Annals of Telecommunications

A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms

--Manuscript Draft--

Manuscript Number:	ANTE-D-21-00155R1
Full Title:	A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms
Article Type:	Open Topics
Corresponding Author:	Nicollas Rodrigues de Oliveira, M.Sc Universidade Federal Fluminense Niteroi, RJ BRAZIL
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Universidade Federal Fluminense
Corresponding Author's Secondary Institution:	
First Author:	João Vitor Valle Silva
First Author Secondary Information:	
Order of Authors:	João Vitor Valle Silva
	Nicollas Rodrigues de Oliveira, M.Sc
	Dianne Scherly Varela de Medeiros
	Martin Esteban Andreoni Lopez
	Diogo Menezes Ferrazani Mattos
Order of Authors Secondary Information:	
Funding Information:	Conselho Nacional de Desenvolvimento Científico e Tecnológico — Not applicable Coordenação de Aperfeiçoamento de Pessoal de Nível Superior — Not applicable Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro — Not applicable Fundação de Amparo à Pesquisa do Estado de São Paulo (2018/23062-5) — Not applicable Rede Nacional de Ensino e Pesquisa — Mr. João Vitor Valle Silva
Abstract:	Machine Learning mechanisms for network intrusion detection systems lack accurate evaluation, comparison, and deployment due to the scarcity of well-constructed datasets. In this paper, we propose a statistical analysis of the features contained in four highly-used security datasets. We conclude that the analyzed datasets should not be used as a benchmark for creating novel anomaly-based mechanisms for intrusion detection systems. The analyzed datasets introduce a biased classification since features are over-correlated, and most of the features are capable of making a complete distinction between normal and attack flows. Our proposed methodology analyzes the correlation among features instead of checking for redundant values or data imbalance. The results align with the performance of three machine learning techniques. We show that biased classification occurs due to a significant difference between attack and normal data. The syntactically generated features are statistically different between normal and attack classes, which implies overfitting in the machine learning approaches.
Response to Reviewers:	Manuscript ID: ANTE-D-21-00155 A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training

Machine Learning Mechanisms

Dear Editor and Reviewers,

We greatly appreciate the effort and time spent on our manuscript. We would like to thank all the reviewers for their suggestions. After all of the reviewer's valuable remarks, we are confident that the paper is in better shape and hopefully ready for publication.

Please consider the revised version of our manuscript entitled "A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms" for publication in the Annals of Telecommunications journal. In this revision, we address the reviewers' comments, highlighting our changes with a blue font in the manuscript. In our response below, we provide specific answers to the reviewers' comments and concerns.

Best regards,

João Vitor Valle Silva,
Nicollas Rodrigues de Oliveira,
Martin Andreoni Lopez,
Dianne Scherly Varela de Medeiros,
Diogo Menezes Ferrazani Mattos

Reviewer reports

Reviewer #2:

Overall, my conclusion from reading the manuscript is that the results tell a different story from the abstract and introduction. The authors indicate that the datasets lead to biased results. Then, they perform some analysis of feature correlation, and remove this correlation using feature selection techniques. However, at the end, the outcome of the models using three feature selection techniques is mostly similar to the outcome of a model trained using all the features. This result, in my point of view, contradicts the argument of the authors.

Response to Reviewers: We thank the reviewer for the attentive read of our paper, but we kindly disagree with the reviewer's statement since the similarity between the results with and without the feature selection procedure does not nullify the bias argument. Indeed, such results corroborate the biased generation of the analyzed datasets since, regardless of the selected features, all algorithms managed to substantially differentiate regular traffic from attack traffic.

Another point worth commenting is that the test methodology requires improvement. There are a number of parameters of the algorithms that should have been provided on the text. I would like to see more details on the chosen hyperparameters, as well as the choice of algorithms.

Response to Reviewers: As requested, we insert Tables 2-4 (new ones) containing more information about the hyperparameter values configured in the Logistic Regression, Decision Tree, and Random Forest algorithms, respectively. We also highlight that the hyperparameters choice was performed as a greedy optimization problem. We performed a greedy search to optimize the analyzed classifiers. However, optimizing classifiers performance is not the scope of the paper.

"Data not related to network attacks are removed at this stage, selecting only DoS or probing attack classes" Please explain why this was done. I assume that a real IDS would also need to classify traffic as normal, so it seems odd to remove normal traffic.

Response to Reviewers: We emphasize that the removal procedure was performed to

discard attacks on layers higher than the transport layer, because the paper scope is on network-related attacks and not on the application-level attacks. The removal procedure refers to attacks other than DOS or probing, not normal traffic. The normal-behaved traffic was kept in the analyzed datasets. To avoid this misinterpretation, we rewrite this snippet clarifying what data is actually kept after the preprocessing stage. “Data not related to network attacks are removed at this stage, maintaining only data related to normal traffic and DoS or probing attacks.”

Tables 2-4 need much more explanation. They provide features without explaining what they are. Also, they do not provide numeric values for the feature importance.

Response to Reviewers: We thank the reviewer for the advice. We removed that Tables that did not aggregate useful information to the paper.

Table 6 considers very simple classifiers. Also, the authors do not explain why they chose these types of classifiers. As an example, I miss more modern techniques such as a deep neural network. Why should someone use decision trees, logistic regression and random forest only as state of the art? There are a number of models that are more popular, including the algorithms based on trees, such as XGBoost.

My proposal for the authors is to show results for one algorithm of each class, e.g. dimensionality reduction, neural networks, tree-based algorithms, etc.

Response to Reviewers: We stress that the focus of the paper is not on evaluating classifiers’ performance, but we focus on testing the existence of bias on well-known security datasets. Hence, we chose to deploy the most common classifiers as they perform accurately for the selected dataset. As suggested by the reviewer, we enriched the comparative analysis by adding results from a tree-based algorithm, XGBoost. The hyperparameters and the results of applying this algorithm to the different datasets are shown in Tables 5 and 7-10, respectively. The inclusion of the algorithm in the analysis was accompanied by a brief explanation of it in Section 3, as seen in the text below. “Although equally decision tree-based, the XGBoost algorithm differs from Random Forest in two main aspects, in which the first is the decision tree creation process. In practice, while Random Forest trains each tree independently and using random samples, XGBoost applies a sequential ensemble technique that builds each tree in turn. In this approach, each data value is weighted according to its probability of being selected by a decision tree for further analysis. The second main difference relies on when the results are combined. Random Forest algorithm traditionally performs an averaging or majority voting at the end of the process, while XGBoost combines results along the way. Such features ensure that XGBoost, when properly tuned, performs better than Random Forest.”

Another important issue on the manuscript is that it lacks information about the hyperparameters used on each model. Furthermore, the authors should have commented how they tackled the issue of hyperparameter optimization.

Response to Reviewers: As mentioned previously, information about the hyperparameters used in the algorithms is expressed in Tables 2-4. It is noteworthy that such hyperparameters are derived from a simple greedy-search process since we intend to demonstrate that such datasets are biased even in the worst case of choice.

The authors do not mention how they split the existing datasets into train-test-validate sets. As an example, the CICBotnet 2014 has only train and test sets, so the authors had to perform their own split for validation.

Response to Reviewers: To clarify the percentage of split adopted, the excerpt below was added in Section 4. Due to the high accuracy, we did not identify the need for a validation step.

“After the removal procedure, each dataset is split with 70% for training and the

remainder for testing.”

Section 6 mentions the results for feature selection, however I did not find in this section themention as to which features were selected? Was it a top-X approach? If so, how did the authors choose the value of X?

Response to Reviewers: In the feature selection stage, we opted for the top-15 approach for each method applied. We choose the 15 most important features as a trade-off between data representation and useful information. Thus, 15 features against all features were necessary and sufficient to show that feature selection is not a confusing factor for the conclusion that the datasets introduce a biased assumption that the classifiers are accurate. Given the analysis of 4 datasets with distinct features, the inclusion of all selected characteristics would occupy an excessive amount of space and would be of little use. Even so, to clarify this issue, we include the excerpt below.

“It is noteworthy that the executed algorithm used the 15th most important feature returned by each feature selection method. This choice was based on the trade-off between data representation and useful information.”

Overall, the results on tables 6-9 are very similar, no matter which method for feature selection, or if feature selection was not used whatsoever. My personal conclusion is that feature selection does not impact the final result. Also, it shows no real difference over any of the feature selection techniques.

Response to Reviewers: We agree with the reviewer. Indeed, the paper aims to show that feature selection has low impact on the final accuracy of classifiers, since the datasets’ construction processes are biased. The convergence of results for similar and high values is another indication that the datasets used, known as a reference in the IDS field, have bias problems. We demonstrated that even applying untuned algorithms, it was possible to achieve high accuracy, precision, and F1-score values regardless of the feature selection method adopted.

Reviewer #3:

The first objective of the paper is to propose a statistical analysis of machine learning datasets for network security. This approach can be considered as an alternative for creating Intrusion Detection Systems. There are several contributions in this paper. First, the statistical analysis of four network security datasets; second, determine the most important shortcoming of the available datasets and finally, evaluate the quality of available datasets for machine learning. For this last point the authors show the weaknesses of the current datasets.

So, the authors deal with the reliability when applying different security datasets: NSL-KDD, the UNSW-NB15, CICIDS 2017, and CICBotnet 2014. The objective is to evaluate Intrusion Detection Systems. They analyze the statistical properties of dataset features. They identified some dataset flaws that introduce bias when applying machine learning classification models. The results show some important variances between the different approaches and methodologies to model the data for predicting the attacks. Then, they generalize the results to handle real-time data. Moreover, they show that running against real-world traffic should consider other datasets to be validated.
The study carried out is extremely important and shows that the use of machine learning techniques can give rise to highly biased results depending on the dataset. In addition, there are more and more attacks against datasets themselves by producing false datasets that lead to results contrary to what the user wants. In the same way, datasets can be manipulated to greatly falsify the results.
This paper is important by evaluating the models' performance and it is shown that a

	higher accuracy is obtained when comparing the results of different works addressing the same problem. The authors conclude that the Random Forest model is the most efficient technique for detecting the attacks. As a conclusion, the paper is well written and may be accepted for publication in the Annals of Telecommunications after improving the paper according to the reviewers remarks Response to Reviewers: We thank the reviewer for recognizing the contribution of the paper.

Noname manuscript No.
(will be inserted by the editor)

A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms

João Vitor V. Silva · Nicollas R. de Oliveira · Dianne S. V. Medeiros · Martin Andreoni Lopez · Diogo M. F. Mattos

Received: date / Accepted: date

Abstract Machine Learning mechanisms for network intrusion detection systems lack accurate evaluation, comparison, and deployment due to the scarcity of well-constructed datasets. In this paper, we propose a statistical analysis of the features contained in four highly-used security datasets. We conclude that the analyzed datasets should not be used as a benchmark for creating novel anomaly-based mechanisms for intrusion detection systems. The analyzed datasets introduce a biased classification since features are over-correlated, and most of the features are capable of making a complete distinction between normal and attack flows. Our proposed methodology analyzes the correlation among features instead of checking for redundant values or data imbalance. The results align with the performance of three machine learning techniques. We show that biased classification occurs due to a significant difference between attack and normal data. The syntactically generated features are statistically different between normal and attack classes, which implies overfitting in the machine learning approaches.

Keywords Machine Learning · Statistics · Network Security · Hypothesis Testing

João Vitor V. Silva · Nicollas R. de Oliveira · Dianne S. V. Medeiros · Diogo M. F. Mattos
MídiaCom/PPGEET/TET/IC
Universidade Federal Fluminense (UFF)
Niterói – RJ – Brazil
Tel.: +55(21)3674-7302
E-mail: {joao_valle, nicollas_rodrigues, diannescherly, diogo_mattos}@id.uff.br

Martin Andreoni Lopez
Technology Innovation Institute
P.O. Box 9639, Masdar City
Abu Dhabi, UAE
E-mail: martin@ssrc.tii.ae

1 Introduction

Intrusion Detection Systems (IDS) aim to monitor and analyze network traffic and automatically discriminate possible threats from normal user behavior [1,2,3]. Network operators rely on these systems as the main pillar of their security tools to avoid disruption of network services and create secure environments for their applications [4]. Intrusion Detection Systems are rule or anomaly-based. Rule-based IDSes depend on a set of policies to identify threats in the network. On the other hand, the anomaly-based approach infers the system's normal behavior and classifies as anomaly all the traffic differing from the inferred normal behavior.

A recent report from *Stott and May* company unveils that IDSes based on machine learning models are an actual trend in securing networks. Almost half of the cybersecurity employees believe that machine learning is a timely solution for security problems since a critical problem is to deploy more than 50 different tools for performing security tasks [1]. Although machine learning-based IDSes may either deploy rule or anomaly-based approaches, both depend on a large dataset as input to train the machine learning model. In this context, the KDD'99, and its variations, is a classical dataset and is still one of the most used datasets by the network security community. Although KDD'99 has known flaws and some alternatives are available in the literature, it is still considered a starting point for many new approaches and proposals of IDSes.

This paper proposes a statistical analysis of network security-intended datasets, which are considered an alternative for creating Intrusion Detection Systems. The statistical analysis considers datasets that aim to surpass the inadequacy of previous KDD-99 and NSL-KDD. We analyze and compare CIC-IDS 2017, UNSW-NB15, and CIC-Botnet 2014 [5,6,7] datasets. Nevertheless, we show that these datasets still contain pitfalls. Our analysis evaluates the data adequacy to the classification problem statement and its overfitting potential. The datasets' intrinsic characteristics are the main focus of the analysis, such as comparing the correlation between the variables and the use of hypothesis tests to verify the statistical relevance of each variable. The analysis is helpful to understand the feasibility of some obtained results. We further discuss why the models have accuracy above the expected threshold and why the classifiers' performance is close to the ideal.

The main contributions of this paper are: i) the statistical analysis of four network security datasets; ii) the highlighting of these datasets' most important shortcomings; and iii) an evaluation of the main datasets through machine learning methods showing the introduction of bias in the models.

The rest of the paper is organized as follows. Section 2.1 explains the motivation and main characteristics of security datasets. Section 3 introduces the machine learning techniques. Section 4 explains the methodology we employ for the datasets' analysis. In Section 5, we statistically analyze the main se-

[1] Available at `https://resources.stottandmay.com/hubfs/Research/Cyber\%20Security\%20in\%20Focus\%202020_web-2.pdf`.

curity datasets. Section 6 presents the results and discusses the performance of the machine learning models. Section 7 discusses the related work. Finally, Section 8 concludes the paper.

2 Security Datasets

Intrusion Detection Systems relate to monitoring and analyzing the events that occur in a computer system or a network aiming to find traces of intrusions or threats [8]. An IDS can be implemented as software that runs on top of hardware dedicated to network analysis [2,3]. There are different approaches to implement an IDS [8,9,10,11], which discriminate into three categories: signature-based detection, anomaly-based detection, or protocol dynamic analysis-based systems. A signature-based detection system aims to find patterns or traces that correspond to a threat, or a known attack, comparing known patterns towards identifying a possible intruder. An anomaly-based detection system seeks to distinguish a malicious from a normal behavior. It typically tracks the network connections, hosts, and users' activities. Besides, a dynamic protocol analysis-based system achieves similar results as anomaly-based systems. However, it performs a behavioral analysis inside the protocol layers, which needs access to generic profiles for each protocol from a development standpoint.

The more prevalent approaches are anomaly detection, and signature detection [12,9,10]. However, other criteria differently categorize IDS approaches [8]. According to Liao *et al.*, there are five types of IDS approaches: statistical, pattern-based, rule-based, state-based, or heuristic approaches. Statistical approaches rely on statistical metrics, such as mean and standard deviation. Pattern-based approaches aim to recognize attacks by comparing network traces against known attack traces. Model-based approaches consider states comparison between different traces using a state machine that depends on *if-then-else* logic to recognize patterns. An approach based on heuristics inspires biological concepts and uses artificial intelligence to determine whether an attack occurs.

2.1 The KDD'99 Dataset and its Variants

In 1997, the Defense Advanced Research Projects Agency (DARPA), the North-American public agency responsible for ensuring national security against cyber-attacks and cyber-terrorism, and Lincoln Labs at Massachusetts Institute of Technology (MIT) developed a program towards assessing the research on network security. They created a dataset [13] that includes the simulation of a series of intrusion techniques in a military network. The network operated according to legitimate behavior, but it suffered various attack attempts for the sake of evaluating the IDS capabilities. KDD'99 dataset [14] is a variant of the original DARPA dataset, whose development focused on the data mining competition "Knowledge Discovery in Databases". KDD'99 dataset counts with approximately 5 million samples, which are traces of the

network raw traffic. Each sample in the dataset is a flow that categorizes as *normal* or any other specific attack type. The simulated attacks are Denial-of-Service (DoS), User-to-Root, Remote-to-Local, and Probing attack. Attacks are labeled and distributed according to Table 1.

Table 1: Attack Types and Respective Classes

Class	Attack Type
DOS	back, land, neptune, pod, smurf, teardrop
U2R	buffer overflow, loadmodule, perl, rootkit
R2L	ftp write, guess password, imap, multihop, phf, spy, warezclient, warezmaster
Probe	ipsweep, nmap, portsweep, satan

In this work, we deploy NSL-KDD dataset [2], which aims to mitigate possible inconsistencies of the KDD'99 dataset, to provide a more realistic dataset, and to assure more realistic prediction results [15]. Hence, the NSL-KDD is an improvement of the original KDD'99 dataset. Tavallaee *et al.* describe a series of misconceptions inherent to the KDD'99 dataset, and propose splitting the data into two datasets, KDD Train+ and KDD Test+ [15]. One of the key advantages of these new datasets is removing redundant samples in train and test sets. In this way, the trained classifiers tend not to be biased, and the performance of the learners tends not to be biased by methods that have better performance on more frequent records, such as Support Vector Machine (SVM).

The number of attacks reduced 93.32% in the train dataset, and 88.26% in the test dataset. However, the reduction in the number of attacks does not impact the models' overall accuracy and improves the processing time, as model training becomes faster than using the original dataset.

2.2 The UNSW-NB15 Dataset

Moustafa *et al.* developed a comprehensive dataset to evaluate Network-based Intrusion Detection Systems, intending to replace the outdated KDD-99 and NSL-KDD datasets [16]. The UNSW-NB15 [6] is the result of a challenge for the cybersecurity research group at the Australian Centre for Cyber Security. The raw network traffic flows of the UNSW-NB 15 dataset were created using the IXIA PerfectStorm tool, generating a hybrid of real modern normal activities and synthetic contemporary attack behaviors.

Using the Tcpdump tool to capture about 100 GB of the raw traffic (e.g., Pcap files), the dataset contains nine types of attacks, including Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. Argus and Bro-IDS tools are used, and twelve algorithms are developed to generate 49 features with the class label.

[2] Available at https://www.unb.ca/cic/datasets/nsl.html.

2.3 CICIDS 2017 Dataset

Sharafaldin *et al.* develop the CICIDS 2017 dataset [5] that contains normal traffic and common attacks, resembling real-world network traffic [17]. To generate realistic traffic data, the B-Profile System [17] was used to profile the abstract behavior of human interactions and generate naturalistic benign background traffic. In the dataset, abstract behaviors were generated for 25 users based on HTTP, HTTPS, FTP, SSH, and email protocols. The implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Hearthbleed, Web Attack, Infiltration, Botnet, and Distributed Denial of Service (DDoS).

The dataset creation took place in a testbed environment, and the testbed implements two networks, namely Attack-Network, and Victim-Network. The Victim-Network is a highly secure infrastructure composed of a firewall, router, switches, and a combination of the most common operating systems and an agent to provide each host's benign behavior. The Attack-Network is a separate infrastructure composed of a router, a switch, and hosts with public IPs and different operating systems for running attack scenarios.

2.4 CICBotnet 2014 Dataset

Beigi *et al.* introduces the CICBotnet Dataset [7] that addresses three primary needs in botnet detection approaches: (i) the low generality of data, (ii) the lack of realism, and (iii) the representativeness of network traffic tracking [18]. Faced with these barriers, the CICBotnet Dataset merges three other datasets in a non-overlapping way. Since there is a wide range of IP addresses in these datasets, the merging process first passes through a mapping. Botnet IPs are correlated with hosts outside the current network using a packet generator. Subsequently, malicious and benign traffic were replayed using TCPReplay [3] and captured by TCPdump as a one-piece dataset. In its final version, CICBotnet Dataset comprises a training and test subset containing 7 and 16 types of botnets distributed in the malicious portion, respectively.

Regarding the CICBotnet's test subset, after collected and labeled adequately as malicious or benign traffic, all data traffic is submitted to flowtbag [4], an open-source software whose purpose is extracting 45 flow features from a given capture file. Among these features, there is information about flow tuple (IP addresses, ports, and transport protocol), packets and byte in forward and backward directions, packets statistics in forward and backward directions, the time between packets in forward and backward directions, flow time statistics, subflow packets and bytes in forward and backward directions, TCP flags, bytes in headers, and type of service [3].

[3] Available at https://tcpreplay.appneta.com/.
[4] Available at https://github.com/DanielArndt/flowtbag.

3 Machine Learning Models

Machine learning techniques are commonly divided into supervised and unsupervised learning. In supervised learning algorithms, the training set contains labeled samples, i.e., there is a dataset in which data are categorized with the correct target class *a priori* to train the classifier [19]. A supervised algorithm learns while reading and processing the labeled data and then adjusts statistical model parameters to predict unseen and non-classified data. Examples of supervised learning algorithms are Classification and Regression models. In unsupervised learning, there is a lack of previous knowledge about the dataset. Hence, the model searches for patterns on the data to extract any information about the dataset. Compared to supervised learning, unsupervised learning algorithms perform a more complex task, and the results tend to be more unstable. Examples of unsupervised learning algorithms are Clusterization and Association Rules algorithms. In this paper, we consider three supervised learning algorithms as follows.

Tree-based classifiers, also called Classification and Regression Trees, or **Decision Trees**, are effective and popular for discriminating network traffic. Decision trees and variations, Boosted Trees, and Random Forests classifiers make the baseline for the most advanced predictive models and are widely used in data science applied to network traffic [2]. Tree-based algorithms start the classification in a root node. The data split into two nodes; each node splits again to generate other nodes, building the tree structure. The tree grows continuously, without restrictions, until it reaches its maximum size, i.e., all the splitting rules have been covered. The criteria for splitting nodes is usually the entropy of features. Decision Trees have two techniques to avoid overfitting. The first technique consists of limiting the maximum size of the tree, reducing its complexity. The second technique defines criteria to split a node, which analyzes whether a leaf has enough size, or splitting it introduces a significant reduction of the Gini Impurity. This metric quantify how much a feature contributes to enhancing nodes' impurity.

Random Forest relies on bagging in decision trees, which introduces a significant advantage. Besides sampling data, it also samples features. Bagging, i.e., a bootstrap aggregator, is a basic grouping algorithm, which, instead of adjusting various models to the same data, adjusts each model to a bootstrapped sample. In each stage, the choice for a data feature is limited to a random subset of features. Compared to the Decision Tree algorithm, a basic Random Forest algorithm adds two more steps, bagging and bootstrap sampling of the features at each division. Random Forest thrives when a model is built for data with many samples and features. It automatically determines what predictors are significant and finds a complex relationship between them. There are two ways to measure the feature importance in Random Forest. The first way is to reduce the model's precision and, thus, increases the entropy, while the second one is to reduce the Gini Impurity. By default, Random Forests calculate the feature importance based on Gini Impurity.

Although equally decision tree-based, the **XGBoost** algorithm differs from Random Forest in two main aspects, in which the first is the decision tree creation process. In practice, while Random Forest trains each tree independently and using random samples, XGBoost applies a sequential ensemble technique that builds each tree in turn. In this approach, each data value is weighted according to its probability of being selected by a decision tree for further analysis. The second main difference relies on when the results are combined. Random Forest algorithm traditionally performs an averaging or majority voting at the end of the process, while XGBoost combines results along the way. Such features ensure that XGBoost, when properly tuned, performs better than Random Forest.

Logistic Regression is a classification algorithm based on the probability and odds of whether an event occurs. Logistic Regression's principal components are the *logistic response function* and the *logit function*, that map the probability for a more expansive scale given a linear modeling. The hypothesis of Logistic Regression limits the Cost Function. The cost function estimates the error between predicted values and expected values and presents it as a single real number in the interval between 0 and 1. Logistic Regression, given its name, uses the logistic function as cost, which derives from the sigmoid function.

4 Pre-processing Setup

Figure 1 depicts the methodology adopted for each dataset employed in this work. Thus, each dataset is first adjusted in a **Pre-Processing Stage** to suit the data to a machine learning model. Data not related to network attacks are removed at this stage, maintaining only data related to normal traffic and DoS or probing attacks. After the removal procedure, each dataset is split with 70% for training and the remainder for testing. Subsequently, all categorical features must be codified to the domain of real numbers or, depending on the method, to a binary matrix with each category's information. Besides, all continuous variables must be normalized since data may present discrepancies in the scale of values from different variables, which effect is mitigated by scaling all variables to the same interval.

Once pre-processed, data are conducted to a **Feature Selection and Training Stage**, in which different methods determine which variables, or features, are the most important to machine learning to predict the correct target class. This selection procedure is crucial to saving time and memory resources when dealing with large datasets. Before the training procedure, the training data is oversampled to balance the classes and consequently reduce the probability of creating a biased model for the dominant class. Finally, in the **Test and Evaluation Stage**, the validation and test data are applied to previously trained models, generating results evaluated through the use information retrieval metrics.

In the paper, we apply three different feature selection techniques and compare them to predict target-class outcomes. The first one uses the Ran-

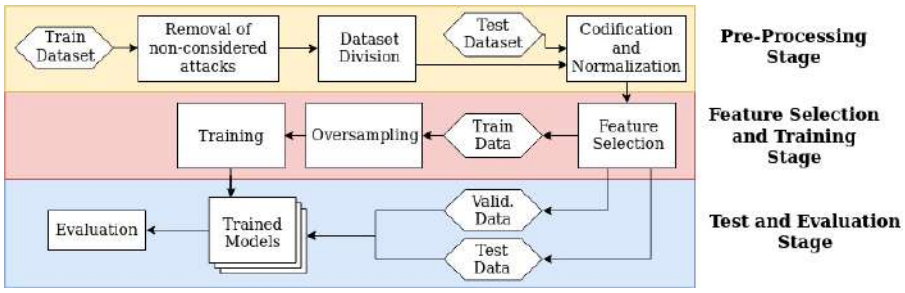


Fig. 1: Dataflow containing all stages from data collection to machine learning test and results.

dom Forest feature importance, which relies on reducing the Gini Impurity, used as a rule to split a decision node [20]. The second one uses Recursive Feature Elimination, which selects features recursively considering smaller sets of features. The third technique applies Mutual Information and returns the feature set that shows the highest dependency between selected features. Whether variables are independent, the mutual information is equal to zero, and as the value of Mutual Information increases, it denotes higher dependency between the variables.

An imbalanced dataset implies that the number of records between classes substantially differs. The class imbalance poses a problem to the machine learning model since statistical models tend to predict inaccurately less prevalent classes, implying a model bias to the majority class. In this regard, Chawla *et al.* proposed the SMOTE (Synthetic Minority Oversampling Technique) method that is widely applied to correct data asymmetry without adding information to the dataset [21]. The SMOTE method resembles with K-Nearest Neighbors (K-NN) and the Support Vector Machine (SVM) algorithms since it selects some samples that are spatially near from others, according to a predefined set limit, and these samples generate new synthetic samples that fill the gap in between two minority data points.

5 Feature Analysis

Besides testing machine learning models and comparing their performance metrics, there are different ways to extract useful information from the studied datasets. Figures 2–5 show the heatmaps of the Pearson correlation coefficient considering the variables with the highest sum of squares of the correlations with the other variables. Besides, we analyze the correlation between variables using the Kendall τ ranking. In parallel, we also employ Mann-Whitney U-test to compare the distribution between attacks and normal traffic samples from all the evaluated datasets, and check whether the two classes share the same distribution.

Table 2: Logistic Regression

Hyperparameters	Value
C	1.0
class_weight	None
dual	False
fit_intercept	True
intercept_scaling	1
max_iter	100
multi_class	auto
penalty	l2
random_state	Non
solver	lbfgs
tol	0.0001
verbose	0
warm_start	False

Table 3: Decision Tree

Hyperparameters	Value
ccp_alpha	0.0
class_weight	None
criterion	gini
max_depth	None
max_features	None
max_leaf_nodes	None
min_impurity_decrease	0.0
min_impurity_split	None
min_samples_leaf	1
min_samples_split	2
min_weight_fraction_leaf	0.0
random_state	None
splitter	best

Table 4: Random Forest

Hyperparameters	Value
bootstrap	True
ccp_alpha	0.0
class_weight	None
criterion	gini
max_depth	None
max_features	auto
max_leaf_nodes	None
max_samples	None
min_impurity_split	None
min_samples_leaf	1
min_samples_split	2
n_estimators	100
n_jobs	None
oob_score	False
random_state	None
verbose	0
warm_start	False

Table 5: XGBoost

Hyperparameters	Value
base_score	0.5
booster	gbtree
colsample_bylevel	1
colsample_bynode	1
colsample_bytree	1
gamma	0
gpu_id	-1
importance_type	gain
learning_rate	0.3
max_depth	6
min_child_weight	1
n_estimators	100
n_jobs	12
num_parallel_tree	1
random_state	0
scale_pos_weight	1
tree_method	exact

5.1 Pearson Correlation Heatmap

A useful tool to evaluate the Pearson correlation between the continuous variables in the datasets is to plot a heatmap, which shows the correlation value between the features' magnitude through a chromatic range. As shown in Figures 2–5 colors vary from red, i.e., positive correlation (heat), to blue, a negative correlation (cold). The figures show that a small subset of features possesses high correlation between themselves.

Figure 2 refers to the NSL-KDD heatmap. It is possible to verify three groups of highly correlated features expressed by regions with a dense concentration of hot spots. It means that the Pearson correlation between the variables have an impact of the choice of the variable as meaningful for classification.

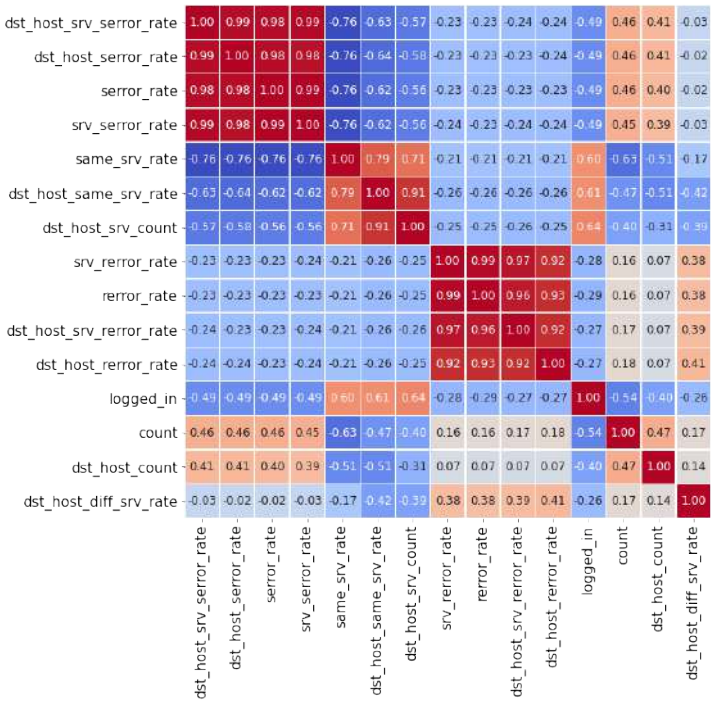


Fig. 2: Heatmap showing the Pearson correlation matrix between the features of the NSL-KDD dataset with the highest sum of squares of the correlation value.

Regarding the CICIDS 2017 dataset, Figure 3 shows a similar behavior, but it contains two groups of features with strong positive correlation among features (greater than 0.9). One group includes three features (*Max Packet Length, Packet Length Std and Bwd Packet Length Max*) and the other, five features (*Bwd Packet Length Mean, Avg Bwd Segment Size, Packet Length Mean, Bwd Packet Length Std* and *Average Packed Size*). A significant difference compared to Figure 2 is cold spots close to 0.5.

According to Figure 4, it is reasonable to state that the heatmap presents two large group of highly correlated features for the UNSW-NB15 dataset. However, only one of the groups has features that are also selected by the three feature selection methods mentioned in Section 4.

Unlike the previous heatmaps, Figure 5 presents a well-defined division and homogeneity of the features most correlated with each other, highlighting a group composed of five features (*sflow_bpackets, total_bpackets, sflow_bpackets, total_fpackets, sflow_bbytes*) and another containing the remaining features. In addition, it is possible to separate this last and most populous group of hot spots into two subsets of features that are even more correlated. When adopting a correlation value greater than 0.96, we find two subsets of five fea-

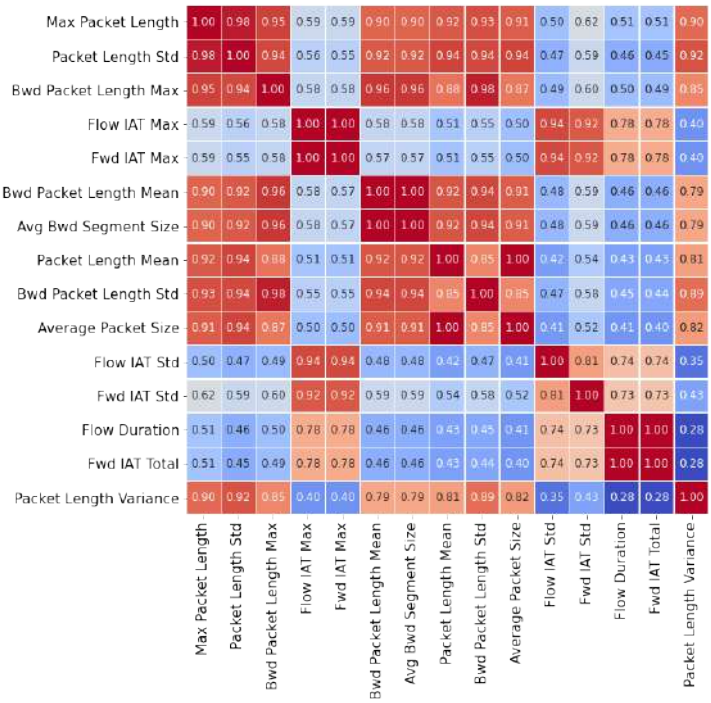


Fig. 3: Heatmap showing the Pearson correlation matrix between the features of the CICIDS 2017 dataset with the highest sum of squares of the correlation value.

tures each, in which the first is composed of *max_active, duration, min_active, mean_active* and the second subset includes *mean_fiat, mean_biat, min_fiat* and *min_biat*.

The analysis indicates a presence of two or more groups with strong positive Pearson correlation in different datasets as verified by Figures 5–4. In each of the heatmaps, highly correlated variables behave as linear combination of them and may be interpreted as one single variable that gives the same information, and since all of these features are chosen to train the machine learning model, it decreases the information gain, while moving the trained model towards overfitting. Thus, these selected variables are not linear independent.

5.2 Kendall Rank Correlation Coefficient

The Kendall τ Coefficient is a statistical measure of the ordinal association between two quantities. Comparing the Kendall τ Coefficient between two variables represents the statistical dependence of the two variables. A Kendall τ test is a non-parametric hypothesis test for statistical dependence. The analysis of the Kendall τ coefficient is important because variables with the highest

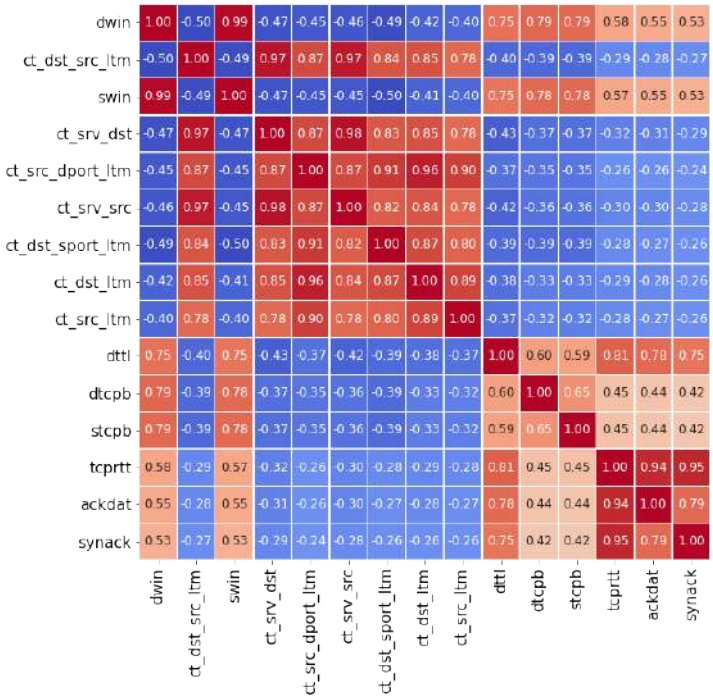


Fig. 4: Heatmap showing the Pearson correlation matrix between the features of the UNSW-NB15 dataset with the highest sum of squares of the correlation value.

coefficient values are highly correlated to the target variable, and, thus, the classifier that uses such variables is biased and cannot be generalized.

Figures 6–9 show the Kendall τ Coefficient for the continuous variables in the datasets. It considers the statistical dependence of each feature concerning the target class. Considering the NSL-KDD, the variable *same_srv_rate* possesses the highest Kendall coefficient value, reaching 0.68. For the CICIDS 2017 dataset, the variable that has the highest Kendall coefficient is *Min Packet Length* and reaches 0.32. Considering the UNSW-NB15 dataset the variable *sttl* possesses the highest Kendall coefficient, that is 0.7. For the CICBotnet dataset, the variable *min_fpktl* reaches the highest Kendall coefficient value, and is equal to 0.25. Therefore, we show that in NSL-KDD and UNSW-NB15 the most correlated variables with the target class carry significant information that leads to artificially accurate models. Indeed, the variable *same_srv_rate* indicates the data rate for the same server, which highly correlates to the DoS attacks and, for UNSW-NB15, the variable *min_fpktl* indicates the minimum length of packets in the forward direction, which relates to synthetic attack generation with small packets.

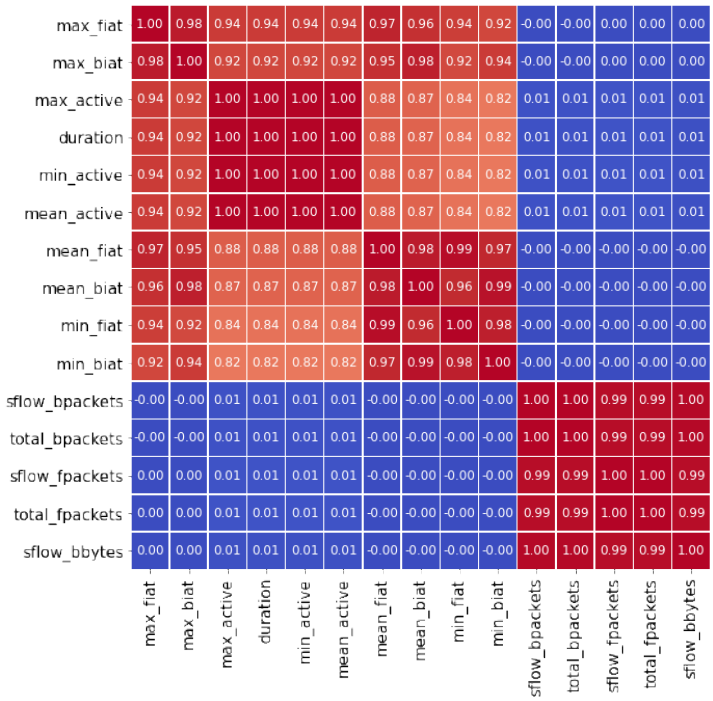


Fig. 5: Heatmap showing the Pearson correlation matrix between the features of the CICBotnet 2014 dataset with the highest sum of squares of the correlation value.

5.3 Mann-Whitney U-Test

Hypothesis tests, also called confirmatory data analysis, are ubiquitous in statistical analysis and work as an inference method. A hypothesis test is a random experiment to evaluate if randomness is a plausible explanation for the difference between two test groups. In executing a hypothesis test, the goal is to verify two affirmatives: the alternative hypothesis and the null hypothesis. The experiment raises the alternative hypothesis because it aims to explain the results obtained by recurring to an alternative pattern. The null hypothesis is that the results obtained are not enough to sustain the first affirmative, which means that event occurred by chance. The goal of any study that involves a hypothesis test is to determine if the null hypothesis can be rejected or not. Discovering that there is a difference in the effects generated by different groups, the null hypothesis can be rejected and accept the alternative hypothesis.

The p-value is an efficient way to measure the statistical significance. A p-value lesser or equal to the significance threshold (commonly used 0.05) is statistically relevant, indicating strong evidence against the null hypothesis

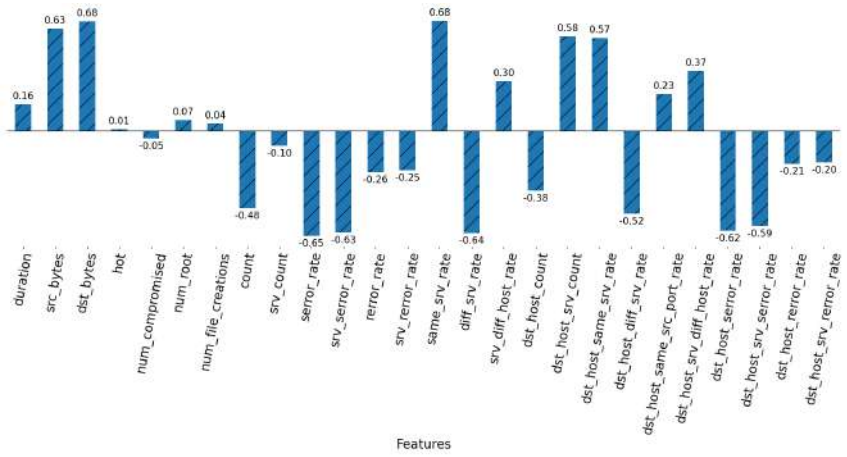


Fig. 6: Kendall τ coefficient between the KDD Train+ continuous features and the target variable. There is a high correlation between the nominal variables and the target class.

because it indicates a probability less than 5% that the null hypothesis is correct.

The Mann-Whitney U-Test is used to evaluate if two comparable samples present the same data distribution. If confirmed, the Null Hypothesis is proven, implying stochastic equality between two samples [22]. In counterpart, the Alternative Hypothesis verifies if two samples are distinct from each other. To check the hypothesis related to the comparison between two samples, three conditions must be respected [22]:

1. The two analyzed samples must be composed of random data obtained from the population. The concept aims to mitigate measuring errors and sample size;
2. Each observation in a sample must be independent, and the samples must be independent of each other;
3. The samples must compose of numeric and continuous values.

In this work, the U-Test aims to compare two groups, attack and normal traffic, considering all the continuous variables in the datasets given a significance threshold of 0.05. Thus, for each variable whose p-value exceeds the threshold, the test fails to reject the null hypothesis. Consequently, it is not possible to assure that the samples are stochastically different.

Table 6 shows the Mann-Whitney hypothesis test results considering each dataset with the respective variables where the p-value is greater than 0.05, for a significance level of 95%, which means that it is not possible to reject the null hypothesis and, then, the attack flows statistically behave like normal flows. From the collected p-values, almost all the considered variables reject

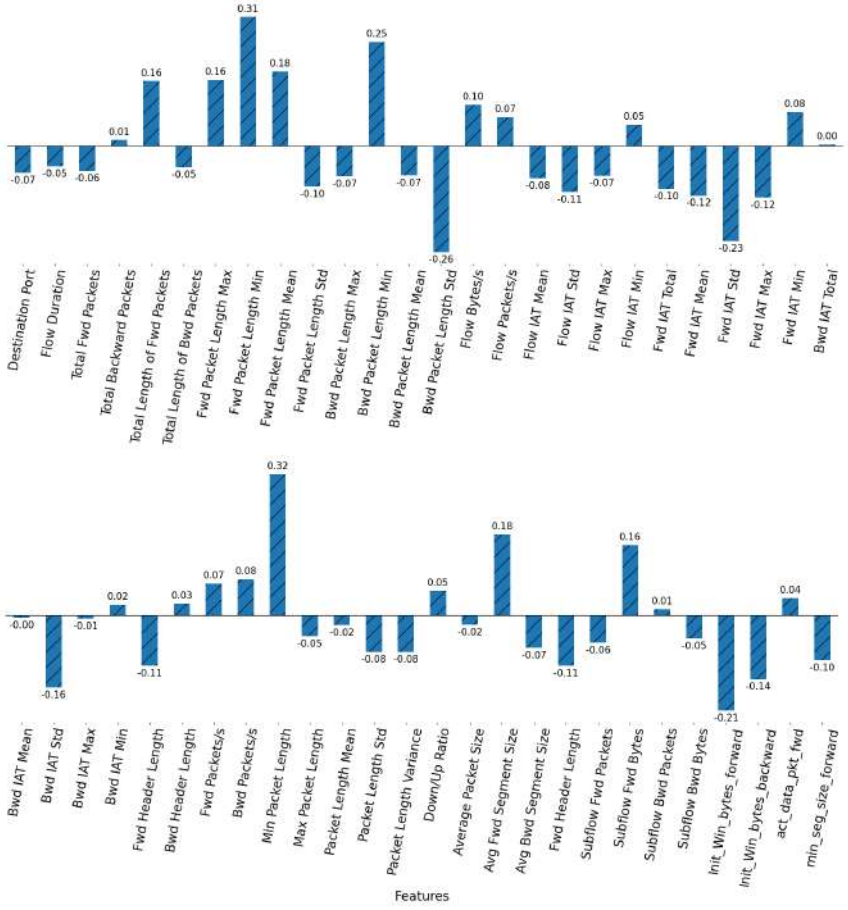


Fig. 7: Kendall τ coefficient between the CICIDS 2017 continuous features and the target variable. There is a high correlation between the nominal variables and the target class.

the null hypothesis, except for the ones in Table 6, meaning that these features distinguish attack traffic from normal traffic.

6 Results and Discussion

To assess each classifier's performance, we consider accuracy, precision, F1-Score, and sensibility (Recall) metrics. Another helpful metric is the Receiver Operating Characteristic (ROC) curve, which exhibits, for a given rate of False Positive values, how does that affect the True Positive Rate in the classifier predictions. The Area Under the ROC Curve provides more explicit information about the ROC curve, as an area that is close to 1 represents a correct,

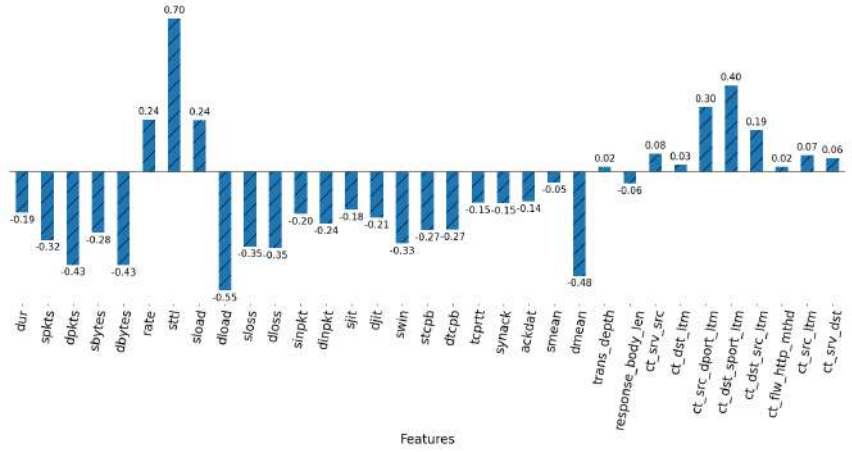


Fig. 8: Kendall τ coefficient between the UNSW-NB15 continuous features and the target variable. There is a high correlation between the nominal variables and the target class.

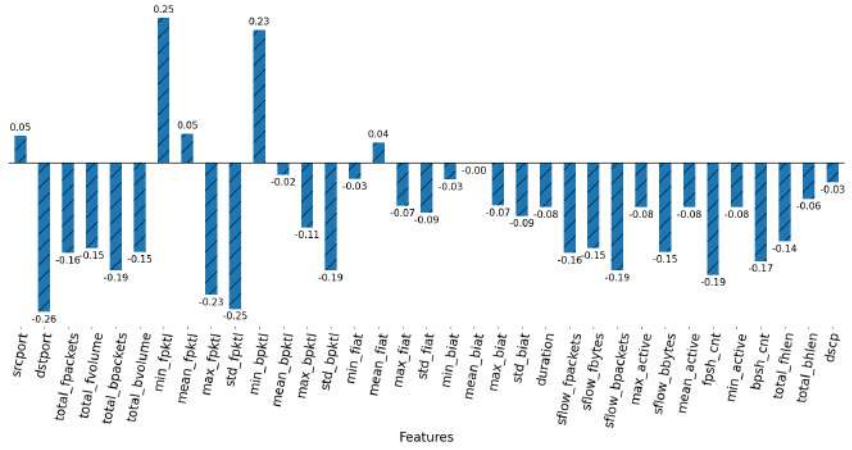


Fig. 9: Kendall τ coefficient between the CICBotnet 2014 continuous features and the target variable. There is a high correlation between the nominal variables and the target class.

but extremely specified, model while an area that is approximately 0.5 corresponds to a random model. Thus, an area that is almost equal to 1 is a strong clue of model overfitting.

Tables 7–10 reunite the results for the best classifiers in combination with each feature selection method (RFE, Random Forest Importance, and Mutual Information), when adopting attack prediction metrics to distinguish the analyzed datasets. Considering Random Forest as the best classifier, Figure 10

Table 6: Continuous numerical variables with Mann-Whitney test results different from zero.

Dataset	Feature	Label	Mean	Std	U-Stat	P-value
Botnet 2014	mean_biat	F20	552.856	3584.327	47600695.0	0.376
NSL-KDD	hot	F4	0.139	1.706	49630951.5	0.429
CICIDS-2017	Bwd IAT Total	F26	9950727	28826015.2	31727053.0	0.266
	Bwd IAT Mean	F27	1795244	8825502.1	31718482.0	0.257
	Bwd IAT Max	F29	4687013	17160506.1	31618835.5	0.165

depicts the associated ROC curves for the analyzed datasets with each curve representing a feature selection method. It is noteworthy that the executed algorithm used the 15^{th} most important feature returned by each feature selection method. This choice was based on the trade-off between data representation and useful information.

Considering the results shown in Tables 7–10, for the UNSW-NB15 dataset, Random Forest Classifier using Mutual Information is considered the best configuration as it has the highest Precision, although the F1-Score is tied with the Random Forest Importance feature selection method. For Botnet 2014, Random Forest Classifier with RFE as feature selection gives the best Recall compared to the other methods. For NSL-KDD, although the results are paired, Random Forest Importance gives the best Recall value. For CICIDS 2017, the Random Forest Classifier combined with the RFE gives the best results. Tables 2–5 depict the machine learning models' hyperparameters after the training step in the Feature Selection and Training Stage.

Table 7: Classification results by Feature Selection Method for CICIDS 2017.

Feature Selection Method	Metrics	Machine Leaning Models			
		Decision Tree	Logistic Regression	Random Forest	XGBoost
No Feature Selection	Accuracy	**1.00**	0.93	**1.00**	**1.00**
	Precision	**1.00**	0.81	**1.00**	**1.00**
	Recall	**1.00**	0.84	**1.00**	**1.00**
	F1-Score	**1.00**	0.83	**1.00**	**1.00**
Random Forest Importance	Accuracy	**0.99**	0.86	**0.99**	**0.99**
	Precision	0.97	0.60	0.97	**0.98**
	Recall	0.98	0.94	**1.00**	**1.00**
	F1-Score	**0.99**	0.73	**0.99**	0.99
Recursive Feature Elimination	Accuracy	**0.99**	0.85	**0.99**	**0.99**
	Precision	**0.99**	0.58	**0.99**	**0.98**
	Recall	0.99	0.93	**1.00**	**1.00**
	F1-Score	**0.99**	0.71	**0.99**	0.99
Mutual Information	Accuracy	**0.99**	0.87	**0.99**	**0.99**
	Precision	**0.98**	0.62	**0.98**	**0.98**
	Recall	**1.00**	0.94	**1.00**	**1.00**
	F1-Score	**0.99**	0.75	0.98	**0.99**

Table 8: Classification results by Feature Selection Method for NSL-KDD.

Feature Selection Method	Metrics	Machine Leaning Models			
		Decision Tree	Logistic Regression	Random Forest	XGBoost
No Feature Selection	Accuracy	**0.98**	0.94	**0.98**	0.98
	Precision	**0.96**	0.91	**0.96**	0.97
	Recall	0.99	0.96	**1.00**	0.99
	F1-Score	0.97	0.93	**1.00**	0.98
Random Forest Importance	Accuracy	0.97	0.92	**0.98**	0.98
	Precision	**0.96**	0.89	**0.96**	0.96
	Recall	0.96	0.90	**1.00**	0.99
	F1-Score	0.96	0.90	**0.98**	0.98
Recursive Feature Elimination	Accuracy	0.97	0.92	**0.98**	0.98
	Precision	**0.96**	0.90	**0.96**	0.96
	Recall	0.96	0.92	**0.99**	1.00
	F1-Score	0.96	0.91	**0.98**	0.98
Mutual Information	Accuracy	0.97	0.91	**0.98**	0.97
	Precision	**0.96**	0.90	**0.96**	0.96
	Recall	0.97	0.88	**0.99**	0.99
	F1-Score	0.96	0.89	**0.98**	0.97

Table 9: Classification results by Feature Selection Method for UNSW-NB15.

Feature Selection Method	Metrics	Machine Leaning Models			
		Decision Tree	Logistic Regression	Random Forest	XGBoost
No Feature Selection	Accuracy	0.86	0.78	**0.87**	0.86
	Precision	**0.82**	0.73	0.81	0.97
	Recall	0.95	0.95	**0.99**	0.73
	F1-Score	0.88	0.82	**0.89**	0.83
Random Forest Importance	Accuracy	0.85	0.78	**0.88**	0.87
	Precision	0.83	0.74	**0.84**	0.94
	Recall	0.93	0.93	**0.96**	0.78
	F1-Score	0.88	0.83	**0.90**	0.85
Recursive Feature Elimination	Accuracy	0.85	0.78	**0.87**	0.89
	Precision	0.82	0.74	**0.84**	0.97
	Recall	0.93	0.93	**0.96**	0.78
	F1-Score	0.87	0.83	**0.89**	0.86
Mutual Information	Accuracy	0.88	0.78	**0.89**	0.89
	Precision	0.85	0.73	**0.86**	0.94
	Recall	0.94	0.94	**0.95**	0.82
	F1-Score	0.89	0.82	**0.90**	0.88

It is noticeable that, according to Tables 7–10, there are differences in the results obtained for each dataset. Figures 10a–10b, which contain the ROC curves considering the CICIDS 2017 and NSL-KDD datasets, show a near-perfect curve. On the contrary, the ROC curves for UNSW-NB15 and Botnet 2014 display a more realistic ROC curve. Considering the ROC curves, Mutual Information displays the best AUC values, meaning that using this feature

Table 10: Classification results by Feature Selection Method for Botnet 2014.

Feature Selection Method	Metrics	Machine Leaning Models			
		Decision Tree	Logistic Regression	Random Forest	XGBoost
No Feature Selection	Accuracy	0.91	0.75	**0.92**	0.91
	Precision	**0.92**	0.73	0.91	0.91
	Recall	0.92	0.94	**0.96**	0.95
	F1-Score	0.92	0.82	**0.94**	0.93
Random Forest Importance	Accuracy	0.89	0.73	**0.91**	0.89
	Precision	**0.92**	0.80	**0.92**	0.91
	Recall	0.90	0.74	**0.94**	0.91
	F1-Score	0.91	0.77	**0.93**	0.91
Recursive Feature Elimination	Accuracy	0.90	0.73	**0.92**	0.90
	Precision	**0.92**	0.79	**0.92**	0.92
	Recall	0.91	0.77	**0.95**	0.93
	F1-Score	0.91	0.78	**0.93**	0.92
Mutual Information	Accuracy	0.89	0.73	**0.91**	0.89
	Precision	0.91	0.79	**0.92**	0.91
	Recall	0.91	0.74	**0.94**	0.92
	F1-Score	0.91	0.77	**0.93**	0.92

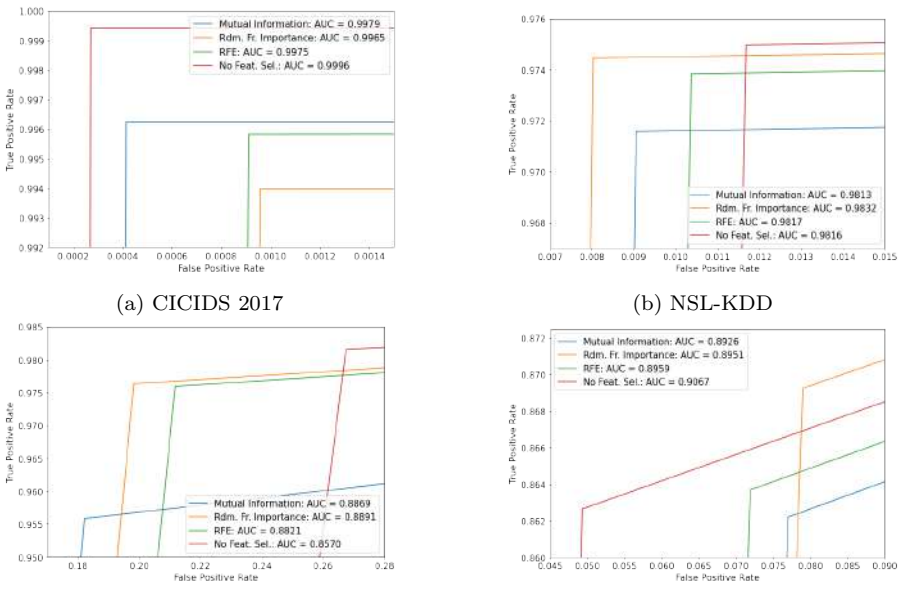
selection method successfully reduces the information loss and gives a more accurate classifier.

According to the previous section, it is possible to understand that the heatmaps explain the ROC curves for the NSL-KDD and CICIDS 2017, Figures 2–3. The heatmaps display clusters of variables that are highly correlated and behave as one single variable. Since a considerable number of highly correlated variables are used to train the machine learning models and are also correlated to the target variable 6–7, the training generates a biased model that leads to overfitting.

The analysis regarding the UNSW-NB15 and Botnet 2014 are explained by analyzing the heatmaps on Figures 4 and 5. The heatmaps show highly correlated variable clusters that may lead to loss of information, reducing the accuracy of the trained models. Besides, the Kendall ranking correlation shows less correlated variables to the target variable.

The Mann-Whitney hypothesis test answers whether the variables are statistically relevant for the target class prediction. Besides, the Mann-Whitney test also shows whether the features of the dataset cause bias in the classification. If the null hypothesis is not rejected, the two considered groups, attack and normal, have an indistinguishable probability distribution, showing that the attacks behave very similar to regular traffic. Nevertheless, as previously verified, the most important selected features show different statistical behaviors between the attack and normal groups, implying distinguishability between classes.

As shown in Table 6, it is possible to understand how the hypothesis test results explain the accuracy of the studied models. For Botnet 2014 dataset,



(a) CICIDS 2017 (b) NSL-KDD

(c) UNSW-NB15 (d) Botnet 2014

Fig. 10: ROC Curve with AUC values for Random Forest Classifier considering different datasets. MI: Mutual Information, RFI: Random Forest Importance, RFE: Recursive Feature Extraction, None: No Feature Selection

there is one variable *mean_biat* that, by analyzing the p-value from the Mann-Whitney hypothesis tests, fails to reject the null hypothesis while all the others reject the null hypothesis, meaning that the probability distribution of these variables for attack traffic and normal traffic differ. The impact from developing an IDS comes from the fact that including variables that clearly distinguish between attack and normal distribution in the feature selection stage to train a classifier increases the classifier probability of overfitting since these variables have a high weight distinguishing attack from normal traffic. The same analysis repeats to the NSL-KDD, which possess only one variable (*hot*) that fails to reject the null hypothesis, and CICIDS-2017, which has more variables that fail to reject the null hypothesis (*Bwd IAT Total, Bwd IAT Mean, Bwd IAT Max*).

Many researchers have pointed out the deficiencies of the KDD'99 dataset. However, we opted for an enhanced version of the dataset and considered other datasets to give state-of-the-art results. It is still noticeable that there are some inefficiencies in features. Factors such as the high correlation between features, the correlation between features and the target class, and the different statistical behavior between the group of samples of each target class cause bias to the prediction algorithms. Therefore, it raises questions about the reliability of the studied datasets to model state-of-the-art IDSs considering real-world network data.

7 Related Work

Tavallaee *et al.* point out some inherent issues in the KDD'99 dataset [15]. The issues include redundant data, unequal distribution of the attacks, which hampers the process of cross-validation. The author also aimed to analyze the precision of seven different machine learning classifiers, J.48, Naive Bayes, NBTree, Random Forest, Random Tree, multi-layer Perceptron, and SVM. According to their analysis, the J.48 model obtained the best results with 98.82% of accuracy.

Olusola *et al.* study the importance of the features from the KDD'99 dataset to discriminate the most important attacks [23]. By applying two metrics, entropy and rough set, the authors consider 10% of the KDD '99 dataset and ensure the redundant registers' removal. To identify the degree of significance of a specific feature, the authors adopted two approaches. The first one focused on calculating the dependency degree for each class of attack based on the number of instances inside the dataset. The second approach was based on comparing the classes and detecting the relevant features that distinguish one class from another. The results are in line with ours and indicate that a particular feature has a great degree of importance, and, for some classes, a single feature is enough to predict whether the sample is an attack or from normal traffic.

Siddiqui *et al.* focus on establishing the relationship between the type of attacks in the KDD'99 dataset and the network protocol. The proposal applies clusterization algorithms [24]. The study shows that TCP networks suffered most of the attacks, exploiting TPC/IP networks' capabilities to execute attacks, such as DoS, taking a long time before being detected. The UDP protocol suffered fewer attacks compared to TCP, about five types of attack. ICMP suffered mostly DoS attacks because of the lack of validation in the messages. ICMP is the target of only seven attack types in the dataset.

Depren *et al.* propose an IDS using a self-organized map to model the behavior on the KDD'99 dataset [12]. They present a hybrid IDS whose architecture consists of a misuse module and an anomaly detection module besides a support system to combine both modules' results. They perform a C4.5 Decision Tree to identify the threats. The results show that the model obtained an accuracy of 99.84%.

Hasan *et al.* identified some deficiencies inherent to the KDD'99 dataset, for example, a significant number of redundant registers, which turn the dataset imbalanced, and may cause the models to become more accurate to predict the majority class [25,26]. A new dataset was developed (KDD99 Test+, KDD99 Train+) that eliminates redundant values and increases the models' performance to mitigate the KDD'99 issues. Hasan *et al.* measure the performance of SVM with different kernel types, and conclude that each kernel is efficient to a specific type of attack [25]. Later, the authors aim to compare SVM's performance with Random Forest Classifier, and they conclude that while SVM has better accuracy, Random Forest takes less time to train the model [26].

Al-Yaseen *et al.* study the original KDD'99 dataset as a basis to propose a multi-level hybrid intrusion detection system that relies on the SVM and extreme learning algorithms to improve detection of known and unknown attacks [10]. The system uses a modified K-means algorithm to build a new small training set, significantly reducing the training time and improving SVM's performance. The results show that the accuracy achieved is above 95.75%, with a false alarm rate of 1.87%.

Panda *et al.* propose a hybrid approach using the combination of two classifiers to enhance the system performance [9]. The procedure firstly applies classification or clusterization methods and then conducts the output data to another classifier, determining whether a sample is an attack. The work relies on the NSL-KDD dataset, the improved version of the KDD'99 dataset. The authors perform some benchmarks over machine learning models, such as Decision Trees, SVM, and Random Forest. The results show that the combination of the meta-classifier END (Ensembles of Balanced Nested Dichotomies for Multi-class Problems) and Random Forest with ten trees produces an out-of-bag error of 0.06 and 100% intrusion detection rate with 0% false alarm rate.

Unlike previous work, our paper focuses on analyzing the correlation between features and the target class. Our work assumes the research hypothesis that the most used network-security datasets present some highly correlated features with the target class and, thus, introduce bias to the trained classifiers. Our results support our hypothesis since, for NSL-KDD and CICIDS 2017 datasets, the high performance of the trained classifier is due to the selection of over correlated features with the target class and the different probability distributions between attack and normal traffic.

8 Conclusion

In this paper, we focused on investigating the reliability of the application of different security datasets such as the NSL-KDD, the UNSW-NB15, CICIDS 2017, and CICBotnet 2014 to build and evaluate Intrusion Detection Systems. We analyzed the statistical properties of dataset features. We identified some dataset flaws that introduce bias when applying machine learning classification models. Our results show that different approaches and methodologies to model the data to predict attack traffic are biased to data. Thus, generalizing the results to handle real-time data or running against real-world traffic should consider other datasets to be validated.

Comparing each machine learning model according to the feature selection method, we conclude that the Random Forest is the most efficient model in detecting the attacks. We found that considering the metrics in the Tables 7–10, Random Forest Classifier displays the best results combined with the RFE feature selection method as it shows the best Recall compared to other feature selection methods in the UNSW-NB15 dataset. For Botnet 2014 and CICIDS 2017, RFE shows the best results along with Random Forest Classifier, and for

NSL-KDD, feature selection using Random Forest Importance is the most efficient. We also compared Receiver Operating Characteristic curves for the best technique for feature selection. The results display the ROC curves considering the Random Forest as the best classifier, differentiating each feature selection method. We show that Mutual Information gives the best AUC value, meaning that this feature selection technique is more efficient in reducing information loss when comparing the results obtained when no feature selection method is applied. It is also notable that, analyzing other aspects of the dataset, a factor that might be the root cause of the high performance is related to the genesis process of features during the initial stages of constructing the datasets. The correlation between most of the relevant features is high, and, thus, whether there is no use of a feature selection method, the performance overall is still noteworthy. In hypothesis tests, since most of the variables in all analyzed datasets succeed in rejecting the null hypothesis, we conclude that most of the datasets' variables differentiate the attack from normal traffic, introducing bias to the machine learning classifier. Therefore, the network attacks in all the datasets analyzed are not stealthy, and then, it introduces bias on training IDS models with them.

Acknowledgements The authors would like to thank CNPq, CAPES, FAPERJ, FAPESP (2018 / 23062-5), and RNP for the funding that made the research possible.

References

1. M. A. Lopez, D. M. Ferrazani Mattos, and O. C. M. B. Duarte, "An elastic intrusion detection system for software networks," *Annals of Telecommunications*, vol. 71, no. 11, pp. 595–605, 2016.
2. M. Andreoni Lopez, D. M. F. Mattos, O. C. M. B. Duarte, and G. Pujolle, "A fast unsupervised preprocessing method for network monitoring," *Annals of Telecommunications*, vol. 74, no. 3, pp. 139–155, 2019.
3. ——, "Toward a monitoring and threat detection system based on stream processing as a virtual network function for big data," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 20, p. e5344, 2019.
4. D. M. F. Mattos, L. H. G. Ferraz, L. H. M. K. Costa, and O. C. M. B. Duarte, "Evaluating virtual router performance for a pluralist future internet," in *Proceedings of the 3rd International Conference on Information and Communication Systems*, ser. ICICS'12. Irbid, Jordan: Association for Computing Machinery, 2012.
5. "Cic ids dataset," accessed: 2020-03-22.
6. "Unsw-nb15 dataset," accessed: 2021-01-26.
7. "Cic botnet 2014 dataset," accessed: 2020-04-02.
8. H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, pp. 16–24, January 2013.
9. M. R. P. Mrutyunjaya Panda, Ajith Abrahamb, "A hybrid intelligent approach for network intrusion detection," vol. 30. Elsevier, 2012.
10. M. Z. A. N. Wathiq Laftah Al-Yaseen, Zulaiha Ali Othman, "Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system," *Expert Systems With Applications*, 2017.
11. I. J. Sanz, D. M. F. Mattos, and O. C. M. B. Duarte, "Sfcperf: An automatic performance evaluation framework for service function chaining," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–9.

12. O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, "An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks," vol. 29. Elsevier, November 2005, pp. 713–722.
13. "1998 darpa intrusion detection evaluation dataset," accessed: 2020-04-02.
14. "Kdd cup 1999 data," accessed: 2020-02-22.
15. M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set." IEEE, 2009.
16. N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.
17. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSp*, 2018, pp. 108–116.
18. E. Biglar Beigi, H. Hadian Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *2014 IEEE Conference on Communications and Network Security*, 2014, pp. 247–255.
19. R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, p. 16, Jun 2018.
20. T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: Data mining, inference, and prediction," *Math. Intell.*, vol. 27, pp. 83–85, 11 2004.
21. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, p. 321–357, Jun. 2002.
22. N. Nachar *et al.*, "The mann-whitney u: A test for assessing whether two independent samples come from the same distribution," *Tutorials in quantitative Methods for Psychology*, vol. 4, no. 1, pp. 13–20, 2008.
23. D. O. A. Adetunmbi A. Olusola, Adeola S. Oladele, "Analysis of kdd '99' intrusion detection dataset for selection of relevance features," vol. 1, 2010.
24. S. N. Mohammad Khubeb Siddiqui, "Analysis of kdd cup 99 dataset using clustering base data mining," vol. 45, 2013, pp. 23–34.
25. B. P. Md. Al Mehedi Hasan, Mohammed Nasser, "On kdd'99 dataset: Support vector machine based intrusion detection system (ids) with different kernels," *International Journal of Electronics Communication and Computer Engineering*, vol. 4, pp. 2278–4209, 2013.
26. M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (ids)," vol. 6, 2014, pp. 45–52.

Cover Letter

Click here to access/download

Supplementary Material

CoverLetter.pdf

Manuscript ID: ANTE-D-21-00155
A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms

Dear Editor and Reviewers,

We greatly appreciate the effort and time spent on our manuscript. We would like to thank all the reviewers for their suggestions. After all of the reviewer's valuable remarks, we are confident that the paper is in better shape and hopefully ready for publication.

Please consider the revised version of our manuscript entitled "A Statistical Analysis of Intrinsic Bias of Network Security Datasets for Training Machine Learning Mechanisms" for publication in the Annals of Telecommunications journal. In this revision, we address the reviewers' comments, highlighting our changes with a blue font in the manuscript. In our response below, we provide specific answers to the reviewers' comments and concerns.

Best regards,

João Vitor Valle Silva,
Nicollas Rodrigues de Oliveira,
Martin Andreoni Lopez,
Dianne Scherly Varela de Medeiros,
Diogo Menezes Ferrazani Mattos

Reviewer reports

Reviewer #2:

Overall, my conclusion from reading the manuscript is that the results tell a different story from the abstract and introduction. The authors indicate that the datasets lead to biased results. Then, they perform some analysis of feature correlation, and remove this correlation using feature selection techniques. However, at the end, the outcome of the models using three feature selection techniques is mostly similar to the outcome of a model trained using all the features. This result, in my point of view, contradicts the argument of the authors.

Response to Reviewers: We thank the reviewer for the attentive read of our paper, but we kindly disagree with the reviewer's statement since the similarity between the results with and without the feature selection procedure does not nullify the bias argument. Indeed, such results corroborate the biased generation of the analyzed datasets since, regardless of the selected features, all algorithms managed to substantially differentiate regular traffic from attack traffic.

Another point worth commenting is that the test methodology requires improvement. There are a number of parameters of the algorithms that should have been provided on the text. I would like to see more details on the chosen hyperparameters, as well as the choice of algorithms.

Response to Reviewers: As requested, we insert Tables 2-4 (new ones) containing more information about the hyperparameter values configured in the Logistic Regression, Decision Tree, and Random Forest algorithms, respectively. We also highlight that the hyperparameters choice was performed as a greedy optimization problem. We performed a greedy search to optimize the analyzed classifiers. However, optimizing classifiers performance is not the scope of the paper.

"*Data not related to network attacks are removed at this stage, selecting only DoS or probing attack classes*" Please explain why this was done. I assume that a real IDS would also need to classify traffic as normal, so it seems odd to remove normal traffic.

Response to Reviewers: We emphasize that the removal procedure was performed to discard attacks on layers higher than the transport layer, because the paper scope is on network-related attacks and not on the application-level attacks. The removal procedure refers to attacks other than DOS or probing, not normal traffic. The normal-behaved traffic was kept in the analyzed datasets. To avoid this misinterpretation, we rewrite this snippet clarifying what data is actually kept after the preprocessing stage.

"Data not related to network attacks are removed at this stage, maintaining only data related to normal traffic and DoS or probing attacks."

Tables 2-4 need much more explanation. They provide features without explaining what they are. Also, they do not provide numeric values for the feature importance.

Response to Reviewers: We thank the reviewer for the advice. We removed that Tables that did not aggregate useful information to the paper.

Table 6 considers very simple classifiers. Also, the authors do not explain why they chose these types of classifiers. As an example, I miss more modern techniques such as a deep neural network. Why should someone use decision trees, logistic regression and random forest only as state of the art? There are a number of models that are more popular, including the algorithms based on trees, such as XGBoost.

My proposal for the authors is to show results for one algorithm of each class, e.g. dimensionality reduction, neural networks, tree-based algorithms, etc.

Response to Reviewers: We stress that the focus of the paper is not on evaluating classifiers' performance, but we focus on testing the existence of bias on well-known security datasets. Hence, we chose to deploy the most common classifiers as they perform accurately for the selected dataset. As suggested by the reviewer, we enriched the comparative analysis by adding results from a tree-based algorithm, XGBoost. The hyperparameters and the results of applying this algorithm to the different datasets are shown in Tables 5 and 7-10, respectively. The inclusion of the algorithm in the analysis was accompanied by a brief explanation of it in Section 3, as seen in the text below.

"*Although equally decision tree-based, the XGBoost algorithm differs from Random Forest in two main aspects, in which the first is the decision tree creation process. In practice, while Random Forest trains each tree independently and using random samples, XGBoost applies a sequential ensemble technique that builds each tree in turn. In this approach, each data value is weighted according to its probability of being selected by a decision tree for further analysis. The second main difference relies on when the results are combined. Random Forest algorithm traditionally performs an averaging or majority voting at the end of the process, while XGBoost combines results along the way. Such features ensure that XGBoost, when properly tuned, performs better than Random Forest*."

Another important issue on the manuscript is that it lacks information about the hyperparameters used on each model. Furthermore, the authors should have commented how they tackled the issue of hyperparameter optimization.

Response to Reviewers: As mentioned previously, information about the hyperparameters used in the algorithms is expressed in Tables 2-4. It is noteworthy that such hyperparameters are derived from a simple greedy-search process since we intend to demonstrate that such datasets are biased even in the worst case of choice.

The authors do not mention how they split the existing datasets into train-test-validate sets. As an example, the CICBotnet 2014 has only train and test sets, so the authors had to perform their own split for validation.

Response to Reviewers: To clarify the percentage of split adopted, the excerpt below was added in Section 4. Due to the high accuracy, we did not identify the need for a validation step.

"*After the removal procedure, each dataset is split with 70% for training and the remainder for testing.*"

Section 6 mentions the results for feature selection, however I did not find in this section themention as to which features were selected? Was it a top-X approach? If so, how did the authors choose the value of X?

Response to Reviewers: In the feature selection stage, we opted for the top-15 approach for each method applied. We choose the 15 most important features as a trade-off between data representation and useful information. Thus, 15 features against all features were necessary and sufficient to show that feature selection is not a confusing factor for the conclusion that the datasets introduce a biased assumption that the classifiers are accurate. Given the analysis of 4 datasets with distinct features, the inclusion of all selected characteristics would occupy an excessive amount of space and would be of little use. Even so, to clarify this issue, we include the excerpt below.

"It is noteworthy that the executed algorithm used the 15th most important feature returned by each feature selection method. This choice was based on the trade-off between data representation and useful information."

Overall, the results on tables 6-9 are very similar, no matter which method for feature selection, or if feature selection was not used whatsoever. My personal conclusion is that feature selection does not impact the final result. Also, it shows no real difference over any of the feature selection techniques.

Response to Reviewers: We agree with the reviewer. Indeed, the paper aims to show that feature selection has low impact on the final accuracy of classifiers, since the datasets' construction processes are biased. The convergence of results for similar and high values is another indication that the datasets used, known as a reference in the IDS field, have bias problems. We demonstrated that even applying untuned algorithms, it was possible to achieve high accuracy, precision, and F1-score values regardless of the feature selection method adopted.

Reviewer #3:

The first objective of the paper is to propose a statistical analysis of machine learning datasets for network security. This approach can be considered as an alternative for creating Intrusion Detection Systems. There are several contributions in this paper. First, the statistical analysis of four network security datasets; second, determine the most important shortcoming of the available datasets and finally, evaluate the quality of available datasets for machine learning. For this last point the authors show the weaknesses of the current datasets.

So, the authors deal with the reliability when applying different security datasets: NSL-KDD, the UNSW-NB15, CICIDS 2017, and CICBotnet 2014. The objective is to evaluate Intrusion Detection Systems. They analyze the statistical properties of dataset features. They identified some dataset flaws that introduce bias when applying machine learning classification models. The results show some important variances between the different approaches and methodologies to model the data for predicting the attacks. Then, they generalize the results to handle real-time data. Moreover, they show that running against real-world traffic should consider other datasets to be validated.
The study carried out is extremely important and shows that the use of machine learning techniques can give rise to highly biased results depending on the dataset. In addition, there are more and more attacks against datasets themselves by producing false datasets that lead to results contrary to what the user wants. In the same way, datasets can be manipulated to greatly falsify the results.
This paper is important by evaluating the models' performance and it is shown that a higher accuracy is obtained when comparing the results of different works addressing the same problem.
The authors conclude that the Random Forest model is the most efficient technique for detecting the attacks.

As a conclusion, the paper is well written and may be accepted for publication in the Annals of Telecommunications after improving the paper according to the reviewers remarks

Response to Reviewers: We thank the reviewer for recognizing the contribution of the paper.

PDF Version (w/ marks)

Click here to access/download
Supplementary Material
Modifications.pdf