Cybersecurity Threat Management
Intelligent Data Reduction

Ana Magalhães Raposo
ana.m.raposo@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal

November 2018

Abstract

Due to the boost of cyber attacks registered in the past decade, many organizations from various sectors as energy, healthcare and even aviation have already implemented cyber security strategies commonly composed of antivirus and firewall technologies and intrusion detection systems. In order to reduce the digital data storage expenses and the complexity of the prevailing systems, this thesis main objectives are to simulate the operation of different intelligent intrusion detection systems, by implementing machine learning algorithms, and to build a filtering tool capable of reducing the digital storage usage to 20% of the present needs while improving data understanding. To achieve these goals, a benchmark database based on cyber events from Kyoto University is explored. Traditionally, intrusion detection systems consist on three distinct elements: data collection, data preprocessing and intrusion detection. Here, a novel approach, containing a data reduction tool between the data preprocessing and the actual intrusion detection, is proposed. In order to guarantee an improvement of data understanding and a fast computation process, the fast correlation-based filter, a feature selection filter based on relevance and redundancy analysis, is implemented as data reduction mechanism. The machine learning models applied to simulate the attack recognition process are artificial neural networks, naive Bayes and random forests. To do so, the models' hyperparameters are selected based on a random search optimization. At last, the detection capabilities of the complete system are inquired in order to demonstrate the benefits of the data reduction mechanism studied.

Keywords: cyber security, intrusion detection, machine learning, feature selection, filter methods

1. Introduction

At the present day, government organizations and business enterprises actively depend on interconnected computer systems to administer several primary services as energy, public transportation, healthcare or water [10]. Despite the operational advantages of the increased connectivity being undeniable once they bring uncountable benefits to the global population on a daily basis, it also has made individuals and organizations extremely more vulnerable to cyber attacks and privacy violations. Due to its operational characteristics, the aviation sector is one of the most critical structural systems that is not only susceptible to physical threats but also cyber threats. This project then addresses this matter from a very practical point of view and exploits an integrated tool capable of enhancing the currently existing cyber security solutions.

Commonly, this cyber security solutions are divided into network and computer (host) security systems each composed of antivirus and firewall technologies and an intrusion detection system (IDS), allied to highly specialized human resources. Summarily, the main purpose of an IDS is to aid in the encounter, determination and identification of any possible threat to the information system of a particular organization. As the use of data mining techniques for cyber security applications has been studied since the decade of 1990 [17], many of the intrusion detection systems being commercialized and implemented today exhibit a fair degree of intelligence. Given the extensive quantity and importance of the data that has to be inspected, the main difficulty associated with this tool is originated by the necessity of reaching a complicated trade-off between detection accuracy, detection speed and processing capabilities. Furthermore, the storage capacities required to achieve a high level performance of the IDS and maintain the compliance required by law imply a great monetary enlargement to the cost of the IDS licensing, representing a substantial portion of the system overall expense.

Thus, the primary goal of this project is to engineer and implement an intelligent filter capable of reducing the quantity of information that has to be maintained in temporary storage in order to be inspected by both human experts and automatic IDSs without compromising detection capacity. From

an academic perspective, this may be defined as a data reduction problem. From the business point of view, the filtering tool is intended to be complementary to the intrusion detection system and, simultaneously, to be independent from it, functioning for different systems based on different technologies.

This project was executed in partnership with EY Portugal and its practical implementation and evaluation was completed using a *python* environment.

This project consists of seven chapters. In the first one, the **introduction**, the real-world motivation behind this dissertation project is explored, the problem is explained and the goals are set. In **modern intrusion detection**, the existing modern techniques available to achieve the objectives defined in the previous are briefly introduced. The **data preparation** chapter is divided into two distinct subsections, dataset description and data preprocessing. The following chapter addresses the theoretical principles on which the **data reduction tool** applied in this thesis is based. Concepts as feature relevance and redundancy are largely explored as well as correlation measures. The fifth chapter, **intrusion detection algorithms**, covers the theoretical concepts behind the machine learning (ML) models applied in this project: artificial neural networks (ANN), naive Bayes (NB) and random forests (RF). The chapter also covers the strategy used to optimize the algorithms' hyperparameters. In the sixth chapter the **results** of the algorithms implementation are presented. The last chapter addresses the **conclusions** drawn from this thesis project. Here, the achievements accomplished are presented and a path for future work is delineated.

2. Modern Intrusion Detection

The following section provides a summary of valuable aspects regarding the state of the art of intrusion detection systems, machine learning and data reduction. It also introduces a novel intrusion detection approach.

2.1. Intrusion Detection Systems

Cyber security systems may be defined as the set of technologies and procedures designed to protect computers, networks, programs and information from attacks, unauthorized accesses, tempering or permanent destruction. These security systems are typically constituted of antivirus and firewall technologies and an intrusion detection system which assists in the encounter, determination and identification of any possible threat.

Regarding the origin of the attacks, they may be executed from both outside (external) or within the organization (internal). Intrusion detection systems, as vital lines of defense against cyber attacks, may also be divided into computer and network based. A computer (or host) based IDS monitors process and file activities associated with the software environment of a particular host, and a network based system, on the other hand, identify intrusions by monitoring the traffic trough network services. Concerning the detection process, IDSs may be categorized into three distinct types: anomaly-based, misused-based and hybrid. Anomaly-based methods consist in modeling the normal behavior of the system and, posteriorly, identifying the anomalies as deviations from that behavior. Misused-based, also designated signature-based techniques, are delineated to detect known attacks by using their electronic signatures. Hybrid techniques combine both the methods introduced above allowing the detection of unknown attacks without raising the false alarm rates [2].

Presently, the prediction abilities of an intrusion detection system are, at most cases, enhanced by a machine learning algorithm which provides it a certain degree of intelligence [24]. The traditional framework of an IDS consists in three distinct blocks: data collection, data preprocessing and intrusion detection. The intrusion detection element comprehends both the classifier training and the attack recognition.

2.2. Machine Learning Algorithms

Machine learning consists on classification and prediction methods based on the properties of the training data, giving computers the ability of learning without previous explicit programing [5].

ML problems may be supervised or unsupervised [21]. Supervised learning requires the agent to observe some example input-output pairs in order to be trained. These observations posteriorly allow the algorithm to map the behavior of the system and predict the outcome of unseen situations. In unsupervised learning, on the other hand, the machine is able to learn patterns about the input without having the need of explicit feedback. As the history of cyber attacks perpetrated against an organization is internally available for the construction of use-cases, supervised learning techniques may be used to implement IDSs [8].

In classic supervised learning, the problem is presented as a set of fixed-length feature vectors, named instances [27]. An instance, or sample, is generally defined as an assignment of values $s = (v_1, ..., v_N)$ to a set of features $F = (f_1, ..., f_N)$ and one of the l possible classes of the class label, $C = (c_1, ..., c_l)$. Here, the main goal is to properly classify (predict) a set of novel instances.

In order to simulate the operation of an intelligent intrusion detection system, distinct machine learning algorithms commonly used to predict cyber attacks were studied: artificial neural networks, association rules, decision trees, naive Bayes, random forests and support vector machines [8].

2.3. Data Reduction

There are two ways of decreasing the quantity of information present in a large dataset: reducing the number of instances (the lines) or reducing the number of features (the columns) [16].

The first method is known as instance selection and the second as feature selection. Usually, both techniques are used as a preprocessing stage before the application of a prediction algorithm in order to strive against what the researchers denominate the curse of dimensionality [3]. In this thesis, however, these methods were studied as a solution to the problem stated in section 1.

Despite also being used as a preprocessing procedure to eliminate noisy and erroneous instances, instance selection is mainly used for coping with the infeasibility of learning from extremely large datasets. When applied properly, this process is capable of reducing the dataset and, at the same time, maintaining, or even increasing the prediction accuracy of a ML algorithm.

Instance selection algorithms are commonly divided into two separate groups: filters and wrappers [11]. Filter evaluation performs data reduction without considering the purpose of the dataset. Identifying and eliminating duplicate instances is an extremely common filtering technique used worldwide [13]. Wrappers, on their turn, depend on ML algorithms to evaluate the results of the dimensionality reduction. Most wrapper methods focus on improving the quality of the training set trough the implementation of processes as identifying and eliminating mislabelled instances before the prediction process [7].

On the other hand, since ML problems exhibit many irrelevant and redundant variables, a feature selection process is typically required. Summarily, the main advantages of feature selection are facilitating data visualization and understanding, reducing computation and storage necessities, reducing training and utilization periods and improving prediction capabilities [12]. Feature selection algorithms may be divided into three different categories: filter, wrapper and embedded. Filters elect subsets of features in a generic pre-processing step which is completely independent of the predictive algorithm used to posteriorly classify the data [12]. Typically implemented due to their computational and statistical scalability, filters apply variable ranking techniques and subset elimination processes based on a specific criteria as linear correlation or information theory [22]. Wrappers perform feature selection by using a predictor as *blackbox* to obtain the variable subset which leads to the highest prediction performance. Here, the output of the machine learning algorithm is used as objective function to evaluate the variable subset generated by the wrapper implying the formulation of a search problem. Embedded methods select features during the training part of the prediction algorithm. These techniques present high efficiency in many aspects since they utilize all the available data, not imposing its division into training and testing set, and are fast in reaching the final solution, avoiding retraining the predictor in every iteration of the algorithm [12].

2.4. Proposed Approach

As introduced in section 1, the main objective of this thesis is to engineer a tool capable of reducing the quantity of information that has to be maintained in temporary storage in a cyber security environment.

The main difference between the novel framework proposed and the traditional IDS framework is the introduction of a separate data reduction element. Here, this element is represented separately from both the preprocessing and the classifier training blocks once it is intended to be independent from them, in the sense of being completely capable of supporting different technologies, not acting as an accessory *blackbox* tool to the classifier.

As one important goal of the filtering tool idealized, allied to the data reduction, is its capacity of improving information understanding, a feature selection approach appeared to be the best candidate. Using FS as an independent tool, instead of having to analyze every variable in detail, both IDSs and human experts may focus mainly on the selected features. In order to assure the self-sufficiency referred above and, at the same time, allow a fast computation process, the category of feature selection explored was the filter.

To guarantee the conservation of the prediction performance after the application of the data reduction tool proposed and evidence its general character, it was necessary to simulate its operation in conjunction with different types of classifiers. To do so, three distinct machine learning algorithms were selected to be implemented: artificial neural networks, naive Bayes and random forests.

An ANN was elected once it is one of the most common methods used to detect cyber attacks, according to literature. NB, on the other hand, were applied due to its simplicity and small computation requirements. As for RF, they were selected because of the high efficiency demonstrated for large datasets [8].

3. Data Preparation

The following section presents the set of benchmark data explored in this thesis, the Kyoto dataset, and introduces the preprocessing mechanisms performed to prepare the data.

3.1. Dataset Description

The benchmark dataset exploited in this thesis is extracted from the raw traffic data obtained by honeypot systems (a very popular tool used to attract

intruders into cyber networks in order to collect information about their malicious activities) deployed in Kyoto University [23]. After being collected, the raw information obtained from the honeypots suffers a complex preprocessing procedure to generate the 23 features displayed in the public dataset.

The 23 features presented are divided into two distinct categories, 15 of them are considered conventional (meaning that they are based on the extremely popular KDD Cup 99 dataset [8]) and the remaining 8 are designated additional (created simply to evaluate the University's network activities). 3 of the features, service, flag and protocol_type, are categorical. The remaining 20 are continuous.

The Kyoto dataset is also labeled, enabling a supervised prediction. The label indicates whether the connection was, or not, considered an attack.

In the original dataset, attacks were also labeled as known or unknown. However, in order to maintain the intrusion detection as a binary classification (yes or no), both known and unknown attacks were aggregated being considered as just one class.

3.2. Data Preprocessing

As real-world data may not always be available on the proper formats, data preprocessing often generates a very significant impact on a machine learning algorithm prediction performance. Here, the preprocessing mechanisms were implemented with the assistance of the *scikit-learn* library [19].

As the features of the Kyoto dataset were artificially generated based on the raw data collected from the University's honeypots, they are already a result of a complex feature construction process. Nonetheless, to solve this particular problem some alterations were introduced: elimination of all the identifying features; elimination of the time variable, start_time; and, separation of each of the detection features (IDS_detection, malware_detection and ashula_detection).

Once the ML algorithms selected to be implemented in this project tend to not perform well in the presence of categorical features [16], the next procedure applied to prepare the dataset was the treatment of those features (service, flag and protocol_type). To do so, a technique named one-hot-encoding was used. This method transforms each of the categories of a categorical variable into a new boolean feature. After the encoding, the dataset is constituted of 47 features, 15 continuous and 32 boolean.

The following preprocessing procedure applied was the normalization of the continuous variables. Normalization is commonly applied to cope with the large difference between the maximum and minimum values displayed in many features. This procedure usually scales the value magnitudes to a small interval, typically [-1,1] or [0,1].

The first logic step to be performed before applying any data normalization method would be the removal of outliers, or, in other words, the observations which lie in an abnormal distance from the remaining values in a random sample from a population [16]. However, after analyzing the remaining 15 continuous features, it was possible to verify that the supposed outliers actually represent valid observations. It was then decided not to remove extreme values, once they may reveal to be essential to this project's study.

The normalization process was then carried out maintaining this values. To do so, the min-max algorithm was implemented:

$$v' = \frac{v - \min_A}{\max_A - \min_A} \tag{1}$$

Where v' and v represent the new and the original value, respectively, and the min and max represent the minimum and maximum values registered for the referred feature (in this case, A). Min-max scales the feature values to the range [0,1].

After the preprocessing stage, the dataset was ready for the implementation of data reduction and prediction algorithms.

4. Data Reduction Tool

The present section explores the theoretical principals behind the feature selection algorithm applied in this thesis. It also presents the pseudo-code used to implement the algorithm in *python*.

4.1. Feature Selection Framework

The traditional FS framework through subset evaluation is presented in figure 1.

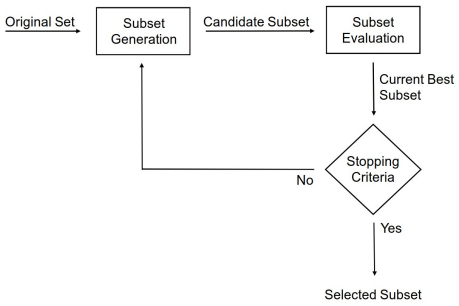

Figure 1: Traditional FS framework [27].

In the traditional framework, the subset generation block produces different subsets according to a specific search strategy. Each candidate subset is posteriorly evaluated and compared to the current best one. If, based on a predefined evaluation measure, the candidate subset is considered better, the current best subset is replaced. To complete the selection, this process of subset generation and evaluation continues until a given stopping criteria is satisfied. Contrary to most individual evaluation approaches, subset evaluation takes into account the existence of redundant features, allowing the obtainment of an approximate optimal subset.

4.2. Relevance and Redundancy

Since many algorithms are based on the search for a set of relevant and yet not redundant features, the concepts of relevance and redundancy are essential in feature selection. Regarding the formal definition of **relevance**, literature establishes three disjoint categories [14]. Strong relevance implies that the feature cannot be removed without comprising the optimality of the subset created. Weak relevance indicates that the feature is not absolutely necessary to form an optimal subset but may became indispensable at specific conditions. Irrelevance, on the other hand, implicates that the feature is not necessary at all.

The definitions presented above may, however, not be sufficient to determine which features should be selected and which ones should be neglected. To do so, it is imperative to define feature redundancy.

To introduce this concept, it is necessary to present the notion of **Markov blanket** [15]:

> given a feature, f_i, let $M_i \subset F(f_i \notin M_i)$, M_i is said to be a Markov blanket for f_i iff $p(F - M_i - \{f_i\}, C|f_i, M_i) = p(F - M_i - \{f_i\}, C|M_i)$.

Considering a prior elimination of irrelevant features, a definition for feature **redundancy**, based on the Markov blanket, was proposed [27]:

> let S be the current set of features, a feature is redundant and hence should be removed from S iff it is weakly relevant and has a Markov blanket M_i within S.

Given the concepts introduced above, the optimal subset, G, may be described as the combination of strongly relevant with weakly relevant but not redundant features.

4.3. Correlation Measures

One of the most applied correlation measures is the linear correlation coefficient also named Pearson correlation coefficient, ρ:

$$\rho = \frac{\sum_i (x_i - \overline{x_i})(y_i - \overline{y_i})}{\sqrt{\sum_i (x_i - \overline{x_i})^2} \sqrt{\sum_i (y_i - \overline{y_i})^2}} \tag{2}$$

Where $\overline{x_i}$ and $\overline{y_i}$ represent the mean value of the variables x and y, respectively. Statistically, the numerator represents the covariance between x and y and the two plots of the denominator represent the variance of the two variables. If two variables are completely linearly correlated, ρ takes the value 1 or -1; if, on the other hand, the variables are independent, ρ is equal to 0.

Despite being one of the most applied relevance indicators in machine learning, it is not always appropriate to assume linear correlation between features once it may not be capable of portray the real non-linear relation between them [12]. In most cases, a non-linear measure is required.

Many of the non-linear correlation measures are based on information theory concepts, as mutual information (MI). MI is an elemental concept in information theory which quantifies the general interdependence between random variables (r.v.) [12]. For two random variables, x and y, the mutual information, $MI(x, y)$, may be defined as:

$$MI(x, y) = H(x) + H(y) - H(x, y) \tag{3}$$

Where H represents the entropy of a random variable and measures the uncertainty related with it [25]. For a continuous r.v., x, the entropy may be represented as:

$$H(x) = -\int p(x) \log p(x)\ dx \tag{4}$$

Being $p(x)$ the marginal probability distribution of x.

According to the definition of H, for a continuous scenario equation 3 may be rewritten in the form:

$$MI(x, y) = \int\int p(x, y) \log \frac{p(x, y)}{p(x)p(y)}\ dxdy \tag{5}$$

As a feature selection tool, mutual information measures the quantity of information provided by a feature to the class label, enabling variable ranking. Theoretically, MI provides two serious advantages over the remaining ranking techniques: it accurately measures general statistical dependency between features and it is invariant to the monotonic transformations performed on variables as a preprocessing procedure. However, empirical results demonstrate that MI tends to favor features with more values [27].

To compensate the bias introduced by this measure and restrict its value to the range [0, 1], the concept of symmetrical uncertainty (SU) was proposed in [20]:

$$SU(X, Y) = 2\left[\frac{MI(X, Y)}{H(X) + H(Y)}\right] \tag{6}$$

SU normalizes the value of MI by dividing it by the corresponding entropies. Here, the value 0 indicates that X and Y are completely independent. The value 1, on the other hand, reveals that one variable totally portrays the value of the other.

4.4. Fast Correlation-Based Filter

Although the framework presented before (fig. 1) has been proven effective in removing irrelevant and redundant features, even with the application of heuristic search strategies the subset generation step requires incredibly powerful computation capacities in order to search through feature subsets. To overcome this issue, enabling an effective feature selection process capable of scaling well to large datasets, a new framework was proposed. The new

FS framework is divided in two steps, a relevance analysis followed by a redundancy evaluation. In the first block, irrelevant features are removed and, in the second, redundancy analysis allows the elimination of the redundant features among the relevant ones. Besides being effective when it comes to reaching an optimal subset, this framework is also extremely useful as a data understanding facilitator, permitting an a priori recognition of irrelevant and redundant features.

As the data reduction tool idealized is intended to be as optimal as possible and, at the same time, expedite the process of information analysis, this framework presented itself as a serious candidate for its practical implementation. So, an effective feature selection algorithm based on this framework and denominated **fast correlation-based filter** was studied [27]. The pseudo-code which describes the *python* implementation of the FCBF is presented below:

Algorithm 1: Fast Correlation-Based Filter

```
Input  : Initial dataset
         S = {f_1, f_2, ..., f_N, C} and a
         predefined parameter δ
Output: A selected subset S_b
begin
initialize S' = {}
calculate SU_c = SU(f_1, f2..., f_N; C)
for i=1 to N do
    if SU_{i,c} > percentile(SU_c, δ) then
        append f_i to S';
end
order S' in descending SU_{i,c} value
f_j = get_first_element(S')
repeat
    f_i = get_next_element(S', f_j)
    if f_i ≠ {} then
        repeat
            calculate SU_{i,j}
            if SU_{i,j} ≥ SU_{i,c} then
                remove f_i from S'
            else
                f_i = get_next_element(S', f_i)
            end
        until f_i == { };
    else
        f_j = get_next_element(S', f_j)
    end
until f_j == {};
S_b = S'
return S_b
```

In this project, the FCBF algorithm was simplified. Instead of using δ as a threshold that defines the absolute value of SU_C from which the features are considered irrelevant, δ is used here as a percentile of SU_C. SU_C is calculated for each feature and afterwards, according to the value of δ, a percentage of the variables are discarded. If δ is equal to 10, for instance, the features which have SU_C values minor than the lower 10^{th} percentile of the complete set of SU_C are excluded. This alteration is presented as a simplification once it does not imply an enlarged study about the statistical meaning of the absolute value of SU_C used as threshold.

5. Intrusion Detection Algorithms

As stated before, this project applies ML algorithms to simulate the functioning of different IDSs. The algorithms' theoretical principles as well as the hyperparameter optimization procedure used to implement them in *python*'s *sckit-learn* library [19] are presented next.

5.1. Artificial Neural Networks

ANN have been developed as generalizations of mathematical models of biological nervous systems. In a simplified mathematical representation of a neuron, the biological synaptic transmission is reproduced by connection weights, w_i, which modulate the effect of the respective input signals, z_i. The non-linear characteristics displayed by neurons is, on the other hand, represented by a transfer function, tf. The learning ability of an artificial neuron is accomplished by adjusting the weights according to the chosen learning algorithm.

In this project, the method implemented to build the first classifier was the backpropagation learning rule applied to a multilayer perceptron (MLP) as the one represented in figure 2.

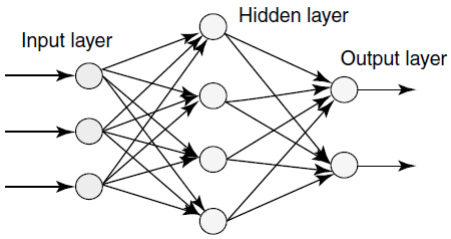

Figure 2: Schematic representation of a multilayer perceptron [1].

As shown in fig. 2, the MLP is constituted by more than the two plain layers, displaying hidden layers. Thus, the signal may flow trough multiple hidden layers before the final non-linear output is produced. Despite not being represented in the image (fig. 2), every arrow connecting the nodes has its respective connection weight.

The backpropagation method used to train the ANN may be synthesised as follows:

1. Selection of the supervised training data;
2. Construction of the multilayer perceptron with the respective connection weights;
3. Random initialization of the connection weights;

4. Selection of the appropriate error function, learning rate and momentum (which determines the effect of the past weight changes on the current calculation);
5. Application of a weight update rule, Δw_{ij}, to each target output for all hidden layers and computation of the associated error value;
6. Continuous repetition of the previous step until the error function is appropriately minimized.

In backpropagation, an adequate choice of both learning rate and momentum is crucial to the success and speed of the ANN training process. Moreover, literature demonstrates that the use of sufficient hidden layers allows the modelation of any non-linear system with an acceptable accuracy [9].

5.2. Naive Bayes

Bayesian networks are statistical classifiers which predict class membership probabilities, such as the possibility of a given instance belonging to a specific class. As the name suggests, bayesian networks are based on Bayes' theorem [26]. A naive bayesian network is a particular case among bayesian classifiers since it assumes that the impact of an attribute value on a given class, c_i, is completely independent of the values of the remaining attributes [18]:

$$p(s|c_i) = \prod_{j=1}^{N} p(v_j|c_i) \tag{7}$$

The NB method implemented in this project is summarized below:

1. Given a sample $s = (v_1, ..., v_N)$, the classifier predicts that s belongs to the class c_i which has the highest a posteriori probability, conditioned on s. In other words, it predicts that s belongs to the class that maximizes $p(c_i|s)$, computed using the Bayes' theorem [26];
2. Since $p(s)$ is constant for all classes, only the numerator, $p(c_i)p(s|c_i)$, has to be maximized;
3. If the class probabilities $p(c_i)$ are not known yet, it is commonly assumed that the classes are equally likely. In that case, the only element to be maximized would be $p(s|c_i)$. Note that the class probabilities may be estimated using $p(c_i) = \frac{\nu(c_i,N)}{N}$, where ν is the class frequency;
4. Apply the class conditional independence assumption to compute $p(s|c_i)$ using the probabilities distributions, Gaussian for continuous datasets or Bernoulli for binary ones;
5. Return the prediction result as the class c_i that maximizes $p(s|c_i)p(c_i)$.

Despite the unrealistic character of the independence assumption, the NB classifier has been proven to be surprisingly effective since its classification decision may often be accurate even if its probability estimates are faulty. This method is also very simple to implement, displaying itself as a powerful compromise to the trade-off between computation speed and performance accuracy [18] .

5.3. Random Forests

RF is considered an ensemble method once it consists in the combination of tree predictors which vote for the most popular class in order to select the final classification result. Here, for the k^{th} tree, a random vector, Ω_k, is generated independently from the previous random vectors, $\Omega_1 ... \Omega_{k-1}$, but with the same mathematical distribution [6]. The RF method implemented in this thesis follows the steps presented below:

1. Elect n different random sets of samples from the initial training data (n= number of trees)
2. For each of the n sets, grow an unpruned classification tree applying the following specification: for each node expansion, rather than choosing the best split using all the prediction variables, randomly select m variables and choose the best split using only those m variables;
3. Expand each tree to the largest extent possible;
4. Estimate the prediction result by aggregating the outcomes of the n trees.

Once the training set is randomly sampled for each tree, about one-third of the information is left out of the prediction process. This portion of the data, denominated out-of-bag (oob), is posteriorly used as test set to perform an unbiased evaluation of the classification error [6]. Thus, RF does not require a cross-validation step. Allied to this factor, there are many other characteristics that lead researchers to consider RF a great candidate to classify large quantities of information: high efficiency on large datasets, capacity of estimating variable importance, high level of robustness with respect to noise and overfitting, and ability to estimate missing data.

5.4. Hyperparameter Optimization

In literature, the problem of identifying the best values for the hyperparameters, λ^*, is known as hyperparameter optimization.

The objective function of this optimization problem may be represented as follows [4]:

$$\lambda^* = \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left[\underset{s \in S^v}{\operatorname{mean}} \left[L(s; A_\lambda S^t) \right] \right] \tag{8}$$

Where Λ is the list of hyperparameters candidates in study, S^v is the validation dataset and L is the lost function associated with the learning algorithm A_λ (which depends on λ) in the mapping of S^t.

The most commonly applied technique to solve the hyperparameter optimization problem is a combination of grid and manual search. Grid search implies an initial selection of the possible candidates for each λ, followed by a complete combinatorial search. Manual search is mostly applied to identify the possible best candidates from the set Λ and to add some intuition to the selection process, being extremely difficult to non-expert users.

Random search was introduced as simpler and yet more practical alternative to the approach introduced above. In random search, the optimization process is carried out within a predefined interval and each trial is statistically independent. Compared with grid search, random search is computationally more effective and is proven of being able to encounter better models [4]. Due to its characteristics, random search was selected as the technique to be used in this thesis to tune the models' hyperparameters. It was implemented in *python* using the respective *scikit-learn* [19] function.

6. Results

The following chapter presents the results obtained after the practical implementation of the algorithms introduced in sections 4 and 5.

6.1. Data Reduction Tool

As the only input parameter to the FCBF algorithm, δ defines the number of features present in the final subset and, therefore, its significance must be carefully evaluated. Table 1 presents the number of relevant features, or, in other words, the number of features existing in the subset before the elimination of the redundant ones, and the number and respective numeric identification of the features displayed in the final subset for 9 different values of δ.

Table 1: FCBF results in function of δ.

δ	# relevant	# final	features in final subset
0	47	15	{9, 5, 46, 30, 27, 47, 4, 33, 24, 20, 21, 28, 31, 1, 29}
10	43	14	{9, 5, 46, 30, 27, 47, 4, 33, 24, 20, 21, 28, 31, 1}
20	38	13	{9, 5, 46, 30, 27, 47, 4, 33, 24, 20, 21, 28, 31}
30	34	12	{9, 5, 46, 30, 27, 47, 4, 33, 24, 20, 21, 28}
40	29	9	{9, 5, 46, 30, 27, 47, 4, 33, 24}
50	24	8	{9, 5, 46, 30, 27, 47, 4, 33}
60	20	8	{9, 5, 46, 30, 27, 47, 4, 33}
70	15	6	{9, 5, 46, 30, 27, 47}
75	13	6	{9, 5, 46, 30, 27, 47}

Table 1 shows that for $\delta = 0$, the final subset consists of 15 features. For $\delta = 75$, however, the number of selected features is reduced to 6, which represents less than 15% of the 47 variables obtained by encoding the original dataset. This signifies that, keeping the 25% of the most relevant variables, the fast correlation-based filter allows a 87% reduction in the number of variables and, consecutively, in the size of the dataset.

6.2. Hyperparameter Optimization

As introduced in section 5, a random search mechanism was utilized to achieve a base-line solution to the problem stated in equation 8. Afterwards, the results were evaluated and tuned manually to assure a fully enhanced performance. The hyperparameters' final values are presented in table 2.

Table 2: Hyperparameters' final values.

Model	Hyperparameters	Final values
ANN	hidden_layer_sizes	(5,3)
	learning_rate	'invscaling'
	momentum	0.55
	max_iter	800
NB	binarize	0.75
	alpha	0.65
RF	max_depth	'None'
	max_features	1
	criterion	'gini'
	n_estimators	78

Next, using the values displayed in table 2, the intrusion detection systems were implemented and their results were closely evaluated.

6.3. Intrusion Detection

The functioning of the novel approach presented in section 1 is mostly based on the qualities of the data reduction tool selected allied to the prediction performance of the ML models used to detect cyber attacks. So, in order to evaluate its performance, the three classifiers explored earlier were implemented (applying the hyperparameters presented above) using the variables selected by the FCBF algorithm as input.

The first step applied to evaluate the efficiency of the FCBF algorithm was the analysis of the area under the receivers operating characteristic (ROC) curve (AUC) value for the range of δ in study, [0,75], to understand how the reduction of variables affects the prediction capabilities of the classifiers.

So, figure 3 exhibits the variation of the AUC with the value of δ.

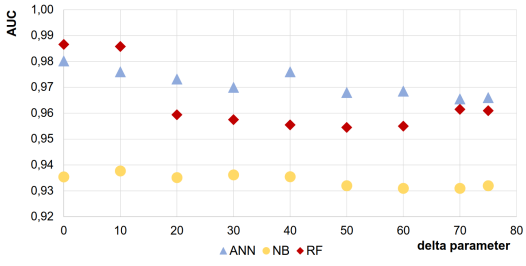

Figure 3: Variation of AUC with δ.

Observing the graphic in figure 3 it is possible to infer that the number of features selected by the algorithm according to the δ parameter does not impact the AUC values obtained on a large scale. The major oscillation verified occurs for the random forest classifier which AUC values vary from 0.986 to 0.955 (a difference of approximately 3%).

To further evaluate the efficiency of the FCBF algorithm, seven performance metrics were utilized: accuracy, recall, precision, specificity, negative predictive value (NPV), false alarm rate (FAR) and AUC [8].

Table 3 displays the results obtained by the three classifiers (ANN, NB and RF) using the 15 variables elected by the FCBF algorithm for $\delta = 0$.

Table 3: FCBF performance evaluation for $\delta = 0$.

Model	Performance Measure (%)						
	Accuracy	Recall	Precision	Specificity	NPV	FAR	AUC
ANN	91.58	73.97	93.82	98.12	90.99	1.82	98.0
NB	81.22	66.68	69.22	90.49	80.88	9.52	93.5
RF	93.57	78.60	96.01	98.79	92.85	1.21	98.7

For $\delta = 0$, despite the great class separation demonstrated by the AUC values, the detection capabilities of the classifiers, expressed by the recall, is not as high as expected. In this scenario, NB is the predictor which exhibits the poorest performance in the detection process. RF, on its turn, reveals itself as the best classifier (by a small margin to ANN).

To completely comprehend what happens to the attack detection capabilities of the ML models when less than 10 variables are used as input (that is, when it is most beneficial to use FCBF in terms of the tread-off AUC-number of features), the 6 features selected by the FCBF algorithm for $\delta = 75$ are utilized as input. Table 4 displays the results of this experiment.

Table 4: FCBF performance evaluation for $\delta = 75$.

Model	Performance Measure (%)						
	Accuracy	Recall	Precision	Specificity	NPV	FAR	AUC
ANN	91.21	73.97	92.42	97.71	90.88	2.29	96.72
NB	80.61	67.80	62.23	88.39	82.53	11.57	93.01
RF	91.41	76.71	93.53	98.12	90.79	1.89	96.12

According to the metrics exhibited in table 4, the detection capabilities of the three models applied, ANN, NB and RF, for $\delta = 75$ are quite similar to the ones presented for $\delta = 0$ (table 3). The class separation indicator, AUC, only decreases 2% when the ANN and RF are applied. For the NB classifier, however, the AUC stays in the 96% range (the same as for $\delta = 0$). So, comparing the metrics in the two tables (3 and 4) it is possible to comprehend that the data reduction strategy used is quite satisfactory. Decreasing the number of features from 15 to 6, the variation perceived in the performance measures is never higher than 3% (which occurs for the RF's precision).

The values obtained for $\delta = 75$ may be considered quite satisfactory, considering the data reduction achieved. The accuracy remains above 90% for the ANN and RF. NB, on its turn, presents an accuracy of 80%. The ability of detecting normal behavior continues clearly high, above 97% for ANN and RF and in 88% for NB. The AUC also remains above 92% for all the predictors which reveal great class separation.

7. Conclusions

This project's primary goal was to build a general data reduction tool capable of maintaining the prediction performance of an IDS and improving data understanding. In addition, the tool idealized was intended to be universal and attachable to any given intrusion detection system already in function, supporting as much technological solutions as possible. The approach presented to implement this tool consisted in adding a data reduction element to the intrusion detection framework, based on a feature selection strategy. To do so, the fast correlation-based filter algorithm was implemented. The only input parameter to this algorithm, apart from the cyber security data in study, is denominated δ and allows the elimination of a given percentage of the most irrelevant variables. Varying δ from 0 to 75, the actual variation in the final number of features was from 15 to only 6 (table 1), representing less than 20% of the original dataset.

Taking into account the data reduction achieved ($\approx$80%) the results obtained with the novel approach proposed are very satisfactory. The class separation indicator, AUC, remains above the 90% for the ANN and RF and above 80% for the NB, the classifier which presents the poorest performance. Better results would probably be achieved with the implementation of a customized feature selection algorithm for each classifier, possibly using a wrapper or embedded technique. However, as the data reduction tool idealized was intended to be universal and attachable to as much as diversified technologies as possible, the results obtained allow to conclude that the FCBF appears to present itself as a extremely viable solution.

References

[1] A. Abraham. *Handbook of Measuring System Design*, chapter Artificial Neural Networks. John Wiley and Sons Ltd., 2005.

[2] M. A. Aydin, A. H. Zaim, and K. G. Seylan. Hybrid intrusion detection system design for computer network detection. *Computers and Electric Engineering*, 35(3):517–526, 2009.

[3] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[4] J. Bergstra and Y. Bengilo. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, December 2012.

[5] C. M. Bishop. Pattern recognition and machine learning. In M. Jordan, K. Kleinberg, and S. Scholkopf, editors, *Information Science and Statistics*. Springer, 2006.

[6] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001.

[7] C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligent Research*, 11:131–167, 1999.

[8] A. L. Buczak and E. Guven. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys and Tutorials*, 18(2):1153–1176, 2016.

[9] M. W. Gardner and S. R. Dorling. Artificial neural networks (the multilayer perceptron) - a review of applications in the atmospheric sciences. *Atmospheric Environment*, 32(14/15):2627–2636, 1998.

[10] K. Gopalakrishnan, M. Govindarasu, D. W. Jacobson, and B. M. Phares. Cyber security for airports. *International Journal for Traffic and Transport Engineering*, 3(4):365–376, 2013.

[11] M. Grochoski and N. Jankowski. Comparison of instance selection algorithms ii. In *Lecture notes in computer science*, June 2004.

[12] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, March 2003.

[13] M. A. Hernandez and S. J. Stolfo. Real-world data is dirty: data cleansing and a merge/purge problem. *Data Mining and Knowledge Discovery*, 2:9–37, 1998.

[14] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In W. W. Cohen and H. Hirsh, editors, *Machine Learning: Proceedings of the Eleventh International Conference*, pages 121–129, San Francisco, CA, 1994. Morgan Kaufmann Publishers.

[15] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 284–292, Bari, Italy, July 1996. Morgan Kaufmann Publishers.

[16] S. B. Kotsiantis, D. Kanellopoulos, and P. E. Pintelas. Data preprocessing for supervised learning. *International Journal of Computer Science*, 1(1), 2006.

[17] S. Kumar and E. H. Spafford. An application of pattern matching in intrusion detection. Technical Report 94-013, Purdue University - Department of Computer Science, 1994.

[18] K. P. Murphy. Naive bayes classifiers, October 2006.

[19] 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. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2^{nd} edition, 1998.

[21] S. Russel and P. Norvig. *Artificial Intelligence - A Modern Approach*, pages 693–767. New Jersey: Prentice Hall, 3^{rd} edition, 2010.

[22] G. C. Shekar and F. Sahin. A survey on feature selection methods. *Computers and Electric Engineering*, 40(1):16–28, January 2014.

[23] J. Song, H. Takakura, and Y. Okabe. Description of kyoto university benchmark data, June 2010.

[24] B. Thuraisingham, L. Khan, and M. M. Masud. Data mining for security applications. In *2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, Shanghai, China, 2008. IEEE.

[25] G. D. Tourassi, E. D. Frederick, M. K. Markey, and C. E. F. Jr. Application of the mutual information criterion for feature selection in computer-aided diagnosis. *The International Journal of Medical Physics Research and Practice*, (28):2394–2402, December 2001.

[26] R. E. Walpole, R. H. Mayers, S. L. Myers, and K. Ye. *Probability and statistics for engineers and scientists*. Prentice Hall, 9 edition, 2011.

[27] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, (5):1205–1224, October 2004.