Impact of Feature Selection Methods on the Classification of DDoS Attacks using XGBoost

Pedro Henrique Hauy Netto de Araujo, Anderson Aparecido Alves da Silva,
Norisvaldo Ferraz Junior, Fabio Henrique Cabrini, Alessandro Santiago dos Santos,
Adilson Eduardo Guelfi, Sergio Takeo Kofuji

***Abstract*— Distributed Denial of Service (DDoS) attacks impose a major challenge for today's security systems, given the variety of its implementations and the scale that the attacks can achieve. One approach for their early detection is the use of Machine Learning (ML) techniques, which create rules for classifying traffic from historical data. However, different types of data contribute unequally to the assertiveness of the trained model. The use of Feature Selection (FS) techniques as a pre-processing step allows identification of the most relevant features for the problem in question. This action reduces training time and can even improve performance when noisy variables are eliminated. The current work is based on a public dataset and the XGBoost algorithm to measure the impact of FS techniques on the DDoS attack classification problem. We consider both techniques independent of the sample labels, as well as methods that use this information to rank the variables in order of importance. We analyzed the problem from the point of view of Binary and Multiclass classification. We also created a benchmark of classification metrics and execution times. Our comparisons involved the Accuracy, Precision, Recall, and F1 Score metrics for different FS methods, in addition to training and execution time. In the results it is possible to verify for both the Binary (78% reduction of the features) and Multiclass classifiers (60% reduction of the features), that the ANOVA method proved to be the most beneficial.**

***Index Terms*— Feature Selection (FS), DDoS, XGBoost, Binary Classifier, Multiclass Classifier**

I. Introduction

DISTRIBUTED Denial of Service (DDoS) attacks are increasingly frequent and voluminous on the Internet. Daily, thousands of attacks are triggered to the most diverse targets: governments, e-commerce companies, telecommunications service providers, multimedia content distributors, among others [1]. The motivations for these attacks are very diverse, such as economic interests, political activism, or even intellectual curiosity.

Currently it is even possible to hire DDoS attacks towards a specific target [2]. Attackers, who have numerous infected devices under their command, charge for the duration and volume of the shots. In addition to the direct economic and social damage caused by the interruption of services, the reputation and credibility of the attack victims are also severely tarnished [3].

Detection of DDoS attacks can be performed by Intrusion Detection/Prevention Systems (IDPS). IDPS have traditionally been divided into two areas: detection by signature and by anomaly [4]. Although attack signatures developed by experts are able to identify threats with great precision, they become ineffective against unprecedented attacks. In contrast, the use of anomaly detection is able to offer some protection even against zero-day attacks. One of the biggest difficulties in implementing DDoS detection systems via anomaly detection is in minimizing the occurrence of false positive or false negative alerts.

There are several anomaly detection techniques. Some are based on the comparison of correlations and gains [5 - 6], while others focus on clustering methods [7]. However, it is Machine Learning (ML) models that have gained more relevance recently, thanks to advances in the availability of computational power, specialized software, and public datasets. The application of ML models for detecting DDoS attacks has been discussed in the literature for some decades [8 – 10]. The problem of anomaly detection has been attacked by both the use of supervised learning (such as classifiers) and unsupervised learning.

However, the selection of features that serve as input to a model is a less explored subject within the context of attack detection. The use of Feature Selection (FS) can bring several benefits, including: (1) a more agile attack detection process; (2) less need for storage and memory when implementing the classifier; and (3) an increase in the ability to interpret the model generated [11].

Pedro Henrique Hauy Netto de Araujo, Norisvaldo Ferraz Junior and Sergio Takeo Kofuji are with the Integrated Systems Laboratory at Escola Politécnica da Universidade de São Paulo (USP), Sao Paulo-SP, Brazil. E-mails: pedrohauy@usp.br, norisjunior@usp.br; kofuji@usp.br.

Anderson A. A. Silva is with the Integrated Systems Laboratory at Escola Politécnica da Universidade de São Paulo (USP), Instituto de Pesquisas Tecnológicas do Estado de São Paulo (IPT), Centro Universitário Senac (SENAC) and Universidade Paulista (UNIP), São Paulo-SP, Brazil. E-mail: anderson.silva@pad.lsi.usp.br.

Fabio Henrique Cabrini is with the Integrated Systems Laboratory at Escola Politécnica da Universidade de São Paulo (USP), Faculdade de Tecnologia de São Paulo (FATEC), Faculdade de Tecnologia Termomecanica (FTT) and Faculdade de Informática e Administração Paulista (FIAP), São Paulo-SP, Brazil. E-mail: fabio.cabrini@gmail.com.

Alessandro Santiago dos Santos is with the Instituto de Pesquisas Tecnológicas do Estado de São Paulo (IPT), São Paulo-SP, Brazil. E-mail: alesan@ipt.br.

Adilson E. Guelfi is with the Universidade do Oeste Paulista (Unoeste), Presidente Prudente-SP, Brazil. E-mail: guelfi@unoeste.br.

The authors would like to thank the partnership between Huawei Technologies Company and Universidade de São Paulo (USP) for supporting this research.

Digital Object Identifier: 10.14209/jcis.2021.22

The objective of this work is, therefore, to measure the impact of FS techniques in the problem of classifying DDoS attacks using Machine Learning. The proposal is to establish a fixed classifier algorithm and evaluate the influence of FS methods on performance metrics. The use of classifiers presupposes a labeled dataset. In particular, in this context, the following are analyzed:

- The impact of FS methods that are independent of the target feature;
- The impact of FS methods that do depend on the target feature;
- The execution time of the whole model (which includes performing FS and training the classifier).

The rest of the article is divided as follows: Section II presents the related works, introducing the main types of FS, datasets, and classifiers found in the literature. Section III details the proposal of the experiments, including the environment used, the data architecture, and the quality metrics of the classification. Section IV presents and analyzes the main results obtained in the computer simulations. Finally, Section V contains the general conclusions of the work, contributions, and future works.

II. Background

This section briefly describes the related works and concepts related to our proposal.

A. Related Works

The FS algorithms are inserted in the context of dimensionality reduction. The objective of these techniques is to find a subset of input features, so that they are closer to the target feature and more distant from each other [12]. In this case, there are several ways to define distance, such as, for example, Pearson's Correlation Coefficient (PCC) or Mutual Information (MI) [13]. An FS technique is characterized by the choice of a subset of features, among the original variables, without any transformation or creation of new variables. It differs, therefore, from feature extraction techniques, such as Principal Component Analysis (PCA), which projects input features in a different space to the original one. An unwanted consequence of methods that transform the original variables is the lack of interpretability of the new variables.

FS methods are divided into: (1) filters; (2) wrappers; (3) embedded; and (4) hybrids [14]. In the work of Polat, Polat, and Cetin [8], the uses of filters, wrappers, and LASSO (Least Absolute Shrinkage and Selection Operator, which is an embedded way of executing FS) are explored as a pre-processing step for the classification of DDoS attacks in Software-Defined Networks (SDN). As classifiers the authors employ Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Naive Bayes (NB), and Artificial Neural Networks (ANN). In all cases, Accuracy improvements occur when the techniques are applied.

The use of filters involves the comparison of a single variable with the target, without taking into account its relationship with the other variables. Because of this, these methods are computationally efficient. They involve assigning a number to each feature and then usually sorting them, choosing the most appropriate K variables. This category includes PCC, Information Gain (IG), and Chi-Square Test, for example. Pattawaro and Polprasert [15] propose a novel technique for classification problems, the Attribute Ratio (AR), which only takes into account counting ratios of the observations belonging to each class, thus being a type of filter.

Wrappers are problems of exhaustive search in the space of variables, directly involving the use of some classifier method. A quality metric is chosen (such as Accuracy), a model is trained using a subset of the features and, depending on the result, new variables are added or removed, iteratively. An example is the work of Gupta [10] who applies Recursive Feature Elimination (RFE) to select the appropriate variables for the classification of DDoS attacks. However, wrappers are computationally very expensive.

The choice of the best variables within the space of all possible combinations is considered an NP-Hard problem [16]. There is even research on ways to optimize the execution of the wrappers. For example, one way is to use Genetic Algorithms [17], that incorporate the idea of objective function (analogous to Natural Selection) to choose the most suitable population of variables. Tree-based classifiers, such as Decision Trees (DT), Random Forest (RF), and XGBoost, already have methods of quantifying the importance of each feature built into their training. An example is in the work of Dhaliwal, Nahid, and Abbas [18], in which an XGBoost model is trained for intrusion detection tasks, using the feature importance score generated by the algorithm to interpret the results. In the work of Wang et. al [14] a hybrid FS model is shown. More specifically, an ensemble of the results obtained by other methods is carried out, aggregating them through arithmetic and geometric means.

Several datasets have been used over the years as a reference for intrusion detection research, with an emphasis on the most popular: KDD 99 [10] [19 - 20] and NSL-KDD [21 - 23]. These attack records are used in several studies, many of them with a focus on detecting anomalous network behaviors using ML. However, these datasets are out of date, since new types of attacks, which are not represented in these sets, have been introduced in the 21st century.

Therefore, it is necessary to use a more up-to-date dataset. The dataset Canadian Institute for Cybersecurity DDoS 2019 (CICDDoS2019), published by Sharafaldin et al. [24], is extracted from a testbed with real equipment, such as routers, switches, firewalls, and several servers. In their work, the authors generate both attacks and benign background traffic. In total, 86 network parameters are collected. Attack traffic is more represented than normal traffic in this dataset. As part of the paper that presents the set, the authors propose the use of ML to perform the classification of attacks, building models with the Iterative Dichotomiser 3 (ID3), RF, NB, and Logistic Regression (LR) algorithms. Since its publication, this dataset has already served as the basis for some works such as that of Hussain [25], which uses resampling to address class imbalance, and Li [26], which makes use of both traditional ML

and Deep Learning (DL) algorithms.

A summary of the main related works is found in Table I. The datasets used are listed for reference. Table I also shows dataset processing techniques, such as Feature Selection and sampling. The methods for training a DDoS attack classifier are also exposed, such as the algorithm used, the type of classification it performs and whether Cross-Validation was used or not. Finally, quality metrics are also listed in Table I. In addition, the experiments that will be shown in this work are placed at the end of the table for comparison with the others. In general, this work seeks to apply a wide range of Feature Selection techniques and performance metrics to assess the impact of the former on the DDoS attack classification problem. We emphasize that this work tackles the problem from both a Binary and a Multiclass point of view. The ratios between metrics and runtimes are also introduced, here called Benefit-Cost Ratios, and will be explained in Section III.D.

Tree-based algorithms have been shown to be efficient as classifiers and regressors for tabular data. In particular, XGBoost [27] drew the attention of industry and academia, for its performance and training agility. It is a gradient boosting algorithm, which produces a strong predictive model through the combination of weak models. Its use is very popular in ML competition platforms, such as Kaggle [28]. In addition, it has been applied to the classification problem of DDoS attacks with good results [15][18][29].

B. Literature Review

As described in [27], XGBoost makes its predictions through an ensemble of decision trees. These trees are constructed so as to minimize a cost function that contains regularization terms (to penalize very complex models). The weight that each new tree will contribute to the final prediction is calculated to go against the gradient of the cost function. As its training depends on the result of the previous iteration, the trees are therefore trained sequentially. XGBoost will be used as the classifier in the experiments that follow.

Several Feature Selection methods are used in the simulations in the next sections. Some of them do not depend on the label of the samples we have in hand. For example, features with low variance (and, consequently, also the constants features, with variance $\boldsymbol{\sigma^2 = 0}$) have no discriminatory power in a decision tree, and therefore can be discarded [30]. Correlated variables end up dividing among themselves the importance they have in a predictive model [13]. They can thus be represented by a single variable.

There are variable selection methods that are label-dependent. We can use Analysis of Variance (ANOVA), for example, to see if a categorical target (as in the case of DDoS attacks) influences the behavior of a numerical variable. ANOVA uses an F-test to determine whether two or more means come from the same distribution or not [31]. A high value of the F-statistic implies that the target categories influence the distribution of the numerical variable and the latter should be added to the feature set.

The idea of Mutual Information (MI) comes from Information Theory and can be used to select features. It is the extent to which knowing a random variable reduces the uncertainty that one has about another random variable. It is usually calculated between two categorical variables, but it can be adapted to the case that we are dealing with a numeric variable and a categorical one [32]. A high MI value between a feature and the target suggests that the former should be included in the model.

The Relief family of algorithms performs variable selection by assigning a weight to each feature [33]. Some samples are taken from the dataset and compared with their closest neighbors: those of the same category (nearest hits) and those of different categories (nearest misses). Weight is computed in

TABLE I
SUMMARY OF RELATED WORKS

Work	Dataset	Feature Selection	Cross-Validation	Sampling	Classifiers	Type of Classification	Metrics
[8]	Their testbed	Relief, Sequential Forward Selection (SFS), LASSO	10-fold CV	No	SVM, NB, ANN, KNN	Multiclass	Accuracy, Sensitivity, Specificity, Precision, F1-Score
[10]	KDD 99	Information Gain, Chi-Squared, RFE, Ensemble	20-fold CV	Undersampling	NB, SVM, DT, RF	Binary	Accuracy, Precision, Recall
[15]	NSL-KDD	Attribute Ratio	No	No	XGBoost	Binary	Accuracy, Precision, Recall, ROC-AUC
[20]	KDD 99	Feature grouping based on linear correlation coefficient (FGLCC), Cuttlefish Algorithm (CFA)	10-fold CV	Undersampling	SVM	Binary	Accuracy, True Positive Rate (TPR), False Positive Rate (FPR)
[25]	CICDDoS2019	Feature Importance of RF	5-fold CV	Undersampling with class rebalance	BayesNet, Bagging, KNN, Sequential Minimal Optimization (SMO), LR	Binary	Precision, Recall, F1-Score, TPR, FPR, Runtime
This Work	CICDDoS2019	Drop Low Variance, Drop High Correlation, ANOVA, MI, ReliefF, Gain of XGBoost, RFE, Ensemble	10-fold CV	Undersampling with class rebalance	XGBoost	Binary, Multiclass, Multiclass One-Vs-One, Multiclass One-Vs-Rest	Accuracy, Precision, Recall, F1-Score, Runtime, Benefit-Cost Ratio

order to favor hits and penalize misses. Features with high weight are selected. ReliefF is an extension of the original Relief algorithm to handle multiple classes.

As a by-product of XGBoost training, rankings of feature importance are created. One of these rankings concerns the Gain of the variables [27], which is a metric that measures the relative contribution of each variable to each tree trained in the model. We can then select only the features considered most important by this metric, which becomes a Feature Selection method embedded in XGBoost.

RFE is a wrapper Feature Selection technique. It starts from a complete feature set and uses a Machine Learning model to list the most important features. From that point on, the worst one is discarded and the process starts again with the remaining set [34], which characterizes the recursive nature of this method. For the experiments in this paper, XGBoost itself is used as the base Machine Learning model for RFE.

III. Methodology

To measure the influence of the use of FS on the DDoS attack classification problem, the CICDDoS2019 dataset was chosen [24]. This set includes labeled traffic samples from 12 modern DDoS attacks (NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, TFTP), in addition to benign traffic. The dataset is processed, subjected to several Feature Selection methods, and serves as input to a classifier that uses the XGBoost algorithm. No new Feature Selection method is being proposed in the experiments that follow. Instead, Feature Selection techniques known in the literature are used in order to compare them numerically.

Tree-based models have been shown to be consistently effective for classification problems using this dataset [25][35].

TABLE II
CLASS DISTRIBUTION OF THE DATASET CICDDOS 2019

Binary		Multiclass	
Class	Samples	Class	Samples
ATTACK	50006249	TFTP	20082580
		DrDoS_SNMP	5159870
		DrDoS_DNS	5071011
		DrDoS_MSSQL	4522492
		DrDoS_NetBIOS	4093279
		DrDoS_UDP	3134645
		DrDoS_SSDP	2610611
		DrDoS_LDAP	2179930
		Syn	1582289
		DrDoS_NTP	1202642
		UDP-lag	366461
		WebDDoS	439
BENIGN	56863	BENIGN	56863

The XGBoost classifier is fixed in the course of the experiments, since the objective of the study is to measure the influence of the FS techniques, not the chosen classification algorithm.

The steps of the simulations carried out are detailed in the rest of this section, as well as the main concepts necessary to evaluate them. Fig. 1 illustrates the general data flow of the experiments in this work. The next subsections detail the implementation of the architecture proposed by this diagram.

A. Resampling and Balancing

Table II presents the distribution of the samples among the different traffic classes for the CICDDoS2019 dataset, demonstrating both the binary and multiclass representation of this set.

Intuitively, we can say that it is an extremely unbalanced dataset, with a number of attacks that far exceeds benign traffic

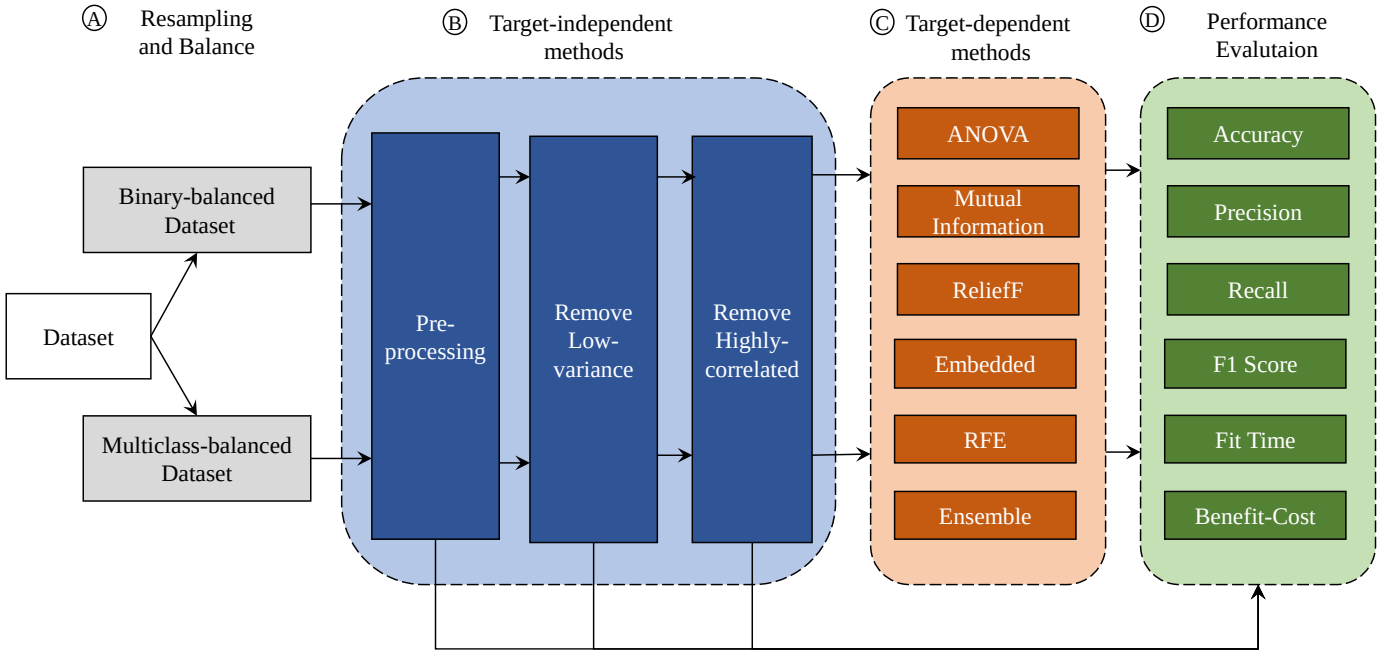

Fig. 1. Data flow proposed for the experiments

and, among these attacks, the TFTP amplification is the most represented. We can formalize this notion of imbalance through the Pielou Index [36]:

$$J = -\frac{1}{\ln S}\sum_{i=1}^{S}\frac{n_i}{N}\ln\frac{n_i}{N} = \frac{H}{\ln S} \qquad (1),$$

where we have S distinct classes, each class C_i has n_i elements and the complete data set has N samples. In equation (1), H is the Shannon Entropy [37]. The Pielou Index normalizes entropy by its maximum ($\ln S$), so that the values are confined between 0 and 1. In this case, 0 represents the largest possible unbalance (all samples are of a single class) and 1 the largest possible balance (the samples are divided equally between classes). Note that the Pielou Index is applicable to both sets with binary and multiclass labels.

An unbalanced dataset tends to favor the majority class(es) [38]. Therefore, as illustrated in Step Ⓐ of Fig. 1, two datasets were generated: one balanced from a binary point of view (in which there is the same number of attacks and benign traffic) and another balanced from a multiclass point of view (with all classes with equal representation). These two ways of resampling the dataset make the new Pielou Index, for both cases, equal to 1. Due to hardware limitation, the resampling process took place by undersampling all classes, in order to efficiently allocate the data in the available memory. Works like those of Hussain [25] and Li [26] also opt for undersampling. Finally, the 'WebDDoS' class was dropped because it had a much smaller number of samples than the other types of attacks.

After the resampling process, the binary and multiclass datasets serve as the basis for training binary and multiclass classifiers, respectively.

B. Label-Independent Feature Selection

The part called "Preprocessing" in Step Ⓑ of Fig. 1 includes several operations that follows. All categorical variables were discarded based on domain knowledge: attributes such as Flow ID, Source IP Address, and Destination IP Address are only identifiers, having no predictive value. Source Port and Destination Port can be treated as numerical variables, but this work will focus on network traffic statistics, choosing not to use the former. All the remaining variables are numeric attributes, such as counters and ratios of these counters over time (such as Bytes/sec). The following variables, even representing counters and measures of packet sizes (thus positive values), had spurious negative values in the dataset: 'Fwd Header Length', 'Bwd Header Length', 'Init_Win_bytes_forward', 'Init_Win_bytes_backward', 'min_seg_size_forward'. It was then decided to ignore the negative sign of these samples (i.e., make them positive). A detailed description of the features of the complete dataset can be found in [26]. Infinite values were replaced by the maximum value of the feature in question. The datasets after these operations serve as input for XGBoost classifiers. We used a 10-fold Stratified Cross-Validation (CV), where the metrics illustrated by Step Ⓓ in Fig. 1 were collected.

The second part of the Step Ⓑ performs the selection of variables through two operations: (1) removal of duplicate columns; and (2) removal of columns with low variance. These methods are called "Basic Methods" in subsequent sections. Note that determining what is a "low" variance is a hyperparameter of the model, which should be defined as a threshold at the time of training. One point of attention is that, in order to verify the influence of the Feature Selection techniques on the classification metrics, it is important that the first ones are performed **within the Cross-Validation process**.

With the dataset reduced by the "Basic Methods", a PCC Matrix is generated between all the predictor variables. The correlation between the predictor variables and the target labels is not adequate, since the former are numerical and the latter categorical [34]. From this operation, groups of features are formed that have a correlation ρ greater than a threshold established in training. This threshold, therefore, is another hyperparameter of the experiment. Within these groups, only one variable is maintained and the others are discarded. The resulting set is also used to train an XGBoost model, within a 10-fold CV. This method will be referred to as "Correlation" in the results of the experiments.

C. Label-Dependent Feature Selection

The methods of Step Ⓒ in Fig. 1 do take into account a relationship between the sample labels and the predictor variables to select the attributes that should be used in the model. In general, all of them are capable of generating rankings of importance of the features. With the ordered variables, the most relevant K are chosen and the others are removed. As in the methods independent of the label, here it is also necessary to carry out the variable selection **within the Cross-Validation process**.

Thus, curves are generated for each of the classification quality metrics according to the value of K, the number of selected features. This operation is repeated for each feature selection method considered.

To save time and memory allocation in the Feature Selection task with methods that depend on the sample label, it was decided to use as input to this block the output of the process of removing highly correlated variables (as in Fig. 1). This procedure facilitated further processing during the experiments, by significantly reducing the number of input features for this block.

As mentioned in section II, the chosen FS methods are divided into the following categories: (1) filters; (2) wrappers; (3) embedded, and (4) hybrid. As Filters, the F-test of ANOVA, MI and ReliefF methods are considered for the experiments. XGBoost Gain is used as an embedded FS technique. The RFE method is also used as a wrapper around XGBoost. All of these techniques are available in Scikit-Learn [39], with the exception of ReliefF [40], which has its own library, but is also compatible with NumPy Arrays.

Finally, an ensemble of the other techniques is performed, as proposed by Gupta [10]. In this method, the rankings of each

variable for each FS technique considered are added in a new vector and the value of this sum is taken into account when ordering the variables. The experiments deal with an ensemble among ANOVA, MI, ReliefF, and XGBoost Gain. The RFE technique was not included in the ensemble due to its high computational cost.

D. Performance Evaluation

Both classifiers trained in Steps Ⓑ and Ⓒ are evaluated according to the performance metrics illustrated in Step Ⓓ from Fig. 1. Metrics differ between binary and multiclass cases.

For a binary classifier, we define one of the labels as positive and the other as negative. This choice is arbitrary, but must be taken into account when interpreting the results. Choosing the class "Benign" as negative and "Attack" as positive, we have the nomenclature in Table III that follows.

TABLE III
RESULTS OF A BINARY CLASSIFICATION

TP (True Positive)	Attack classified as Attack
FP (False Positive)	Benign classified as Attack
FN (False Negative)	Attack classified as Benign
TN (True Negative)	Benign classified as Benign

For a multiclass classifier, consider T samples in the test set and S distinct C_i classes. TP_i is the number of elements correctly classified with the label of C_i. FP_i are elements classified as C_i, but that belong to another class. FN_i are the elements of C_i, but mistakenly classified in another class.

1) Accuracy (AC): is the ratio between the correct classifications of the model and the total classifications made. For the binary case*:*

$$AC = \frac{TP + TN}{TP + FP + TN + FN} \qquad (2)$$

For the multiclass case:

$$AC = \frac{1}{T}\sum_{i=1}^{S} TP_i \qquad (3)$$

2) Precision (PR): is the ratio of the correct predictions and the total predictions for a given class. A high Precision value is linked to fewer false alarms. For the binary case*:*

$$PR = \frac{TP}{TP + FP} \qquad (4)$$

For the multiclass case, we have some possible aggregation methods [41]. The Macro-Average of Precision was chosen, in order not to privilege any specific class. We have:

$$PR_M = \frac{1}{S}\sum_{i=1}^{S} \frac{TP_i}{TP_i + FP_i} = \frac{1}{S}\sum_{i=1}^{S} PR_i \qquad (5)$$

3) Recall (RC): is the ratio of the correct predictions and the total elements of a given class. A high Recall value implies that most samples in a class have been recognized. For the binary case*:*

$$RC = \frac{TP}{TP + FN} \qquad (6)$$

For the multiclass case, again with the Macro-Average:

$$RC_M = \frac{1}{S}\sum_{i=1}^{S} \frac{TP_i}{TP_i + FN_i} = \frac{1}{S}\sum_{i=1}^{S} RC_i \qquad (7)$$

4) F1 Score (F1): the PR and RC metrics are conflicting requirements, since by increasing one of them the other is compromised. The F1 Score is the harmonic mean among the latter. For the binary case*:*

$$F1 = \frac{2 * PR * RC}{PR + RC} \qquad (8)$$

For the multiclass case, the most recommended calculation [42] is given by:

$$F1_M = \frac{1}{S}\sum_{i=1}^{S} \frac{2 * PR_i * RC_i}{PR_i + RC_i} = \frac{1}{S}\sum_{i=1}^{S} F1_i \qquad (9)$$

The metrics for Accuracy, Precision, Recall, and F1 Score vary between 0 (worst) and 1 (best), for both binary and multiclass classifiers. Since we are using the Cross-Validation process, we can calculate the Mean and Standard Deviation for each of these metrics.

5) Fit Time: is the sum of the time spent with the execution of the Feature Selection process and the training of the classifier.

6) Benefit-Cost Ratio (BCR): is a family of ratios given by dividing one of the metrics from 1) to 4) by the required Fit Time. For an X metric, we have*:*

$$BCR(X) = \frac{X}{Fit\_Time} \qquad (10)$$

E. Multiclass Classifier Training Considerations

XGBoost training involves building new decision trees in order to go against the gradient of a cost function. For a Multiclass classifier, this cost function is traditionally the Categorical Cross-Entropy. However, it is also possible to train a Multiclass classifier from several Binary classifiers, in meta-learning strategies such as One-Vs-Rest (OvR) [43] and One-Vs-One (OvO) [44]. In the latter cases, the cost function to be minimized is Binary Cross-Entropy. For the experiments, these meta-learning strategies were also considered.

The OvR strategy consists of transforming the classification problem between S classes into S binary classification problems. Each one of them determines whether the sample belongs to a certain class C_i or not (and in this case it is part of

the rest) [43]. Each Binary classifier must return a probability and the classifier with the highest probability indicates the predicted class for the sample.

The OvO strategy demands more computational power, as it compares all possible 2-element combinations among the S classes [44]. This gives a total of $\boldsymbol{S(S-1)/2}$ binary classifiers. Each classifier then votes for a class. The class with the highest number of votes is predicted by the OvO model.

F. Analysis Scope

The present work focuses on numerically characterizing the influence of Feature Selection techniques on the performance metrics of a DDoS attack classifier. As all results are obtained using 10-fold Cross-Validation, we potentially have 10 distinct feature groups for each simulated data point. This situation is the same for both label-independent and label-dependent variable selection methods. It is beyond the scope of this article to list which features were selected in each round of the experiments.

G. Test environment

For all experiments, a notebook with an Intel Core i7 3630QM processor, 8GB DDR3 RAM, 500GB Solid State Drive (SSD) and Windows 10 operating system was used. Through the Anaconda package manager, the following programs (as well as their dependencies) were installed: Python (v3.8.5), NumPy (v1.19.5), Pandas (v1.2.0), Scikit-learn (v0.24.0), Seaborn (v0.11.1), ReliefF (v0.1.2), and XGBoost (v1.3.1). It is, therefore, a traditional environment for research in Machine Learning based on the Python language and its libraries.

TABLE IV
UNDERSAMPLED AND BALANCED DATASETS

New Binary		New Multiclass	
Class	Samples	Class	Samples
ATTACK	2640	TFTP	440
		DrDoS_SNMP	440
		DrDoS_DNS	440
		DrDoS_MSSQL	440
		DrDoS_NetBIOS	440
		DrDoS_UDP	440
		DrDoS_SSDP	440
		DrDoS_LDAP	440
		Syn	440
		DrDoS_NTP	440
		UDP-lag	440
BENIGN	2640	BENIGN	440

IV. RESULTS

The results and analyses of the works described above are presented as follows.

A. Resampling and Balancing

The dataset described in Table II is unbalanced, which can favor the majority classes in a classification problem. From a binary point of view, it has a Pielou Index **J = 0.012748** (according to equation 1). From a multiclass point of view, this same dataset has **J = 0.764539**.

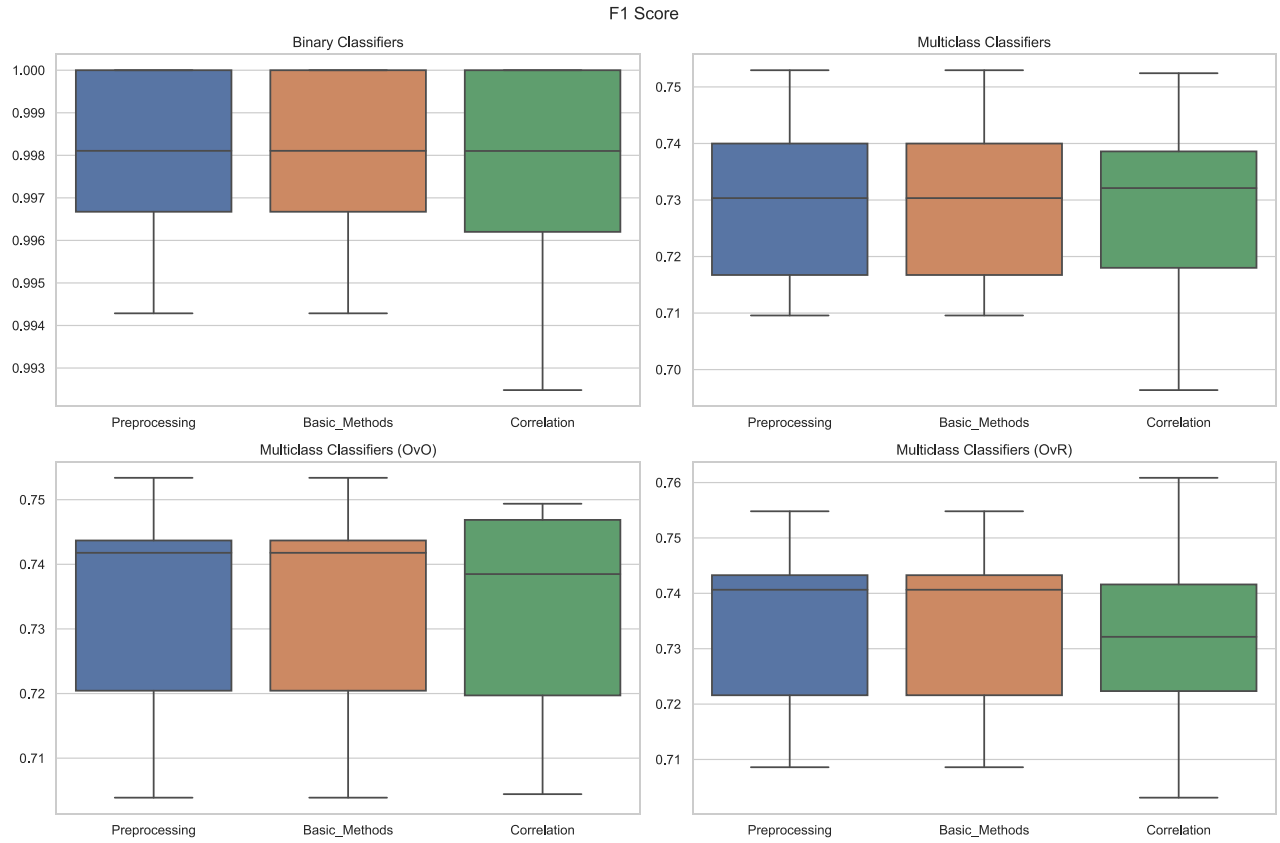

Fig. 2. F1 Score for the methods of Label Independent Feature Selection

Two **distinct** datasets are generated, sub-samples of the original, in such a way that they become perfectly balanced data sets (**J = 1**). Sampling is performed in such a way that the final two datasets have the same number of elements.

The resulting datasets are shown in Table IV. Despite having different class distributions, both sets of data generated have 5280 elements.

B. Label-Independent Feature Selection

Fig. 2 shows the F1 Score result for the experiments with selection of variables independent of the sample labels. For the "Preprocessing" phase, 9 variables were eliminated, namely: Flow ID, Source IP, Source Port, Destination IP, Destination Port, Protocol, Timestamp, SimillarHTTP, and Inbound. The source and destination IP addresses only serve as identifiers of the devices in the testbed. The datasets were collected in [24] distributing attacks evenly across multiple victim servers. Works like [26] and [35] also chose to discard IP addresses in their predictive modeling, since the statistical behavior of the network parameters is being taken into account for the detection of attacks. For the "Basic Methods", all features with variance $\boldsymbol{\sigma^2 < 0.01}$ were eliminated. For "Correlation", features with $\boldsymbol{\rho > 0.9}$ between them were grouped. The values of $\boldsymbol{\sigma^2}$ and $\boldsymbol{\rho}$ were taken based on the method developed in [30] and proved to be adequate for the data sets used in the experiments.

Analogously, Accuracy, Precision, and Recall were also measured. The behavior of all metrics was very similar in relation to the use of Feature Selection. The results of "Preprocessing" and "Basic Methods" were the same, proving that the removal of variables with low variance does not change the performance of a classifier (both Binary and Multiclass). The performance in removing the highly correlated variables also did not change much, with a slight increase in the standard deviation of the metrics when using Cross-Validation for the Binary, Multiclass, and Multiclass OvR classifiers.

The results of the classifiers subjected to label-independent FS methods are summarized in Table V ahead. Overall, Binary classifiers performed better than Multiclass classifiers. Among the Multiclass classifiers, those which used the meta-learning strategies (OvO and OvR) achieved slightly higher performances than the traditional Multiclass classifier.

TABLE V
CLASSIFICATION METRICS FOR THE LABEL INDEPENDENT FEATURE SELECTION METHODS

Binary Classifiers			
Metric	Preprocessing	Basic methods	Correlation
Accuracy	99.79 ± 0.23	99.79 ± 0.23	99.75 ± 0.27
Precision	99.85 ± 0.36	99.85 ± 0.36	99.81 ± 0.47
Recall	99.73 ± 0.40	99.73 ± 0.40	99.7 ± 0.39
F1 Score	99.79 ± 0.23	99.79 ± 0.23	99.75 ± 0.27
Features	77	63	41
Multiclass classifiers			
Metric	Preprocessing	Basic methods	Correlation
Accuracy	73.9 ± 1.6	73.9 ± 1.6	73.8 ± 1.8
Precision	74.3 ± 2.0	74.3 ± 2.0	74.2 ± 2.2
Recall	73.9 ± 1.6	73.9 ± 1.6	73.8 ± 1.8
F1 Score	73.0 ± 1.5	73.0 ± 1.5	72.8 ± 1.7
Features	77	61	36
Multiclass classifiers - One vs One			
Metric	Preprocessing	Basic methods	Correlation
Accuracy	74.3 ± 1.8	74.3 ± 1.8	74.3 ± 1.7
Precision	74.5 ± 2.2	74.5 ± 2.2	74.6 ± 2.1
Recall	74.3 ± 1.8	74.3 ± 1.8	74.3 ± 1.7
F1 Score	73.3 ± 1.7	73.3 ± 1.7	73.3 ± 1.6
Features	77	61	36
Multiclass classifiers - One vs Rest			
Metric	Preprocessing	Basic methods	Correlation
Accuracy	74.4 ± 1.7	74.4 ± 1.7	74.1 ± 1.8
Precision	74.8 ± 2.2	74.8 ± 2.2	74.4 ± 2.3
Recall	74.4 ± 1.7	74.4 ± 1.7	74.1 ± 1.8
F1 Score	73.4 ± 1.6	73.4 ± 1.6	73.1 ± 1.7
Features	77	61	36

We can see in Table V that, even though there is stability in the classification metrics, the number of variables could be reduced considerably. There is a **mean** reduction of **46%** in the number of variables for the Binary classifiers (from 77 to 41 features), while the Multiclass classifiers have their variables reduced by **53%** (from 77 to 36 features). Although the total number of samples is the same, the Binary and Multiclass datasets were built in different ways, with different goals (as already shown in Table IV). The Binary dataset has more elements of the benign traffic class. By having more samples of this type of traffic, which is closer to the normal behavior of users (that is unpredictable), the Binary dataset has fewer columns with low variance, which would be eliminated by the

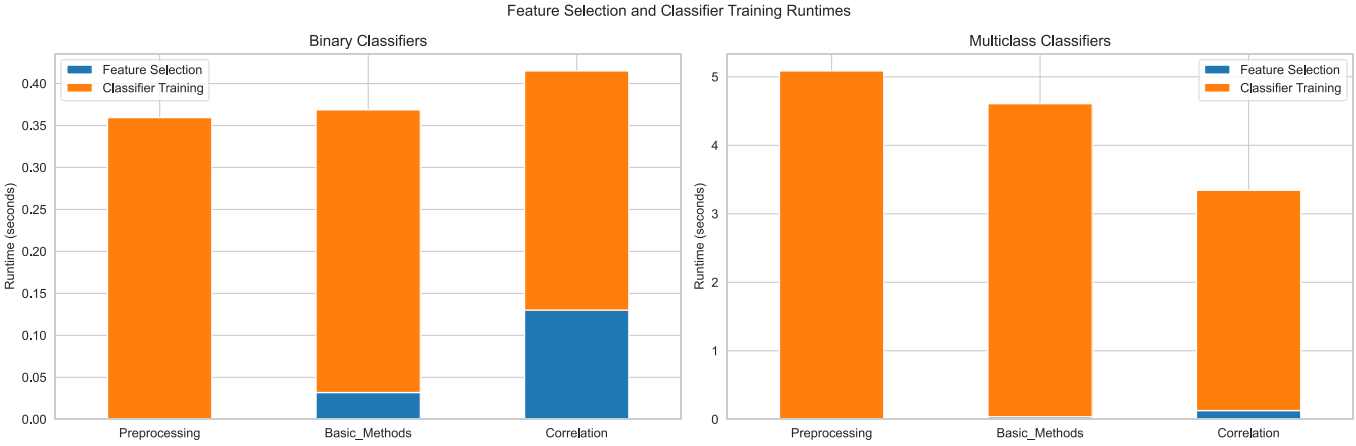

Fig. 3. Fit Times: Feature Selection and Classifier Training

"Basic Methods" described prior. Benign traffic also lacks the repetitive patterns of attack traffic, which would be grouped and removed in a correlation analysis.

The three Multiclass classifiers have the same number of end features (36 on average). The experiments were coded using the same random seed to divide the dataset into 10 folds, to make them reproducible. In this way, as the dataset is the same for the three Multiclass classifiers, it is divided in the same way within the Cross-Validation and goes through variable selection processes using the same hyperparameters, the Multiclass classifiers end up being trained with the same number of features.

The difference in the number of variables is reflected in the Fit Time, as shown in Fig. 3. The Fit Time was defined as the time to execute both the Feature Selection and the training of the classifier. Fig. 3 shows two different situations: for the Binary classifiers, the Fit Time has increased with the use of variable selection methods, even though the number of features has decreased. As for the Multiclass classifiers, we can observe a sharp decrease in the Fit Time. For the Multiclass OvO and Multiclass OvR classifiers, the results are similar to those of traditional Multiclass classifiers, and thus omitted in Fig. 3.

By separating the time for the selection of variables from the training time of the Machine Learning model, we can verify the influence of each step of the process. Table VI shows that the training time for Binary classifiers is already low in the initial models. When we remove variables using the other methods, the **training** time for Binary classifiers drops only **20%**. For Multiclass classifiers, this drop is **37%**.

However, the runtimes to execute Feature Selection steps are about the same order of magnitude of the training times of the Binary classifiers, but significantly less than the Multiclass classifiers training times. This makes the Fit Time of the Binary classifiers more sensitive to the increase in complexity between the "Basic Methods" (which remove low variance features) and the removal of highly correlated variables, which requires more arithmetic operations to be performed. Note that, because they are target-independent Feature Selection methods, their execution time depends only on the number of predictor variables.

C. *Label-Dependent Feature Selection*

The methods of selecting variables dependent on the label of the samples produce rankings of attributes according to a well-established statistic. From this, it is possible to choose the best K variables according to this criterion. Figures 4 to 11 illustrate the change in classification metrics as a function of the K value.

TABLE VI
FIT TIMES (IN SECONDS) FOR THE LABEL INDEPENDENT FEATURE SELECTION METHODS

Binary Classifiers		
Method	Feature Selection	Classifier Training
Preprocessing	0.0 ± 0.0	0.3594 ± 0.0092
Basic Methods	0.0319 ± 0.0037	0.3368 ± 0.0097
Correlation	0.130 ± 0.022	0.285 ± 0.040
Multiclass Classifiers		
Method	Feature Selection	Classifier Training
Preprocessing	0.0 ± 0.0	5.08 ± 0.28
Basic Methods	0.0368 ± 0.0073	4.57 ± 0.32
Correlation	0.127 ± 0.018	3.21 ± 0.11

1) Binary Classifiers: Accuracy, Precision, Recall and F1 Score curves were constructed for the Feature Selection methods considered. Fig. 4 illustrates the Cross-Validation Mean curves for the F1 Score. The other metrics have curves with similar behavior and therefore omitted.

For Binary classifiers, quality metrics remain stable up to approximately 17 features, for all variable selection techniques. This represents a 78% reduction in the number of features, compared to the first trainings, where only "Preprocessing" had been carried out and 77 features had been used. When choosing less than 17 features, we noticed a performance degradation in almost all methods, in particular for ReliefF. As we reduce the multi-dimensional space by removing features, the points in this space become closer, that is, the distance measurements decrease. In this way, the particular choice of points that ReliefF makes is more sensitive to fluctuations in the data, which can cause errors that are reflected in the F1 Score. For less than 5

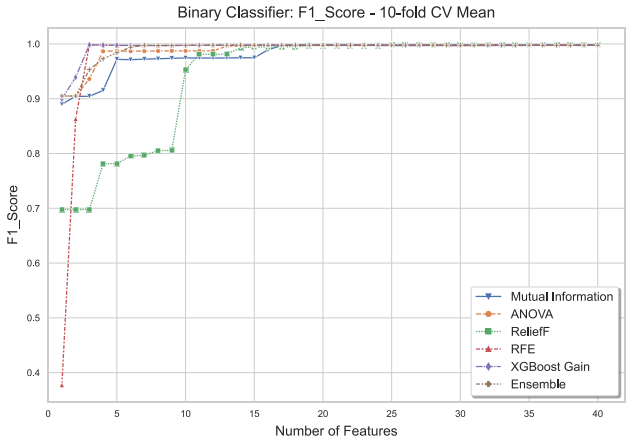

Fig. 4. Mean of F1 Score as a function of number of variables for Binary Classifiers.

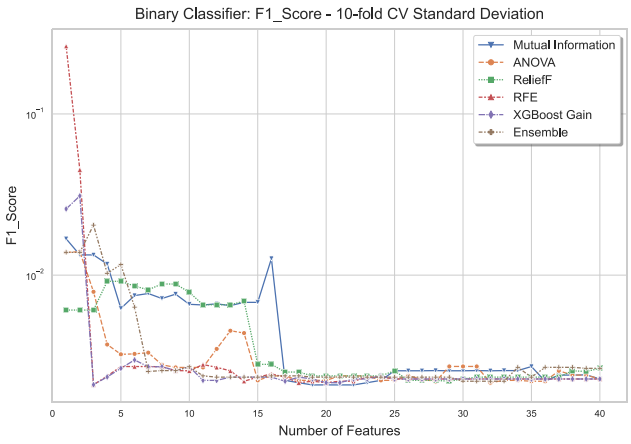

Fig. 5. Standard Deviation of F1 Score as a function of the number of variables for Binary Classifiers.

features, RFE suffers an abrupt worsening in metrics, which remains high when using more variables. The preparation of the RFE selector is done considering the features individually. It is possible that a variable has a greater predictive power when it is used together with another in the training of a Machine Learning model. However, in RFE, if this other variable is prematurely discarded, the classification result will be worse when we choose a small number of features.

Fig. 5 shows the F1 Score Standard Deviation for all Feature Selection techniques. The graph was plotted on a logarithmic scale on the vertical axis. Lower values for the Standard Deviation indicate less fluctuations for the metric in question. Again, when choosing more than 17 features, all techniques produce results with low variance. Below this value, MI and ReliefF have greater variance. In the case of MI and its adaptation to numerical variables, its calculation is influenced by the data binning process. This process, in turn, is influenced by the Cross-Validation steps. As it is not possible to guarantee homogeneity in the sets generated in Cross-Validation, the features with more information also differ, which ends up increasing the variance of the performance metrics. For ReliefF, we must take into account that the choice of points that will have its neighbor checked is random, contributing to the increase in variance.

The Fit Time for each of the Feature Selection methods is shown in Fig. 6. There are few intersections in this graph, since most variable selection techniques take a fixed time to choose K attributes. The exception is the RFE method, which, given its recursive nature, takes more time when it needs to eliminate more variables. In the worst cases, it takes up to 4 times longer than the other methods. We can also highlight the Ensemble method, which is executed, approximately, in the sum of the 4 times of the methods that compose it. The ANOVA method has the fastest execution of all the considered methods. This is because ANOVA performs fewer arithmetic operations, in addition to not performing iterative or binning processes, like the other variable selection methods considered.

The benefit-cost ratio graph for the mean of the F1 Score is plotted on a logarithmic scale in Fig. 7. Since the F1 Score (and similarly the other metrics) remains approximately constant when removing attributes, it is more efficient to choose Feature Selection methods that will be executed more quickly to achieve this result. For Binary classifiers, ANOVA is the one that best meets this criterion, regardless of the value of K. The XGBoost Gain as a variable selector appears as a second option, since the Binary classification has a low training time. As we remove variables, the RFE is progressively degrading its Benefit-Cost Ratio because, although it does not suffer such a sharp drop in Accuracy, Precision, Recall, and F1 Score, its cost in execution time grows rapidly.

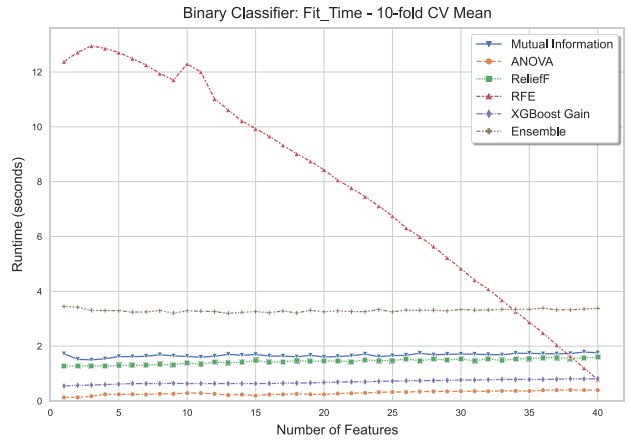

Fig. 6. Fit Time as a function of the number of variables for Binary Classifiers.

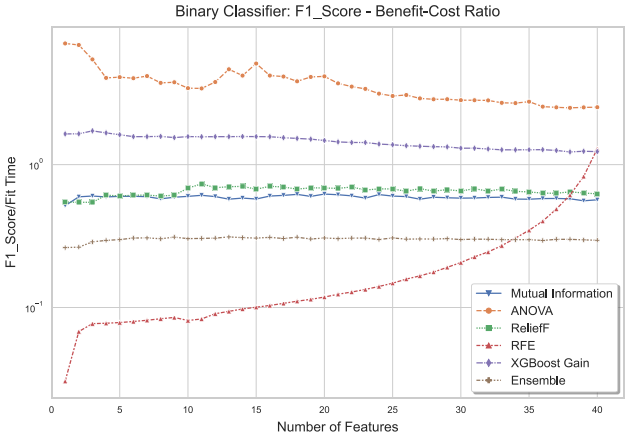

Fig. 7. F1 Score Benefit-Cost Ratio as a function of the number of variables for Binary Classifiers.

2) Multiclass Classifiers: Accuracy, Precision, Recall and F1 Score curves were also generated for the Feature Selection methods used before Multiclass classifiers. Fig. 8 shows the Cross-Validation Mean for the F1 Score. The other metrics have similar graphs with similar results.

There is more variety of Multiclass results compared to that in Binary case. For Multiclass classifiers with less than 31 features, the chosen variable selection method has more influence. Note that this represents a 60% decrease in the number of features used in the original model. Mutual Information is the method that obtains the best results for the F1 Score Mean. The use of Mutual Information is particularly beneficial for multiclass classification, since the greater number of classes decreases the probability of occurrence of each one of them individually. The logarithm in the Mutual Information calculation formula highlights these small differences, making the values obtained for each class farther apart and more suitable for ranking. We can also note that the performance of XGBoost Gain is degraded quickly when we choose less than 10 features. These gains are calculated with the widest dataset (result of filtering it by label-independent methods). The individual contribution of each feature may not be as determinant for the final prediction of a sample class as its contribution combined with the others, resulting in worse performance with fewer features. Finally, RFE again suffers a sharp drop in performance with less than 5 features. This happens because of the premature elimination of features (as in the case of binary classifiers) and also because it incorporates the gain of XGBoost in the calculation of importance of

features, with the latter having a poor performance with few variables.

Fig. 9 presents the F1 Score Standard Deviation for all variable selection techniques, on a logarithmic scale. As there are more classes than in the Binary case, the random cuts of samples and columns performed by XGBoost [27] have more influence on the methods of selecting variables that are based on it. RFE and XGBoost Gain, which have this characteristic, end up presenting a higher Standard Deviation (of all metrics) when selecting less than 15 variables.

The training time for a Multiclass classifier also directly influences the increase in the RFE Fit Time, as can be seen in Fig. 10. When used to choose less than 5 features, this method takes up to 11 times longer to be executed than the others. The computational complexity of the Multiclass classifier also makes XGBoost Gain have a higher execution time than all Filter methods.

The difference in Fit Times between the Binary and Multiclass classifiers directly influences the Benefit-Cost Ratio curve. In Fig. 11, we see this ratio for the F1 Score of the Multiclass classifiers, on a logarithmic scale. RFE requires a much longer execution time than the others, being below all other methods (in terms of the Benefit-Cost Ratio) for almost all K values. The XGBoost Gain and, consequently, the Ensemble that contains it, ends up worse than all Filters according to this measure. The best performances are, in this order, ANOVA, ReliefF, and Mutual Information, since they do not use the training of a classifier to select variables. The relative order between the Filters is the same as the Binary classification.

3) Multiclass OvO and Multiclass OvR Classifiers: again, Accuracy, Precision, Recall, and F1 Score curves were constructed for the methods of selection of Multiclass classifier variables. However, this time the training of the classifiers was performed according to the OvO and OvR strategies. The metrics, in general, came close to those obtained with traditional Multiclass classifiers (whose cost function is the Categorical Cross-Entropy).

As an example, Table VII presents the results of Accuracy, F1 Score, and Fit Times for all variable selection methods,

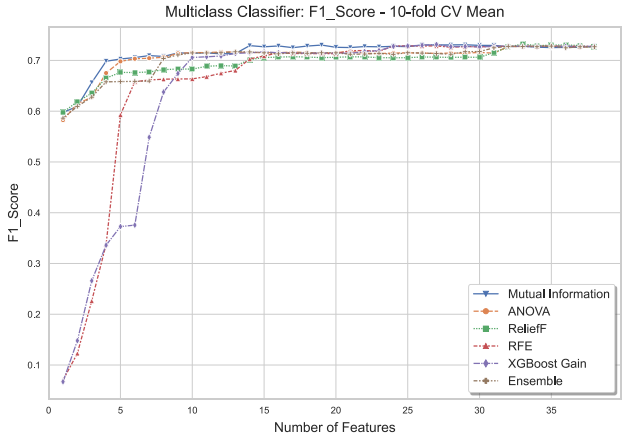

Fig. 8. Mean of F1 Score as a function of the number of variables for Multiclass Classifiers.

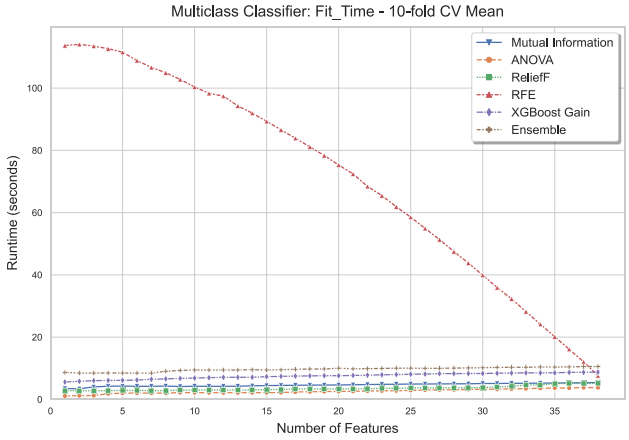

Fig. 10. Fit Time as a function of the number of variables for Multiclass Classifiers.

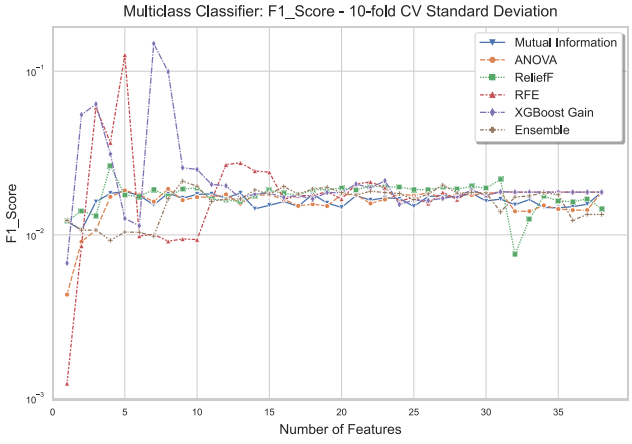

Fig. 9. Standard Deviation of F1 Score as a function of the number of variables for Multiclass Classifiers.

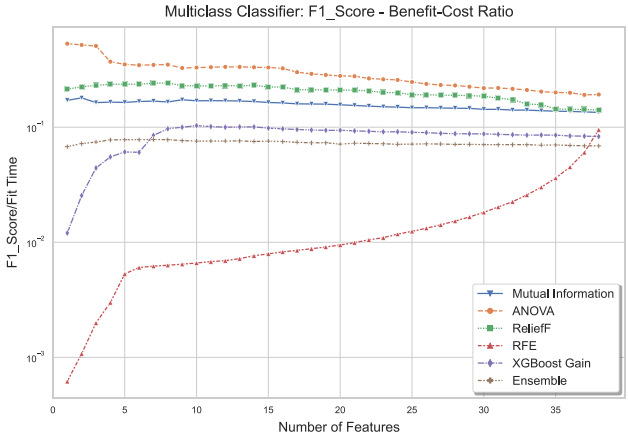

Fig. 11. F1 Score Benefit-Cost Ratio as a function of the number of variables for Multiclass Classifiers.

focusing on the analysis for K = 15 selected variables (quantity arbitrarily chosen). In the table we can compare the performance of the traditional Multiclass classifiers, Multiclass OvO, and Multiclass OvR, in addition to Binary classifiers. In general, a Feature Selection method provides close classification results for the three types of Multiclass classifiers. However, the Fit Time can differ significantly between them. Given the large number of classes (S = 12), the OvO and OvR strategies become computationally more costly than traditional Multiclass training, even though they only perform binary XGBoost training. In several methods of Feature Selection (such as ANOVA and Mutual Information), twice the Fit Time was spent to train an OvO Multiclass classifier in relation to a traditional Multiclass classifier. Similar results were obtained for the other K values.

4) Comparison of ANOVA and Label-Independent Feature Selection Methods: ANOVA was above all other label-dependent methods in the benefit-cost ratio curves that were constructed. For this reason, in this section it is compared with **label-independent** methods in terms of the number of features selected, F1 Score and Fit Time.

Three K values (K=30, K=20, K=10) were arbitrarily chosen to illustrate the variation in F1 Score and Fit Time as a more restricted number of features is selected. The results are shown in Table VIII. By the way the experiment is designed, the input of the ANOVA block corresponds to the output of the step of removing highly correlated variables. With this in mind, it is possible to comment on the general process of removing variables performed.

The decrease from 77 to 10 variables (87% reduction) brought with it a subtle performance degradation in all cases. For Binary, Multiclass, Multiclass OvO and Multiclass OvR classifiers, there was a reduction in the F1 Score of, respectively, **1.06**%, **2.05**%, **2.86**% and **2.31**%. However, the reduction in Fit Times is significantly more expressive. Binary, Multiclass, Multiclass OvO and Multiclass OvR classifiers had their times reduced by **19.86**%, **57.30**%, **35.39**% and **59.84**%, respectively. It is reasonable, therefore, to eliminate some variables to obtain such runtime gains, especially on large datasets.

V. Conclusion

In this work, the CICDDoS2019 dataset was used as a basis for verifying the impact of FS on the DDoS attack classification quality metrics. The reference classifier was XGBoost. FS techniques that are independent of the label of the samples were considered, such as the use of domain knowledge, and removal of attributes with low variance and high correlation. Methods that used the information contained in the sample label were also used, such as ANOVA, Mutual Information, ReliefF, XGBoost Gain, and RFE, in addition to an Ensemble technique involving the rankings of variables generated by other methods.

The removal of attributes with low variance did not alter the classification metrics. It was possible to verify the robustness

TABLE VII
CLASSIFICATION METRICS FOR THE LABEL DEPENDENT FEATURE SELECTION METHODS (WITH K=15 FEATURES SELECTED)

Binary Classifiers			
Method	Accuracy	F1 Score	Fit Times (s)
ANOVA	99.73 ± 0.22	99.73 ± 0.22	0.195 ± 0.018
MI	97.44 ± 0.69	97.48 ± 0.68	1.70 ± 0.11
ReliefF	99.43 ± 0.28	99.43 ± 0.28	1.48 ± 0.14
Gain	99.77 ± 0.23	99.77 ± 0.23	0.635 ± 0.018
RFE	99.77 ± 0.23	99.77 ± 0.23	9.93 ± 0.29
Ensemble	99.77 ± 0.23	99.77 ± 0.23	3.26 ± 0.12
Multiclass Classifiers			
Method	Accuracy	F1 Score	Fit Times (s)
ANOVA	73.0 ± 1.7	71.6 ± 1.8	2.167 ± 0.020
MI	73.7 ± 1.8	72.7 ± 1.5	4.434 ± 0.027
ReliefF	72.3 ± 1.8	70.6 ± 1.9	3.154 ± 0.028
Gain	73.1 ± 1.7	71.7 ± 1.8	7.303 ± 0.094
RFE	72.5 ± 2.0	70.9 ± 2.4	89.34 ± 0.72
Ensemble	72.9 ± 1.7	71.5 ± 1.8	9.443 ± 0.059
Multiclass Classifiers - One vs One			
Method	Accuracy	F1 Score	Fit Times (s)
ANOVA	72.8 ± 1.8	71.3 ± 1.9	5.730 ± 0.028
MI	73.9 ± 1.9	72.9 ± 1.8	7.647 ± 0.035
ReliefF	72.1 ± 1.5	70.3 ± 1.6	6.726 ± 0.024
Gain	73.0 ± 1.7	71.5 ± 1.8	10.938 ± 0.079
RFE	72.6 ± 1.9	70.9 ± 2.3	94.6 ± 1.1
Ensemble	72.9 ± 1.9	71.4 ± 2.0	12.752 ± 0.045
Multiclass Classifiers - One vs Rest			
Method	Accuracy	F1 Score	Fit Times (s)
ANOVA	73.3 ± 1.7	71.9 ± 1.8	2.94 ± 0.13
MI	73.9 ± 1.8	72.9 ± 1.6	5.136 ± 0.025
ReliefF	72.3 ± 1.6	70.5 ± 1.6	3.925 ± 0.051
Gain	73.2 ± 1.5	71.7 ± 1.6	8.22 ± 0.23
RFE	73.0 ± 1.8	71.3 ± 2.2	92.2 ± 1.8
Ensemble	73.4 ± 1.9	71.9 ± 2.1	10.40 ± 0.23

TABLE VIII
COMPARISON OF ANOVA AND LABEL-INDEPENDENT FEATURE SELECTION METHODS

Binary Classifiers			
Method	Features	F1 Score	Fit Times (s)
Preprocessing	77	99.79 ± 0.23	0.3594 ± 0.0092
Basic methods	63	99.79 ± 0.23	0.3686 ± 0.0090
Correlation	41	99.75 ± 0.27	0.415 ± 0.059
ANOVA (K=30)	30	99.70 ± 0.27	0.3528 ± 0.0096
ANOVA (K=20)	20	99.73 ± 0.22	0.240 ± 0.030
ANOVA (K=10)	10	98.73 ± 0.27	0.288 ± 0.038
Multiclass Classifiers			
Method	Features	F1 Score	Fit Times (s)
Preprocessing	77	73.0 ± 1.5	5.08 ± 0.28
Basic methods	61	73.0 ± 1.5	4.60 ± 0.32
Correlation	36	72.8 ± 1.7	3.34 ± 0.11
ANOVA (K=30)	30	71.6 ± 1.8	3.273 ± 0.033
ANOVA (K=20)	20	71.5 ± 1.8	2.56 ± 0.026
ANOVA (K=10)	10	71.5 ± 1.7	2.169 ± 0.022
Multiclass Classifiers - One vs One			
Method	Features	F1 Score	Fit Times (s)
Preprocessing	77	73.3 ± 1.7	8.371 ± 0.083
Basic methods	61	73.3 ± 1.7	7.66 ± 0.38
Correlation	36	73.3 ± 1.6	6.716 ± 0.045
ANOVA (K=30)	30	71.2 ± 1.8	6.633 ± 0.064
ANOVA (K=20)	20	71.2 ± 1.5	6.133 ± 0.052
ANOVA (K=10)	10	71.2 ± 1.9	5.408 ± 0.029
Multiclass Classifiers - One vs Rest			
Method	Features	F1 Score	Fit Times (s)
Preprocessing	77	73.4 ± 1.6	7.005 ± 0.046
Basic methods	61	73.4 ± 1.6	6.34 ± 0.34
Correlation	36	73.1 ± 1.7	4.412 ± 0.056
ANOVA (K=30)	30	71.4 ± 1.8	4.22 ± 0.025
ANOVA (K=20)	20	71.5 ± 1.6	3.393 ± 0.019
ANOVA (K=10)	10	71.7 ± 1.7	2.813 ± 0.014

of the XGBoost, which maintains stable performance results even without 78% of the original variables, for the Binary classifier, and 60%, in the Multiclass case. The training of a Multiclass classifier takes considerably more time than a Binary, for the same number of samples and attributes. Methods of selecting variables that incorporate a classifier in their execution, such as wrapper (RFE) and embedded methods (XGBoost Gain), need more time to be executed in the Multiclass case. Due to its fast execution time, ANOVA proved to be the most advantageous label-dependent FS technique, both for Binary and Multiclass classifiers.

As main contributions made in this work, we can highlight:

- The formalization of the notion of dataset unbalance, using the Pielou Index;
- Addressing the DDoS attack classification problem from both a Binary and Multiclass standpoint;
- The use of meta-learning strategies such as One-Vs-One and One-Vs-Rest;
- The use of a wide range of Feature Selection techniques and the creation of a benchmark of classification metrics according to them.

This study does not exhaust the FS theme in the classification of DDoS attacks, but brings a perspective that can serve as a basis for future work. A limitation of the work was the use of only one dataset (CICDDoS2019), artificially generated in a testbed by the authors of [24]. A dataset extracted from a production environment or one that is more balanced could have generated different results. All experiments were performed via CPU, but the parallelization of several GPU operations can bring several gains in execution time [45]. An interesting point would be to list which variables were selected in each Cross-Validation round, list which were the most frequent and whether the choice depends on the Feature Selection method used.

Most of the Feature Selection algorithms considered were available in Scikit-Learn, but there are other interesting techniques described in the literature such as mRMR [46], BORUTA [47], or even the use of SHAP values [48] to rank the attributes. To verify the ability to generalize the results described here, the use of the same methodology is proposed with other base classifiers, such as KNN and SVM, or non-linear models, such as Neural Networks.

The programs and graphs created for this article, as well as the additional results mentioned in the text and not illustrated for the sake of brevity are fully available at the site https://github.com/pedrohauy/ddos_feature_selection.

Acknowledgement: this paper was supported by Huawei-USP research partnership.

REFERENCES

[1] "Global DDoS Summary, March 2021 | NETSCOUT Cyber Threat Horizon," NETSCOUT [Online]. Available: https://horizon.netscout.com/?atlas=summary, Accessed on: Apr. 16, 2021.

[2] L. Newman, "DDoS-For-Hire' Is Fueling a New Wave of Attacks | WIRED," WIRED [Online]. Available: https://www.wired.com/story/ddos-for-hire-fueling-new-wave-attacks/, Accessed on: Apr. 16, 2021.

[3] C. Lowe, "Cybersecurity and Reputation | Stronger International Inc. | Cyber Security Training | IT Training," Stronger [Online]. Available: https://stronger.tech/cybersecurity-and-reputation/, Accessed on: Apr. 16, 2021.

[4] N. Bindra and M. Sood, "Detecting DDoS Attacks Using Machine Learning Techniques and Contemporary Intrusion Detection Dataset," *Automatic Control and Computer Sciences*, Vol. 53, No. 5, pp. 419–428, 2019, doi: 10.3103/S0146411619050043

[5] Silva A. A. A, Zhou F., Pontes E., Simplicio M. A., Aguiar R., Guelfi A. and Kofuji S. T. "Energy-efficient node position identification through payoff matrix and variability analysis," *Telecommunication Systems*, Vol. 65, No. 3, pp. 459-477, 2017, doi: 10.1007/s11235-016-0245-4

[6] A.A.A. Silva, N. Ferraz Jr., A.E. Guelfi, S.H.I. Barboza and S.T. Kofuji "Grouping detection and forecasting security controls using unrestricted cooperative bargains," *Computer Communications*, Vol. 146, pp. 155-173, 2019, doi: 10.1016/j.comcom.2019.07.022

[7] Ferraz Jr. N., Silva A. A. A, Guelfi A. and Kofuji S. T. "IoT6Sec: reliability model for Internet of Things security focused on anomalous measurements identification with energy analysis," *Wireless Networks*, Vol. 25, No. 4, pp. 1533-1556, 2019, doi: 10.1007/s11276-017-1610-2

[8] H. Polat, O. Polat and A. Cetin, "Detecting DDoS Attacks in Software-Defined Networks Through Feature Selection Methods and Machine Learning Models," *Sustainability*, 12, 1035, 2020, doi: 10.3390/su12031035

[9] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, Vol. 25, pp.152–160, 2018, doi: 10.1016/j.jocs.2017.03.006

[10] A. Gupta, "Distributed Denial of Service Attack Detection Using a Machine Learning Approach," M.S. Thesis, University of Calgary, Calgary, AB, Canada, 2018, doi: 10.11575/PRISM/32797

[11] M. J. Post, P. Putten and J. N. Rijn, "Does Feature Selection Improve Classification? A Large Scale Experiment in OpenML," *Advances in Intelligent Data Analysis XV. IDA 2016*, Vol. 9897, 2016, doi: 10.1007/978-3-319-46349-0_14

[12] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers and Electrical Engineering*, Vol. 40, pp. 16–28, 2014, doi: 10.1016/j.compeleceng.2013.11.024

[13] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, Vol. 3, pp. 1157-1182, 2003.

[14] J. Wang, J. Xu, C. Zhao, Y. Peng and H. Wang, "An ensemble feature selection method for high-dimensional data based on sort aggregation." *Systems Science & Control Engineering*, Vol.7, No.2, pp. 32-39, 2019, doi: 10.1080/21642583.2019.1620658

[15] A. Pattawaro and C. Polprasert, "Anomaly-Based Network Intrusion Detection System through Feature Selection and Hybrid Machine Learning Technique," *Sixteenth International Conference on ICT and Knowledge Engineering*, 2018, doi: 10.1109/ICTKE.2018.8612331

[16] N. Lindqvist and T. Price, "Evaluation of Feature Selection Methods for Machine Learning Classification of Breast Cancer," Degree Project, Kth Royal Institute of Technology, School of Electrical Engineering and Computer Science, Stockholm, Sweden, 2018.

[17] C. Wang, H. Yao and Z. Liu, "An efficient DDoS detection based on SU-Genetic feature selection," *Cluster Computing*, Vol. 22, pp. 2505–2515, 2019, doi: 10.1007/s10586-018-2275-z

[18] S. S. Dhaliwal, A. Nahid and R. Abbas, "Effective Intrusion Detection System Using XGBoost," *Information*, Vol. 9, No. 149, 2018, doi: 10.3390/info9070149

[19] Z. Chen, F. Jiang, Y. Cheng, X. Gu, W. Liu and J. Peng, "XGBoost Classifier for DDoS Attack Detection and Analysis in SDN-based Cloud," *IEEE International Conference on Big Data and Smart Computing*. 2018, doi: 10.1109/BigComp.2018.00044

[20] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," *Journal of Information Security and Applications*, Vol. 44, pp. 80–88, 2019, doi: 10.1016/j.jisa.2018.11.007

[21] S. K. Dey, M. M. Rahman and M. R. Uddin, "Detection of Flow Based Anomaly in OpenFlow Controller: Machine Learning Approach in Software Defined Networking," *4th International Conference on Electrical Engineering and Information & Communication Technology*, 2018, doi: 10.1109/CEEICT.2018.8628105

[22] K. S. Hoon, K. C. Yeo, S. Azam, B. Shanmugam and F. De Boer, "Critical review of machine learning approaches to apply big data analytics in DDoS forensics," *International Conference on Computer Communication and Informatics*, Coimbatore, India, 2018, doi: 10.1109/ICCCI.2018.8441286

[23] S. Das, A. M. Mahfouz, D. Venugopal and S. Shiva, "DDoS Intrusion Detection through Machine Learning Ensemble," *IEEE 19th International Conference on Software Quality, Reliability and Security Companion*, 2019, doi: 10.1109/QRS-C.2019.00090
[24] I. Sharafaldin, A. H. Lashkari, S. Hakak and A. A. Ghorbani, "Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy," *IEEE 53rd International Carnahan Conference on Security Technology*, Chennai, India, 2019, doi: 10.1109/CCST.2019.8888419
[25] Y. S. Hussein, "Network Intrusion Detection for Distributed Denial-of-Service (DDoS) Attacks using Machine Learning Classification Techniques," M. S. Thesis, University of Victoria, Victoria, BC, Canada, 2020.
[26] J. Li, "Detection of Ddos Attacks Based On Dense Neural Networks, Autoencoders And Pearson Correlation Coefficient," M. S. Thesis, Dalhousie University, Halifax, NS, Canada, 2020.
[27] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," *KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, doi: 10.1145/2939672.2939785
[28] I. Reinstein, "XGBoost, a Top Machine Learning Method on Kaggle, Explained," KDnuggets [Online]. Available: https://www.kdnuggets.com/2017/10/xgboost-top-machine-learning-method-kaggle-explained.html, Accessed on: Apr. 16, 2021.
[29] A. Husain, A. Salem, C. Jim and G. Dimitoglou, "Development of an Efficient Network Intrusion Detection Model Using Extreme Gradient Boosting (XGBoost) on the UNSW-NB15 Dataset," *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2019, doi: 10.1109/ISSPIT47144.2019.9001867
[30] J. Xie, V. Rojkova, S. Pal and S. Coggeshall, "A Combination of Boosting and Bagging for KDD Cup 2009 - Fast Scoring on a Large Database," *JMLR: Workshop and Conference Proceedings*, Vol. 7, pp. 35-43, 2009.
[31] M. Kuhn and K. Johnson, *Feature Engineering and Selection: A Practical Approach for Predictive Models*, 1st ed., USA: Chapman and Hall/CRC Press, 2019, p. 242.
[32] B. C. Ross, "Mutual Information between Discrete and Continuous Data Sets," *PLoS ONE*, Vol. 9, 2014, doi: 10.1371/journal.pone.0087357
[33] R. J. Urbanowicz, M. Meeker, W. La Cava and J. H. Moore, "Relief-based feature selection: Introduction and review," *Journal of Biomedical Informatics*, Vol. 85, 2018, pp. 189-203, doi: 10.1016/j.jbi.2018.07.014
[34] M. Kuhn and K. Johnson, *Applied Predictive Modeling*, 1st ed., USA: Springer, 2013.
[35] M. Marvi, A. Arfeen and R. Uddin, "A generalized machine learning-based model for the detection of DDoS attacks," *Int J Network Mgmt*, 2020, doi: 10.1002/nem.2152E
[36] E. C. Pielou, "The Measurement of Diversity in Different Types of Biological Collections," *Journal of Theoretical Biology*, Vol. 13, pp. 131-144, 1966, doi: 10.1016/0022-5193(66)90013-0
[37] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, Vol. 27, pp. 379–423, 623–656, 1948, doi: 10.1002/j.1538-7305.1948.tb01338.x
[38] H. Liu, M. Zhou, X. S. Lu and C. Yao, "Weighted Gini Index Feature Selection Method for Imbalanced Data," *IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018, doi: 10.1109/ICNSC.2018.8361371
[39] "scikit-learn: machine learning in Python — scikit-learn 0.24.1 documentation," scikit-learn [Online]. Available: https://scikit-learn.org/stable/index.html, Accessed on: Apr. 16, 2021.
[40] "ReliefF," PyPI [Online]. Available: https://pypi.org/project/ReliefF/, Accessed on: Apr. 16, 2021.
[41] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, Vol. 45, Issue 4, pp. 427-437, 2009, doi: 10.1016/j.ipm.2009.03.002
[42] J. Opitz and S. Burst, "Macro F1 and Macro F1," *arXiv*, 2019, arXiv:1911.03347
[43] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: The MIT Press, 2012, p. 503.
[44] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006, p. 183.
[45] "XGBoost GPU Support," xgboost 1.4.0-SNAPSHOT documentation [Online]. Available: https://xgboost.readthedocs.io/en/latest/gpu/index.html, Accessed on: Apr. 16, 2021.
[46] H. Peng, F. Long and C. Ding, "Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy," *IEEE Transactions On Pattern Analysis And Machine Intelligence*, Vol. 27, No. 8, 2005, doi: 10.1109/TPAMI.2005.159
[47] M. B. Kursa and W. R. Rudnicki, "Feature Selection with the Boruta Package," *Journal of Statistical Software*, Vol. 36, Issue 11, 2010, doi: 10.18637/jss.v036.i11
[48] S. M. Lundberg and S. Lee, "A Unified Approach to Interpreting Model Predictions," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, 2017, arXiv:1705.07874v2

Pedro Araujo is a Master's degree student at the University of Sao Paulo. He also holds a Bachelor's Degree in Electronics Engineering from the University of Sao Paulo (2013), where he has done undergraduate research in Signal Processing. He has over 7 years of experience in telecommunications, IP network planning and automation. His research interests include Machine Learning, Statistics, Software-defined Networks and Security Systems.

Anderson Silva received his PhD. Degree in Computer Engineering from USP in 2016, his Master's degree in Computer Engineering from IPT in 2010, his MBA in Systems Analysis from FECAP in 1993 and his Data Processing degree from FIEO in 1991. Anderson has over 25 years of experience in the IT field. Currently he is a professor in several undergraduate and graduate courses in Sao Paulo and develops his second postdoctoral project in machine learning security at USP.

Norisvaldo Ferraz Junior is a PhD student at USP. He received his Master's Degree in Computer Engineering from IPT in 2016, his MBA in the security of systems and environments from FASP in 2007 and his Bachelor degree in systems analysis in 2003. He works at FUNDACENTRO since 2005 as an analyst in science and technology. His research interests include security in Internet of things, wireless sensor networks, machine learning, and intrusion prevention systems.

Fabio Cabrini holds a Master's degree in Electrical Engineering from Universidade de São Paulo (2006), graduated in Materials Technology, Processes and Electronic Components at FATEC São Paulo (1996). Currently he is a Ph.D. student at Universidade de São Paulo (USP). His research interests include security and resilience in backend platforms for IoT and Smart Environments.

With a PhD in Transport Engineering (Poli/USP), a Master's in Computer Science (USP) and a Bachelor's Degree in Computer Science (UFMG), **Alessandro Santiago** is currently the business support manager for digital technologies at IPT. He is the coordinator and professor of the Professional Master's course in Applied Computing at IPT.

Adilson Guelfi is a Doctor in Eletric Engineering at the Electronic Engineering Department (USP). He has over 22 years of experience in training, education, research, project development, professional and innovation skills in information security area. Until the year 2017, Adilson held the position of regular professor in the master's program in Computer Engineering at the FIPT. Currently, Adilson has the position of Dean of Research and Graduate Studies at UNOESTE since 2014.

Sergio Kofuji holds a Bachelor's Degree in Physics from the University of Sao Paulo (1985), a Master's Degree in Electrical Engineering from the University of Sao Paulo (1988) and a PhD in Electrical Engineering from the University of Sao Paulo (1995), He is currently Professor Dr. at Polytechnic School of the University of Sao Paulo. Currently He has been focusing in Projects related to IoT, 5G communications, Smart Objects, Machine Learning and Artificial Intelligence, applied to Smart Cities and Smart Farms.