A Comparative Analysis of Undersampling Techniques for Network Intrusion Detection Systems Design

Bruno R. S. Silva, Ricardo J. N. Silveira, Manuel G. da Silva Neto, Paulo César Cortez and Danielo G. Gomes

Abstract—**Intrusion Detection Systems (IDS) figure as one of the leading solutions adopted in the network security area by preventing intrusions and ensuring data and services security. A current approach to detect network intrusions is IDS development by employing Machine Learning (ML) techniques. Due to a variety of strategies used, the IDS project needs to be assertive and has an efficient processing time. Undersampling techniques allow ML classifiers to be evaluated from smaller dataset subsets in a representative manner, aiming for high assertive metrics. There are several literature solutions for IDS conception, but there is a lack of some criteria such as replicability. In this work, we evaluated three undersampling methodologies: Random, Cluster-centroids, and NearMiss1 in two novel unbalanced datasets (CICIDS2017 and CICIDS2018). We evaluated Nearest Centroid, Naive Bayes, Random Forest, K-Nearest Neighbor, and Support Vector Machines classifiers using 5x2-Fold cross-validation and Wilcoxon signed-rank statistical test. Our results indicated that distance-based classifiers performed well when applied to the undersampled datasets by the Cluster centroids technique. Moreover, the adoption of the undersampling schemas allows the evaluation of cost-processing classifiers into a competitive time.**

Index Terms—**Intrusion Detection Systems, Undersampling, CICIDS2017, CICIDS2018.**

I. Introduction

INTRUSION Detection Systems (IDS) are responsible for detecting network anomalies that firewalls cannot handle, such as unauthorized access and malicious traffic. IDS monitors traffic in real-time for anomalies and, if so, alerts network administrators to take appropriate counter-measures. These actions can be, to name a few: (i) blocking specific ports or IPs, (ii) rejecting services to a node that is sending malicious requests, and (iii) flooding services commonly used for attacks [1].

Therefore, networks that do not have any traffic analysis mechanism can not ensure security to them who use it because they have no guarantee that they can operate efficiently. Also, unmonitored networks are subject to functionality loss or even to be successfully attacked by a malicious network agent [1]. These mechanisms aim to detect malicious activity through classifiers, distinguishing traffic in two ways: binary or multi-class classification. In the former, the IDS classifies the network traffic as normal or abnormal. On the later, with multiple classes, the system detects the type of attack in question.

Machine learning techniques allow a computer to learn about specific rules and infer unseen data in the learning phase. Thus, machine learning-based IDS provide a learning-based methodology for discovering attacks according to the behavior trained in the system [2].

IDS designers deal with several challenges in adopting machine-based approaches to IDS. For example, its performance depends on the quality, size, and generalization of knowledge databases used for training and testing. Another issue is the classifier's choice for the network traffic standards recognition. For this, the trade-off between the intrinsic classifier' computational cost, the amount of available data, and the desired model's generalization capacity must be taken into account.

The designers have additional issues related to data unbalance-factor. Unbalancing in datasets is a condition in which the proportion between each class's samples, e.g., normal and abnormal, is unequally distributed. This condition influences the classifier's behavior due to a disproportional number between one or more categories. Therefore, undersampling techniques have been used to narrow this gap in the class-imbalance learning area [3]. With undersampling techniques, one can only take part of the data in the evaluation process while maintaining the model representativeness and generalization. In this way, it is possible to consider characteristics such as the number of records in the databases, time needed for training/testing the classifiers, and the computational cost used in its evaluations [4].

Several literature works adopt different approaches to evaluating classifiers in the IDS project. Some recent works have been published using obsolete datasets NSL KDD, DARPA, and CAIDA [5]–[11]. However, such bases have no recent attacks, such as SQL Injection and Heartbleed, which are common in modern scenarios.

Silva Neto *et al.* [12] evaluated the performance of eight machine learning-based intrusion detection algorithms in CICIDS2017 database [13]. The authors applied two sampling techniques to represent two scenarios and Mean Decrease Impurity to select the most relevant features. However, by using the whole database in their evaluation, it was impossible to evaluate computationally complex algorithms such as Support Vector Machine (SVM) in a time-efficient way.

The authors are with Department of Teleinformatics Engineering (DETI), Federal University of Ceará (UFC), Campus do Pici, Ceará, Brazil e-mails: ({bruno,cortez}@lesc.ufc.br, manuel@ifpi.edu.br, danielo@ufc.br).

The authors would like to thank the Brazilian National Council for Scientificand Technological Development (CNPq) under Grant No. 313599/2019-0 and Coordination for the Improvement of Higher Education Personnel (CAPES)-Finance Code 001 for the support to this research.

Digital Object Identifier: 10.14209/jcis.2021.3

The work in [14] used Principal Component Analysis (PCA) for feature reduction to evaluate computationally complex algorithms.

In this same direction, Liu *et al.* [15] used 10% of KDD99 database, maintaining different proportions between classes.

Moreover, D'hooge *et al.* [16] conducted a recent study on the NSL-KDD, CICIDS2017, and CICIDS2018 databases, but the undersampling approach used, as well as the flow of experiments performed, need more details for replication purposes. Thus, none of these works systematically uses undersampling, with different techniques to evaluate subsets' generalization.

Therefore, this work aims to evaluate IDS machine learning-based on subsets generated from different undersampling techniques found in the literature. Afterward, we compare the performance of the classifiers for different scenarios using Wilcoxon's statistical test. We also discuss the trade-off between assertiveness, classifiers' computational cost, and time spent to train/test the models for intrusion detection.

Our main contributions in this work are:

We present an exploratory analysis and performance evaluation of two up-to-dated IDS datasets: CICIDS2017 and CICIDS2018;

We present sub-bases from three different undersampling techniques applied in CICIDS2017 and CICIDS2018 datasets, namely, Random, Cluster Centroids and NearMiss; and

We compare the performance of five machine learning approaches, namely, Nearest Centroid, Naive Bayes, Random Forest, K-Nearest Neighbors, and Support Vector Machines, between the all-data schemas and each of the undersampled datasets.

This work is organized as follows: Section II presents the related works. We detail the experiments' methodology in Section III. In Section IV, we present the results and discussions of experiments. Finally, in Section V, we synthesize the conclusions, contributions, and future work.

II. Related Works

In the IDS machine learning-based design, the classification algorithm decision-making is a crucial activity that involves a specific set of criteria. In this process, systems must handle data efficiently because, in an ideal scenario, training time, testing, and the computational load of processing for intrusion detection must be minimized. Different approaches have been used to evaluate the trade-off between IDS processing time and assertiveness in this context. There are works in the literature that addresses the design of machine learning IDS, in which techniques are adopted to optimize the selection process of classification algorithms. We describe some of these works below.

Another approach using 20% of the NSL-KDD can be found in work done by Aljawarneh *et al.* [17]. The authors use binary classification in the database. Thus, the class labels referring to attacks were unified, representing an abnormal traffic pattern. Also, a voter-based detection model among seven classical algorithms was proposed, and the use of the Information Gain algorithm to reduce the number of characteristics from 41 to 8. The results obtained by the authors indicate that the proposed model has high accuracy rates, between 90% and 99%, and low false-positive rates, between 0.003% and 0.102%. However, in real scenarios where there are processing restrictions, this model is computationally complex, requiring seven different algorithms to perform the voting.

The work proposed by Bhaskar *et al.* [18] proposes a feature selection technique to evaluate in the NSL-KDD database, seeking to minimize false positive and negative rates, as well as maximize the detection rate. However, no evaluation methodology has been applied, either by evaluation algorithms or metrics. Also, the experiments were performed using an obsolete database and did not evaluate IDS in their methodology.

Gao *et al.* [19] used the NSL-KDD database[1] as a research object to propose a combined learning algorithm, similar to the work discussed above. They evaluated five classifiers for the composition of the voting system. The experiments were performed using 2-Folds in cross-validation, obtaining results between 73% and 79% for accuracy and sensitivity, 80%, and 84% specificity, and 69% and 80% F1. Moreover, to deal with database unbalance, the random undersampling technique was used. However, the number of randomly undersampled databases was not explicited. It may lead to the belief that the data are unrepresented because of the randomness of only one sub-base generated.

In this same direction, Bedi *et al.* [11] used the NSL-KDD database to evaluate Siamese Neural Network to deal with the imbalance problem in intrusion datasets. The authors do not compare different strategies for undersampling in the design of their experiment. Moreover, the authors' solution covers only some attacks, namely, Remote to Local (R2L) and User to Root (U2R).

A feature selection algorithm based on two objective functions called MOEDAFS was proposed by Maza *et al.* [20]. The experiments were conducted by creating different sub-databases with a number of characteristics between 5 and 22, using the NSL-KDD as the study's object and seven machine learning algorithms. The results presented in terms of accuracy vary between 81% and 98%.

Liu *et al.* [15] performed a 10% undersampling in the KDD99 database in the evaluation of algorithms for smart home IDS based on the Convolutional Neural Network (CNN) classifier and Principal Component Analysis (PCA) for feature selection. However, the number of randomly undersampled databases was not made explicit. It may lead to the belief that the data are not well represented because of the randomness of only one sub-base generated.

Ullah *et al.* [21] presented a framework for IDS with a focus on Internet of Things (IoT) scenarios applied to Smart-Grid, in which there are resource constraints such as power and processing capacity. The authors used an undersampling approach to evaluate machine learning algorithms to detect attacks in this environment. Thus, 20% of ISCX2012 [22] was used for training, and 80% for testing, varying different values of K-Fold cross-validation. Note that the approach

[1] https://www.unb.ca/cic/datasets/nsl.html

is unrecommended in evaluating algorithms that have high computational cost in their testing phase.

Sharafaldin *et al.* [13] applied a feature selection technique called Mean Decrease Impurity (MDI) in the CICIDS2017 dataset for evaluation of seven machine learning algorithms. This technique allowed selecting the best features for each of the 15 traffic types, reducing the amount of data used for training and testing the classifiers in the algorithm selection process. The experiments' results were from 77% to 98% for accuracy, 4% and 98% for recall, and 4% and 94% for F1. However, only selecting the best characteristics still resulted in a significant amount of information and high time for training and testing of the classifiers.

Silva Neto *et al.* [12] evaluated the performance of eight machine learning-based intrusion detection algorithms in the CICIDS2017 database [13], using different sampling techniques. Besides, the MDI technique was used to reduce the number of database characteristics, achieving high detection rates in terms of accuracy, recall, F1, and processing time. The experiments' results were from 75% to 99% for precision, 55% and 98% for recall, and 63% and 99% for F1. However, because the whole database was employed in its evaluation, it was hard to evaluate algorithms with high computational load, such as Support Vector Machines (SVM).

The work presented by D'hooge *et al.* [16] evaluates 12 machine learning algorithms in four databases comprehensively: NSL-KDD, ISCX2012, CICIDS2017, and CICIDS2018. The experimental design at work consisted of vertical (samples) and horizontal (feature) reduction, obtaining expressive results (up to 99% accuracy) using 1% of data for training and 99% for testing. However, the work is difficult to replicate for some reasons:

undersampling technique used was not made explicit, it is unknown which samples were selected by the authors;

characteristics were reduced based on empirical methods, i.e., the characteristics removed were justified in work. They contaminated the results or were redundant or even problematic; it was not specified details about this "contamination";

codes of the experiments and the CSV files of the databases used were made available for consultation. However, it is noted that: the database CICIDS2018 is not complete, and techniques not explained in the article, such as Principal Component Analysis (PCA), are used.

A brief comparison between the related papers and this work is found in Table I. We highlight that none of these works presented a systematic and replicable comparison among undersampling techniques in the IDS project design.

III. Methodology

Fig. 1 shows a high-level overview of the methodology employed. The pre-processing phase followed the methodology proposed by Silva Neto *et al.* [12], which resulted in the concatenation of all dataset files, followed by the removal of records containing invalid values, as well as the removal of characteristics with mean and zero standard deviation for all samples. In the undersampling phase, the main objective is to reduce unbalance. To achieve this, we consider that given a number n of desired sub-dataset samples, a pre-selection is needed to preserve the minority classes. Tables II and III present the comparison between original and undersampled datasets in terms of the number of samples, class percentage, and unbalanced Ratio (UR).

We used three undersampling techniques to evaluate the classifiers: Random, Cluster Centroids, and NearMiss1. Moreover, the choices of techniques were based on the following reasons:

The Random undersampling is the standard approach widely used in the literature.

The Cluster centroid undersampling uses the concept of selection of samples based on each class's centroid, seeking a grouping among the samples selected

The NearMiss1 technique uses the concept of borderline among the classes to select the most representative samples.

Such techniques are used in the literature in different contexts such as voice [25], financial [26], among others.

In the random technique, we generated ten Random undersampled datasets. Stability is aimed at sample selection, which explains the random technique context. Therefore, we consider that performance evaluation is the average of all trained/tested sub-datasets. We used the centroid analysis in the second approach for the generation of the other undersampled base. Thus, each class calculates the centroid to all samples. The n samples, which have the smallest distances to the centroid, will be selected. It is worth mentioning that n is the size of the desired base; in this case, the same size as the previous base. NearMiss1 selects from the majority class in the third approach, the instances in which the average distances to the three closest minority instances are the smallest. Thus the instances selected by NearMiss-1 are close to some of the minority class instances [27].

In the scaling-change phase, the technique called Min-max scaler proposed by [28] is used based on the following formula: Let X = $[X_1,X_2,...,X_n]$ be a sample described b n features, Y = $[y_1, y_2,...,y_k]$ the set of k classes and $\hat{Y}$ = $[\hat{y}_1,\hat{y}_2,...,\hat{y}_n]$ the set of k classifier predicted classes. $X_{MinMax} = \frac{X \quad X_{min}}{X_{max} \quad X_{min}}$, where X_{MinMax} is the vector with scale between 0 (zero) and 1 (one), X is the original vector, X_{min} and X_{max} are the smallest and largest elements of the vector X, respectively.

After that, the performance evaluation of Naive-Bayes (NB), KNN, Nearest-Centroid (NC), Random Forests (RF), and Support Vector Machine (SVM) classifiers is performed. We choose these classifiers following three criteria: three distance-based algorithms - NC, linear, while SVM and KNN non-linear, Naive Bayes, which is probability-based and Random Forests, which handles well with unbalanced bases, disadvantaging more balanced subsamples.

In the classification phase, we train and test each algorithm 10 (ten) times using the cross-validation technique. The data is separated into two mutually exclusive folds repeating five times: one for training and one for testing. We adopt this procedure according to Dietterich *et al.* [29], which recommends experiments with five repetitions of two folds in cross-

TABLE I: A summary of related works

Work	Dataset	Sampling technique	Undersampling technique	Evaluated classifiers	Evaluation metrics	Statistical test
[17]	NSL-KDD	10-Fold CV	20% NSL-KDD train portion	J48, Meta Pagging, RandomTree, REPTree, AdaBoostM1, DecisionStump and Naive Bayes	Accuracy, FPR, FNR, TP, TN	None
[18]	NSL-KDD	None	None	None	None	None
[19]	NSL-KDD	2-Fold CV	Random Undersampling	DecisionTree, RandomForest, KNN, LR, SVM, DNN, AdaBoost	Accuracy, Precision, Recall and F1	None
[20]	NSL-KDD	80/20 Train Test Split	None	Naive Bayes, MLP, SVM, KNN, DecisionTree	Accuracy, Precision, Recall and F1	None
[11]	NSL-KDD	50/50 Train Test Split	None	Siamese Neural Network	Precision, Recall	None
[15]	KDD99	Not clear	Not clear (10%)	CNN	Accuracy, MRR Processing time	None
[21]	ISCX2012	80/20, 3-Fold CV, 5-Fold CV, 10-Fold CV, 15-Fold CV, 20-Fold CV	None	J48, JRip, Naive Bayes, SVM, MLP	Precision, Recall and F1	None
[13]	CICIDS2017	Not clear	None	KNN, RandomForest, ID3, AdaBoost, MLP, Naive-Bayes, QDA	Precision, Recall, F1 and execution time	None
[12]	CICIDS2017	10-Fold CV, 10-Fold Stratified CV	None	Nearest Centroid, NaiveBayes, AdaBoost, MLP, Decision Tree, KNN, Random Forest and QDA	Precision, Recall, F1 and Test time	Wilcoxon
[23]	CICIDS2017	67/33 Train/Test Split	None	DNN and SVM	Accuracy, Precision, Recall and F1	None
[24]	CICIDS2017	70/30 Train/Test Split	None	RF, Naive Bayes, LDA, QDA	False Alarm Rate, Accuracy, Detection Rate, Precision, Recall, F1	None
[16]	NSL-KDD, ISCX2012, CICIDS2017, CICIDS2018,	1/99, 10/90, 20/80, 30/70, 40/60, 50/50 Stratified Train/Test Split	Not clear	DecisionTree, Bagging, AdaBoost, Gradient Boosted Trees, Regularized Gradient Boosting, RF, ExtraTrees, KNN, Nearest Centroid, Linear SVM, RBF SVM, Logistic Regression	Accuracy, Precision, Recall, F1 and ROC	None
This work	CICIDS2017, CICIDS2018	5x 2CV	Random, Cluster Centroids, Near Miss	Nearest Centroid, Naive Bayes, Random Forest, KNN, SVM	Precision, Recall, F1 and processing time	Wilcoxon

CV - Cross Validation.

NC - Nearest Centroid, NB - Naive Bayes, KNN - K-Nearest Neighbor, RF - Random Forest, SVM - Support Vector Machines, LR - Logistic Regression, DNN - Deep Neural Networks, MLP - Multi Layer Perceptron, CNN - Convolutional Neural Networks, QDA - Quadratic Discriminant Analysis, LDA - Linear Discriminant Analysis, RBF - Radial Basis Function.

FPR - False Positive Rate, FNR - False Negative Rate, TP - True Positive, TN - True Negative, ROC - Receiver Operating Characteristic, MRR - Missing Report Rate.

validation, aiming at statistical tests such as Wilcoxon and Friedman.

A. Environment

The experiments were executed on a computer with a Linux operating system, Ubuntu 16.04 LTS distribution, Intel Core i7-6700-K processor (8 cores), and 48GB of RAM. The Python3 programming language and the scikit-learn package [28] were used to implement the classifiers. The imbalanced-learn package [30] was used for the undersampling techniques.

B. Evaluation metrics

In this section, the metrics used in this work are presented. In this work, the datasets contain 14 malicious traffic and one normal in an unbalanced way, as previously discussed. Therefore, it is necessary for an evaluation that considers the assertiveness among all types of network flow in a weighted way. According to [31] and [32], the unbalance between classes can ”mask” the results and consequently lead to precipitate conclusions. Thus, the following weighted metrics are presented.

1) Accuracy (AC): It is the most intuitive metric of classifiers' evaluation because it represents a measure of the proportion of correct predictions without considering false positives or negatives. Given X as a set of test elements for each label of the classes that were predicted and $corr(:)$, a function that counts the number of correct predictions, the accuracy can be defined by $Acc(X) = \frac{corr(X)}{jXj}$

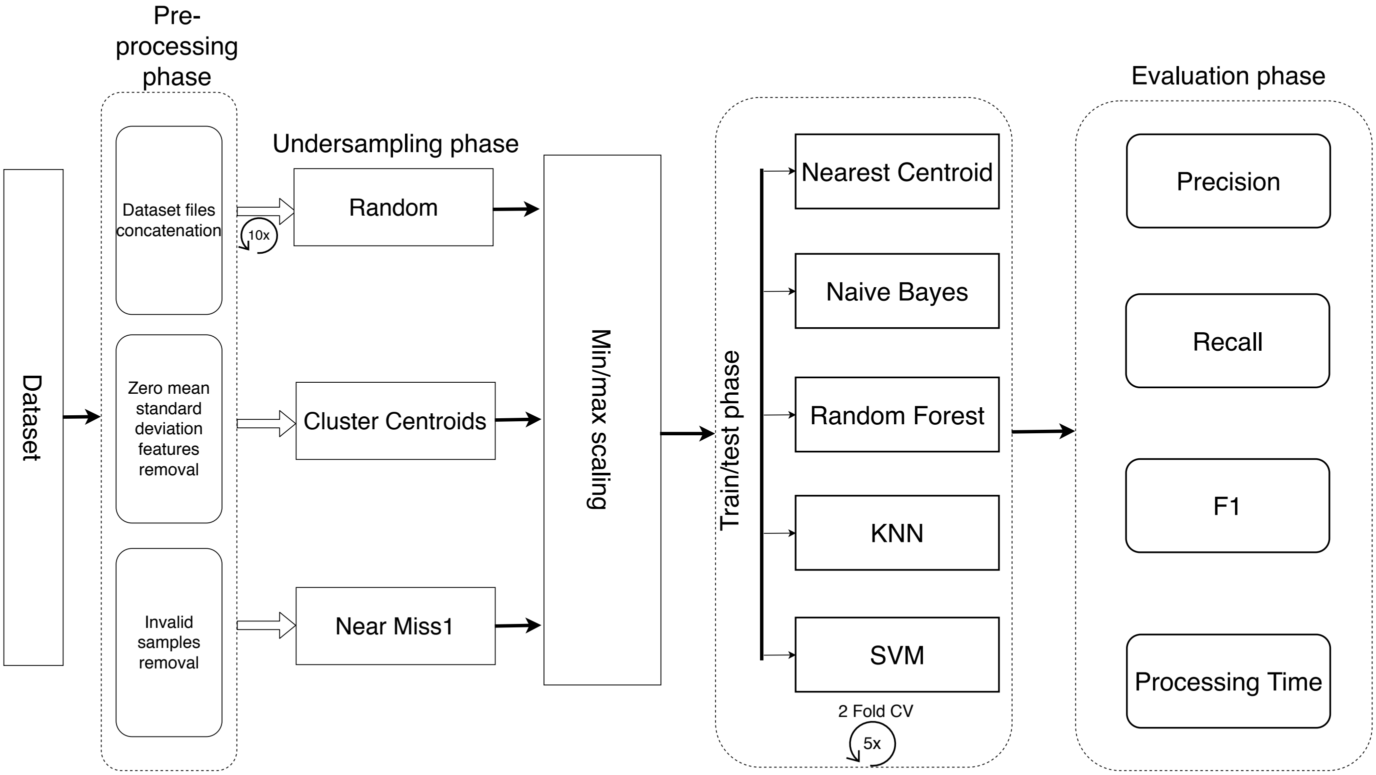


Fig. 1: High-level experiments workflow.

TABLE II: Original and undersampled datasets comparison for CICIDS2017 dataset

Class	Samp.	CL / UR	# of Sub.	CL / UR
Benign	2273097	80.3 / 1:1	33526	19 / 1:1
DoS Hulk	231073	8.1 / 9:1	33526	19 / 1:1
PortScan	158930	5.6 / 14:1	33526	19 / 1:1
DDoS	128027	4.5 / 17:1	33526	19 / 1:1
GoldenEye	10293	0.36 / 220:1	10293	5.9 / 1:3
FTP-Patator	7938	0.28 / 286:1	7938	4.5 / 1:4
SSH-Patator	5897	0.20 / 385:1	5897	3.3 / 1:5
Slowloris	5796	0.20 / 392:1	5796	3.3 / 1:5
SlowHTTPtest	5499	0.19 / 413:1	5499	3.3 / 1:16
Bot	1966	0.069 / 1156:1	1966	1.1 / 1:17
Brute Force	1507	0.053 / 1508:1	1507	0.8 / 1:22
XSS	652	0.0023 / 3486:1	652	0.3 / 1:52
Infiltration	36	0.00012 / 63141:1	36	0.2 / 1:931
SQL Injection	21	0.000074 / 108242:1	21	0.1 / 1:1596
Heartbleed	11	0.000038 / 206645:1	11	0.06 / 1:3047
Total	2830743	100	173707	100

CL - Percentage of class label, UR - Unbalanced ratio: proportion of minority class with respect to majority ones.

TABLE III: Original and undersampled datasets comparison for CICIDS2018 dataset

Class	Samp.	CL / UR	# of Sub.	CL / UR
Benign	13609917	83.20 / 1:1	16006	9.2 / 1:1
HOIC	686012	4.19 / 19:1	16006	9.2 / 1:1
LOIC-HTTP	576191	3.52 / 23:1	16006	9.2 / 1:1
DoS Hulk	461912	2.82 / 29:1	16006	9.2 / 1:1
Bot	286191	1.74 / 47:1	16006	9.2 / 1:1
FTP-Brute	193354	1.18 / 70:1	16006	9.2 / 1:1
SSH-Brute	187589	1.14 / 72:1	16006	9.2 / 1:1
Infiltration	160639	0.98 / 84:1	16006	9.2 / 1:1
SlowHTTPTst	139890	0.85 / 97:1	16006	9.2 / 1:1
GoldenEye	41508	0.25 / 327:1	16006	9.2 / 1:1
Slowloris	10990	0.067 / 1238:1	10990	6.3 / 1:1
LOIC-UDP	1730	0.0105 / 7867:1	1730	0.99 / 1:9
Brute-Force	611	0.0037 / 22274:1	611	0.35 / 1:26
XSS	230	0.0014 / 59173:1	230	0.13 / 1:69
SQL-Injection	87	0.00053 / 156435:1	87	0.05 / 1:183
Total	16356851	100	173708	100

CL - Percentage of class label, UR - Unbalanced ratio: proportion of minority class with respect to majority ones.

2) Recall: Is the proportion of positive cases correctly identified. The following formulas are required:

y is the set of samples predicted by the model
$\hat{y}$ is the set of test labels
L is the set of labels
y_l is the subset of y with label l

$$R_{weighted}(y,\hat{y}_l) = \frac{1}{\sum_{l\in L}|\hat{y}_l|}\sum_{l\in L}|\hat{y}_l|R(y_l,\hat{y}_l), \qquad (1)$$

where $R(y_l,\hat{y}_l) = \frac{|y_l \cap \hat{y}_l|}{|\hat{y}_l|}$

3) Precision: This metric represents the fraction of correct positive predictions. It is a complementary metric to the previous one. The equation 2 presents its formulation:

$$Pr_{weighted}(y,\hat{y}_l) = \frac{1}{\sum_{l\in L}|\hat{y}_l|}\sum_{l\in L}|\hat{y}_l|P(y_l,\hat{y}_l) \qquad (2)$$

where $P(y_l,\hat{y}_l) = \frac{|y_l \cap \hat{y}_l|}{|\hat{y}_l|}$.

4) F1: Aiming to evaluate the trade-off between the two metrics mentioned above, the efficiency measurement, also called F , is used.

$$F_{weighted}(y;\hat{y}_l) = \frac{1}{\sum_{l \in L} |\hat{y}_l|} \overset{max}{\sum_{l \in L}} |\hat{y}_l| F(y_l;\hat{y}_l); \quad (3)$$

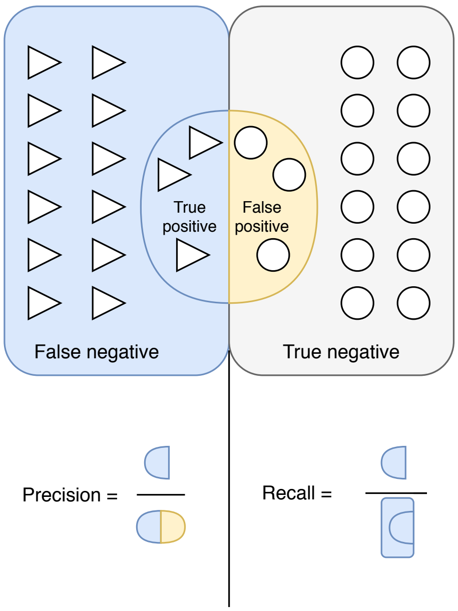
where,

$$F = (1 + {}^2)\frac{P(A;B) \quad R(A;B)}{P(A;B) + R(A;B)} \quad (4)$$

According to [31], the parameter can be any positive value but is usually set to 1. In this case, the metric is simplified and is given by $F = \frac{2 \cdot Pr \cdot R}{Pr + R}$, which is the harmonic mean between Precision and Recall.

Each metric, except the accuracy, has its weight calculated based on the number of samples. Besides, precision and Recall are complementary metrics. Fig. 2 illustrates this behavior, where one notices that the Recall is calculated by the ratio between the number of true positives and the total of triangles. At the same time, the precision takes into account the amount of triangle for the sum of true and false positives.

Fig. 2: Precision x Recall



Thus, precision is the ratio between the number of elements of a class correctly predicted by the classifier (true positives), by the sum of the quantities of elements of a class correctly predicted and the number of elements of other classes wrongly predicted as elements of it (false positives). The Recall is the ratio between the number of elements of a class correctly predicted by the classifier (true positives) and the sum of the rate of true positives and the number of elements of the same class wrongly predicted (False negatives). Thus, precision can be used in situations where False positives are considered more harmful than False negatives. In the case of Recall, it is used when false negatives are more harmful than false positives. This work uses metrics weighted according to the number of samples of each class according to the Equations 1, 2 and 3.

C. Wilcoxon Signed-Ranks Test

The comparison between classifiers using the statistical non-parametric Wilcoxon test aims to evaluate differences between the pairs of metrics the same classifier in each sub-sample. This statistical test is an alternative to the paired T-test, which ranks the differences ignoring the signs [33]. In this work, we compare each classifier in pairs of sub-sampling different scenarios A and B. There are two hypotheses to be evaluated for classifier given sub-sampling scenarios A and B:

> Null hypothesis: Is the default assumption for the test and means that there is no difference between a classifier metric in A and B sub-sample scenarios.
>
> Alternative hypothesis: The classifier scores in the B scenario are higher than the A ones.

The T and z values represent the minimum value of the sum of the ranks used and the test's value. Suppose the T value is lower than the critical T according to the chosen significance level, and the absolute value of z is higher than the critical value. In that case, we can reject the null hypothesis. Consequently, the alternative hypothesis becomes valid. In the latter, the samples are statistically different, and we can infer that the classifier B metrics have higher medians than the A ones.

IV. Results

This section presents the results concerning the employing of undersampling techniques for IDS. We also show an exploratory analysis of the databases considered in this study. Moreover, we compare results for each undersampled sub-bases from Random, Cluster centroids, and NearMiss1 techniques. We applied the Wilcoxon statistical test to assert the inter-classifiers performances. In this way, we can find statistical differences between the metric pairs, as described in the previous section. At the same time, we give specific discussions for each result and general critical comments about classifiers' characteristics and undersampled datasets.

Exploratory analysis of CICIDS2017 and CICIDS2018 datasets

We present an analysis of the CICIDS2017 and CICIDS2018 as follows. The analysis of the CICIDS2017 dataset shows that for benign traffic, there are 53,788 unique ports in use where the most employed ports were 53 (DNS) and 443 (HTTPS). On the other hand, for attacks, there existed 1,686 ports in use were the most employed was 53 (DNS) and 443 (HTTPS).

Table IV shows the mean, minimum, and maximum values on a feature subset of CICIDS2017. We can note that some predictors such as flow_duration and flow_bytes/s presents amplitude variations along with the samples, evidencing the need for scaling to a given range. Furthermore, some features such as bwd_psh_flags, bwd_urg_flags, and fwd_avg_bytes/bulk have zero values, indicating that there is no variation between them along the flow traffics. We also present the data skew in the dataset. Thus, we can note that most features have the distribution of predictors shifted to the right of the mean. Such

TABLE IV: A CICIDS2017 fragment of the feature set description

Feature	Mean	Min	Max	Skew
Total_fwd_packets	9.36	1	$2{:}1 \times 10^{5}$	244.25
Total_backward_ packets	10.40	0	$2{:}0 \times 10^{5}$	244.55
Total_length_of_fwd_packets	549.85	0	$1{:}2 \times 10^{7}$	805.16
Subflow_fwd_bytes	549.84	0	$1{:}2 \times 10^{7}$	803.19
Bwd_psh_flags	0	0	0	0
Bwd_urg_flags	0	0	0	0
Fwd_avg_bytes/bulk	0	0	0	0
Fwd_avg_packets/bulk	0	0	0	0
Fwd_avg_bulk_rate	0	0	0	0
Bwd_avg_bytes/bulk	0	0	0	0
Bwd_avg_packets/bulk	0	0	0	0
Bwd_avg_bulk_rate	0	0	0	0

TABLE V: A CICIDS2018 fragment of the feature set description

Feature	Mean	Min	Max	Skew
Flow_iat_std	$1{:}2 \times 10^{6}$	0	$4{:}7 \times 10^{11}$	1,213.0
Pkt_len_var	$4{:}1 \times 10^{4}$	0	$5{:}1 \times 10^{8}$	1,866.9
Idle_mean	$5{:}1 \times 10^{6}$	0	$3{:}9 \times 10^{11}$	1,266.3
Idle_min	$4{:}7 \times 10^{6}$	0	$2{:}3 \times 10^{11}$	3,335
Bwd_psh_flags	0	0	0	0
Bwd_urg_flags	0	0	0	0
Fwd_avg_bytes/bulk	0	0	0	0
Fwd_avg_packets/bulk	0	0	0	0
Fwd_avg_bulk_rate	0	0	0	0
Bwd_avg_bytes/bulk	0	0	0	0
Bwd_avg_packets/bulk	0	0	0	0
Bwd_avg_bulk_rate	0	0	0	0

behavior may interfere with probability-based classifiers, such as Naive Bayes.

The analysis of the CICIDS2018 dataset shows that for benign traffic, there are 64,665 unique ports in use where the most employed ports were 53 (DNS) and 443 (HTTPS). On the other hand, for attacks, there existed 8,374 ports in use were the most employed was 80 (HTTP) and 21 (FTP).

Table V, presents the mean, maximum, and minimum values of a feature subset on the CICIDS2018 dataset. Note that the same features found in the previous dataset have all zero values, requiring dropping these predictors. In respect to skewness, note that the features have right-shifted asymmetry for both attacks and benign traffics.

A. CICIDS2017 dataset experiments

Table VI highlights the results from experiments concerning the CICIDS2017 dataset. Note that for this database on the all-data schema, the KNN algorithm achieved the best metrics for assertiveness. However, their processing time achieved low performance when compared to the other classifiers. The SVM classifier was not evaluated on the all-data schema due to algorithm inherent computational cost.

The all-data schema results show that despite both KNN and Nearest centroid models was distance-based ones. These models presented performance differing up to 43% for the accuracy metric. The data geometry of the all-data schema is sparse and impairs the class's representativity by its centroid.

Regarding accuracy and recall of the Naive Bayes algorithm, the poor performance is justified by the skewness in the entire database since the classifier uses probability as its basis for model recognition and prediction. Further investigations concerning data preprocessing is a promising way.

The Random Forest model achieved good performance on the all-data schema. Here, we highlighted the relationship between model training time and assertiveness. In this scenario, the most appropriate classifier would be the Random Forest, which has a lower assertiveness than KNN, but their training time has attractive values.

Concerning the Random undersampling scenario, the NC classifier presents a lower assertiveness. Thus, indicating that the undersampled dataset has overlapping centroids. On the other hand, the NB algorithm presented better assertiveness metrics into Random schema than the all-data one. We also evaluated the SVM classifier in this scenario. This model achieved intermediate performance and higher processing time. The Random Forests model obtained less assertiveness than all-data due to the lower number of samples in the undersampled base.

On Cluster centroids undersampled schema, we note that the classifiers have greater assertiveness than the previous scenarios, indicating that the undersampling technique has better representativity than the previous ones. Note that the NC, NB, and SVM algorithms increased up the performance metrics. Moreover, we highlight that those distance-based algorithms performed well in terms of the evaluation metrics.

In the results of the experiments using the NearMiss1 undersampling approach, the classifiers' performance is generally higher than in the Random and lower than Cluster centroids undersampling scenario, except for the KNN and SVM classifiers, which have similar performance on assertiveness and time.

Comparison of classifiers under different undersampling techniques

In this sub-section, we present a comparison between the classifiers under different undersampling techniques. We further supported our results with statistical tests. The results of the experiments are presented in Table VI. We discuss two metrics for scenario comparison: accuracy, which represents an overview of the classifier, and F1 score, which is the harmonic mean between recall and precision.

The comparison between the sub-sets and the entire CICIDS2017 dataset for the NC classifier was made initially by accuracy metric. The undersampling by Cluster centroids presented the best metric for this classifier, followed by NearMiss1, indicating that these techniques applied to the evaluated dataset selected the most representative samples. We can observe that the Random undersampling has similar results to the all-data schema for this classifier, indicating that centroids in a Random sub-sampling scenario do not represent the generated data.

Concerning the F1 metric, the cluster centroid undersampling technique is the most suitable for the NC classifier, above to 80%. In contrast, Random undersampling is the least suitable since the classifier produced a higher false-positive rate than in an entire base scenario.

TABLE VI: CICIDS2017 - performance evaluation

Algorithm	Accuracy mean/std		Precision mean/std		Recall mean/std		F1 mean/std		Time (sec.) mean/std	
Entire dataset										
NC	0.55	$1{:}30 \times 10^{3}$	0.85	$5{:}19 \times 10^{4}$	0.55	$8{:}23 \times 10^{4}$	0.64	$1{:}30 \times 10^{3}$	4.55	$3{:}33 \times 10^{2}$
NB	0.44	$7{:}64 \times 10^{2}$	0.96	$1{:}28 \times 10^{3}$	0.44	$7{:}64 \times 10^{2}$	0.57	$8{:}46 \times 10^{2}$	5.15	$6{:}10 \times 10^{2}$
KNN	**0.98**	$5{:}56 \times 10^{4}$	**0.98**	$5{:}55 \times 10^{4}$	**0.98**	$5{:}56 \times 10^{4}$	**0.98**	$5{:}54 \times 10^{4}$	**1625.54**	$8{:}30 \times 10^{1}$
RF	0.88*	$1{:}87 \times 10^{4*}$	0.82*	$3{:}13 \times 10^{4*}$	0.88*	$1{:}87 \times 10^{4*}$	0.85*	$2{:}56 \times 10^{4*}$	64.03*	$1{:}60 \times 10^{1*}$
SVM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Random undersampling										
NC	0.57	$1{:}08 \times 10^{2}$	0.64	$5{:}29 \times 10^{3}$	0.57	$1{:}08 \times 10^{2}$	0.56	$1{:}51 \times 10^{2}$	0.36	$4{:}59 \times 10^{3}$
NB	0.84	$5{:}08 \times 10^{3}$	0.94	$4{:}18 \times 10^{3}$	0.84	$5{:}08 \times 10^{3}$	0.88	$4{:}40 \times 10^{3}$	0.42	$5{:}03 \times 10^{3}$
KNN	**0.98**	$3{:}26 \times 10^{4}$	**0.98**	$3{:}17 \times 10^{4}$	**0.98**	$3{:}26 \times 10^{4}$	**0.98**	$3{:}25 \times 10^{4}$	**3.21**	$2{:}66 \times 10^{1}$
RF	0.70	$6{:}13 \times 10^{3}$	0.55	$2{:}28 \times 10^{3}$	0.70	$6{:}13 \times 10^{3}$	0.61	$5{:}08 \times 10^{3}$	3.04	$2{:}73 \times 10^{2}$
SVM	0.88	$1{:}82 \times 10^{2}$	0.86	$1{:}77 \times 10^{2}$	0.88	$1{:}82 \times 10^{2}$	0.86	$1{:}82 \times 10^{2}$	198.76	$1{:}69 \times 10^{1}$
Cluster Centroids undersampling										
NC	0.81	8.47e-03	0.85	$8{:}78 \times 10^{3}$	0.81	$8{:}47 \times 10^{3}$	0.82	$8{:}36 \times 10^{3}$	0.28	$5{:}14 \times 10^{3}$
NB	0.96*	$5{:}83 \times 10^{4*}$	0.97*	$7{:}41 \times 10^{4*}$	0.96*	$5{:}83 \times 10^{4*}$	0.96*	$6{:}54 \times 10^{4*}$	0.31*	$2{:}43 \times 10^{3*}$
KNN	**0.99**	$1{:}09 \times 10^{4}$	**0.99**	$1{:}23 \times 10^{4}$	**0.99**	$1{:}09 \times 10^{4}$	**0.99**	$1{:}09 \times 10^{4}$	**2.85**	1:20
RF	0.76	$2{:}37 \times 10^{3}$	0.60	$1{:}53 \times 10^{2}$	0.76	$2{:}37 \times 10^{3}$	0.67	$4{:}96 \times 10^{3}$	3.23	$1{:}36 \times 10^{2}$
SVM	0.93	$4{:}81 \times 10^{3}$	0.93	$4{:}68 \times 10^{3}$	0.93	$4{:}81 \times 10^{3}$	0.92	$5{:}82 \times 10^{3}$	92.60	1.45
NearMiss1 undersampling										
NC	0.71	$9{:}42 \times 10^{3}$	0.78	$1{:}62 \times 10^{2}$	0.71	$9{:}42 \times 10^{3}$	0.70	$1{:}27 \times 10^{2}$	0.28	$3{:}71 \times 10^{3}$
NB	0.89	$2{:}01 \times 10^{3}$	0.95	$1{:}83 \times 10^{3}$	0.89	$2{:}01 \times 10^{3}$	0.91	$2{:}19 \times 10^{3}$	0.32	$1{:}87 \times 10^{2}$
KNN	**0.99**	$1{:}50 \times 10^{4}$	**0.99**	$1{:}65 \times 10^{4}$	**0.99**	$1{:}50 \times 10^{4}$	**0.99**	$1{:}47 \times 10^{4}$	**3.41**	$9{:}04 \times 10^{1}$
RF	0.78	$3{:}59 \times 10^{3}$	0.65	$1{:}47 \times 10^{2}$	0.78	$3{:}59 \times 10^{3}$	0.70	$7{:}05 \times 10^{3}$	3.07	$6{:}43 \times 10^{3}$
SVM	0.93	$1{:}45 \times 10^{3}$	0.93	$1{:}19 \times 10^{3}$	0.93	$1{:}45 \times 10^{3}$	0.92	$1{:}31 \times 10^{3}$	77.04	1.25

NC - Nearest Centroid, NB - Naive Bayes, KNN - K-Nearest Neighbor, RF - Random Forest, SVM - Support Vector Machines

The results for the Naive Bayes classifier in the evaluated scenarios reveal some aspects of the classifier and the dataset. We observe that the NB classifier performance in the all-data schema is less than the undersampled schemas and produces the most significant variation around the mean for all metrics. We believe that such behavior is due to the database's skewness and unbalance factors since the classifier is a probabilistic one. A large number of the majority class elements make the model inaccurate in its predictions. In this sense, the undersampling approach tends to mitigate this effect since reducing the samples occurs in the majority classes, making the model calculate the probabilities better and consequently be more assertive. The Cluster centroid and NearMiss1 approaches stand out concerning Random strategy for accuracy metric because selecting the samples in the undersampled datasets is more criterion than the latter. The same behavior occurs to the F1 metric. The classifier does not deal well with a large amount of data of the entire base, and the solution by undersampling presents significant differences in precision and recall.

For the Random Forest classifier, their accuracy values performed well in the entire database. When comparing with the sub-samples schemas, the RF classifier evaluated in Random sub-sampling has the lowest average accuracy. Concurrently, for Cluster centroids and NearMiss1 scenarios, the algorithm achieved similar assertiveness.

K-nearest neighbor's classifier presented levels of around 99% on all metrics for all scenarios. However, the processing time is higher than almost all classifiers since it is an instance-based algorithm that suffers from the amount of data [12].

Regarding the SVM classifier's accuracy for undersampled datasets, the sub-samples by Cluster centroids and NearMiss1 show results above 90% accuracy for this classifier. However, in the Random sub-base, the accuracy is lower than the ones. Processing time is an issue for this algorithm because their compute and storage requirements increase rapidly with the number of training vectors [28].

Table VII illustrates the values found for the Wilcoxon statistical test for evaluated classifiers concerning all undersampling techniques.

Concerning Random and Cluster centroids / Random and NearMiss1 undersampling scenarios comparisons, the values of T and z absolute found are, respectively, lower and higher for all classifiers in all metrics with the significance of 99%, statistically evidencing the feasibility of using undersampling by Cluster centroids and NearMiss1 compared to the Random approach.

When comparing the sub-bases by NearMiss1 and Cluster centroids through the Wilcoxon test, we can note that the later sub-base metrics are statistically different. The T and z values in the table correspond to the rejection of the null hypothesis and adoption of the alternative hypothesis, which indicates that the median of the NC's metrics, NB classifiers are higher than those of the other subsample. However, for the KNN

TABLE VII: Wilcoxon statistical test for all undersampling techniques considering CICIDS2017 dataset

Values of T and z for n = 10.								
Algorithm	Accuracy T / z		Precision T / z		Recall T / z		F1 T / z	
Random and Cluster centroids / Random and NearMiss1 comparisons								
NC	0	2.8	0	2.8	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
KNN	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8
SVM	0	2.8	0	2.8	0	2.8	0	2.8
Cluster centroids and NearMiss1 comparison								
NC	0	2.8	0	2.8	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8
SVM	0	2.5	10	1.78	0	2.5	21	0.66
Entire dataset and Cluster Centroids/NearMiss1 comparison								
NC	0	2.8	0	2.8	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
KNN	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8
Entire dataset and Random comparison								
NC	0	2.8	0	2.8	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8

critical T = 5, critical z = 2.33 and 0.001 of significance

classifier, no numerical or statistical differences were found for the number of decimal precision equal to three. Therefore it is not possible to reject the null hypothesis. Also, the precision and F1 score metrics in the SVM classifier are statistically equal for both sub-bases.

Concerning comparisons between the all-data schema and sub-bases by Cluster centroids, Random and NearMiss1, the test indicates the following behaviors:

When comparing the Random with Cluster centroid and NearMiss1 scenarios, the latter presented the best medians of all classifiers' metrics.

For Cluster Centroids and NearMiss1 comparison, the former scenario presented the best medians of metrics for all classifiers, except KNN.

Concerning the comparison between the entire dataset and Cluster centroids/NearMiss1 scenarios, we highlight that the latter presented the best medians of metrics for all evaluated classifiers.

Regarding the all-data schema and Random scenario, the latter presented the best medians of metrics for all evaluated classifiers, except KNN.

Therefore, we can infer that the use of undersampling techniques favors most classifiers' metrics in all scenarios.

B. CICIDS2018 dataset experiments

We present the experiments' results of CICIDS2018 database in Table VIII. We observed that the Nearest centroid classifier does not perform well on the all-data schema, indicating that the centroids do not represent the classes. Moreover, by comparing both CICIDS2017 and CICIDS2018 experiments, the latter is five times larger than the former, and the unbalance factor is more evident in CICIDS2018.

The metrics for the Naive Bayes classifier performed well in CICIDS2018 than the previous one. Moreover, we suggest caution when using the results. The high imbalanced factor aligned with the amount of data has some drawbacks concerning weighted metrics usage. The Naive Bayes classifier has, for CICIDS2018, the most significant weight for the legitimate traffic class. In the test portion, these samples category possibly occur more often due to their large number. Thus, it requires more experimental analysis to give us insights into the data imbalance factor.

On the all-data schema, we observe that the Random Forests classifier performed well. The results differ from the previous ones. Here the most suitable classifier is Naive Bayes, which has acceptable assertiveness in less time. The SVM and KNN classifiers were not evaluated in this scenario due to the high computational complexity needed to execute the experiments into a dataset five times higher than CICIDS2017.

Concerning the Random sub-base, we discuss the results by relating the assertiveness with the all-data schema as follows: The NC classifier maintained the low assertiveness by comparing the results with all-data schema. Despite an increase of 21% into accuracy, their low performance in precision and recall still indicating that the undersampled dataset has poor representation by the centroid of each class.

The NB algorithm presents lower results than in the all-data schema in terms of the evaluated metrics. It reveals a behavior on the entire dataset: we adopted weighted metrics by the number of elements in the class. Thus, the classifier that is assertive for a specific majority class obtains acceptable results in assertiveness for the overall results. In this case, the entire CICIDS2018 data's undersampling technique reduced their imbalance factor and indirectly impacted the classification assertiveness.

The Random Forests algorithm obtained less assertiveness concerning the all-data scenario, maintaining the same behavior in the CICIDS2017 experiments. On the other hand, the KNN and SVM classifiers obtained results close to each other.

Regarding Cluster centroid undersampled sub-base, the classifiers present greater assertiveness concerning the previous scenarios. It is worth noting that the NC, NB, and SVM algorithms presented the best assertiveness concerning the other scenarios presented above. We observed that the distance-based algorithms such as NC, KNN, and SVM, performed well within the Cluster-centroid undersampling technique.

For the NearMiss1 undersampling technique, the complete classifiers' set achieved overall performance higher than in the Random scenario. On the other hand, Nearmiss1 assertiveness is lower than the cluster-centroid schema. Thus, indicating that the cluster-centroid undersampling approach has representativeness than their counterparts.

Comparison between classifiers under sub-samples strategies

In this sub-section, we show a comparison between the classifiers under different undersampling techniques and support our observations with statistical tests. The results of the experiments are presented in Table VIII. The same metrics of the previous scenario were chosen for comparison. Taking

TABLE VIII: CICIDS2018 - performance evaluation

Algorithm	Accuracy mean/std		Precision mean/std		Recall mean/std		F1 mean/std		Time (sec.) mean/std	
Entire dataset										
NC	0.26	$7{:}74 \times 10^{3}$	0.90	$6{:}20 \times 10^{3}$	0.26	$7{:}74 \times 10^{3}$	0.35	$6{:}72 \times 10^{3}$	3.75	1.35
NB	**0.80**	$8{:}54 \times 10^{5}$	**0.86**	$1{:}68 \times 10^{4}$	**0.80**	$8{:}54 \times 10^{5}$	**0.81**	$1{:}30 \times 10^{4}$	**5.86**	$5{:}61 \times 10^{1}$
RF	0.83	$1{:}33 \times 10^{4}$	0.69	$2{:}21 \times 10^{4}$	0.83	$1{:}33 \times 10^{4}$	0.75	$1{:}87 \times 10^{4}$	410.86	$7{:}21 \times 10^{1}$
KNN	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A*
SVM	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Random undersampling										
NC	0.47	$5{:}64 \times 10^{3}$	0.54	$2{:}23 \times 10^{2}$	0.47	$5{:}64 \times 10^{3}$	0.45	$6{:}58 \times 10^{3}$	0.39	$3{:}52 \times 10^{2}$
NB	0.76	$1{:}52 \times 10^{3}$	0.85	$5{:}08 \times 10^{3}$	0.76	$1{:}52 \times 10^{3}$	0.76	$4{:}35 \times 10^{3}$	0.39	$3{:}15 \times 10^{2}$
KNN	**0.85**	$2{:}35 \times 10^{3}$	**0.83**	$8{:}40 \times 10^{3}$	**0.85**	$2{:}35 \times 10^{3}$	**0.82**	$5{:}75 \times 10^{3}$	**6.87**	**1.20**
RF	0.74	$4{:}15 \times 10^{3}$	0.73	$1{:}02 \times 10^{2}$	0.74	$4{:}15 \times 10^{3}$	0.68	$6{:}44 \times 10^{3}$	2.58	$8{:}15 \times 10^{2}$
SVM	0.81	$6{:}25 \times 10^{4}$	0.84	$7{:}55 \times 10^{4}$	0.81	$6{:}25 \times 10^{4}$	0.79	$8{:}13 \times 10^{4}$	151.50	1.44
Cluster Centroids undersampling										
NC	0.88	$3{:}71 \times 10^{3}$	0.89	$4{:}42 \times 10^{3}$	0.88	$3{:}71 \times 10^{3}$	0.87	$3{:}00 \times 10^{3}$	0.34	$2{:}41 \times 10^{2}$
NB	0.98*	$5{:}37 \times 10^{3}$*	0.98*	$4{:}44 \times 10^{3}$*	0.98*	$5{:}37 \times 10^{3}$*	0.98*	$5{:}08 \times 10^{3}$*	0.36*	$2{:}41 \times 10^{3}$*
KNN	**0.99**	$1{:}67 \times 10^{4}$	**0.99**	$1{:}65 \times 10^{4}$	**0.99**	$1{:}67 \times 10^{4}$	**0.99**	$1{:}66 \times 10^{4}$	**6.70**	**3.26**
RF	0.85	$5{:}11 \times 10^{3}$	0.82	$2{:}06 \times 10^{2}$	0.85	$5{:}11 \times 10^{3}$	0.80	$8{:}02 \times 10^{3}$	2.87	$2{:}97 \times 10^{1}$
SVM	0.97	$6{:}95 \times 10^{4}$	0.97	$7{:}77 \times 10^{4}$	0.97	$6{:}95 \times 10^{4}$	0.96	$7{:}61 \times 10^{4}$	68.98	7.10
NearMiss1 undersampling										
NC	0.86	$8{:}13 \times 10^{4}$	0.88	$6{:}44 \times 10^{4}$	0.86	$8{:}13 \times 10^{4}$	0.86	$8{:}12 \times 10^{4}$	0.37	$6{:}19 \times 10^{3}$
NB	0.91*	$3{:}86 \times 10^{3}$*	0.92*	$3{:}23 \times 10^{3}$*	0.91*	$3{:}86 \times 10^{3}$*	0.90*	$3{:}90 \times 10^{3}$*	0.44*	$2{:}35 \times 10^{2}$*
KNN	**0.92**	$1{:}64 \times 10^{2}$	**0.92**	$3{:}32 \times 10^{2}$	**0.92**	$1{:}64 \times 10^{2}$	**0.90**	$2{:}95 \times 10^{2}$	**9.17**	**3.73**
RF	0.84	$8{:}24 \times 10^{3}$	0.85	$8{:}91 \times 10^{3}$	0.84	$8{:}24 \times 10^{3}$	0.81	$8{:}17 \times 10^{3}$	1.83	$5{:}77 \times 10^{2}$
SVM	0.84	$6{:}66 \times 10^{4}$	0.81	$7{:}77 \times 10^{4}$	0.84	$6{:}66 \times 10^{4}$	0.81	$7{:}88 \times 10^{4}$	73.24	$6{:}85 \times 10^{1}$

NC - Nearest Centroid, NB - Naive Bayes, KNN - K-Nearest Neighbor, RF - Random Forest, SVM - Support Vector Machines

the accuracy metric as parameter for comparison between the sub-sets and the entire CICIDS2018 dataset for the NC classifier. As the previous scenario, the classifier has lower metrics for the all-data schema compared to the undersampling approaches. The undersampling by Cluster centroids presented the best metrics for this classifier, followed by NearMiss1, indicating that these techniques provide higher representativity to the database. However, the Random undersampling has a different behavior from their counterparts due to its Random characteristics.

As for F1 score metrics, Cluster centroids' undersampling is the most appropriate for this classifier, which has a score above 87%. Simultaneously, the all-data approach presents the worst F1 score values since the classifier produced a higher false positives rate than in other scenarios. We highlighted the metric recall in Table VIII to evidence this behavior.

Concerning the Naive Bayes classifier, the accuracy metric performance in the Random sub-base is lower than in other scenarios. This fact occurs due to the different U_R in the entire base and the sub-bases, causing the classifier to generate many false positives. Thus, as occurred in the CICIDS2017 base, the approaches by Cluster centroids and NearMiss1 stand out concerning Random undersampling approaches because the sample selection in this scenario has a well-defined sample selection schema than in the latter.

The same behavior occurs to the F1 metric. The classifier does not deal well with the large mass of data of the entire database, and the solution by undersampling presents significant differences in accuracy/recall.

Regarding the Random Forest classifier's accuracy results, the all-data schema achieved higher performance than the Random undersampling strategy. This behavior is due to two reasons: the Random Forest classifier, when dealing with unbalanced data, favors the majority class for the prediction. In the tests, the majority class appears more often. Therefore, the Random sub-base schema does not have specific criteria for sample selection, making it challenging to represent the entire database for classification purposes.

When comparing the to other sub-samples schemas, the Random Forest classifier reached assertiveness above 85% for the Cluster centroids and NearMiss1 schemas. The first presents higher metrics for this classifier. Concerning the F1 score, the RF classifier maintained the same behavior as explained regarding data representativity.

For the KNN classifier's accuracy on each undersampling schema, we can observe that the Cluster centroids approach presents greater accuracy than their counterparts. Thus, indicating that schema selects the most representative samples for the CICIDS2018 dataset. Despite the feasibility of the KNN's processing time for an IDS project on subsampled datasets, the testing time is too high in the all-data schema that considers this classifier unfeasible in a real context. The same behavior occurs for the F1 score.

Concerning the SVM classifier's accuracy for the undersam-

TABLE IX: Wilcoxon statistical test for all undersampling techniques considering CICIDS2018 dataset

T and z values for n = 10								
Algorithm	Accuracy T / z		Precision T / z		Recall T / z		F1 T / z	
NearMiss1 x Cluster Centroids Random x NearMiss1 comparisons								
NC	0	2.8	0	2.8	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
KNN	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8
SVM	0	2.8	0	2.8	0	2.8	0	2.8
Entire dataset x Cluster centroids comparison								
NC	0	2.8	**0**	**2.65**	0	2.8	0	2.8
NB	0	2.8	0	2.8	0	2.8	0	2.8
RF	0	2.8	0	2.8	0	2.8	0	2.8

critical T = 5, critical z = 2.33 and 0.001 of significance

pled datasets, the Cluster centroids and NearMiss1 schemas show results above 84% accuracy for the SVM. At the same time, in the Random undersample sub-bases, the accuracy is lower. The sub-bases generated by Cluster centroids has the best representativity, being possible to achieve rates of up to 97%. The same behavior occurs for the F1 score.

We compare our experiments through Wilcoxon statistical test. Table IX shows the test all scenarios.

The statistical test confirms that the metrics in NC, NB, and KNN, RF, and SVM classifiers are superior in a scenario of undersampling by Cluster centroids comparing to other approaches.

Concerning NearMiss1 x Cluster Centroids / Random x NearMiss1 undersampling scenarios comparisons, the absolute values of T and z are, respectively, lower and higher for all classifiers in all metrics with the significance of 99%. They are statistically evidencing the feasibility of using undersampling by Cluster centroids and NearMiss1 compared to the Random approach.

When comparing the results between the all-data schema and Cluster centroids undersampling through the Wilcoxon test, we can note that the latter metrics are statistically different. The T and z values correspond to the null hypothesis rejection and the alternative hypothesis adoption, indicating that the median of all classifier metrics is higher than those of the all-data schema.

Therefore, we can infer that the use of undersampling techniques overcomes the all-data scenarios for most classifiers, particularly the Cluster centroid schema overcomes all counterparts.

Based on the obtained results, we can observe that the use of representative undersampling techniques attempts to deal with unbalanced databases and the use of cost-intensive classifiers, aiming at increasing the assertiveness in classifiers as the basis for Intrusion Detection Systems design. In our experiments, KNN presented the best metrics. However, it is observed some drawbacks on the trade-off with processing time and assertiveness.

Our results indicate that the Cluster centroid approach favored in all scenarios the distance-based classifiers as well as the Naive Bayes one, significantly increasing the assertiveness metrics and decreasing the model recognition time. The results also showed that, through statistical testing, the classifiers' metrics in their majority are higher in classifiers evaluated under the undersampled dataset from this approach. In this scenario, we recommend the use of Cluster centroid-based undersampling when evaluating distance-based algorithms. We employed five metrics to evaluate the classifiers for the undersampling techniques.

To support the decision-making for the best classifier, the IDS designer should use one of the evaluation metrics or a combination. In this paper, we used five metrics to evaluate machine learning algorithms. Two assertiveness metrics was adopted as global decision-criteria: accuracy and F1-score. We also used the processing time as an additional criterion for the feasibility of IDS design.

In the complete and undersampled datasets, the KNN classifier obtained the highest assertiveness metrics. However, the compromise between assertiveness and processing time is observed, as discussed in previous sections. Our findings agree with the work of Silva Neto [12], which obtains competitive results for the KNN classifier. However, the testing time was decisive for the choice of other classifiers for the IDS project.

Based on decision-criteria for classifiers choice, the all-data schema for the CICIDS2017 database shows that the Random Forest classifier obtained the best results, with average scores of 88.6%, 85.0%, 64s for accuracy, F1, and time respectively. For the all-data schema, the CICIDS2018 database shows average results of 83.2%, 75.6%, and 410s for accuracy, F1, and time respectively.

As for Random undersampling schema on CICIDS2017 and CICIDS2018 datasets, the KNN classifier achieved the best performance values of accuracy, F1 score, and processing time, with scores equal to 98.7%, 98.7%, 3.2 seconds for the sub-sampled CICIDS2017 and 85.5%, 82.8%, 6.8 seconds for the sub-sampled CICIDS2018 dataset. Although the Naive Bayes classifier obtained overall low assertiveness metrics in each sub-base, we observe that it is 8 (eight) times faster for undersampled CICIDS2017 and 12 (twelve) times faster than the undersampled CICIDS2018 datasets in terms of processing time.

In the undersampled datasets by Cluster centroids, in both CICIDS2017 and CICIDS2018 datasets, the Naive Bayes classifier obtained the best results by achieving up to 98% in their scores and similar overall rates in each sub-base schema.

For sub-bases by NearMiss1, we observed that for the undersampled CICIDS2017, the KNN classifier is considered the best by achieving 99.4%, 99.4%, and 3.4 seconds for accuracy, F1 score, and processing time. For the undersampled CICIDS2018 dataset, we consider the Naive Bayes classifier the best one. This classifier achieved performances of 91.1%, 90.5%, and 0.4 seconds for accuracy, F1 score, and training time.

V. Conclusion

We presented an evaluation of three undersampling techniques in two up-to-date IDS databases: the CICIDS2017 and CICIDS2018 [13]. The performance of Nearest Centroid, Naive Bayes, Random Forests, K-Nearest Neighbors (KNN), and Support Vector Machines (SVM) algorithms were evaluated in the two complete datasets named as all-data schema, as well as in sub-bases generated by the random selection, Cluster centroids, and NearMiss undersampling techniques.

Our evaluation process aims to support the decision-making for the best couple: classifier and undersampling technique, into the IDS project lifecycle. Thus, the results obtained in both for all-data schema and sub-bases from the KNN classifier presented the best assessments except for the time metric. However, based on the decision-criteria adopted for selecting the best classifiers in the IDS project, for the CICIDS2017 and CICIDS2018 all-data schemas, the classifier Random Forests obtains the best results. As for the sub-base generated from the CICIDS2017 database by the random undersampling, the KNN was considered the best classifier for its average accuracy, efficiency, and training time. In the sub-base using the Cluster centroids technique, generated from CICIDS2018, the Naive Bayes classifier produced the best results. As for the subbases generated from CICIDS2017 and CICIDS2018, using the NearMiss1 undersampling technique, the best classifiers, for their average metrics of accuracy, efficiency and training time, were KNN and Naive Bayes, respectively.

The undersampling process presents sufficient conditions for evaluating different classifiers, including those with high processing time. Moreover, these techniques allow improving the unbalance at the bases and consequently reduce the skewness in them.

The results suggest that Cluster centroids' undersampling technique presents the best performance when applied to distance-based classifiers. Our analysis indicated that the undersampling techniques influence the decision-making for the best classifier in the IDS design process.

The main contributions of this work are:

- The evaluation of undersampling techniques in a systematic way to support the IDS design decision-making for the best classifier;
- Exploratory analysis of the CICIDS2017 and CICIDS2018 datasets;
- An updated analysis of undersampling strategies in IDS designer-domain;
- Production of ready-to-use undersampled datasets by using CICIDS2017 and CICIDS2018 as a basis.

This work did not exhaust the possibilities of researching undersampling techniques and the choice of classifiers in the IDS design process. In future works, we recommend a cluster-based exploratory analysis to understand data geometry. We also suggest the evaluation of data preprocessing schemas to leverage the performance evaluation. Since we evaluate specific attack types, we also recommend further investigations on binary classification schemas, in which one can merge all the attacks to represent an abnormal behavior pattern and reduce the effects of unbalance. There are open questions for further investigations on real-time attack detection and a cross-dataset evaluation among the studied databases. These future works can leverage IDS design-domain.

References

[1] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, July 2019. doi: 10.1186/s42400-019-0038-7. [Online]. Available: https://doi.org/10.1186/s42400-019-0038-7

[2] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 2019. doi: 10.1109/COMST.2018.2847722

[3] M. Peng, X. Xing, T. Gui, X. Huang, Y.-G. Jiang, K. Ding, and Z. Chen, "Trainable undersampling for class-imbalance learning," in *Proc. AAAI Conf. on Artif. Intell.*, vol. 33, July 2019, pp. 4707–4714. doi: 10.1609/aaai.v33i01.33014707

[4] J. Lee and K. Park, "GAN-based imbalanced data intrusion detection system," *Pers. and Ubiquitous Comput.*, Nov. 2019. doi: 10.1007/s00779-019-01332-y. [Online]. Available: https://doi.org/10.1007/s00779-019-01332-y

[5] P. S. Bhattacharjee, A. K. Md Fujail, and S. A. Begum, "A comparison of intrusion detection by K-means and fuzzy C-means clustering algorithm over the NSL-KDD dataset," in *IEEE Int. Conf. ICCIC*, 2017, pp. 1–6. doi: 10.1109/ICCIC.2017.8524401

[6] G. Meena and R. R. Choudhary, "A review paper on IDS classification using KDD99 and NSL-KDD dataset in WEKA," in *IEEE Int. Conf. COMPTELIX*, 2017, pp. 553–558. doi: 10.1109/COMPTELIX.2017.8004032

[7] C. Zhang, F. Ruan, L. Yin, X. Chen, L. Zhai, and F. Liu, "A deep learning approach for network intrusion detection based on NSL-KDD dataset," in *Proc. 13th IEEE Int. Conf. ASID*, 2019, pp. 41–45. doi: 10.1109/ICASID.2019.8925239

[8] R. Thomas and D. Pavithran, "A survey of intrusion detection models based on NSL-KDD data set," in *Proc. 5th IEEE Int. Conf. CTIT*, 2018, pp. 286–291. doi: 10.1109/CTIT.2018.8649498

[9] D. Stiawan, S. Sandra, E. Alzahrani, and R. Budiarto, "Comparative analysis of K-means method and Naïve Bayes method for brute force attack visualization," in *Proc. 2nd IEEE Int. Conf. ICACC*, 2017, pp. 177–182. doi: 10.1109/Anti-Cybercrime.2017.7905286

[10] J. Jiao, B. Ye, Y. Zhao, R. J. Stones, G. Wang, X. Liu, S. Wang, and G. Xie, "Detecting TCP-based DDoS attacks in Baidu cloud computing data centers," in *Proc. 36th IEEE Symp. SRDS*, 2017, pp. 256–258. doi: 10.1109/SRDS.2017.37

[11] P. Bedi, N. Gupta, and V. Jindal, "Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network," *Procedia Comp. Sci.*, vol. 171, pp. 780–789, 2020. doi: 10.1016/j.procs.2020.04.085. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920310541

[12] M. Silva Neto and D. G. Gomes, "Network intrusion detection systems design: A machine learning approach," in *Proc. 37th Brazilian Symp. (SBRC)*, 2019, pp. 932–945. doi: 10.5753/sbrc.2019.7413. [Online]. Available: https://sol.sbc.org.br/index.php/sbrc/article/view/7413

[13] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *Proc. 3rd Int. Conf. ICISSP*, 2018, pp. 108–116. doi: 10.5220/0006639801080116

[14] M. R. Parsaei, S. M. Rostami, and R. Javidan, "A hybrid data mining approach for intrusion detection on imbalanced NSL-KDD dataset," *Int. J. Adv. Comp. Sci. Appl.*, vol. 7, no. 6, 2016. doi: 10.14569/IJACSA.2016.070603. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2016.070603

[15] K. Liu, Z. Fan, M. Liu, and S. Zhang, "Hybrid intrusion detection method based on K-means and CNN for smart home," in *Proc. 8th IEEE Int. Conf. CYBER*, 2018, pp. 312–317. doi: 10.1109/CYBER.2018.8688271

[16] L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, "Classification hardness for supervised learners on 20 years of intrusion detection data," *IEEE Access*, vol. 7, pp. 167 455–167 469, 2019. doi: 10.1109/ACCESS.2019.2953451

[17] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *J. Comp. Sci.*, vol. 25, pp. 152–160, 2018. doi: 10.1016/j.jocs.2017.03.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877750316305099

[18] T. Bhaskar, T. Hiwarkar, and K. Ramanjaneyulu, "Adaptive jaya optimization technique for feature selection in NSL-KDD data set of intrusion detection system," in *Proc. Int. Conf. ICCIP*, 2019. doi: 10.2139/ssrn.3421665

[19] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," *IEEE Access*, vol. 7, pp. 82 512–82 521, 2019. doi: 10.1109/ACCESS.2019.2923640

[20] S. Maza and M. Touahria, "Feature selection for intrusion detection using new multi-objective estimation of distribution algorithms," *Appl. Intell.*, vol. 49, no. 12, pp. 4237–4257, Dec. 2019. doi: 10.1007/s10489-019-01503-7. [Online]. Available: https://doi.org/10.1007/s10489-019-01503-7

[21] I. Ullah and Q. H. Mahmoud, "An intrusion detection framework for the smart grid," in *Proc. 30th IEEE Can. Conf. CCECE*, 2017, pp. 1–5. doi: 10.1109/CCECE.2017.7946654

[22] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012. doi: 10.1016/j.cose.2011.12.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404811001672

[23] D. Aksu and M. Ali Aydin, "Detecting port scan attempts with comparative analysis of deep learning and support vector machine algorithms," in *Proc. Int. Congr. IBIGDELFT*, 2018, pp. 77–80. doi: 10.1109/IBIGDELFT.2018.8625370

[24] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features dimensionality reduction approaches for machine learning based network intrusion detection," *Electronics*, vol. 8, no. 3, 2019. doi: 10.3390/electronics8030322. [Online]. Available: https://www.mdpi.com/2079-9292/8/3/322

[25] T. Kathiresan, D. Maurer, and V. Dellwo, "Highly spectrally undersampled vowels can be classified by machines without supervision," *J. Acoust. Soc. Am.*, vol. 146, no. 1, pp. EL1–EL7, July 2019. doi: 10.1121/1.5111154. [Online]. Available: https://doi.org/10.1121/1.5111154

[26] C.-F. Tsai, W.-C. Lin, Y.-H. Hu, and G.-T. Yao, "Under-sampling class imbalanced datasets by combining clustering analysis and instance selection," *Inf. Sci.*, vol. 477, pp. 47–54, 2019. doi: 10.1016/j.ins.2018.10.029. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025518308478

[27] A. Kasem, A. A. Ghaibeh, and H. Moriguchi, "Empirical study of sampling methods for classification in imbalanced clinical datasets," in *Advances in Intelligent Systems and Computing*. Springer International Publishing, Oct. 2016, pp. 152–162. [Online]. Available: https://doi.org/10.1007/978-3-319-48517-1_14

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[29] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, 1998. doi: 10.1162/089976698300017197

[30] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning," *J. Mach. Learn. Res.*, vol. 18, no. 1, p. 559–563, Jan. 2017. doi: 10.5555/3122009.3122026

[31] S. Vluymans, *Dealing with Imbalanced and Weakly Labelled Data in Machine Learning using Fuzzy and Rough Set Methods*. Springer International Publishing, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-04663-7

[32] P. A. Flach and M. Kull, "Precision-recall-gain curves: PR analysis done right," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst. - Vol. 1*, 2015, p. 838–846. doi: 10.5555/2969239.2969333

[33] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. of Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006. doi: doi/10.5555/1248547.1248548

Bruno Riccelli dos Santos Silva is a PhD candidate in Teleinformatics Engineering at Federal University of Ceará (UFC), Brazil. He received a Master degree in Teleinformatics Engineering from Federal University of Ceará, Brazil, in 2020. His research interests are in the areas of real-time systems, Network security, Artificial Intelligence. Currently, he participates in two research projects and he works as a researcher at the Computer Systems Engineering Laboratory (LESC). ORCID 0000-0001-8189-7187 ; IDLattes: 9288483499965859

Ricardo Jardel Nunes da Silveira is an Assistant Professor at Teleinformatics Department (DETI), Federal University of Ceará (UFC), Brazil, since 2011. He received a Master degree in Teleinformatics Engineering from Federal University of Ceará, Brazil, in 2008. His research interests are in the areas of security, fault tolerance, and real-time systems. Currently, he is a doctorate student in DETI/UFC and participates in two research projects and he works as a researcher at the Computer Systems Engineering Laboratory (LESC). ORCID 0000-0002-8199-5737; IDLattes: 4664100616809407

Manuel Gonçalves da Silva Neto is a PhD candidate in Teleinformatics Engineering at Federal University of Ceará (UFC), Brazil. He received his Master degree in Software Engineering from Cesar-edu, Brazil (2014). His research interests are: evidence-based computing, machine learning, health informatics, cyber-physical systems and computer networks. He is a student member of the GREat research group (www.great.ufc.br). ORCID 0000-0002-4959-6912; ID Lattes: 9433835294642844.

Paulo Cesar Cortez received the B.Sc.degree in electrical engineering from the Federal University of Ceará, Brazil, in 1982, and the M.Sc.and Ph.D. degrees in electrical engineering from the Federal University of Paraíba, in 1992 and 1996, respectively. He is currently a Full Professor at the Department of Teleinformatics Engineering, Federal University of Ceará. His fields of interest include image and signal analysis, computer vision, biomedical signal processing, and biomedical systems. ORCID 0000-0002-4020-3019 ; IDLattes: 5024602152304064

Danielo G. Gomes is an associate professor at the Department of Teleinformatics Engineering of the Federal University of Ceará (UFC), Brazil. He received his Ph.D. degree in Réséaux et Télécoms from the University of Evry, France (2004). His research interests include sensing and data science in precision beekeeping, urban computing, IoT, environmental monitoring. ORCID 0000-0002-8285-4629; ID Lattes: 6303297687237256