; Babar, M.A. Architectural Tactics for Big Data Cybersecurity Analysis Systems: A Review. *J. Syst. Softw.* **2019**, *15*, 81–118. [CrossRef]
36. Luque, A.; Carrasco, A.; Martin, A.; de las Heras, A. The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognit.* **2019**, *91*, 216–239. [CrossRef]
37. Xie, M.; Hu, J.; Chang, E. Evaluating Host-Based Anomaly Detection Systems: Application of the Frequency-Based Algorithms to ADFA-LD. In *Network and System Security. NSS 2015. Lecture Notes in Computer Science 8792*; Au, M.H., Carminati, B., Kuo, J., Eds.; Springer: Cham, Switzerland, 2014. [CrossRef]
38. Borisniya, B.; Patel, D.R. Evaluation of Modified Vector Space Representation Using ADFA-LD and ADFA-WD Datasets. *J. Inf. Secur.* **2015**, *6*, 250. [CrossRef]
39. Vijayakumar, S.; Sannasi, G. Machine Learning Approach to Combat False Alarms in Wireless Intrusion Detection System. *Comput. Inf. Sci.* **2018**, *11*, 67. [CrossRef]
40. Proebstel, E.P. Characterizing and Improving Distributed Network-Based Intrusion Detection Systems (NIDS): Timestamp Synchronization and Sampled Traffic. Master's Thesis, University of California DAVIS, Davis, CA, USA, 2008.
41. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy, Madeira, Portugal, 22–24 January 2018; pp. 108–116.
42. Levy, J.L.; Khoshgoftaar, T.M. A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *J. Big Data* **2020**, *7*, 104. [CrossRef]

43. Lippmann, R.P.; Cunningham, R.K.; Fried, D.J.; Graf, I.; Kendal, K.R.; Webster, S.E.; Zissman, M.A. Results of DARPA 1998 Offline Intrusion Detection Evaluation. In Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX), Hilton Head, SC, USA, 25–27 January 2000; IEEE Computer Society Press: Los Alamitos, CA, USA, 2000.
44. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A. A Detailed Analysis of the KDD Cup '99 dataset. In Proceedings of the 2009 IEEE Symposium of Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6. [CrossRef]
45. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 Military Communications and Information Systems Conference, Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6. [CrossRef]
46. Song, J.; Takakura, H.; Okabe, Y.; Eto, M.; Inoue, D.; Nakao, K. Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation. In Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, Salzburg, Austria, 10–13 April 2011; pp. 29–36. [CrossRef]
47. Singh, R.; Kumar, H.; Singla, R.K. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. *Expert Syst. Appl.* **2015**, *42*, 8609–8624. [CrossRef]
48. SIGKDD-KDD Cup. KDD Cup 1999: Computer Network Intrusion Detection. 1999. Available online: https://www.kdd.org/kdd-cup/view/kdd-cup-1999/Tasks (accessed on 20 May 2016).
49. Park, K.; Song, Y.; Cheong, Y. Classification of Attack Types for Intrusion Detection Systems Using a Machine Learning Algorithm. In Proceedings of the 2018 IEEE 4th International Conference on Big Data Computing Service and Applications, Bamberg, Germany, 26–29 March 2018; pp. 282–286. [CrossRef]
50. Ting, K.M. Confusion matrix. In *Encyclopedia of Machine Learning*; Sammut, C.J., Webb, G.I., Eds.; Springer: Boston, MA, USA, 2011. [CrossRef]
51. Ring, M.; Wunderlich, S.; Scheuring, D.; Landes, D.; Hotho, A. A Survey of Network-based Intrusion Detection Data Sets. *arXiv* **2019**, arXiv:1903.02460. [CrossRef]
52. Demertzis, K. The Bro Intrusion Detection System. Machine Learning to Cyber Security. 2018. Available online: https://www.researchgate.net/publication/329217161_The_Bro_Intrusion_Detection_System (accessed on 27 May 2020).
53. McCarthy, R. Network Analysis with the Bro Security Monitor. 2014. Available online: https://www.admin-magazine.com/Archive/2014/24/Network-analysis-with-the-Bro-Network-Security-Monitor (accessed on 14 February 2019).
54. Papamartzivanos, D.; Marmol, F.G.; Kamblourakis, K. Introducing deep learning self-adaptive misuse network intrusion detection system. *IEEE Access* **2019**, *7*, 13546–13560. [CrossRef]
55. Sahu, A.; Mao, Y.; Davis, K.; Goulart, A.E. Data Processing and Model Selection for Machine Learning-based Network Intrusion-Detection. In Proceedings of the 2020 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR), Stevenson, WA, USA, 14 May 2020; pp. 1–6.
56. Weston, J.; Elisseff, A.; Schoelkopf, B.; Tipping, M. Use of the zero norm with linear models and kernel methods. *J. Mach. Learn. Res.* **2003**, *3*, 1439–1461.
57. Song, L.; Smola, A.; Gretton, A.; Borgwardt, K.; Bedo, J. Supervised feature selection via dependence estimation. In Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA, 20–24 June 2007; pp. 823–830. [CrossRef]
58. Dy, J.G.; Brodley, C.E. Feature selection for unsupervised learning. *J. Mach. Learn. Res.* **2005**, *5*, 845–889.
59. Mitra, P.; Murthy, C.A.; Pal, S. Unsupervised feature selection using feature similarity. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 301–312. [CrossRef]
60. Zhao, Z.; Liu, H. Semi-supervised feature selection via spectral analysis. In Proceedings of the SIAM International Conference on Data Mining, Minneapolis, MN, USA, 26–28 April 2007; pp. 641–646. [CrossRef]
61. Xu, Z.; Jin, R.; Ye, J.; Lyu, M.; King, I. Discriminative semi-supervised feature selection via manifold regularization. *IEEE Trans. Neural Netw.* **2010**, *21*, 1033–1047.
62. Porkodi, R. Comparison on filter-based feature selection algorithms: An overview. *Int. J. Innov. Res. Technol. Sci.* **2014**, *2*, 108–113.
63. Artur, A. Review the performance of the Bernoulli Naïve Bayes Classifier in Intrusion Detection Systems using Recursive Feature Elimination with Cross-validated Selection of the Best Number of Features. *Procedia Comput. Sci.* **2021**, *190*, 564–570. [CrossRef]
64. Osnaiye, O.; Ogundile, O.; Aina, F.; Peniola, A. Feature Selection for Intrusion Detection System in cluster-based heterogeneous wireless sensor networks. *Facta Univ. Ser. Electron. Energetics* **2019**, *32*, 315–330. [CrossRef]
65. Aamir, A.; Zaidi, S.M.A. DDoS Attack detection with feature engineering and machine learning, the framework and performance evaluation. *Int. J. Inf. Secur.* **2019**, *18*, 761–785. [CrossRef]
66. Khammassi, C.; Krichen, S. A GA-LR wrapper approach for feature selection in network intrusion detection. *Comput. Secur.* **2017**, *70*, 255–277. [CrossRef]
67. Umar, M.A.; Zhanfang, C.; Liu, Y. Network Intrusion Detection Using Wrapper-based Decision Tree for Feature Selection. In Proceedings of the 2020 International Conference on Internet Computing for Science and Engineering, Male, Maldives, 14–16 January 2020; pp. 5–13. [CrossRef]
68. Venkateswaran, N.; Umadevi, K. Hybridized Wrapper Filter Using Deep Neural Network for Intrusion Detection. *Comput. Syst. Sci. Eng.* **2022**, *42*, 1–14. [CrossRef]
69. Thakkar, A.; Lohiya, R.A. A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions. *Artif. Intell. Rev.* **2021**, *55*, 453–563. [CrossRef]

70. Almomani, O. A Feature Selection Model for Network Intrusion Detection System Based on PSO, GWO, FFA and GA Algorithms. *Symmetry* **2020**, *12*, 1046. [CrossRef]
71. Choudhury, A. What are Feature Selection Techniques in Machine Learning. Available online: https://analyticsindiamag.com/what-are-feature-selection-techniques-in-machine-learning/ (accessed on 21 May 2022).
72. Biswas, S.K.; Bordoloi, M.; Purkayastha, B. Review of Feature Selection and Classification using Neuro-Fuzzy Approaches. *Int. J. Appl. Evol. Comput.* **2016**, *7*, 28–44. [CrossRef]
73. Rosely, N.F.L.M.; Sallehuddin, R.; Zain, A.M. Overview Feature Selection Algorithm. *J. Phys. Conf. Ser.* **2019**, *1192*, 012068. [CrossRef]
74. Musheer, R.A.; Verma, C.K.; Srivastava, N. Dimension reduction methods for microarray data: A review. *AIMS Bioeng.* **2017**, *4*, 179–197. [CrossRef]
75. Ahmed, I.; Shin, H.; Hong, M. Fast Content-Based File Type Identification. In *Advances in Digital Forensics VII. Digital Forensics 2011*; Springer: Berlin/Heidelberg, Germany, 2011. [CrossRef]
76. Maza, S.; Touahria, M. Feature Selection Algorithms in Intrusion Detection System: A Survey. *KSII Trans. Internet Inf. Syst.* **2018**, *12*, 5079–5099. [CrossRef]
77. Zhao, F.; Zhao, J.; Niu, X.; Luo, S.; Xin, Y. A Filter Feature Selection Algorithm Based on Mutual Information for Intrusion Detection. *Appl. Sci.* **2018**, *8*, 1535. [CrossRef]
78. Seok, J.; Seon Kang, Y. Mutual information between discrete variables with many categories using recursive adaptive partitioning. *Sci. Rep.* **2018**, *5*, 10981. [CrossRef] [PubMed]
79. Battiti, R. Using mutual information for selecting features in supervised neural net learning. *IEEE Trans. Neural Netw.* **1994**, *5*, 537–550. [CrossRef] [PubMed]
80. Peng, H.; Long, F.; Ding, C. Feature Selection based on Mutual Information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1226–1238. [CrossRef]
81. Kwak, N.; Choi, C.H. Input feature selection for classification problems. *IEEE Trans. Neural Netw.* **2002**, *13*, 143–159. [CrossRef]
82. Novovicova, J.; Somol, P.; Haindl, M.; Pudil, P. *Conditional Mutual Information Based Feature Selection for Classification Tasks*; Springer: Berlin/Heidelberg, Germany, 2007.
83. Amiri, F.; Yousefi, M.M.R.; Lucas, C.; Shakery, A.; Yazani, N. Mutual information-based feature selection for intrusion detection systems. *J. Netw. Comput. Appl.* **2011**, *34*, 1184–1195.
84. Bindumadhavi, P.; Sandhya Rani, B. Building an intrusion detection system using a filter-based feature selection algorithm. *Int. J. Innov. Res. Stud.* **2017**, *7*, 24–35.
85. Alanazi, R.; Aljuhani, A. Anomaly Detection for Industrial Internet of Things Cyberattacks. *Comput. Syst. Sci. Eng.* **2022**, *44*, 2361–2378. [CrossRef]
86. Mitchel, T. *Machine Learning*; McGraw Hill: New York, NY, USA, 1997.
87. Schroeber, P.; Boer, C.; Schwarte, L. Correlation coefficients: Appropriate use and interpretation. *Anesth. Analg.* **2018**, *126*, 1763–1768. [CrossRef]
88. Kalavadekar, P.N.; Sane, S.S. Building an Effective Intrusion Detection System using combined Signature and Anomaly Detection Techniques. *Int. J. Innov. Technol. Explor. Eng.* **2019**, *8*, 429–435. [CrossRef]
89. Perez, D.; Alonso, S.; Moran, A.; Prada, M.A.; Fuentes, J.J.; Domingez, M. Comparison of Network Intrusion Detection Performance Using Feature Representation. In *Engineering Applications of Neural Networks. EANN 2019. Communications in Computer and Information Science (1000)*; Macintyre, J., Illadis, L., Maglogoiannis, I., Jayne, C., Eds.; Springer: Cham, Switzerland, 2019. [CrossRef]
90. Salo, F. Towards Efficient Intrusion Detection Using Hybrid Data Mining Techniques. Ph.D. Thesis, Western University, London, ON, Canada, 2019.
91. Ahmad, T.; Aziz, M.N. Data preprocessing and feature selection for machine learning intrusion detection systems. *ICIC Int. Express Lett.* **2019**, *13*, 93–101.
92. Patro, S.G.K.; Sabu, K.K. Normalization: A Preprocessing Stage. 2015. Available online: https://arxiv.org/ftp/arxiv/papers/1503/1503.06462.pdf (accessed on 19 September 2022).
93. Panda, S.K.; Jana, P.K. An Efficient Task Scheduling Algorithm for Heterogeneous Multi-cloud Environment. In Proceedings of the 3rd International Conference on Advances in Computing, Communications & Informatics (ICACCI), Delhi, India, 24–27 September 2014; pp. 1204–1209.
94. Deb, K.; Pratap, A.; Agarwal, S. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
95. Mienye, I.D.; Sun, Y.; Wang, Z. Prediction performance of improved decision tree-based algorithms: A review. *Procedia Manuf.* **2019**, *35*, 698–703. [CrossRef]
96. Singh, S.; Banerjee, S. Machine Learning Mechanisms for Network Anomaly Detection System: A Review. In Proceedings of the International Conference on Communication and Signal Processing (ICCSP), Chennai, India, 28–30 July 2020; pp. 0976–0980. [CrossRef]
97. Mohammadi, S.; Mirvaziri, H.; Ahaeea, M.G.; Karimipour, H. Cyber intrusion detection by the combined feature selection algorithm. *J. Inf. Secur. Appl.* **2019**, *44*, 80–88. [CrossRef]

98. Keserwani, P.K.; Govil, M.C.; Pilli, E.S. An effective NIDS framework based on a comprehensive survey of feature optimization and classification techniques. *Neural Comput. Appl.* **2023**, *35*, 4993–5013. [CrossRef]
99. Mishra, A.; Cheng, A.; Zhang, Y. Intrusion Detection Using Principal Component Analysis and Support Vector Machines. In Proceedings of the 2020 IEEE 16th International Conference on Control and Automation (ICCA), Singapore, 9–11 October 2020; pp. 907–912.
100. Vallejo-Huanga, D.; Ambuludi, M.; Morillo, P. Empirical Exploration Machine Learning Techniques for Detection of Anomalies Based on NIDS. *IEEE Lat. Am. Trans.* **2021**, *19*, 772–779. [CrossRef]
101. Seelammal, C.; Devi, K.V. Multi-criteria decision support for feature selection in network anomaly detection system. *Int. J. Data Anal. Tech. Strateg.* **2018**, *10*, 334–350. [CrossRef]
102. Wang, S.; Cai, C.X.; Tseng, Y.W.; Lin, K.S.M. Feature Selection for Malicious Traffic Detection with Machine Learning. In Proceedings of the 2020 International Computer Symposium (ICS), Tainan, Taiwan, 17–19 December 2020; pp. 414–419. [CrossRef]
103. Idhammad, M.; Afdel, K.; Belouch, M. Semi-supervised machine learning approach for DDoS detection. *Appl. Intell.* **2018**, *48*, 3193–3208. [CrossRef]
104. Hamid, Y.; Balasaraswahti, V.R.; Journaux, L.; Sugumaran, M. Benchmark Dataset for Network Intrusion Detection: A review. *Int. J. Netw. Secur.* **2018**, *20*, 645–654.
105. Protic, D.; Stankovic, M.; Antic, V. WK-FNN design for detection of anomalies in the computer network traffic. *Facta Univ. Ser. Electron. Eng.* **2022**, *35*, 269–282. [CrossRef]
106. Thakkar, A.; Lohiya, R. Attack classification using feature selection techniques: A comparative study. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *12*, 1249–1266. [CrossRef]
107. Rahman, A.; Islam, Z. AWST: A novel attribute weight selection technique for data clustering. In Proceedings of the 13th Australasian Data Mining Conference, Sidney, Australia, 8–9 August 2015; pp. 51–58.
108. Rahman, M.A.; Islam, M. CRUDAW: A novel fuzzy technique for clustering records following user-defined attribute weights. In Proceedings of the 10th Australasian Data Mining Conference, Sydney, Australia, 5–7 December 2012; Volume 134, pp. 27–42.
109. Torres, J.M.; Comesana, C.I.; Garcia-Nieto, P. Review: Machine learning techniques applied to cybersecurity. *Int. J. Mach. Learn. Cybern.* **2019**, *10*, 2823–2836. [CrossRef]
110. Wang, Z.; Cai, L.; Su, Y.; Peng, Z. An inexact affine scaling Levenberg-Marquardt method under local error bound conditions. *Acta Math. Appl. Sin. Engl.* **2019**, *35*, 830–844. [CrossRef]
111. Xue, X.; Zhang, K.; Tan, K.; Feng, L.; Wang, L.; Chen, G.; Zhao, X.; Zhang, L.; Yao, J. Affine transformation-enhanced multi factorial optimization for heterogeneous problems. *IEEE Trans. Cybern.* **2020**, *52*, 6217–6231. [CrossRef] [PubMed]
112. Protic, D.; Gaur, L.; Stankovic, M.; Rahman, M.A. Cybersecurity in smart cities: Detection of opposing decisions of anomalies in the computer network behavior. *Electronics* **2022**, *11*, 3718. [CrossRef]
113. Singh, P.; Singh, N.; Kant Singh, K.; Singh, A. Chapter 5—Diagnosing of disease using machine learning. In *Machine Learning and the Internet of Medical Thing in Healthcare*; Academic Press: Cambridge, MA, USA, 2021; pp. 89–111.
114. Kulkarni, A.; Chong, D.; Bataresh, F.A. Data democracy. In *Nexus of Artificial Intelligence, Software, Development, and Knowledge Engineering*; Academic Press: Cambridge, MA, USA, 2020; pp. 83–106.
115. Tyagi, N. What is Confusion Matrix? 2021. Available online: https://www.analyticssteps.com/blogs/what-confusion-matrix (accessed on 30 March 2022).
116. Split Software: False Positive Rate. 2020. Available online: https://www.split.io/glossary/false-positive-rate/ (accessed on 27 August 2021).