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

Prioritizing Positive Feature Values: A New Hierarchical Feature Selection Method

Pablo Nascimento da Silva · Alexandre Plastino · Alex A. Freitas

Received: date / Accepted: date

Abstract In this work, we address the problem of feature selection for the classification task in hierarchical and sparse feature spaces, which characterise many real-world applications nowadays. A binary feature space is deemed hierarchical when its binary features are related via generalization-specialization relationships, and is considered sparse when in general the instances contain much fewer "positive" than "negative" feature values. In any given instance, a feature value is deemed positive (negative) when the property associated with the feature has been (has not been) observed for that instance. Although there are many methods for the traditional feature selection problem in the literature, the proper treatment to hierarchical feature structures is still a challenge. Hence, we introduce a novel hierarchical feature selection method that follows the lazy learning paradigm – selecting a feature subset tailored for each instance in the test set. Our strategy prioritizes the selection of features with positive values, since they tend to be more informative – the presence of a relatively rare property is usually a piece of more relevant information than the absence of that property. Experiments on different application domains have shown that the proposed method outperforms previous hierarchical feature selection methods and also traditional methods in terms of predictive accuracy, selecting smaller feature subsets in general.

Keywords Hierarchical Feature Spaces · Feature Selection · Classification

P. N. da Silva (✉)
Institute of Computing, Fluminense Federal University (UFF), Niterói-RJ, Brazil
E-mail: pablosilva@id.uff.br

A. Plastino
Institute of Computing, Fluminense Federal University (UFF), Niterói-RJ, Brazil
E-mail: plastino@ic.uff.br

A. A. Freitas
School of Computing, University of Kent, UK
E-mail: a.a.freitas@kent.ac.uk

1 Introduction

The classification task is one of the most relevant types of supervised learning in the knowledge discovery scenario [8]. A previously trained classification model automatically assigns a class label to an instance, based on the values of its features. In many important real-world problems, each instance in the dataset can be described as a binary feature vector, such that each feature takes either a "positive" or a "negative" value, indicating the presence or the absence of a property, respectively, in the object being classified. It should be noted that in this scenario, intuitively positive values are more informative than negative values in general. After all, a positive feature value has a clear and well-defined meaning, whilst the negative value of a feature represents very vague information, in the sense that it just tell us that the object being classified does not have a certain property, without providing any clue about the object's properties. Therefore, in this work we prioritize the selection of positive feature values over negative feature values, when learning classification models.

More specifically, this work addresses hierarchical feature spaces, where binary features are related via generalization-specialization relationships. In addition, the addressed feature spaces are sparse, i.e., in general the instances contain much fewer positive than negative feature values. In a generalization-specialization hierarchy, also known as "IS-A" hierarchy, for any given instance t, if a feature x has positive value in t, denoted $(x = 1)$, then all ancestors of x in the feature hierarchy also have positive value in t. In contrast, if a feature x has negative value in t, denoted $(x = 0)$, then all descendants of x in the feature hierarchy also have negative value in t.

Some examples of data commonly characterized by hierarchical and sparse feature spaces, where positive feature values are in general more informative than negative values, are text [11] and biological data [29, 31], which are two of the most investigated types of machine learning applications.

For example, in the text classification problem, an article may be characterized by a set of tags describing its content. In this case, one general feature (e.g., News) may be associated with one or more specialized features (e.g., Economy, Politics and Sports). In addition, knowing that a document contains a certain word like Economics (positive feature value) provides us with clear information about the document's contents, whilst knowing that the document does not contain a certain word like Economics (negative feature value) provides us with much less information about the document's contents.

Similarly, in bioinformatics problems where each instance represents a gene, each gene may be associated with terms derived from an ontology of biological processes or functions. Hence, a general feature (e.g., biological process) would be the ancestor of more specific features (e.g., reproduction or biological regulation). In addition, a gene annotation indicating that the gene is involved in say DNA repair (positive feature value) provides us with much more information than a lack of DNA repair annotation (negative feature value) for that gene.

Many important real-world datasets have a large number of features, many of which are not crucial for predicting the correct class. Some features can be redundant (highly correlated with each other) or irrelevant for predicting the class variable, decreasing the classifier's predictive accuracy, making the learning process slower, and reducing the comprehensibility of the results.

Feature selection methods have been successfully employed to cope with these problems. They aim at selecting a reduced subset of features to predict the target class, yet increasing the predictive accuracy of the classifier [17]. Although many methods address this problem [6, 12, 15, 17, 18, 22, 32], only few of them explore the hierarchical information in order to improve their effectiveness [11, 20, 24, 29, 30, 31]. Existing hierarchical feature selection methods usually find a suitable subset of features by keeping those features with higher values of relevance and removing redundancy among hierarchically related features.

In this work, we focus on hierarchical and sparse feature spaces from different domains that share a singular characteristic; a positive feature value is always much more informative than a negative feature value, as briefly discussed earlier and discussed in more detail later. Despite this interesting characteristic of positive feature values, none of the previously proposed feature selection methods for hierarchical and sparse feature spaces has prioritized the selection of positive feature values. Hence, in this work, we hypothesize that the selection of positive feature values tends to increase the predictive accuracy of the classifier, and we propose feature selection methods prioritizing positive feature values.

The main contribution of this work is the proposal of a novel lazy feature selection method for hierarchical and sparse feature spaces which relies on the higher relevance of features with positive values for the classification task. The basic idea of this method is to select, for each test instance, a subset with the most specific positive feature values in the hierarchy as well as its relevant ancestors. To assess the quality of the subset of positive feature values, we introduce a new lazy version of a relevance measure that evaluates the predictive relevance of a feature value for the current test instance.

The main related work involves two other lazy feature selection methods proposed in the literature, the HIP and MR methods [31]. In essence, the proposed feature selection method – called Select Relevant Positive Feature Values (RPV) – differs from HIP and MR in three major ways: First, RPV uses a new relevance measure, proposed in the current work. Second, RPV selects only positive feature values for each instance, whilst the HIP and MR methods select both positive and negative feature values for each instance. Third, comparing RPV vs. HIP, HIP selects only the most specific positive features, whilst RPV selects not only the most specific features but also some of their relevant ancestors; and comparing RPV vs. MR, RPV removes only features which are hierarchically redundant, whilst MR in general removes features that are not hierarchically redundant. These differences are explained in more detail in Section 3 (Related Work), after a description of HIP, MR and other hierarchical feature selection methods.

The proposed feature selection method prioritizing positive feature values is evaluated in 17 datasets. These datasets are mainly from the area of bioinformatics, but they also include other types of application domains, in particular two datasets involving the classification of sports tweets, one dataset involving the classification of news headings, one dataset involving the classification of URLs, and finally one dataset classifying cities into categories of "liveability". The results of experiments with these diverse application domains show that the proposed hierarchical feature selection method outperforms both traditional feature selection methods and recent state-of-the-art hierarchical feature selection methods regarding predictive accuracy, whilst also selecting smaller feature subsets.

The remainder of this paper is organized as follows. Section 2 defines the hierarchical feature selection problem and briefly discusses feature selection methods. Section 3 reviews related work. Section 4 introduces the new relevance measure and the novel lazy restrictive hierarchical feature selection method. Section 5 presents experimental results. Section 6 presents conclusions and research directions.

2 Hierarchical Feature Selection for Classification

The classification problem can be defined as follows. Let $X = \{X_1, \ldots, X_d\}$ be a set of d predictive features and $L = \{l_1, \ldots, l_q\}$ be a set of q class labels, where $q \geq 2$. Let $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ be a dataset with N instances, where x_i corresponds to a vector $(x_{i1}, x_{i2}, \ldots, x_{id})$, for the i-th instance, which stores values for the d features in X and each $y_i \in L$ corresponds to a single target class. The goal of the classification task is to learn a classifier from D that, given an unlabelled instance $t = (x, ?)$, predicts its class label y.

The quality of the feature set has a huge impact on the predictive performance of classification algorithms [17]. Feature selection methods aim at improving the predictive performance of the classifier by selecting a subset containing relevant and non-redundant features. Relevant features are those that are useful for predicting the target class variable, and non-redundant features are those that are not highly correlated with other features.

Feature selection methods can be categorized into embedded, wrapper and filter methods [17]. Embedded methods are incorporated into the classification algorithm, selecting features during the learning of a classification model. Wrapper and filter methods are instead used in a data pre-processing step. Wrapper methods measure the relevance of a feature subset by evaluating the predictive accuracy of a classifier built using that subset. Hence, they select features tailored for the target classification algorithm, but they tend to be very time-consuming. By contrast, filter methods evaluate the predictive power of features in a generic way, by using a relevance measure that is independent of the target algorithm. Filter methods tend to be much faster and more scalable than wrapper methods. We focus on filter methods in this work. For an example of a wrapper approach see [3].

In some scenarios, the i-th instance is defined as a d-dimensional binary feature vector $(x_{i1}, x_{i2}, \ldots, x_{id})$ with $x_{ij} \in \{0, 1\}$ for all $1 \leq j \leq d$. When the feature set X is hierarchically structured, we call it a hierarchical feature space, which can be represented as a Direct Acyclic Graph (DAG). In this DAG a vertex (node) represents a feature and an edge represents a generalization-specialization relationship between features. In this sense, an edge $(X_a \rightarrow X_b)$ indicates that X_a is a parent of X_b and X_b is a child of X_a. More generally, a feature X_a is an ancestor (descendant) of a different feature X_b if and only if there is a sequence of edges leading from X_a to X_b (from X_b to X_a) in the feature DAG – or feature tree, in some scenarios. The root node is the most general feature, while the leaf nodes are the most specific ones. Note that this structure produces a hierarchical redundancy among features, since a specific feature value logically implies the values of all its ancestors or descendants: all ancestors of positive-valued features have positive values and all descendants of negative-valued features have negative values.

For example, to classify an instance where the feature set is formed of Gene Ontology (GO) terms, if the instance is annotated with the GO term “multicel-

lular organism reproduction", then that instance is considered annotated with the more general GO terms "reproduction" and "multicellular organism process". Conversely, if an instance is not annotated with the GO term "reproduction" (i.e., the feature "reproduction" has a negative value), then the instance is considered not to be annotated with the GO term "multicellular organism reproduction" (i.e., the child feature is guaranteed to have a negative value too, in that instance).

Hierarchical feature selection methods exploit characteristics of the feature DAG to improve the predictive accuracy. This is typically done by removing hierarchically redundant features [24, 31]. Note that, even though we are dealing with hierarchical features, the information about the hierarchical structure (represented by a feature DAG) is only used by the hierarchical feature selection methods. I.e., the hierarchical structure is used to enhance the feature selection process, helping to identify a better set of features to be selected. After the feature selection step, the data is treated as a "flat" dataset (the hierarchical structure is not considered anymore), then we can use traditional classification methods to make predictions.

Feature selection methods (as well as classification methods) are categorized as eager or lazy. Eager methods select a subset of features based on the training instances. Then, a model trained with the selected features is used to predict the class of any test instance. By contrast, lazy methods select a feature subset tailored for each test instance [1, 22], by observing the feature values (but not the class, of course) in that test instance. In this work, the main motivation to adopt the lazy learning approach is the ability to select a set of relevant positive feature values specifically tailored for each testing instance.

Based on these definitions, our proposed hierarchical method can be categorized as a filter feature selection method which follows the lazy learning paradigm.

3 Related Work

Traditional (non-hierarchical) feature selection methods, like the well-known eager Correlation-based Feature selection (CFS) [6] and ReliefF [12] methods, can be used in hierarchical feature spaces by ignoring the hierarchical relationships among features. However, this is intuitively a sub-optimal approach. Hence, a few methods that directly exploit such hierarchical relationships to improve performance have been recently proposed, as follows.

SHSEL [24] is a hierarchical feature selection method that performs eager learning. SHSEL assumes that, if two features are directly hierarchically related (one is a parent of the other), they are usually highly correlated and tend to be similarly relevant for classification. Hence, for each pair of directly hierarchically related features, SHSEL removes the most specific feature if the correlation between them is higher than a user-defined threshold. Then, using only the remaining features, it keeps for each path in the hierarchy the features whose relevance is higher than the average relevance of features in that path. Moreover, Lu et al. proposed the Greedy Top-Down (GTD) search strategy [20], which selects the most relevant features in each path from each leaf to the root node in the hierarchy. Likewise, an eager learning hierarchical method called Tree-Based Feature Selection (TSEL) [11] has been used in the special case of tree-structured features. Previous work showed that SHSEL achieves better performance than TSEL and GTD [24].

Some hierarchical methods proposed in the literature are based on the lazy learning paradigm, such as the Select Hierarchical Information-Preserving Features (HIP) method [31], the Select Most Relevant Features (MR) method [31], and the hybrid Select Hierarchical Information-Preserving and Most Relevant Features (HIP-MR) method [31]. Since the hybrid HIP-MR obtained worse results than its base methods HIP and MR in [31], it is no longer considered. Next, we briefly describe HIP and MR.

The HIP method eliminates hierarchical redundancy by selecting only the "core" features in the current test instance – i.e., features whose values are non-redundant since they cannot be inferred from the values of other features. In other words, HIP selects the subset of the most specific positive-valued features (which imply their ancestors) and the most general negative-valued features (which imply their descendants). The values of the features selected by HIP for an instance imply the values of all other features for that instance, so it ensures that hierarchical redundancy is completely eliminated. However, HIP does not take into account the relevance of the selected features.

In a similar vein, the MR method not only eliminates hierarchical redundancy but also selects features with higher relevance. For each feature in the DAG, MR considers all paths between the feature and the root (for positive feature values) or between the feature and the leaves (for negative values). Then, the most relevant feature in each path is kept. However, unlike HIP, in general MR does not select all "core" features, i.e., it removes some hierarchically non-redundant features.

The proposed RPV method (described in Section 4) shares with HIP a certain focus on more specific positive feature values, but there are three important differences between these methods. First, HIP selects both positive and negative feature values, whereas RPV only selects positive feature values. Second, among positive feature values, HIP selects only the most specific ones; whilst RPV selects not only the most specific feature values, but also some of their relevant ancestors in the feature hierarchy. Third, RPV uses a new measure of feature value relevance (introduced in this paper), whilst HIP does not use any such relevance measure.

In this work, we compare our proposed method against the state-of-the-art hierarchical feature selection methods HIP, MR and SHSEL, as well as against the traditional (non-hierarchical) feature selection methods CFS and ReliefF.

Note there are also other types of hierarchical feature selection methods, often discussed in the literature under the name of structured feature selection, as reviewed in [5, 19]. However, in general those methods have been proposed for the regression task (using a variation of the Lasso method that produces a sparse linear model), rather than for the classification task addressed in this paper.

It is important to highlight that the hierarchical feature selection task addressed in this paper should not be confused with the kind of hierarchical feature learning performed in deep learning processes. Deep neural networks involve hierarchical feature construction, where, during the training of the neural net, features are hierarchically learnt across the layers of the network [23]. On the other hand, in the problem discussed in this work, the hierarchy of features is predefined, and it is provided as an input to the feature selection algorithm. The point is not to learn or construct new features; the point is to select the best possible subset of features, among the original feature set, exploiting generalization-specialization information associated with the predefined feature hierarchy.

4 The Proposed Hierarchical Feature Selection Method

This section presents our new relevance measure and the new feature selection method for hierarchical and sparse feature spaces.

4.1 Lazy Feature Relevance Measure

In general, how to assess the relevance (or predictive power) of a feature plays an important role in the design of a good feature selection method. Many different functions have been proposed to cope with this issue, such as the Information Gain [2], the Mutual Information [28], the R measure [26], etc.

The R measure, first proposed by [26], was adjusted by [31] to assess the predictive power of features in hierarchical feature selection. As shown in Equation 1, where k is the number of classes, the R measure calculates the relevance of a binary feature X based on the differences between the conditional probabilities of each class c_i given feature values x_1 and x_2.

$$R(X) = \sum_{i=1}^{k} [P(c_i|x_1) - P(c_i|x_2)]^2 \tag{1}$$

Note that Equation 1 is an eager relevance measure, but features may be useful or not depending on the feature values of the test instance being currently classified [22]. Our proposed feature selection method considers that taking into account the feature values (specifically positive values) of the current test instance may contribute to identifying a subset of high-quality features for that particular instance, in the spirit of lazy learning. For this reason, we propose a new feature relevance measure, named Lazy Relevance Measure (LazyR), which assesses the predictive power of a given feature X taking a specific value x in the current test instance. Defined in Equation 2, LazyR calculates the relevance of X with value x as a function of the sum of differences in the conditional probabilities of each class (c_i) given the specific feature value x and the class probability $\frac{1}{k}$ associated with a uniform distribution – ignoring the other values of X, since they do not occur in the current test instance. This measure has the highest value when the feature value x is perfectly correlated with one of the k classes, and presents the lowest value when the conditional probability of each class c_i is exactly $\frac{1}{k}$.

$$LazyR(X = x) = \sum_{i=1}^{k} \left[P(c_i|x) - \frac{1}{k} \right]^2 \tag{2}$$

The LazyR measure has some benefits over eager measures. Eager relevance measures (e.g., R and Information Gain) assess the relevance of all values of a feature to discriminate among class labels. In contrast, LazyR assesses the relevance of a specific feature value. Consider, e.g., a feature A with positive and negative values, where the positive value discriminates well among class labels and the negative value does not. An eager relevance measure could assign a low score to feature A, leading to its removal. In contrast, our lazy relevance measure would keep A in the model if the instance being classified has a positive value for A, and remove it if the instance has a negative value, a principled data-driven decision.

4.2 The Proposed Lazy and Restrictive Hierarchical Feature Selection Method

We designed a new feature selection method for hierarchical and sparse feature spaces called Select Relevant Positive Feature Values (RPV). The intuition for this method is twofold. First, in sparse feature spaces, positive feature values are more informative and easier to interpret than negative values. That is, since positive feature values are quite rare, they provide more relevant and more meaningful information than negative values. For instance, in text mining, typically a document is described by features representing the presence (positive value) or absence (negative value) of words in that document, and the class indicates a document's subject. The presence of the word "teacher" is relevant for predicting that the document's class is "Education", but the absence of the word "teacher" is not relevant for classification nor meaningful, it is too broad information. Second, the generalization-specialization structure of hierarchical feature spaces creates hierarchical redundancy among features, which intuitively reduces predictive accuracy. RPV exploits generalization-specialization relationships in order to eliminate hierarchical redundancy, which should improve predictive accuracy.

More specifically, our method adopts the following ideas: (i) it relies on a restrictive selection approach, where selecting only positive feature values might increase the accuracy of the classifier; (ii) it tries to identify a specific subset of relevant positive features for each instance t in the test set – using the lazy paradigm; (iii) taking into account the hierarchy, it selects the most specialized positive feature values as well as those positive feature values whose relevance value is higher than (or equal to) the relevance of all its positive descendants.

We now show, theoretically, that Naïve Bayes – a classifier used in related work [24, 29, 31] and also employed in our experiments – tends to give larger influence to positive feature values than to negative feature values in sparse datasets, which is in agreement with the ideas behind the proposed feature selection method.

Consider the log-odds ratio form of Naïve Bayes (for binary classes c_1 and c_2):

$$ln\frac{P(c_1|X)}{P(c_2|X)} = ln\frac{P(c_1)}{P(c_2)} + \sum_{i=1}^{d} ln\frac{P(x_i|c_1)}{P(x_i|c_2)}, \tag{3}$$

which predicts class c_1 for the current instance if $ln\frac{P(c_1|X)}{P(c_2|X)} > 0$, and predicts class c_2 otherwise. The summation term of this formula can be divided into two parts: $\sum_{i+=1}^{d+} ln\frac{P(x_{i+}|c_1)}{P(x_{i+}|c_2)}$ and $\sum_{i-=1}^{d-} ln\frac{P(x_{i-}|c_1)}{P(x_{i-}|c_2)}$, where $i+$ and $i-$ index the set of positive and negative feature values in the current instance, respectively; $d+$ and $d-$ are the number of positive and negative feature values in the current instance, respectively; and $d- + d+ = d$. In the case of very sparse features, each term in the second summation (over the $d-$ negative feature values) will tend to zero. This is because, since the vast majority of instances take the negative value for a highly sparse feature, both the terms $P(x_{i-}|c_1)$ and $P(x_{i-}|c_2)$ will tend to have similar values (both will tend to be close to 1), and therefore each term $ln\frac{P(x_{i-}|c_1)}{P(x_{i-}|c_2)}$ will tend to be close to zero. I.e., negative feature values will have little influence in the Naïve Bayes formula. On the other hand, for positive feature values, the terms $P(x_{i+}|c_1)$ and $P(x_{i+}|c_2)$ will have quite different values in general, and so the summation of the terms $ln\frac{P(x_{i+}|c_1)}{P(x_{i+}|c_2)}$ over the $d+$ positive feature values will tend

to be a large number, rather than close to zero. I.e., positive feature values tend to have a larger influence than negative feature values in the Naïve Bayes formula.

The RPV method works as follows. Given a test instance t, first, it evaluates the relevance of each feature in t. Then, it identifies the list of ancestors for each positive feature value, using the feature DAG. After that, RPV marks every negative feature value in t for removal. For each positive feature X_i in t, RPV evaluates each of its ancestors and marks for removal those whose relevance is lower than the relevance of X_i. At the end of the process, RPV removes every feature marked for removal, and the remaining features are used in the lazy classification of the current test instance.

Algorithm 1 describes how RPV works in detail. This algorithm produces as output a subset of features named $SelectedFeatSubSet$. In the initialization phase (lines 1 to 5), the ancestors and the relevance value (measured by LazyR) for each feature in the DAG are computed and stored into the respective $Ancestors$ and $Relevance$ arrays (indexed by the features' ids). Also, the $Status$ array is initialized with the "Selected" value for all features.

Algorithm 1 Select Relevant Positive Feature Values (RPV)

Input : D (training dataset), t (test instance) and DAG (feature hierarchy)
Output: a subset of features $SelectedFeatSubSet$

```
 1: for each feature X_i in DAG do
 2:    Ancestors[X_i] ← list of ancestors of X_i in the DAG
 3:    Relevance[X_i] ← LazyR(X_i = positive) {computed using the training set D}
 4:    Status[X_i] ← "Selected"
 5: end for
 6: for each feature X_i in DAG do
 7:    if Value(X_i, t) is positive then
 8:       for each feature A_j ∈ Ancestors[X_i] do
 9:          if Relevance[A_j] < Relevance[X_i] then
10:             Status[A_j] ← "Removed"
11:          end if
12:       end for
13:    else
14:       Status[X_i] ← "Removed" {since Value(X_i, t) is negative}
15:    end if
16: end for
17: SelectedFeatSubSet ← features with Status set to "Selected"
18: return SelectedFeatSubSet
```

The main phase of RPV works as follows. In line 7, for each feature X_i in DAG, the function $Value(X_i, t)$ returns the value of X_i in the test instance t. If the returned value is positive, RPV looks at each ancestor A_j of X_i in the DAG and marks for removal (setting the $Status$ flag) those with relevance value lower than the relevance of X_i (lines 8 to 12). In line 14, every feature with negative value in t is marked for removal, since negative values are much less informative than positive values, as discussed earlier. In lines 17 and 18, the feature subset $SelectedFeatSubSet$ receives all features whose $Status$ is still "Selected" and this subset is returned by the algorithm. Then, a lazy classifier is executed for test instance t using only the selected features. Note that, after initializing each feature's $Status$ with "Selected", the $Status$ of a feature can only be changed to "Removed" in lines 10 and 14, and once this change is made, that feature's $Status$

is never set back to "Selected" by the algorithm. Hence, the result of the algorithm does not depend on the order in which the features are processed.

The RPV algorithm is executed for each test instance in a lazy learning fashion, but note that, in order to save time, the values of the *Ancestors* and *Relevance* arrays can be pre-computed in an eager fashion and stored to be accessed whenever a new instance needs to be classified.

Figure 1 illustrates how RPV works. In this figure, each vertex represents a feature, and the numbers on the right and left side of each node represent, respectively, the feature's value (1 for positive, 0 for negative) and the relevance of that feature value. After RPV's initialization phase, each feature in the DAG (denoted by letters A to N) is processed in turn. When A, B, D, E, F and I are processed, their *Status* will be set to "Removed", since their values are "0". When C (with value "1") is processed, RPV sets to "Removed" the *Status* of C's ancestors in the DAG whose relevance (LazyR) value is lower than C's relevance – i.e., G and N are marked for removal. When H is processed, L and N (ancestors with lower relevance than H) are marked for removal, and M will also be marked for removal when J is processed. After processing all features, the only ones selected (never marked for removal) are features C, K, H and J.

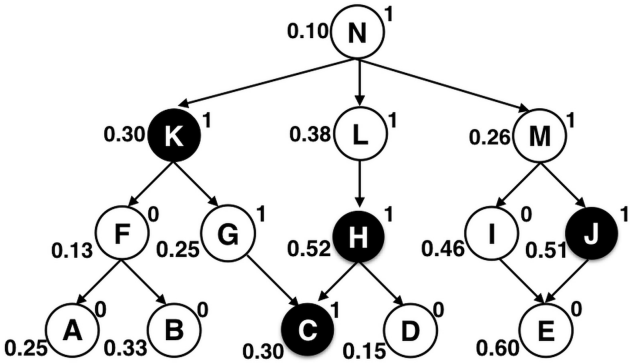


Fig. 1: Example of a feature DAG showing the subset of features selected by RPV.

The reader might be questioning whether the rule "if a given feature A is an ancestor of a feature B then $LazyR(A) > LazyR(B)$" is always true. We demonstrate that it is not always true by using the counter-example presented in Figure 2. Considering that feature A is an ancestor of feature B (the value $B = 1$ in an instance implies the value $A = 1$ in that instance), then we will show that in this case $LazyR(B) > LazyR(A)$. Note that we evaluate the relevance of the value of the feature and how well it is correlated with the class variable. So, LazyR is higher when the feature values that appear in the instance to be classified can describe the class well. In this specific case, the features are binary and then the possible feature values are 0 or 1. We evaluate the relevance of each one of these two feature values for each feature in the datasets. According to equation 2, we have that $k = 2$, so $1/k = 0.5$, then $LazyR(B = 1) = (1 - 0.5)^2 + (0 - 0.5)^2 = 0.5$ and $LazyR(A = 1) = (1 - 0.5)^2 + (0.5 - 0.5)^2 = 0.25$, i.e., $LazyR(B = 1) > LazyR(A = 1)$. In a second example, in Figure 2, consider that C is an ancestor

of D (the value $D = 1$ in an instance implies the value $C = 1$ in that instance). In this case, $LazyR(C = 1) = (0 - 0.5)^2 + (1 - 0.5)^2 = 0.5$ and $LazyR(D = 1) = (0-0.5)^2+(0.5-0.5)^2 = 0.25$, so $LazyR(C = 1) > LazyR(D = 1)$, since C is much more discriminative to the class variable than D. These values demonstrate that the value of LazyR is not linked with the position of the feature in the hierarchy, it is related with the discriminative power of the feature value. The position of the feature in the hierarchy is much more related to the redundancy among feature values than to their relevance.

	A	B	C	D	Class
Instance 1	1	1	0	0	C1
Instance 2	1	0	1	0	C2
Instance 3	0	0	1	1	C2

Fig. 2: Example of a dataset with two features and three instances where A is an ancestor of B and $LazyR(B = 1) > LazyR(A = 1)$.

Figure 3 depicts the end-to-end process of employing the RPV feature selection method in the pre-processing stage of the classification procedure. First of all, since the RPV method is a lazy filter feature selection method, it uses the feature hierarchy and the training dataset to automatically and intelligently select a subset of features for posterior use in the classification of a given test instance t. Note that this feature selection process follows the lazy paradigm, i.e., the filter procedure is tailored to each instance that passes through the RPV method. After the feature selection procedure is executed, a lazy classifier (such as the Naïve Bayes or the Nearest Neighbor classifier) is used to predict the class of the instance t using only the subset of the original features selected by the RPV method.

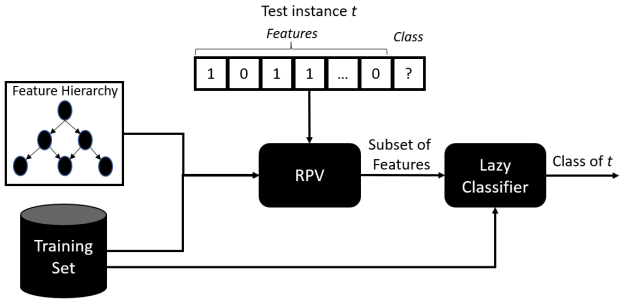


Fig. 3: Diagram illustrating the inputs and output of the RPV method for a given test instance, and also showing that the selected feature subset (RPV's output) is used by a lazy classifier to classify that instance.

The RPV method presents some appealing characteristics: (i) it selects only positive feature values, which are more informative than negative values; (ii) it

uses a lazy relevance measure specifically adapted to assess the relevance of a feature value in the current test instance; (iii) since it selects only positive values, it tends to select fewer features than the other methods used in our experiments (as shown later); (iv) it selects the most specific positive feature values and some of their most relevant ancestors.

4.3 An analysis of the worst-case time complexity of RPV

The worst-case time complexity of RPV can be calculated as follows. First of all, note that, in the worst case scenario, the feature DAG has N nodes (features) and $\frac{N \cdot (N-1)}{2}$ edges. I.e., each feature is linked to all features but its own descendants. Hence, line 2 of Algorithm 1 has worst-case time complexity $O(N)$, but this line is executed N times since it is within a *for* loop, which takes $O(N^2)$. Line 3 of Algorithm 1 requires the value of the LazyR (Lazy Relevance) measure for each feature, which is precomputed before Algorithm 1 is run, with a time complexity of $O(N \cdot M)$, where N is the number of features and M is the number of training instances. Lines 3 and 4 take constant time (simple assignment), and so they can be ignored in the analysis, since their time complexity is dominated by the one of line 2. During RPV's feature elimination procedure, performed by the nested loop starting in line 6, in the worst case, in line 9 the relevance value of the most specific feature is compared to the relevance of $N - 1$ ancestors, the second most specific feature is compared to $N - 2$ ancestors and so on, until all features have been evaluated. The other lines in the nested loop, lines 10 and 14, do not change the time complexity associated with this loop. In total, in the worst case, the nested loop starting at line 6 of Algorithm 1 performs $\frac{N \cdot (N-1)}{2}$ comparisons, i.e., $O(N^2)$.

Hence, in addition to the time $O(N \cdot M)$ to pre-compute all LazyR values, Algorithm 1 takes $O(N^2) + O(N^2)$, which in total is $O(N \cdot M + N^2)$. Note that Algorithm 1 is executed once for each test instance to be classified. Hence, the total worst-case time complexity of the RPV feature selection method is $O(N \cdot M + t \cdot N^2)$, where N is the number of features, M is the number of training instances, and t is the number of test instances to be classified.

Note, however, that in practice the time taken by RPV tends to be much smaller than suggested by this worst-case analysis, because in practice the number of ancestors of each feature is usually much smaller than the theoretical maximum of $N - 1$ (a key assumption in the above analysis).

5 Computational Experiments

5.1 Datasets

In this work, the proposed method was evaluated on 17 distinct datasets, 12 from the bioinformatics domain and 5 from other classification domains.

Following the same methodology described in [29, 31], we created 12 datasets of ageing-related genes, involving the effect of genes on an organism's longevity. These datasets were created by integrating data from the Human Ageing Genomic Resources (HAGR) GenAge database (version: Build 17) [21] and the Gene Ontology (GO) database (version: 2015-10-10) [27]. HAGR is a database of ageing-

and longevity-associated genes in four model organisms: *C. elegans* (worm), *D. melanogaster* (fly), *M. musculus* (mouse) and *S. cerevisiae* (yeast). The GO database provides information about three ontology types: biological process (BP), molecular function (MF) and cellular component (CC). Each ontology contains a separate set of GO terms (features), i.e., a distinct feature hierarchy (a DAG). For each of the 4 model organisms, we created 3 datasets, one for each feature type (feature hierarchy), denoted by BP, CC and MF. Hence, each dataset contains instances (genes) from a single model organism. Each instance is formed of a set of binary features indicating whether or not the gene is annotated with each GO term in the GO hierarchy and a binary class variable indicating if the instance is either positive ("pro-longevity" gene) or negative ("anti-longevity" gene) according to the HAGR database. That is, the class variable indicates whether a gene has the effect of extending or reducing the lifespan of an organism, which is some important information for biologists trying to understand the process of ageing. In order to avoid overfitting, GO terms which occurred in less than three genes were discarded.

Note that GO terms are a particularly suitable type of predictive feature in the context of our experiments, because they have a clear hierarchical structure, they are available in different types of biological hierarchies (involving biological processes, molecular functions and cellular localization information), and they have also been used in benchmark datasets used in previous research in this area, as mentioned earlier. In addition, GO terms are popular in bioinformatics, because they allow biologists to describe biological properties of genes in a standardized way, as well as allowing the description of gene properties at different levels of abstraction. Hence, genes whose properties are known in detail can be annotated with very specific GO terms (around the bottom of the GO hierarchy), whilst genes whose properties are mainly unknown can be annotated with very high level GO terms (around the top of the GO hierarchy). This makes GO terms a flexible approach for representing biological knowledge.

The remaining 5 datasets, previously used in a related work [24], represent different classification tasks with features and hierarchies extracted from either the Open Directory Project[1] or DBpedia [14]. These datasets are described below.

- Tweets T and Tweets C: in these datasets, the task is to identify sports-related tweets, where each tweet can be either related to sports (positive class) or not related to sports (negative class). The hierarchy and features were generated by extracting types (in Tweets T) and categories (in Tweets C) from DBpedia.
- NY Daily: this dataset is a set of news headings augmented with DBpedia's types, where the classification task is to identify a sentiment variable (positive/negative).
- Stumbleupon (Stb.upon): it is a user-curated web content discovery engine that recommends relevant, high quality pages and media to its users, based on their interests. The aim is to classify webpages into the "ephemeral" class if they are visited on specific periods of time or into the "evergreen" class if they are visited for a long period of time.
- Cities: this dataset was generated from a list of the most and the least liveable cities according to the Mercer survey augmented with DBpedia types. The task is to classify each city into low, medium and high liveability, where liveability

[1] http://www.dmoz.com

is a variable that combines many factors (such as political stability, medical supplies and services, censorship, among others) to measure to what extent it is desirable to live in a city.

Information about the datasets is shown in Table 1. For each of the four model organisms, each of the three rows shows information about a specific dataset. The first column identifies the group of datasets for each of the four organism or the general-domain datasets. The second column shows the feature hierarchies used to build the bioinformatics datasets or, in the last five rows, the names of the datasets from general (non-bioinformatics) domains. The other columns show, respectively, the number of features (#features), the number of edges in the feature DAGs (#edges), the number of instances (#instances), the percentage of positive-class instances (% Pos), the percentage of negative-class instances (% Neg), and the percentage of positive feature values (% Pos feat values). In the last row, columns 6 and 7 show the class distribution (low, medium and high) for the dataset Cities.

Table 1: Detailed information about the datasets used in the experiments.

Group	Dataset	#features	#edges	#instances	% Pos	% Neg	%Pos feat values
CE	BP	991	1707	657	34.40	65.60	4.50
	CC	178	277	484	36.36	63.64	6.49
	MF	263	331	504	37.70	62.30	5.07
DM	BP	800	1355	132	71.97	28.03	8.32
	CC	89	130	122	70.49	29.51	12.02
	MF	146	182	126	70.63	29.37	7.72
MM	BP	1333	2406	109	68.81	31.78	10.65
	CC	143	214	107	68.22	31.78	16.41
	MF	240	289	106	67.92	32.08	9.73
SC	BP	844	1511	331	13.29	86.71	5.35
	CC	145	230	331	13.29	86.71	9.04
	MF	221	277	331	13.29	86.71	5.73
General	Tweets T	4082	36019	1179	55.64	44.36	1.14
	Tweets C	10883	15189	1179	55.64	44.36	1.02
	NY Daily	5145	44152	1016	57.09	42.91	1.21
	Stb.upon	3976	12354	3020	45.36	54.64	1.17
	Cities	727	7051	212	18.40/50.00/31.60		3.31

5.2 Experimental Methodology

We implemented our RPV method and other methods used in this work within the open-source WEKA data mining tool [7]. The datasets used in the experiments and the program code of the RPV method will be freely available on the web after the publication of the paper. The methods were evaluated on the 17 datasets described earlier. The lazy k-NN with Euclidean distance (with $k = 1$) and a lazy version of Naïve Bayes (NB) (both from WEKA) were used as classification algorithms for all evaluated feature selection methods, and the predictive accuracy was measured by 10-fold cross-validation.

It is worth mentioning that Naïve Bayes has also been used in previous work on hierarchical feature selection [20, 24, 30, 31], as well as k-NN [30, 31]. In addition, in some preliminary experiments with the datasets used in this work and without employing any feature selection method, Naïve Bayes achieved the best predictive accuracy, followed by 1-NN, when compared with other traditional classification algorithms, namely SVM (with various types of kernel), Random Forest and Decision Trees (C4.5). The results of these preliminary experiments are reported in Appendix B.

As shown in Table 1, the majority of the datasets have imbalanced class distributions, so we evaluated the methods' predictive accuracy by using the Geometric Mean (GM) of sensitivity and specificity, the Area Under the Precision-Recall Curve (AUCPR), and the Area Under the Receiver Operating Characteristics curve (AUROC) measures [8, 10]. The GM is defined in Equation 4, which was also used in [29, 30, 31].

$$GM = \sqrt{Sensitivity * Specificity} \tag{4}$$

GM takes into account the balance between the sensitivity and specificity of the classifier. Sensitivity (or true positive rate) is the proportion of positive class instances correctly predicted as positive, whereas specificity (or true negative rate) is the proportion of negative class instances correctly predicted as negative. The AUCPR plots the precision of the classifier as a function of its recall, then the area under this curve is used to evaluate the classifier (the higher the better). The ROC curve is created by plotting the true positive rate (TPR) – also know as Sensitivity – against the false positive rate (FPR), and again, the higher the area under this curve, the better the performance of the classifier.

To determine whether the differences in performance are statistically significant, we ran the Friedman test and the Holm post-hoc test [9], as recommended by Demsar [4]. First, the Friedman test was executed with the null hypothesis that the performances of all methods are equivalent. The alternative hypothesis is that there is a difference between the results of all methods as a whole, without identifying specific pairs of methods with significantly different results. If the null hypothesis is rejected, we run the Holm post-hoc test (which corrects for multiple hypothesis testing) to compare the results of the proposed RPV method (with the LazyR measure) against each of the other methods. Both the Friedman and Holm test were used at the 0.05 significance level in all our experiments.

5.3 Results

Tables 2 through 9 report the predictive accuracy results for three experiments: the first one (Tables 2, 3, and 4) compares our proposed RPV method to baseline approaches; the second one (Tables 5 and 6) compares RPV against the well-known traditional (non-hierarchical) feature selection methods CFS and ReliefF; and the third one (Tables 7, 8 and 9) compares RPV against state-of-the-art hierarchical features selection methods. These results are discussed in the following three subsections.

Table 2: Comparing RPV with different feature relevance measures against baseline methods in terms of AUCPR – in %.

		Naïve Bayes						1-NN					
		Baseline			RPV			Baseline			RPV		
	Datasets	No FS	All-Neg	All-Pos	IG	R	LazyR	No FS	All-Neg	All-Pos	IG	R	LazyR
CE	BP	55.1	53.1	54.6	55.7	55.3	**56.4**	48.0	41.8	48.3	49.4	47.4	**50.4**
	CC	56.3	**56.5**	55.1	53.1	54.5	54.3	49.4	53.5	54.5	53.2	53.9	**54.6**
	MF	50.2	47.8	50.5	50.6	51.4	**51.7**	47.6	49.9	**51.7**	**51.7**	50.5	50.1
DM	BP	83.1	80.2	83.4	**83.5**	82.2	82.5	78.2	76.2	80.0	**80.7**	80.2	79.3
	CC	87.6	84.0	87.8	88.5	89.8	**90.0**	79.6	81.9	85.2	83.5	**84.0**	83.4
	MF	81.9	79.7	81.9	82.0	**82.8**	81.1	79.3	80.0	81.9	82.1	82.2	**82.4**
MM	BP	82.5	82.1	84.4	**85.5**	85.1	85.2	77.1	75.9	76.0	**78.2**	77.7	77.6
	CC	84.5	82.2	86.0	**86.5**	85.9	84.4	74.1	69.6	75.1	77.8	**78.9**	76.6
	MF	**87.1**	83.9	86.5	85.8	86.3	85.6	77.6	74.1	**81.8**	78.7	79.2	79.1
SC	BP	45.6	43.2	40.2	42.8	38.3	**46.4**	29.2	25.9	32.8	**39.5**	35.5	36.4
	CC	34.0	32.5	33.9	34.6	31.6	**35.3**	30.6	31.7	34.5	36.9	37.3	**37.6**
	MF	26.8	20.4	**27.0**	20.8	20.7	25.2	28.9	26.3	32.0	**36.0**	34.3	35.8
General	Tweets T	81.6	71.1	82.2	82.7	82.8	**83.3**	76.5	58.8	81.0	81.6	82.4	**82.6**
	Tweets C	98.3	95.4	98.3	98.4	**98.5**	**98.5**	94.5	90.2	96.6	96.9	97.0	**97.5**
	NY Daily	64.1	60.7	64.1	64.1	63.5	**64.8**	**60.4**	57.9	59.5	58.9	58.8	60.1
	Stb.upon	77.8	75.7	78.0	77.9	77.7	**78.7**	71.6	**74.8**	74.1	74.1	74.2	74.4
	Cities	70.7	64.8	71.7	71.2	70.9	**74.1**	60.2	61.2	**69.5**	69.0	69.2	69.4
	Avg. Rank	3.6	5.5	3.2	2.8	3.6	**2.3**	5.1	5.3	3.1	2.7	2.6	**2.2**
	#Win	1.0	1.0	1.0	3.0	1.5	**9.5**	1.0	1.0	3.5	4.5	1.0	**6.0**

Naïve Bayes: {RPV-LazyR} $\succ$ {No FS, All-Neg, RPV-R}
1-NN: {RPV-LazyR} $\succ$ {No FS and All-Neg}

5.3.1 Comparison against baseline feature selection approaches

This first experiment aims to evaluate the effectiveness of the two main characteristics of the proposed RPV method, i.e., its focus on selecting only a subset of positive feature values and its new lazy relevance measure.

Tables 2, 3 and 4 report the AUCPR, AUROC and GM results, respectively. In these Tables, columns 3 to 8 show the results of six methods using Naïve Bayes and the last six columns show the results of the same six methods using 1-NN. The 6 methods are our RPV method using the LazyR relevance measure, two RPV versions with different relevance measures and three baseline methods. The first baseline is the base classifier using no feature selection method (No FS). I.e., it uses the full set of predictive features. We also implemented two baseline lazy non-hierarchical feature selection methods: one that selects all features with positive value in the current test instance (All-Pos); and another one that selects all features with negative values in the current test instance (All-Neg). Moreover, in order to evaluate the benefit of our proposed feature relevance measure (LazyR), we compare our RPV (with the LazyR measure) against two other RPV versions. The first version uses the original eager relevance measure R (RPV-R), defined in Equation 1, and the second version uses the traditional Information Gain measure (RPV-IG).

Table 3: Comparing RPV with different feature relevance measures against baseline methods in terms of AUROC – in %.

		Naïve Bayes						1-NN					
		Baseline			RPV			Baseline			RPV		
	Datasets	No FS	All-Neg	All-Pos	IG	R	LazyR	No FS	All-Neg	All-Pos	IG	R	LazyR
CE	BP	**69.9**	66.6	69.7	69.7	69.8	69.8	63.4	58.3	62.7	63.3	63.2	**64.4**
	CC	**70.4**	66.8	69.2	66.0	67.2	67.4	64.2	65.4	**68.9**	66.9	68.3	68.2
	MF	62.8	58.4	63.2	63.1	62.8	**64.0**	62.2	64.4	64.2	**64.7**	62.9	63.6
DM	BP	63.2	59.6	**63.4**	62.9	61.1	61.0	62.0	52.5	**64.4**	63.7	61.4	62.2
	CC	76.3	69.0	76.6	78.2	**79.0**	78.7	66.4	66.6	**69.6**	68.1	69.2	67.8
	MF	64.8	59.7	**64.8**	64.6	64.6	63.0	59.9	59.2	65.6	66.2	66.0	**66.5**
MM	BP	69.7	65.7	71.1	**74.8**	74.2	74.2	65.3	59.8	62.8	**66.4**	64.8	64.5
	CC	69.4	65.6	72.4	**73.3**	71.6	70.9	55.7	45.1	56.4	57.5	**59.2**	56.8
	MF	**72.2**	64.8	71.3	71.0	71.7	70.9	61.9	51.4	**69.7**	64.6	65.2	65.0
SC	BP	74.7	73.6	**75.2**	72.1	72.7	74.8	69.7	60.5	67.1	**72.1**	68.1	69.4
	CC	66.6	65.2	66.8	66.3	64.5	**67.0**	63.8	65.4	64.3	66.1	**70.5**	64.8
	MF	**61.4**	53.3	**61.4**	57.4	57.0	60.5	67.0	62.5	67.8	69.6	66.7	**70.0**
General	Tweets T	80.6	73.0	81.1	82.7	**83.5**	83.2	70.2	70.4	**73.6**	72.0	73.1	71.7
	Tweets C	73.1	72.0	**73.6**	70.6	71.1	73.1	68.1	59.2	65.6	**70.5**	66.7	67.9
	NY Daily	55.1	49.4	52.5	52.0	52.1	**56.1**	46.2	48.8	46.8	47.7	49.1	**52.8**
	Stb.upon	78.1	74.1	**78.2**	**78.2**	78.1	78.1	76.6	**78.9**	78.2	78.1	78.0	78.1
	Cities	66.7	58.2	68.9	69.6	**69.8**	69.7	60.9	53.6	**67.6**	67.2	67.2	67.3
	Avg. Rank	3.1	5.6	**2.4**	3.6	3.4	2.8	4.6	4.9	3.1	**2.4**	3.2	2.8
	#Win	3.5	0.0	**5.0**	2.5	3.0	3.0	0.0	1.0	**6.0**	4.0	2.0	4.0

Naïve Bayes: {RPV-LazyR} $\succ$ {All-Neg}
1-NN: {RPV-LazyR} $\succ$ {No FS and All-Neg}

The last two rows of Tables 2, 3 and 4 show, for each method, its average rank (Avg. Rank) and its number of wins (#Win). The lower the Avg. Rank, the better (higher) the AUCPR, AUROC or GM value. Note that the Avg. Rank and #Win values for the 6 methods are computed separately for each of the two classifiers (NB and 1-NN). For each classifier, the highest AUCPR, AUROC or GM value for each dataset is highlighted in bold type. In the row right below Tables 2, 3 and 4, the symbol $\succ$ represents a statistically significant difference between one or more methods, such that $\{a\} \succ \{b, c\}$ means that a is significantly better than b and c.

Considering the results for the AUCPR measure in Table 2, RPV-LazyR obtained the best #Win and the best Avg. Rank values for both Naïve Bayes and 1-NN. For these two algorithms, the Holm post-hoc test indicated that RPV-LazyR is significantly better than No FS and All-Neg. Additionally, for NB, RPV-LazyR is significantly better than RPV-R.

The AUROC results in Table 3 show that All-Pos obtained the best #Win for both NB and 1-NN. For NB, the best Avg. Rank was obtained by All-Pos, while RPV-IG obtained the best result for 1-NN. For both classifiers, RPV-LazyR obtained the second best Avg. Rank among all other baselines. The Holm post-hoc test indicated that RPV-Lazy is significantly better than All-Neg for both classifiers and significantly better than No FS for 1-NN. Note that, for both classi-

Table 4: Comparing RPV with different feature relevance measures against baseline methods in terms of GM – in %.

		Naïve Bayes						1-NN					
		Baseline			RPV			Baseline			RPV		
	Datasets	No FS	All-Neg	All-Pos	IG	R	LazyR	No FS	All-Neg	All-Pos	IG	R	LazyR
CE	BP	62.0	0.0	59.4	60.5	60.9	**65.7**	58.4	20.6	**61.0**	60.9	59.3	60.1
	CC	**65.7**	39.9	65.5	63.6	65.3	63.6	59.9	62.9	**64.0**	62.1	63.2	63.0
	MF	57.6	44.2	**59.1**	57.0	54.1	56.7	**53.4**	24.4	45.4	46.2	44.6	44.4
DM	BP	**59.4**	0.0	51.4	53.8	54.6	55.5	58.8	18.1	**62.4**	58.6	56.8	59.6
	CC	66.7	0.0	75.6	**76.2**	75.7	74.4	**71.9**	50.3	71.0	69.5	69.5	68.9
	MF	58.0	0.0	67.0	63.8	63.3	**67.2**	51.3	0.0	70.1	70.5	70.5	**70.9**
MM	BP	59.1	25.7	66.3	**69.9**	68.4	67.9	**65.1**	33.8	57.9	61.6	62.1	59.5
	CC	64.1	28.1	68.3	69.0	66.4	**69.4**	55.0	37.0	54.1	56.1	**56.8**	54.6
	MF	63.5	57.6	65.9	65.3	64.7	**67.9**	61.3	42.1	**68.1**	63.9	61.8	66.1
SC	BP	61.5	0.0	54.7	56.2	52.4	**61.6**	54.7	0.0	57.3	**67.1**	60.1	57.2
	CC	57.6	0.0	59.3	58.9	59.2	**59.9**	**52.5**	0.0	38.7	39.0	39.0	38.8
	MF	34.2	0.0	**57.8**	54.2	44.1	54.6	**43.5**	0.0	34.0	34.2	34.3	34.0
General	Tweets T	68.2	8.8	72.2	73.0	73.1	**73.6**	73.0	0.0	74.2	74.7	74.8	**75.1**
	Tweets C	87.6	0.0	94.5	94.8	**95.0**	94.8	91.2	59.1	**95.0**	94.5	94.4	**95.0**
	NY Daily	50.3	0.0	56.7	**57.1**	55.9	56.6	**53.0**	0.0	51.3	51.1	51.5	52.6
	Stb.upon	68.7	15.1	70.6	70.6	**71.0**	70.7	70.6	**71.8**	71.0	70.9	70.7	71.1
	Cities	73.5	0.0	**73.6**	63.9	71.0	71.8	59.3	24.7	61.2	61.2	60.8	**61.4**
	Avg. Rank	3.8	6.0	2.9	3.0	3.2	**2.2**	3.5	5.6	3.0	3.0	3.1	**2.9**
	#Win	2.0	0.0	3.0	3.0	2.0	**7.0**	**6.0**	1.0	4.5	1.0	1.0	3.5

Naïve Bayes: {RPV-LazyR} $\succ$ {No FS and All-Neg}
1-NN: {RPV-LazyR} $\succ$ {All-Neg}

fiers, there is no significant difference between the AUROC results of All-Pos and RPV-LazyR.

The GM results in Table 4 show that, for Naïve Bayes, RPV-LazyR obtained the smallest (best) Avg. Rank among all six methods. It also obtained the highest GM value in 7 out of the 17 datasets. The Holm post-hoc test indicated that RPV-LazyR is significantly better than No FS and All-Neg. For 1-NN, RPV-LazyR achieved the best Avg. Rank, but the No FS baseline achieved the highest #Win. Moreover, the Holm post-hoc test indicated that RPV-LazyR is significantly better than All-Neg.

In summary, the results reported in Tables 2, 3 and 4 involve six comparison settings, i.e., three predictive performance measures times two classifiers. Regarding the Avg. Ranks, RPV-LazyR was the best method in four settings (for both the NB and 1-NN classifiers with both the AUCPR and GM measures), All-Pos was the best method in one setting (for NB with AUROC) and RPV-IG was the best in one setting (for 1-NN with AUROC). Regarding the #Wins, RPV-LazyR was the best method in three settings (for both classifiers with AUCPR and for NB with GM), All-Pos was the best method in two settings (for both classifiers with AUROC), and No FS was the best approach in one setting (with 1-NN and GM).

Note that the full set of positive feature values has much higher predictive power than the full set of negative values and the full set of features, since All-Pos

Table 5: Comparing RPV with traditional feature methods CFS and ReliefF in terms of AUCPR, AUROC and GM using NB as base classifier – in %.

		Naïve Bayes								
		AUCPR			AUROC			GM		
	Datasets	CFS	ReliefF	RPV	CFS	ReliefF	RPV	CFS	ReliefF	RPV
CE	BP	47.4	47.4	**56.4**	**71.5**	63.3	69.8	47.2	46.2	**50.4**
CE	CC	48.6	50.7	**54.3**	**69.4**	67.4	64.0	50.8	50.3	**54.6**
CE	MF	41.8	40.6	**51.7**	62.8	51.9	**64.0**	40.2	42.1	**50.1**
DM	BP	**82.8**	81.5	82.5	**65.3**	52.5	61.0	**83.2**	82.8	79.3
DM	CC	82.3	82.1	**90.0**	76.0	63.2	**78.7**	81.3	80.0	**83.4**
DM	MF	76.9	76.8	**81.1**	58.0	62.3	**63.0**	76.2	78.4	**82.4**
MM	BP	72.6	70.7	**85.2**	66.9	60.4	**74.2**	75.3	72.7	**77.6**
MM	CC	71.5	70.4	**84.4**	60.6	66.8	**70.9**	68.7	72.9	**76.6**
MM	MF	74.3	70.2	**85.6**	**72.4**	65.2	70.9	**80.0**	69.0	79.1
SC	BP	28.1	25.4	**46.4**	77.6	**78.2**	74.8	27.9	30.6	**36.4**
SC	CC	13.3	13.3	**35.3**	65.7	**67.3**	67.0	11.2	14.9	**37.6**
SC	MF	19.4	19.4	**25.2**	60.7	**62.8**	60.5	15.7	19.4	**35.8**
General	Tweets T	74.9	78.1	**83.3**	74.1	66.8	**83.2**	74.9	72.1	**82.6**
General	Tweets C	90.0	87.8	**98.5**	71.0	68.1	**73.1**	87.8	86.4	**97.5**
General	NY Daily	59.2	59.0	**64.8**	55.1	55.1	**56.1**	59.0	57.3	**60.1**
General	Std.upon	70.5	66.1	**78.7**	75.8	68.3	**78.1**	66.4	65.8	**74.4**
General	Cities	57.1	56.9	**74.1**	62.3	58.0	**69.7**	57.1	58.4	**69.4**
	Avg Rank	2.1	2.8	**1.1**	2.0	2.5	**1.5**	2.3	2.5	**1.2**
	#Wins	1.0	0.0	**16.0**	4.0	3.0	**10.0**	2.0	0.0	**15.0**

AUCPR: {RPV} $\succ$ {CFS and ReliefF}
AUROC: {RPV} $\succ$ {ReliefF}
GM: {RPV} $\succ$ {CFS and ReliefF}

obtained much better AUCPR, AUROC and GM Avg. Rank values than All-Neg and No FS. This suggests that positive feature values are more informative than negative feature values. Also, RPV-LazyR has both the best average rank and the highest number of wins for Naïve Bayes using AUCPR and GM, and for 1-NN using AUCPR. RPV-LazyR also obtained the second highest number of wins for both classifiers using AUROC. So, selecting the most relevant features using the LazyR relevance measure increases the predictive power when compared with the baselines methods. Also, this result indicates the benefit of using our proposed LazyR measure, rather than the R or IG measures. Hence, the RPV method using LazyR was chosen to be used in the next experiments due to its highest predictive accuracy overall.

5.3.2 Comparison against traditional feature selection methods

In the second experiment, the RPV method (RPV with LazyR) is evaluated against two non-hierarchical feature selection methods: CFS using WEKA's default parameters and best-first search (with lookup set to 1 and search termination set to 5) and ReliefF (with a threshold set to 0.01, sigma set to 2 and k set to 10). Tables 5 and 6 show the results for the Naïve Bayes and 1-NN classifiers, respectively. These

Table 6: Comparing RPV with traditional feature methods CFS and ReliefF in terms of AUCPR, AUROC and GM using 1-NN as base classifier – in %.

		1-NN								
		AUCPR			AUROC			GM		
	Datasets	CFS	ReliefF	RPV	CFS	ReliefF	RPV	CFS	ReliefF	RPV
CE	BP	61.6	54.4	**65.7**	**66.7**	64.8	64.4	59.3	59.1	**60.1**
	CC	63.2	**65.5**	63.6	**69.6**	61.1	68.2	63.1	**63.6**	63.0
	MF	53.2	38.8	**56.7**	62.1	51.9	**63.6**	**45.9**	**45.9**	44.4
DM	BP	60.6	**69.3**	55.5	**64.4**	57.8	62.2	**66.2**	62.5	59.6
	CC	64.8	69.8	**74.4**	**75.2**	58.3	67.8	**70.9**	62.4	68.9
	MF	51.9	54.7	**67.2**	58.4	53.9	**66.5**	54.3	57.8	**70.9**
MM	BP	54.4	44.1	**67.9**	**73.9**	58.4	64.5	47.5	47.1	**59.5**
	CC	50.4	44.8	**69.4**	60.5	**65.1**	56.8	45.2	42.7	**54.6**
	MF	62.7	47.4	**67.9**	61.9	61.8	**65.0**	47.6	38.8	**66.1**
SC	BP	**64.8**	50.2	61.6	69.1	68.1	**69.4**	50.9	50.2	**57.2**
	CC	46.2	0.0	**59.9**	**70.7**	65.5	64.8	0.0	15.6	**38.8**
	MF	26.1	26.3	**54.6**	57.7	66.7	**70.0**	26.3	26.3	**34.0**
General	Tweets T	70.6	70.8	**73.6**	59.3	63.3	**71.7**	74.8	68.8	**75.1**
	Tweets C	87.6	89.4	**94.8**	66.3	65.2	**67.9**	91.9	88.5	**95.0**
	NY Daily	48.7	43.8	**56.6**	50.8	51.2	**52.8**	49.0	42.9	**52.6**
	Std.upon	67.1	67.3	**70.7**	74.9	68.2	**78.1**	**71.9**	66.7	71.1
	Cities	59.5	55.0	**71.8**	58.8	59.7	**67.3**	56.8	56.3	**61.4**
	Avg Rank	2.4	2.4	**1.2**	1.9	2.5	**1.6**	1.9	2.6	**1.5**
	#Wins	1.0	2.0	**14.0**	6.0	1.0	**10.0**	3.5	1.5	**12.0**

AUCPR: {RPV} $\succ$ {CFS and ReliefF}
AUROC: {RPV} $\succ$ {ReliefF}
GM: {RPV} $\succ$ {ReliefF}

tables are divided into three partitions, showing results for the AUCPR, AUROC and GM measures.

As shown in the last two rows of the tables, RPV (with LazyR) obtained the best Avg. Rank and by far the highest #Win for all six combinations of the two classifiers and the three performance measures. The Holm post-hoc test indicated that RPV is significantly better than both CFS and ReliefF for three of those six combinations, whilst in the remaining three combinations RPV is significantly better than ReliefF only.

5.3.3 Comparison against state-of-the-art hierarchical feature selection methods

In this experiment, the RPV method is evaluated against five recent hierarchical feature selection methods: GTD, TSEL, SHSEL (using Information Gain, with a threshold set to 0.99 [24]), HIP and MR – all reviewed in Section 3. GTD, TSEL, HIP and MR have no user-defined parameters. MR could also use the LazyR instead of the original R measure. However, previous experiments (not reported here) have shown that the R measure is the best relevance measure to MR. So, we use the original MR with R in the following experiments.

Table 7 shows that RPV achieves both the best Avg. Rank and the highest #Win for both NB and 1-NN, in terms of AUCPR. For NB, the Holm post-hoc

Table 7: Comparing the proposed RPV against state-of-the-art feature selection methods in terms of AUCPR – in % values.

		Naïve Bayes						1-NN					
	Datasets	GTD	TSEL	SHSEL	HIP	MR	RPV	GTD	TSEL	SHSEL	HIP	MR	RPV
CE	BP	55.1	53.1	56.5	**58.4**	55.6	56.4	45.3	47.4	52.5	**56.6**	52.6	50.4
	CC	56.3	56.5	49.6	**57.1**	54.5	54.3	53.1	47.5	48.4	50.6	52.8	**54.6**
	MF	50.2	47.8	45.6	50.6	50.0	**51.7**	49.3	47.2	45.5	49.2	47.3	**50.1**
DM	BP	83.1	80.2	84.1	**87.6**	82.0	82.5	77.2	80.8	**82.2**	78.3	79.3	79.3
	CC	87.6	84.0	88.3	88.6	89.6	**90.0**	78.3	76.9	**87.6**	83.7	82.4	83.4
	MF	81.9	79.7	79.3	**82.9**	80.4	81.1	76.3	78.7	81.5	80.0	80.6	**82.4**
MM	BP	82.5	82.1	85.7	**86.0**	85.3	85.2	79.3	77.0	**79.7**	73.7	75.6	77.6
	CC	84.5	82.2	82.8	80.0	**85.0**	84.4	73.5	**81.1**	78.9	70.0	78.3	76.6
	MF	**87.1**	83.9	85.9	85.5	82.7	85.6	80.6	79.7	**82.9**	81.2	82.8	79.1
SC	BP	45.6	43.2	41.1	46.3	41.2	**46.4**	33.8	33.7	**41.9**	31.8	28.0	36.4
	CC	34.0	32.5	26.3	30.4	29.9	**35.3**	33.4	28.7	30.2	31.7	28.3	**37.6**
	MF	26.8	20.4	24.0	**27.0**	25.8	25.2	29.9	33.8	23.6	**36.8**	36.0	35.8
General	Tweets T	81.6	71.1	73.7	77.2	82.1	**83.3**	77.6	77.9	75.4	77.7	79.4	**82.6**
	Tweets C	98.3	95.4	82.3	85.3	87.8	**98.5**	90.2	91.7	84.2	94.1	87.8	**97.5**
	NY Daily	64.1	60.7	60.8	64.2	64.5	**64.8**	56.2	54.7	55.9	59.4	59.4	**60.1**
	Stb.upon	77.8	75.7	76.7	75.9	76.2	**78.7**	71.2	73.4	73.6	73.4	72.6	**74.4**
	Cities	70.7	64.8	62.8	73.2	69.8	**74.1**	67.8	63.7	62.6	64.8	64.5	**69.4**
	Avg. Rank	2.9	5.0	4.3	2.7	3.6	**2.2**	4.1	4.3	3.4	3.4	3.6	**2.2**
	#Win	1.0	0.0	0.0	6.0	1.0	**9.0**	0.0	1.0	5.0	2.0	0.0	**9.0**

Naïve Bayes: {RPV} $\succ$ {TSEL, SHSEL and MR}
1-NN: {RPV} $\succ$ {GTD, TSEL, SHSEL, HIP and MR}

test indicates that RPV is statistically better than TSEL, SHSEL and MR. For 1-NN, the Holm test indicates that RPV is significantly better than all the other five hierarchical feature selection methods.

Considering the results for the AUROC measure in Table 8, RPV obtained the best Avg. Rank and #Win for both NB and 1-NN classifiers. According to the Holm post-hoc test, for NB, RPV is statistically superior to four out of five state-of-the-art feature selection methods (GTD, TSEL, SHSEL and MR), while, for 1-NN, RPV obtained statistically superior results to three methods (GTD, TSEL and HIP).

Table 9 shows that RPV achieves both the best Avg. Rank and by far the highest #Win for both Naïve Bayes and 1-NN, in terms of GM. For NB, the Holm post-hoc test indicates that RPV is statistically superior to all five hierarchical methods. For 1-NN, the post-hoc test indicates that RPV had a statistically better GM than GTD, TSEL, SHSEL and HIP.

Note that there is a large difference of performance between RPV and HIP, the two best methods overall. RPV achieves both a higher number of wins and a better average rank than HIP in all experiments reported in Tables 7, 8 and 9. Also, RPV achieved statistically better results than HIP in four out of the six comparison scenarios (three measures times two classifiers).

Summarizing, using AUCPR, AUROC and GM measures, and both Naïve Bayes and 1-NN classifiers, RPV achieved better average rank and higher number of wins than the hierarchical methods GTD, TSEL, SHSEL, HIP and MR. In all

Table 8: Comparing the proposed RPV against state-of-the-art feature selection methods in terms of AUROC – in % values.

		Naïve Bayes						1-NN					
	Datasets	GTD	TSEL	SHSEL	HIP	MR	RPV	GTD	TSEL	SHSEL	HIP	MR	RPV
CE	BP	62.8	65.3	70.0	**71.5**	70.5	69.8	57.6	62.9	66.9	**68.6**	67.4	64.4
	CC	63.8	65.4	64.0	**70.9**	68.9	67.4	65.8	62.4	62.8	67.0	67.2	**68.2**
	MF	55.2	60.4	59.6	62.8	62.5	**64.0**	59.8	59.2	59.6	63.6	60.8	**63.6**
DM	BP	65.8	50.3	62.2	**70.3**	64.5	61.0	55.3	59.7	61.6	57.8	**62.4**	62.2
	CC	60.1	68.4	77.1	75.4	75.2	**78.7**	56.8	60.0	**77.2**	70.1	67.0	67.8
	MF	63.5	59.5	59.7	**64.7**	61.7	63.0	50.7	58.1	62.6	60.5	60.0	**66.5**
MM	BP	55.1	68.2	73.3	**74.5**	71.8	74.2	62.9	63.4	**65.2**	50.2	55.2	64.5
	CC	66.6	66.3	68.7	64.1	**70.9**	**70.9**	51.7	**63.5**	57.8	43.1	61.8	56.8
	MF	54.9	65.2	68.4	69.9	66.0	**70.9**	63.9	64.2	**72.9**	68.2	70.7	65.0
SC	BP	68.6	75.3	76.0	**77.8**	72.9	74.8	62.3	65.9	66.9	66.5	61.3	**69.4**
	CC	58.9	**68.5**	64.5	66.0	63.2	67.0	61.4	63.4	**67.3**	62.8	62.9	64.8
	MF	56.1	57.1	57.3	59.6	59.3	**60.5**	63.5	67.8	56.1	**71.6**	69.4	70.0
General	Tweets T	68.3	74.6	78.2	79.0	74.6	**83.2**	58.5	65.5	65.3	48.7	69.8	**71.7**
	Tweets C	63.4	69.6	70.2	71.9	67.4	**73.1**	58.8	65.8	65.7	49.0	62.8	**67.9**
	NY Daily	50.6	46.5	55.1	51.8	55.7	**56.1**	69.4	71.7	**76.1**	71.0	71.1	52.8
	Stb.upon	69.9	77.4	77.7	77.4	77.8	**78.1**	72.8	77.2	77.9	77.4	76.8	**78.1**
	Cities	63.4	57.3	60.7	**71.4**	68.5	69.7	65.3	62.6	61.0	64.5	64.4	**67.3**
	Avg. Rank	5.2	4.6	3.5	2.2	3.3	**2.0**	5.1	3.9	2.9	3.6	3.3	**2.2**
	#Win	0.0	1.0	0.0	7.0	0.5	**8.5**	0.0	1.0	5.0	2.0	1.0	**8.0**

Naïve Bayes: {RPV} $\succ$ {GTD, TSEL, SHSEL and MR}
1-NN: {RPV} $\succ$ {GTD, TSEL and HIP}

comparisons RPV was statistically superior to TSEL. In five of six comparisons RPV was statistically superior to SHSEL. When compared to GTD, HIP and MR, in some cases, the differences were not statistically significant, however, in all these cases, RPV clearly outperformed these hierarchical methods in terms of both average rank and number of wins.

In addition, by analyzing the results of RPV in Tables 2 through 9, we can observe that overall RPV performed particularly well in the five 'General' datasets, which broadly speaking are the largest datasets used in our experiments, in terms of number of features, number of edges in the feature hierarchy and number of instances – as can be observed in Table 1.

5.3.4 Running time performance

Figure 2 shows the results of the experiments that measure the methods' runtimes on a computer with 4 GB of RAM and an Intel Core i5 1.6GHz CPU. This figure reports the mean, over the 21 datasets, of the ratio of the runtime of each method over the runtime of the RPV method as a baseline. This mean ratio is used because the runtimes vary greatly in magnitude across the 21 datasets, so a direct mean over the raw runtime values would be misleading. Subfigure 4(a) shows the mean ratio of the training time of each method over RPV's training time. For eager methods, training time includes the time spent selecting features and building the Naïve Bayes model. For lazy methods, it includes the time for pre-computing the

Table 9: Comparing the proposed RPV against state-of-the-art feature selection methods in terms of GM – in % values.

		Naïve Bayes						1-NN					
	Datasets	GTD	TSEL	SHSEL	HIP	MR	RPV	GTD	TSEL	SHSEL	HIP	MR	RPV
CE	BP	58.6	56.7	61.5	61.4	63.4	**65.7**	51.2	54.3	50.8	52.8	57.9	**60.1**
	CC	64.0	63.5	62.6	**68.6**	65.3	63.6	**63.6**	57.7	62.3	60.2	63.4	63.0
	MF	50.7	47.2	48.4	50.9	54.1	**56.7**	45.7	44.5	30.5	**51.3**	48.3	44.4
DM	BP	61.1	52.8	58.4	**66.1**	57.8	55.5	53.4	**61.2**	49.7	53.6	60.9	59.6
	CC	58.0	63.2	61.6	68.4	61.4	**74.4**	61.8	66.3	**75.0**	68.1	69.0	68.9
	MF	51.6	56.3	53.3	57.1	51.6	**67.2**	50.2	54.3	48.5	48.4	46.8	**70.9**
MM	BP	62.9	59.1	**68.8**	67.3	59.1	67.9	**62.8**	60.9	60.3	44.2	56.0	59.5
	CC	64.7	66.4	60.6	58.3	61.9	**69.4**	44.4	56.4	48.1	45.2	**56.7**	54.6
	MF	62.4	60.5	62.8	65.8	61.1	**67.9**	64.5	64.6	65.4	53.0	65.9	**66.1**
SC	BP	56.3	63.7	52.1	68.8	**69.1**	61.6	47.5	46.3	51.4	44.2	38.4	**57.2**
	CC	48.2	**60.2**	33.1	47.8	42.2	59.9	**41.9**	32.5	33.3	36.8	38.9	38.8
	MF	39.0	29.6	26.3	42.8	31.9	**54.6**	33.3	38.3	21.1	**38.5**	36.2	34.0
General	Tweets T	70.6	68.2	63.1	73.0	71.8	**73.6**	70.2	72.1	64.2	72.5	72.2	**75.1**
	Tweets C	88.4	90.2	89.7	72.6	77.8	**94.8**	90.4	88.2	0.0	84.2	84.1	**95.0**
	NY Daily	49.6	52.3	39.1	52.4	49.8	**56.6**	50.6	49.8	41.8	52.1	52.5	**52.6**
	Stb.upon	70.4	69.3	68.2	66.4	66.4	**70.7**	68.0	70.3	68.8	70.5	**71.3**	71.1
	Cities	62.4	57.1	66.0	**86.1**	73.9	71.8	**62.0**	60.7	57.5	61.1	58.2	61.4
	Avg. Rank	4.0	4.2	4.4	2.9	3.7	**1.8**	3.6	3.6	4.6	3.8	3.0	**2.3**
	#Win	0.0	1.0	1.0	3.0	1.0	**11.0**	4.0	1.0	1.0	2.0	2.0	**7.0**

Naïve Bayes: {RPV} $\succ$ {GTD, TSEL, SHSEL, HIP and MR}
1-NN: {RPV} $\succ$ {GTD, TSEL, SHSEL and HIP.}

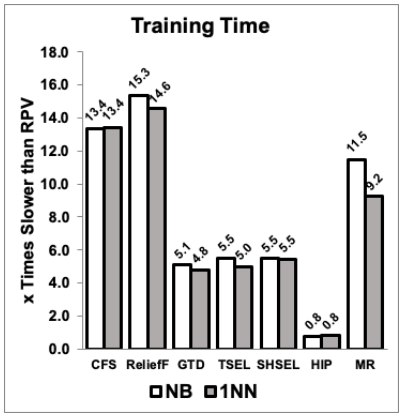
probabilities used by Naïve Bayes and computing for each feature in the dataset: its ancestors and descendants (for HIP and RPV), the paths from a feature to root and leafs (for MR) and the relevance value (for MR and RPV).

Subfigure 4(b) shows the mean ratio of the testing time of each method over RPV's testing time. For eager methods, testing time includes the time required for classifying one instance with Naïve Bayes or 1-NN. For lazy methods, it includes the time required for feature selection and classifying one instance.

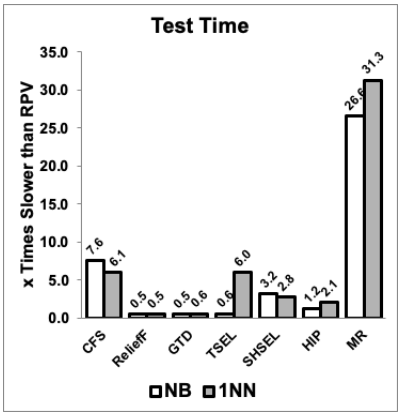
CFS, ReliefF, GTD, TSEL and SHSEL are eager methods and find a single subset of features during the training step which will be used to classify all test instances, while the lazy methods HIP, MR and RPV find a subset of features for each test instance in the classification step. Hence, as can be observed in Figure 4, RPV has the best training time when compared with all other methods but HIP, which is 20% faster to train than RPV. Regarding the testing times, RPV is the fastest among the three lazy methods, but it is slower than most eager methods.

5.3.5 Evaluating the feature space compression

The selection of a small subset of relevant features for each instance may improve the interpretability of the model's predictions, since only relevant features are used to justify each prediction. So, we report the percentage of features selected by each method in Table 10. Again, the table's first two columns show the feature hierarchies (GO term types) used to build the datasets and the name of the general



(a) Mean ratio of each method's training time over RPV's training time.



(b) Mean ratio of each method's test time over RPV's test time.

Fig. 4: Mean ratios of a method's training and test times over RPV's corresponding times, evaluated for CFS, ReliefF, GTD, TSEL, SHSEL, HIP and MR methods.

Table 10: Average percentage of features selected by CFS, ReliefF, GTD, TSEL, SHSEL, HIP, MR, All-Neg, All-Pos and RPV.

	Dataset	CFS	ReliefF	GTD	TSEL	SHSEL	HIP	MR	All-Neg	All-Pos	RPV
CE	BP	4.6	12.3	52.8	30.6	**1.6**	7.0	17.8	95.6	4.4	1.7
	CC	8.9	10.0	64.8	38.6	4.6	16.3	33.6	93.9	6.1	**2.8**
	MF	10.2	8.3	59.5	22.9	3.0	11.3	23.7	95.3	4.7	**2.5**
DM	BP	**3.7**	19.0	55.2	29.5	12.5	11.9	22.7	91.8	8.2	4.2
	CC	13.7	38.9	57.5	40.3	15.3	21.7	35.1	88.1	11.9	**5.8**
	MF	8.8	38.5	52.0	23.8	14.1	15.4	22.6	93.0	7.0	**4.8**
MM	BP	12.8	25.8	55.2	32.7	15.2	11.7	21.5	89.4	10.6	**4.5**
	CC	10.9	36.1	58.5	37.8	18.1	28.4	32.7	84.0	16.0	**8.4**
	MF	7.5	28.2	59.2	22.4	16.0	20.1	25.2	90.7	9.4	**6.2**
SC	BP	3.6	35.9	52.5	26.3	2.9	7.0	17.9	94.8	5.2	**2.3**
	CC	15.2	36.3	65.3	35.3	**4.9**	19.7	33.3	90.6	9.4	5.9
	MF	12.9	43.2	55.9	24.4	**3.1**	10.8	26.0	94.7	5.3	3.4
General	Tweets T	4.3	8.0	56.7	33.4	4.1	10.1	36.8	98.2	1.8	**0.9**
	Tweets C	2.1	13.3	55.9	35.4	48.6	39.0	43.1	99.0	1.0	**0.8**
	NY Daily	8.8	1.9	67.0	33.2	5.3	14.7	20.3	97.8	2.2	**1.0**
	Stb.upon	2.4	5.1	52.0	55.6	26.2	30.3	63.8	98.3	1.2	**0.8**
	Cities	18.0	24.3	60.3	42.4	29.0	45.9	59.7	76.3	23.8	**10.6**
	#Win	1.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0	**13.0**
	Avg. %	8.7	22.7	57.7	33.2	13.2	18.9	31.5	92.4	7.5	**3.9**

domain datasets. The following columns show the percentage of features selected by each method. The last two rows show the number of wins (#Win) and the average percentage of features selected (Avg. %) across all 17 datasets.

The results show that, in general, RPV selects a feature subset smaller than the one selected by all other nine methods. RPV (with the LazyR measure) selects,

on average, only 3.9% of all features per instance, whilst achieving the highest predictive accuracy in general. Note that the percentage of features selected by RPV is about a half of those selected by All-Pos. It means that not all positive feature values are relevant and that the correct identification of relevant positive feature values increases the predictive accuracy of Naïve Bayes and 1-NN. In this work, we introduced the LazyR measure to identify such features. Although the HIP method (which selects the set of the most specific positive-valued features and the most general negative-valued features) achieved a good predictive performance, the RPV method obtained both a higher predictive performance and a much smaller selected feature subset.

Note that all lazy feature selection methods (RPV, HIP and MR) select a separate set of features for each test instance. This has the advantage of improving the user's interpretability of the classification of individual instances, when a user wants to identify the most relevant features for the current instance; but it has the disadvantage of not providing a unique set of relevant features for the dataset as a whole.

5.3.6 Brief remarks on the most frequently selected features in the bioinformatics datasets

We have ranked all features (GO terms) in each bioinformatics dataset in decreasing order of selection frequency by the RPV method. The full rankings are available in http://github.com/pablonsilva/RPV. In this subsection we briefly focus only on the top 15 features in the four datasets (one per organism) containing only BP GO terms, since this type of GO terms is broadly easier to interpret than MF and CC GO terms. The top 15 BP GO terms for each of these datasets are available in the Supplementary Tables S1-S4 in the above GitHub website.

Two examples of the relevance of some of these GO terms to the biology of ageing are as follows. First, the term GO:0006412 ("Translation") was ranked seventh in the "yeast" dataset. As evidence supporting the relevance of this term, a reduction in the levels of 60S ribosomal subunits led to a significant increase in yeast's replicative lifespan as shown in [25]. As a second example, in the "fly" dataset, several stress-response related GO terms – e.g., GO:0033554 ("cellular response to stress") and GO:0006950 ("response to stress") – were among the top-15 terms. Stress response has been shown to be enhanced in a mutant line with extended longevity [16], and stress-response functions are enriched in genes with enhanced rhythmicity of expression in late life ("late-life cyclers") [13].

6 Conclusion and Future Work

This paper presented a novel lazy method for hierarchical and sparse feature selection based on the hypothesis that positive feature values provide more meaningful and accurate information, even though they are present to a small extent in each instance. Our method, named Select Relevant Positive Feature Values (RPV), has some interesting properties: (i) it selects rare but informative and relevant positive features; (ii) it selects smaller feature subsets; and (iii) it is based on a new lazy

feature relevance measure (LazyR) which assesses the predictive power of a feature value specifically in the current test instance being classified.

The computational experiments involved 17 real-world datasets and six different classification scenarios, namely six combinations of two different classifiers (Naïve Bayes and 1-NN) times three different predictive accuracy measures (AUCPR, AUROC and the Geometric Mean of Sensitivity and Specificity). The results have shown that the proposed RPV method obtained in general the best predictive accuracy across those six classification scenarios. Overall, the proposed RPV method (with the LazyR measure) was compared, across the above six scenarios, against 12 other feature selection approaches: five hierarchical feature selection methods, two traditional (non-hierarchical) feature selection methods, three baseline approaches, and two other variants of RPV (not using the LazyR measure).

The results of statistical significance tests have shown that RPV obtained predictive accuracies significantly better than another approach in 44 out of the 72 cases (61.1% of all the cases). In addition, in none of those cases RPV's predictive accuracy was significantly worse than the accuracy of any other feature selection approach. Furthermore, RPV selected in general the smallest subset of features, among all evaluated feature selection methods. This is also desirable, since each instance is classified using its own small set of relevant features; hence each instance's classification is justified by a more specific feature subset, improving prediction interpretability.

In addition, the hypothesis that selecting positive feature values might increase the predictive accuracy of the classifier (mentioned earlier) is supported by two types of results. First, the fact that RPV, which selects only a subset of positive feature values (i.e., it never selects negative feature values) obtained by far the best predictive accuracy results. Second, the fact that the results of the All-Pos baseline method, which selects all positive feature values (and no negative values) were clearly better than the results of the All-Neg method, which selects all negative feature values (and no positive values), as discussed in Section 5.3.1.

In future work, we plan to evaluate the behaviour of the proposed RPV method in each application domain analysing the usefulness of the selected features with the assistance of specialists from those domains.

7 Acknowledgements

We thank Dr. Cen Wan for kindly providing us with the program code of the HIP and MR methods. This work was supported by the Brazilian government's agencies CAPES and CNPq.

References

1. D. Aha. *Lazy Learning.* Kluwer Academic Publishers, 1997.
2. T. Cover and J. Thomas. *Elements of Information Theory.* Wiley-Interscience, second edition, 2006.
3. P. da Silva, A. Plastino, and A. Freitas. A novel genetic algorithm for feature selection in hierarchical feature spaces. In *Proc. of the 2018 SIAM International Conference on Data Mining*, pages 738–746. SIAM, 2018.

4. J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
5. J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan. Feature selection based on structured sparsity: a comprehensive study. *IEEE Transactions on Neural Networks and Learning Systems*, 28(7):1490 – 1507, 2016.
6. M. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proc. 17th International Conference on Machine Learning (ICML)*, pages 359–366, 2000.
7. M. Hall, E. Frank, G. Holmes, B. Pfahringer, and P. Reutemann. The weka data mining software: an update. *ACM SIGKDD Exploration Newsletter*, 11(1):10–18, 2009.
8. J. Han, M. Kamber, and J. Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2011.
9. S. Holm. A simple sequential rejective method procedure. *Scandinavian Journal of Statistics*, 6:65–70, 1979.
10. N. Japkowicz and M. Shah. *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge University Press, 2011.
11. Y. Jeong and S.-H. Myaeng. Feature selection using a semantic hierarchy for event recognition and type classification. In *Proc. 6th Intl. Joint Conf. on NLP (IJCNLP)*, pages 136–144, 2013.
12. I. Kononenko. Estimating attributes: Analysis and extensions of relief. In *Proc. 7th European Conference on Machine Learning*, 1994.
13. R. e. a. Kuintzle. Circadian deep sequencing reveals stress-response genes that adopt robust rhythmic expression during aging. *Nature Communications*, 8:943–946, 2017.
14. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015.
15. J. Li, K. Cheng, S. Wang, F. Morstatter, R. Trevino, J. Tang, and H. Liu. Feature selection: A data perspective. *arXiv preprint arXiv:1601.07996*, 2016.
16. Y. e. a. Lin. Extended life-span and stress resistance in thedrosophila mutant methuselah. *Science*, 282:943–946, 1998.
17. H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Springer, 2012.
18. H. Liu and R. Setiono. A probabilistic approach to feature selection: A filter solution. In *Proc. 13th International Conference on Machine Learning (ICML)*, pages 319–327, 1996.
19. J. Liu and J. He. Moreau-yosida regularization for grouped tree structure learning. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1459–1467. Curran Associates, Inc., 2010.
20. S. Lu, Y. Ye, R. Tsui, H. Su, R. Rexit, S. Wesaratchakit, X. Liu, and R. Hwa. Domain ontology-based feature reduction for high dimensional drug data and its application to 30-day heart failure readmission prediction. In *Proc. 9th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing*, pages 478–484, 2013.
21. J. Magalhães, A. Bukovsky, G. Lehmann, J. Costa, Y. Li, V. Fraifeld, and G. Church. The human ageing genomic resources: Online databases and tools for biogerontologistis. *Ageing Cell*, 8(1):65–72, 2009.

22. R. Pereira, A. Plastino, B. Zadrozny, L. Merschmann, and A. Freitas. Lazy attribute selection: Choosing attributes at classification time. *Intelligent Data Analysis*, 15(5):715–732, 2011.
23. C. Qi, L. Yi, H. Su, and L. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 5099–5108. Curran Associates, Inc., 2017.
24. P. Ristoski and H. Paulheim. Feature selection in hierarchical feature spaces. In S. Dzeroski, P. Panov, D. Docev, and L. Todorovski, editors, *Proc. Discovery Science 2014*, volume 8777 of *LNCS*, pages 288–300. Springer, 2014.
25. K. e. a. Steffen. Yeast life span extension by depletion of 60s ribosomal subunits is mediated by gcn4. *Cell*, 133(2):292–302, 2008.
26. C. Stencil and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1986.
27. The GO Consortium. Gene ontology: Tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.
28. J. Vergara and P. Estévez. A review of feature selection methods based on mutual information. *Neural Computing and Applications*, 24(1):175–186, 2014.
29. C. Wan and A. Freitas. Two methods for constructing a gene ontology-based feature network for a bayesian network classifier and applications to datasets of ageing-related genes. In *Proc. 6th ACM Conf. on Bioinfo., Comp. Biology and Health Informatics (BCB)*, pages 27–36, 2015.
30. C. Wan and A. Freitas. An empirical evaluation of hierarchical feature selection methods for classification in bioinformatics datasets with gene ontology-based features. *Artificial Intelligence Review*, 50(2):201–240, 2017.
31. C. Wan, A. Freitas, and J. Magalhães. Predicting the pro-longevity or anti-longevity effect of model organism genes with new hierarchical feature selection methods. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 12(2):262–275, 2015.
32. L. Wang, Y. Wang, and Q. Chang. Feature selection methods for big data bioinformatics: A survey from the search perspective. *Methods*, 111(1):21–31, 2016.

Appendix A - Statistical Analysis

This appendix shows more detailed results of the statistical analysis for the experiments reported in Sections 5.3.1, 5.3.2 and 5.3.3. Table A.1 shows the detailed results of the Friedman test and the Holm post-hoc test. This table is organized as follows. Each one of the six parts of the table shows the results of the statistical test from a different table in the results Section (5.3.1, 5.3.2 or 5.3.3). The left-handed side of each part shows the statistical results for the Naïve Bayes classifier, while the right-handed side shows the results for the 1-NN classifier. The five columns in the left and right parts of Table A.1 represent, respectively, the feature selection method's name, its average rank, the p-value obtained by the Holm test, the adjusted α and whether or not the comparison between RPV and the given method is statistically significant (Sig?) according to the Holm post-hoc test. The last row in each of the six parts of the table shows the value of the computed Friedman's statistic (X_f^2) and whether or not the test's result is statistically significant.

For the Friedman test, a significant difference is found when the value of X_f^2 is greater than the critical value of 12.83 (this number is defined for the comparison with $k = 6$ methods, $n = 17$ datasets and significance level of 5%) or 7.41 specifically for the results of Tables 5

Table A.1: Results from the statistical analysis of the experiments presented in Sections 5.3.1, 5.3.2 and 5.3.3.

AUCPR (Results for Table 2)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	2.3				RPV	2.2			
RPV-IG	2.8	2.2E-01	0.017	No	RPV-R	2.6	2.6E-01	0.017	No
All-Pos	3.2	8.0E-02	0.017	No	RPV-IG	2.7	2.1E-01	0.017	No
RPV-R	3.6	1.1E-02	0.017	Yes	All-Pos	3.1	8.0E-01	0.017	No
No FS	3.6	1.4E-03	0.013	Yes	No FS	5.1	1.0E-05	0.013	Yes
All-Neg	5.5	1.0E-05	0.010	Yes	All-Neg	5.3	1.0E-05	0.010	Yes
Friedman's $X_f^2 = 28.20$ (Yes)					Friedman's $X_f^2 = 43.25$ (Yes)				

AUROC (Results for Table 3)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	2.8				RPV	2.8			
All-Pos	2.4	3.2E-01	0.050	No	RPV-IG	2.4	3.2E-01	0.050	No
No FS	3.1	2.7E-01	0.025	No	All-Pos	3.1	2.7E-01	0.025	No
RPV-R	3.4	1.7E-01	0.017	No	RPV-R	3.2	2.7E-01	0.017	No
RPV-IG	3.6	1.1E-01	0.013	No	No FS	4.6	2.5E-03	0.013	Yes
All-Neg	5.6	1.0E-05	0.010	Yes	All-Neg	4.9	5.3E-04	0.010	Yes
Friedman's $X_f^2 = 31.83$ (Yes)					Friedman's $X_f^2 = 25.27$ (Yes)				

GM (Results for Table 4)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	2.2				RPV	2.9			
All-Pos	2.9	1.4E-01	0.017	No	All-Pos	3.0	4.4E-01	0.013	No
RPV-IG	3.0	1.1E-01	0.017	No	RPV-IG	3.0	4.4E-01	0.013	No
RPV-R	3.2	6.0E-02	0.017	No	RPV-R	3.1	3.8E-01	0.013	No
No FS	3.8	6.3E-03	0.013	Yes	No FS	3.5	1.7E-01	0.013	No
All-Neg	6.0	1.0E-05	0.010	Yes	All-Neg	5.6	1.3E-05	0.010	Yes
Friedman's $X_f^2 = 43.28$ (Yes)					Friedman's $X_f^2 = 26.44$ (Yes)				

AUCPR (Results for Table 5 and 6)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	1.1				RPV	1.2			
CFS	2.1	1.0E-04	0.050	Yes	CFS	2.4	2.0E-02	0.050	Yes
ReliefF	2.8	1.1E-04	0.025	Yes	ReliefF	2.4	1.2E-02	0.025	Yes
Friedman's $X_f^2 = 26.15$ (Yes)					Friedman's $X_f^2 = 14.94$ (Yes)				

AUROC (Results for Table 5 and 6)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	1.5				RPV	1.6			
CFS	2.0	1.5E-01	0.050	No	CFS	1.9	2.7E-01	0.050	No
ReliefF	2.5	2.0E-02	0.025	Yes	ReliefF	2.5	3.2E-03	0.025	Yes
Friedman's $X_f^2 = 7.53$ (Yes)					Friedman's $X_f^2 = 7.88$ (Yes)				

GM (Results for Table 5 and 6)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	1.2				RPV	1.5			
CFS	2.3	1.2E-02	0.050	Yes	CFS	1.9	1.5E-01	0.050	No
ReliefF	2.5	3.7E-03	0.025	Yes	ReliefF	2.6	2.0E-02	0.025	Yes
Friedman's $X_f^2 = 17.76$ (Yes)					Friedman's $X_f^2 = 10.71$ (Yes)				

AUCPR (Results for Table 7)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	2.2				RPV	2.2			
HIP	2.7	2.2E-01	0.025	No	SHSEL	3.4	1.5E-02	0.050	Yes
GTD	2.9	1.4E-01	0.025	No	HIP	3.4	1.5E-02	0.030	Yes
MR	3.6	1.5E-02	0.017	Yes	MR	3.6	1.5E-02	0.017	Yes
SHSEL	4.3	5.3E-04	0.013	Yes	GTD	4.1	1.3E-03	0.013	Yes
TSEL	5.0	1.3E-05	0.010	Yes	TSEL	4.3	5.3E-04	0.010	Yes
Friedman's $X_f^2 = 27.96$ (Yes)					Friedman's $X_f^2 = 12.92$ (Yes)				

AUROC (Results for Table 8)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	2.0				RPV	2.2			
HIP	2.2	3.8E-01	0.025	No	SHSEL	2.9	1.7E-01	0.050	No
MR	3.3	2.1E-02	0.025	Yes	MR	3.3	6.0E-02	0.030	No
SHSEL	3.5	9.7E-03	0.017	Yes	HIP	3.6	8.1E-03	0.017	Yes
TSEL	4.6	2.5E-05	0.013	Yes	TSEL	3.9	6.3E-03	0.013	Yes
GTD	5.2	1.0E-05	0.010	Yes	GTD	5.1	1.0E-05	0.010	Yes
Friedman's $X_f^2 = 39.45$ (Yes)					Friedman's $X_f^2 = 23.42$ (Yes)				

GM (Results for Table 9)

Naïve Bayes					1-NN				
Methods	Avg. Rank	p-value	adjusted α	Sig?	Methods	Avg. Rank	p-value	adjusted α	Sig?
RPV	1.8				RPV	2.3			
HIP	2.9	4.3E-02	0.050	Yes	MR	3.0	1.4E-01	0.050	No
MR	3.7	1.5E-03	0.025	Yes	TSEL	3.6	1.1E-02	0.025	Yes
GTD	4.0	3.0E-04	0.017	Yes	GTD	3.6	1.1E-02	0.017	Yes
TSEL	4.2	9.2E-05	0.013	Yes	HIP	3.8	9.7E-03	0.013	Yes
SHSEL	4.4	1.0E-05	0.010	Yes	SHSEL	4.6	1.7E-04	0.010	Yes
Friedman's $X_f^2 = 23.43$ (Yes)					Friedman's $X_f^2 = 15.31$ (Yes)				

Table B.1: Predictive performance of Naive Bayes, 1-NN, 3-NN, 5-NN, 7-NN, C4.5 (decision tree algorithm), SVM (Support Vector Machines) and RF (Random Forests) in terms of the Geometric Mean of Sensitivity and Specificity – GM (%). The best result for each dataset is shown in boldface.

	Dataset	NaiveBayes	1-NN	3-NN	5-NN	7-NN	C4.5	SVM	RF
CE	BP	**62.0**	58.4	50.0	46.7	41.1	56.6	56.6	54.6
	CC	**65.7**	59.8	63.4	62.3	62.5	62.2	60.9	59.4
	MF	57.6	53.4	56.3	56.4	53.2	46.9	45.0	**58.2**
DM	BP	59.4	58.8	47.1	44.6	31.9	**65.5**	46.0	46.8
	CC	66.7	**71.8**	**71.8**	49.4	50.7	69.3	68.0	68.1
	MF	**58.0**	51.3	39.7	29.8	23.7	54.3	30.9	53.4
MM	BP	59.1	65.1	62.2	49.2	41.4	43.3	**65.9**	56.0
	CC	**64.1**	55.0	55.3	57.3	50.1	45.2	29.6	52.4
	MF	**63.4**	61.3	57.9	47.9	37.4	46.5	54.2	60.6
SC	BP	**61.5**	54.7	52.7	33.1	21.2	49.9	47.3	46.7
	CC	**57.6**	52.5	41.5	25.1	21.0	0.0	32.1	41.7
	MF	34.2	43.5	**43.9**	42.2	0.0	26.3	30.6	43.7
General	Tweets T	68.2	73.0	70.7	68.0	68.3	74.2	73.8	**75.3**
	Tweets C	87.6	91.2	85.9	83.9	83.7	91.6	94.5	**94.9**
	NY Daily	50.3	**53.0**	52.6	52.8	46.5	49.8	47.7	50.3
	Stb.upon	68.7	70.6	71.4	71.5	71.5	71.7	71.2	**72.1**
	Cities	**73.5**	59.3	59.5	57.8	58.1	56.5	60.1	63.9
	Avg. Rank	**2.2**	3.3	4.2	6.1	7.2	4.6	4.4	4.0

and 6 (with $k = 3$, $n = 17$ and significance level of 5%). We first run the Friedman test to verify if there is a significant difference among those k methods' results. After that, if the result of the Friedman test is statistically significant, we execute the Holm test to identify the pair of methods with statistically different results. Since the value of each one of the eight comparisons shown is greater than the critical value, we can say that all experiments have detected significant differences among at least one pair of methods. So, we applied the Holm test to all scenarios. Likewise, the results presented for the Holm test are statistically significant when the p-value is lower than the adjusted α. Both the p-value and the adjusted α were internally calculated by the Holm test.

Appendix B - Predictive Performance of a Number of Classification Algorithms (Without Any Feature Selection)

This appendix reports the detailed results of the preliminary evaluation used to select the best two classification algorithms, which were then used as classifiers in all experiments reported in this paper. Table B.1 shows the results for Naïve Bayes, k-NN (with $k = 1, 3, 5$ and 7), the decision tree algorithm C4.5, an optimised version of SVM (using the RBF kernel with a grid search procedure to optimize the cost C varying from 2^{-5} to 2^{15} and gamma varying from 2^{-15} to 2^{3}) and the Random Forest (RF). These results are shown in terms of Geometric Mean between Sensitivity and Specificity (GM).

The results in Table B.1 show that Naive Bayes has achieved the best predictive performance among the evaluated methods, achieving the best average ranking (2.2). Naive Bayes is followed by 1-NN, which obtained the second best average rank (3.3). So, since Naive Bayes and 1-NN were the two best methods and both follow the lazy paradigm (like the new feature selection method proposed in this work), we decided to use them in the experimental evaluation of this work.