Information Sciences 181 (2011) 3766–3782

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Towards improving cluster-based feature selection with a simplified silhouette filter

Thiago F. Covões, Eduardo R. Hruschka *

Department of Computer Sciences, University of São Paulo (USP) at São Carlos, Av. Trabalhador São-carlense, 400, Centro Caixa Postal 668, CEP 13.560-970, São Carlos, SP, Brazil

ARTICLE INFO

Article history:
Received 8 May 2009
Received in revised form 11 February 2011
Accepted 29 April 2011
Available online 12 May 2011

Keywords:
Feature selection
Filters
Clustering
Classification

ABSTRACT

This paper proposes a filter-based algorithm for feature selection. The filter is based on the partitioning of the set of features into clusters. The number of clusters, and consequently the cardinality of the subset of selected features, is automatically estimated from data. The computational complexity of the proposed algorithm is also investigated. A variant of this filter that considers feature-class correlations is also proposed for classification problems. Empirical results involving ten datasets illustrate the performance of the developed algorithm, which in general has obtained competitive results in terms of classification accuracy when compared to state of the art algorithms that find clusters of features. We show that, if computational efficiency is an important issue, then the proposed filter may be preferred over their counterparts, thus becoming eligible to join a pool of feature selection algorithms to be used in practice. As an additional contribution of this work, a theoretical framework is used to formally analyze some properties of feature selection methods that rely on finding clusters of features.

1. Introduction

Successful data mining applications depend on several factors. The availability of suitable feature selection methods is one of such factors. Feature selection involves choosing a subset of the original variables (attributes) by eliminating the redundant, uninformative, and noisy ones. This issue has been broadly investigated in supervised learning tasks for which datasets with many features are available, like in text mining and gene expression data analysis. Under this perspective, there are many potential benefits of feature selection like, for instance [13]: facilitating data visualization and understanding; reducing the measurement and storage requirements; reducing training and running times; and defying the curse of dimensionality to improve prediction performance.

A comprehensive survey of feature selection algorithms has been presented in [26]. In brief, there are three fundamentally different approaches for feature selection [13]: *wrapper*, *filter* and *embedded*. Wrapper methods evaluate the subset of selected features by using criteria based on the results of learning algorithms, while filters select features based on intrinsic properties of the data, being independent of the learning algorithm to be used. In other words, wrapper methods assess subsets of variables according to their usefulness to a given predictor (e.g. classifier). These methods are often criticized because they require a high computational time [13]. In data mining applications one usually faces very large datasets, and so the complexity of the problem is aggravated. In such cases, it is interesting to use feature selection methods called filters [24], which are usually faster than wrappers [13,27]. Embedded methods [28] integrate the selection of features in model

* Corresponding author. Tel.: +55 (16) 3373 8182; fax: +55 (16) 3373 9751.
E-mail addresses: tcovoes@icmc.usp.br (T.F. Covões), erh@icmc.usp.br, eduardo.hruschka@pq.cnpq.br (E.R. Hruschka).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.04.050

building. The readers interested in filters are referred to references [3,4,10,13,17,23–25,29,30,36,39,43] and the bibliography therein. While some filters involve some kind of transformation of the feature space (e.g., principal component analysis and factor analysis), the present work focuses on finding subsets of features of the original space, mainly because this often allows simpler and comprehensible results, maintaining the physical interpretation of the selected features.

Feature assessments performed by filters are based on intrinsic properties of the data. These properties are presumed to affect the ultimate performance of the classification algorithm, but the feature set is filtered without considering the classification algorithm that will be ultimately employed. As already observed, in general filters are less computationally expensive than wrappers, but these may be superior in relation to the quality of the subset of features achieved. Therefore, it may be interesting to consider combinations of filters and wrappers, i.e., hybrid approaches from which one can expect to have a reasonable tradeoff between efficiency (computational effort) and efficacy (classification accuracy). Even so, filters are usually preferred when computational efficiency is of particular interest [13,24].

The feature selection algorithm proposed in this paper is based on the filters developed by Mitra et al. [29] and Au et al. [3]. These filters remove redundant features by partitioning the original feature set into some distinct clusters, which are formed by similar features. In brief, after grouping features into clusters, such filters select a subset of features from each cluster, allowing dimensionality reduction. The filter by Mitra et al. [29] is considered a state of the art method for feature selection in the statistical pattern recognition field [43]. Even so, it is important to have in mind that its performance may strongly depend upon a user-defined parameter that indirectly controls the cardinality of the subset of selected features, as will be shown in this paper. In the approach developed by Au et al. [3], by its turn, the user is left with the task of selecting a number of features from the induced clusters. Although this approach can be viewed as a more direct way of exploratory data analysis, the user ultimately needs to decide how many features to choose. In summary, although the algorithms described in [29] and [3] can be useful for exploratory data analysis (as it will be illustrated in this paper), in general there is no clear guidance on how to choose their respective user-defined parameter values, which can be critical for some applications. In this context, more automatic tools may be preferred. The approach proposed in this paper is also based on clustering similar features. Nevertheless, we use a clustering algorithm that induces a set of k clusters (k is estimated directly from data) from which a number of representative features is automatically selected. From this perspective, the proposed algorithm selects a subset of features in a more automatic fashion when compared to their counterparts. Also, a variant of this algorithm that considers attribute-class correlations is proposed. As an additional contribution of this work, we elaborate on a theoretical framework to formally analyze some properties of feature selection methods that rely on finding clusters of features (e.g., the filters proposed in [29,3], as well as the filter proposed in our current work).

The remainder of this paper is organized as follows. The next section discusses theoretical properties of methods conceived to select a subset of features from clusters of similar features. To the best of our knowledge, such discussion is new in the literature. Section 3 elaborates on related work. Section 4 describes the proposed feature selection method, which has been briefly introduced in [7], and its variant that considers attribute-class correlations. Analyses in terms of computational complexity are provided. Section 5 analyzes empirical results obtained by our filter in ten datasets, comparing them to the results achieved by two state of the art filters from the literature, namely: the filters developed by Mitra et al. [29] and Au et al. [3]. Finally, Section 6 concludes the paper and points out some future work.

2. Feature clustering

Assume a training set of N vectors $\chi = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, where each $\mathbf{x}_j$ is an observation vector consisting of M measurements of the feature set $X = \{X_1, X_2, \ldots, X_M\}$. In classification problems, each of these vectors is categorized into one of l possible classes from $C = \{c_1, c_2, \ldots, c_l\}$. We can consider these observation vectors as rows of a matrix, $\mathbf{X}$, in which x_{ji} refers to the value of feature i for vector j. A data instance is then a tuple $\langle \mathbf{x}_j, c_m \rangle$ where the vector $\mathbf{x}_j$ is assigned to class c_m, $m \in \{1,2,\ldots,l\}$. A classifier built from training data takes as input an unseen data vector and classifies it into one of the possible classes – c_1, c_2, …, c_l. In theory, this vector should fully determine the correct classification. In practice, however, this is rarely the case because, among other reasons, we usually do not have enough features to make a deterministic decision. Moreover, as discussed in Section 1, real-world datasets may contain irrelevant and redundant features. In this scenario, a probability distribution that models the classification function can be used. More precisely, it is assumed that all of the data vectors are drawn from some probability distribution over the space of observation vectors, and that for each vector we have a probability distribution on the different possible classes. In this context, for each assignment of values $\mathbf{x}$ to X there is a probability distribution $\mathbf{P}(C|X = \mathbf{x})$. We use $X = \mathbf{x}$ as a shorthand for $X_1 = x_{j1} \wedge X_2 = x_{j2} \wedge \ldots \wedge X_M = x_{jM}$, for any $j \in \{1,2,\ldots,N\}$.

Let G be some subset of $X = \{X_1, X_2, \ldots, X_M\}$. Following [43,22], the goal of feature selection can be formalized as selecting a minimum subset G such that $\mathbf{P}(C|G)$ is equal or as close as possible to $\mathbf{P}(C|X)$, where $\mathbf{P}(C|G)$ is the probability distribution of different classes given the features in G, and $\mathbf{P}(C|X)$ is the (true) original distribution given the features in X. We shall note that, in practice, we cannot really compare $\mathbf{P}(C|G)$ to $\mathbf{P}(C|X)$, since this precise probability distribution is unknown. Indeed, there is only a training set that provides a rough approximation to it. Having this caveat in mind, the definitions of feature (ir)relevance given by [18,21,43] can be formalized as follows. Let S_i be the set of all features except X_i, i.e., $S_i = X - \{X_i\}$. We then denote $S_i = \mathbf{s}_{ji}$ as a shorthand for any value-assignment to all features in S_i for vector $j \in \{1,2,\ldots,N\}$.

Definition 1 (*Strong relevance*). A feature X_i is strongly relevant *iff* for some $j \in \{1,2,\ldots,N\}$ there exists x_{ji} and $\mathbf{s}_{ji}$ such that $\mathbf{P}(C|S_i = \mathbf{s}_{ji}) \neq \mathbf{P}(C|X_i = x_{ji}, S_i = \mathbf{s}_{ji})$, where $C = \{c_1, c_2, \ldots, c_l\}$ is the set of l possible classes.

Definition 2 (*Weak relevance*). A feature X_i is weakly relevant *iff* it is not strongly relevant and for some $j \in \{1,2,\ldots,N\}$ there exists x_{ji} and $\mathbf{s}_{ji}$ such that $\mathbf{P}(C|S_i = \mathbf{s}_{ji}) = \mathbf{P}(C|X_i = x_{ji}, S_i = \mathbf{s}_{ji})$ and $\exists S'_i \subset S_i$, such that $\mathbf{P}(C|S'_i = \mathbf{s}'_{ji}) \neq \mathbf{P}(C|X_i = x_{ji},\ S'_i = \mathbf{s}'_{ji})$.

Features can be deemed relevant if they are either strongly relevant or weakly relevant. As a consequence of the above definitions, irrelevant features can be formally defined as:

Definition 3 (*Irrelevance*). A feature X_i is irrelevant *iff* $\forall S'_i \subseteq S_i$ there is $\mathbf{P}(C|S'_i = \mathbf{s}'_{ji}) = \mathbf{P}(C|X_i = x_{ji}, S'_i = \mathbf{s}'_{ji}),\ \forall j \in \{1,2,\ldots,N\}$.

Roughly speaking, strong relevance states that feature X_i cannot be removed without changing the original conditional class distribution. Weak relevance indicates that X_i is not always necessary to keep $\mathbf{P}(C|G)$ as close as possible to $\mathbf{P}(C|X)$, but it may become necessary sometimes [43]. Finally, irrelevance suggests that X_i is not necessary at all. As observed by Yu and Liu [43], the goal of feature selection is to find a minimum subset G that includes all strongly relevant features, a subset of weakly relevant features, and none of irrelevant features. These authors argue that, among all the weakly relevant features, there are redundant features that can be identified and consequently removed. Feature redundancy is usually defined in terms of feature correlation. As observed by Yu and Liu [43]: *"It is widely accepted that two features are redundant to each other if their values are completely correlated"*. From this observation we formalize feature redundancy as:

Definition 4 (*Redundancy*). Let $\psi(X_i,X_j)$ be an index that captures the correlation between features X_i and X_j. Further, let $\psi(X_i,X_j) \in [\psi_{min}, \psi_{max}]$, where ψ_{min} and ψ_{max} are the minimum and maximum correlation values for such an index, respectively. X_i and X_j are redundant *iff* $\psi(X_i,X_j) = \psi_{\max}$.

Example. Let $X_i = \{x_{1i}, x_{2i}, \ldots, x_{Ni}\}$ and $X_j = \{x_{1j}, x_{2j}, \ldots, x_{Nj}\}$. For the well-known Pearson correlation coefficient, $\rho(X_i,X_j) \in [-1,+1]$, $\psi(X_i,X_j)$ can be conveniently defined as $\psi(X_i,X_j) = |\rho(X_i,X_j)| \in [0,1]$. Since this index measures the degree of linear correspondence between the entries of X_i and X_j, the maximum correlation value is obtained *iff* there exist two real-valued scalars $c \neq 0$ and d such that $x_{ki} = c \cdot x_{kj} + d$ for $k = 1, \ldots, N$ [5]. If this condition holds, X_i and X_j are then deemed *redundant*.

Definition 4 is useful for feature selection methods aimed at finding clusters of correlated features. Indeed, it is implicitly used as an underlying assumption of two state of the art feature selection methods proposed in the literature [29,3], which do not explicitly address feature (ir)relevance and redundancy in formal analyses. Due to this, before going into the details of the proposed method, let us analyze some theoretical properties derived from clustering completely correlated features in the light of the previously addressed definitions. In practice, it is expected that completely correlated features will hardly be found. Actually, redundancy (Definition 4) is much more a matter of degree. Even so, it is interesting to study some theoretical properties that hold under some idealized assumptions, with the purpose of better understanding the expected behavior of feature selection methods based on finding clusters of features.

Definition 5 (*Feature clustering*). Feature clustering is the partitioning of a set X of attributes $X = \{X_1, X_2, \ldots, X_M\}$, into a collection $C^X = \{C_1, C_2, \ldots, C_k\}$ of mutually disjoint subsets of completely correlated[1] features C_i of X, where k is the number of clusters of features, such that $C_1 \cup C_2 \cup \ldots \cup C_k = X$, $C_i \neq \emptyset$, and $C_i \cap C_j = \emptyset$ for $i \neq j$.

Proposition 1. *Strongly relevant features can only be found in singleton clusters.*

Proof (*By contradiction*). For simplicity, let us firstly assume that a given cluster C_s is formed by two strongly relevant features X_i and X_j, i.e. $C_s = \{X_i, X_j\}$. By Definition 5, if X_i and X_j are part of the same cluster, then they are completely correlated. Thus, X_i subsumes the information provided by X_j and vice-versa – i.e., they are redundant according to Definition 4. This contradicts Definition 1, which states that neither X_i nor X_j can be removed without changing the original conditional class distribution $\mathbf{P}(C|X)$. Similarly, let us assume that cluster C_s is formed by r ($r > 2$) strongly relevant features. From Definition 5, such features are completely correlated and, according to Definition 4, they are redundant. Consequently, $(r-1)$ of such features can be removed without changing $\mathbf{P}(C|X)$. This implies that such r features are not strongly relevant, contradicting the premise. □

Proposition 2. *Irrelevant features and weakly relevant features cannot be found in the same cluster.*

Proof (*By contradiction*). Let us assume a given cluster C_r formed by completely correlated features. By hypothesis, consider that $X_i \in C_r$ is irrelevant (Definition 3) and that the remaining features from C_r are weakly relevant (Definition 2). From those weakly relevant features, let us take into account any feature $X_j \in C_r - \{X_i\}$. Following Definition 3, X_i is irrelevant *iff* $\forall S'_i \subseteq S_i$

[1] Clustering methods are usually based on similarity measures, which, in this case, are correlation measures.

we have $\mathbf{P}(C|S'_i = \mathbf{s}'_{ji}) = \mathbf{P}(C|X_i = x_{ji}, S'_i = \mathbf{s}'_{ji}), \forall j \in \{1, 2, \ldots, N\}$. However, provided that X_i and X_j are completely correlated, X_i subsumes the information provided by X_j, and vice-versa. Consequently, the same condition for Definition 3 holds for X_j, thus being in contradiction with the definition of weak relevance (Definition 2). □

Corollary 1. *If there are irrelevant features in X, at least one of them will be included in G.*

Justification: From the above propositions it is possible to see that a given cluster obtained from the process of clustering completely correlated features does not encompass different kinds of features (i.e., strongly relevant, weakly relevant, and irrelevant). Consequently, if there are irrelevant features in X, at least one of them will be included in G (e.g., when some representative feature(s) of each cluster is/are selected for G). □

Corollary 1 suggests a possible further refinement for methods aimed at feature selection from finding clusters of similar features, namely: employing wrappers to select the ultimate feature set to be used for building the classifier. This approach may be particularly appealing in applications for which the datasets contain a significant number of redundant features. Provided that these features are removed, only a small set of features will be subject to further processing, possibly justifying the use of a wrapper without significantly reducing computational efficiency. However, a word of caution is in order. In theory, feature selection methods aimed at finding clusters of correlated features (e.g., [29,3]) can indeed select at least one of the irrelevant features to be used for classification purposes. In practice, however, completely irrelevant features (Definition 3) will hardly be found, and the undesirable property described in Corollary 1 may not strictly hold. Actually, (ir)relevance and redundancy are much more a matter of degree. From this point of view, heuristic methods based on feature clustering frequently provide good results by *approximating* the set of features G from the removal of (partially) redundant features of X.

3. Related work

3.1. Filter proposed by Mitra et al. [29] – MMP

The filter proposed by Mitra, Murthy, and Pal [29] (here named MMP, for short) involves two main steps, namely: (i) the partitioning of the complete feature set into clusters and; (ii) the selection of a representative feature from each cluster. Linear dependency is used to assess similarities between two features and, consequently, to induce clusters. The authors argue that there are several benefits of choosing linear dependency as a feature similarity measure – see [29] for a detailed discussion on this issue. In particular, it is shown [29] that the proposed *Maximal Information Compression Index* – $\lambda(X_i,X_j)$ – in Eq. (1) is a measure of the minimum amount of information loss (or the maximum amount of information compression) possible. Hence, it is a dissimilarity measure that may be suitably used for redundancy reduction. Let X_i and X_j be two random variables (here called features). $\lambda(X_i,X_j)$ is defined as:

$$2\lambda(X_i, X_j) = \xi - \sqrt{\xi^2 - 4\mathrm{var}(X_i)\mathrm{var}(X_j)(1 - \rho(X_i, X_j)^2)}, \tag{1}$$

where $\xi = \mathrm{var}(X_i) + \mathrm{var}(X_j)$, $\mathrm{var}(X_i)$ denotes the variance of X_i, and $\rho(X_i,X_j)$ is the correlation coefficient between X_i and X_j:

$$\rho(X_i, X_j) = \frac{\mathrm{cov}(X_i, X_j)}{\sqrt{\mathrm{var}(X_i)\mathrm{var}(X_j)}}, \tag{2}$$

where $\mathrm{cov}(X_i,X_j)$ denotes the covariance between X_i and X_j. The value of $\lambda(X_i,X_j)$ is 0 (zero) when the features are linearly dependent and increases as the amount of dependency decreases [29].

Clusters of features are obtained via the well-known k-Nearest Neighbors (k-NN) principle [29]. Initially (first iteration of the algorithm), the k nearest neighbors (k_{NN}) of each feature are computed. Among them, the feature that has the most compact cluster is selected, and its k_{NN} neighboring features are discarded. The distance of a given feature to its farthest neighbor measures the *lack of compactness* of a given cluster. The process is repeated for the remaining features, iterating until all of them are classified as either selected or discarded. During the execution of the algorithm, the k_{NN} value is indirectly controlled by a parameter called *constant error threshold*, ε, which is set equal to the distance of the kth nearest-neighbor of the feature selected in the first iteration. In subsequent iterations, the *lack of compactness* value is checked to verify whether it is greater than ε or not. If that is true, the k_{NN} value is decreased. It is important to note that the initial value of k_{NN} is chosen by the user, and it controls the cardinality of the subset of selected features. As claimed by the authors [29], on the one hand it may be useful to control the representation of the data at different levels of details, performing some kind of exploratory data analysis. In this sense, the results reported in [29] show that the filter based on $\lambda(X_i,X_j)$ has superior performance when compared to four related methods (branch and bound [8], sequential floating forward search [33], sequential forward search [8], and stepwise clustering [20]) on nine real-world datasets. On the other hand, the choice of the value of such a parameter may be hard to be accomplished in practice, for the user is left to estimate a critical parameter of the algorithm. The method proposed in this paper is aimed at further improving the algorithm proposed in [29] by making it capable of selecting features in a more automatic fashion. Moreover, the proposed algorithm has been designed to be at least as computationally efficient as the Mitra et al.'s algorithm – whose overall computational complexity for a given value of k_{NN} is estimated in

[29] as $O(M^2N)$, where M and N are the number of features and instances, respectively. For scenarios in which the value of the parameter k_{NN} is unknown, its values can be varied over a given range $[k_{NNmin}, k_{NNmax}]$. In this case, we estimate the computational complexity of the filter as:

$$O\left(M^2 \cdot N + \sum_{k=k_{NN\,min}}^{k_{NN\,max}} (M + (k-1)\cdot(M-k))\right). \tag{3}$$

For instance, making an exploratory data analysis for $k_{NNmin} = 1$ and $k_{NNmax} = M - 1$ leads to a computational cost of $O(M^2N + M^3)$.

3.2. Filter proposed by Au et al. [3] – ACA

Au et al. [3] proposed a filter named *Attribute Clustering Algorithm* (ACA) that uses a non-linear correlation measure to group features. In principle, the proposed correlation measure assumes that all features of the dataset are discrete. In particular, let us assume that $X = \{X_1, X_2, \ldots, X_M\}$ is such a set of discrete features and that $\forall X_i \in X, dom(X_i) = \{v_{i1}, v_{i2}, \ldots, v_{im_i}\}$. The correlation measure used by ACA is the *Interdependence Redundancy Measure*, $R(X_i, X_j)$, defined in [3] as:

$$R(X_i, X_j) = \frac{I(X_i, X_j)}{H(X_i, X_j)}, \tag{4}$$

where $I(X_i, X_j)$ and $H(X_i, X_j)$ are the *mutual information* and the *joint entropy* between the attributes X_i and X_j, respectively:

$$I(X_i, X_j) = \sum_{k=1}^{m_i}\sum_{l=1}^{m_j} P(X_i = v_{ik}, X_j = v_{jl}) \log \frac{P(X_i = v_{ik}, X_j = v_{jl})}{P(X_i = v_{ik})P(X_j = v_{jl})}, \tag{5}$$

$$H(X_i, X_j) = -\sum_{k=1}^{m_i}\sum_{l=1}^{m_j} P(X_i = v_{ik}, X_j = v_{jl}) \log P(X_i = v_{ik}, X_j = v_{jl}). \tag{6}$$

Eq. (4) reflects the dependence between X_i and X_j. The denominator is basically a normalization term aimed at avoiding a *bias* towards features with a large number of values. $R(X_i, X_j) \in [0, 1]$, where 1 (one) indicates full dependence (correlation) between features and 0 (zero) the opposite [3].

The authors in [3] also introduced the concept of a *mode feature*, which is the feature most correlated with the other features of a given group. More precisely, the *mode* of a group C_r is computed as:

$$\eta_r = \arg\max_{X_i \in C_r} \sum_{X_j \in C_r} R(X_i, X_j). \tag{7}$$

For clustering features, they use a *variant* of k-means called k-modes. The main differences between them are: (i) the replacement of the centroid by the *mode feature*; (ii) the replacement of the Euclidean distance by the *Interdependence Redundancy Measure* – $R(X_i, X_j)$ – in Eq. (4). The k-modes algorithm terminates when the *mode features* of two consecutive iterations are equal. In other words, k-modes can be seen as a variant of the well-known k-medoids algorithm, used in ACA with a specific similarity measure – $R(X_i, X_j)$ – instead of the widely used Euclidean distance.

In principle, the clustering algorithm used by ACA needs the definition of k *a priori*, but the authors [3] suggest that the sum of the multiple significant interdependence redundancy measure in (8) can indicate the best value for k.

$$k = \arg\max_{k \in \{k_{\min}, \ldots, k_{\max}\}} \sum_{r=1}^{k} \sum_{X_i \in C_r - \{\eta_r\}} R(X_i, \eta_r). \tag{8}$$

After partitioning the features into distinct clusters, the authors propose to select, from each cluster, the r features with the highest correlation with the other features of the cluster, being r a user-defined value. Although this approach may sound persuasive at a first glance, note that it favors the selection of redundant features. For the sake of illustration, let us assume that ACA has found a cluster formed by three features $\{X_1, X_2, X_3\}$, for which $R(X_1, X_2) = 0.9$, $R(X_1, X_3) = 0.5$, and $R(X_2, X_3) = 0.4$. Then, provided that ACA selects the r features most correlated with the other features of the cluster, for $r = 2$ it would select X_1 and X_2, which are highly correlated, and consequently highly redundant. Aimed at circumventing this potential problem, we propose a different approach in our filter (to be described in the next section).

In what concerns computational efficiency issues, the computational complexity of ACA is estimated in [3] as $O(kNM^2t)$, where t is the number of k-modes iterations and k is the given number of clusters. Let us now consider a scenario in which k is unknown. In this case, as suggested in [3] one can run ACA for all $k \in \{k_{min}, k_{min} + 1, \ldots, k_{max} - 1, k_{max}\}$. By doing so, and assuming that each cluster will have approximately M/k features, we estimate the overall computational complexity of ACA as:

$$O\Big((k_{\min} + k_{\max})(k_{\max} - k_{\min}) \cdot M + M^2 \cdot \ln(k_{\max} - k_{\min}) + (k_{\max} - k_{\min}) \cdot M\Big). \tag{9}$$

For instance, making an exploratory data analysis for $k_{min} = 2$ and $k_{max} = M - 1$ leads to a computational cost of $O(M^3)$. Finally, if one considers the commonly used rule of thumb for cluster analysis for which $k_{min} = 2$ and $k_{max} = M^{1/2}$ then the computa-

1. Generate a random initial partition of features into k nonempty clusters;
2. Compute the cluster medoids – using Equation (11) – of the current partition;
3. Assign each feature to the cluster with the nearest medoid;
4. If the medoids from two consecutive iterations are equal, then stop; else, go to Step 2.

Fig. 1. k-medoids algorithm.

tional complexity of ACA is estimated as $O(M^2 lnM)$. For evaluating correlations between features, the computation of the entropy has $O(N)$ complexity when the popular PKID algorithm [40] is used for discretization (as done in this work).

4. Simplified silhouette filter (SSF)

Recalling Definition 5 (Section 2), after partitioning a set of features $X = \{X_1, X_2, \ldots, X_M\}$ into a collection $C^X = \{C_1, C_2, \ldots, C_k\}$ of k mutually disjoint subsets of correlated features C_i of X, it is expected that features that belong to the same cluster should be more similar (correlated) to each other than features that belong to different clusters. Therefore, it is necessary to devise means of evaluating similarities (correlations) between feature sets. This problem can also be tackled indirectly, i.e. distance measures can be used to quantify dissimilarities (*lack of correlation*) between features. To facilitate the comparisons between the filters analyzed in this work, we employ in SSF the correlation measures used by the filters described in Section 3, namely: the *Maximal Information Compression Index* in Eq. (1) and the *Interdependence Redundancy Measure* in Eq. (4). For convenience, we define a dissimilarity measure, $f(c(X_i,X_j))$, as a function of a particular correlation measure – $c(X_i,X_j)$. For example, $f(\lambda(X_i,X_j)) = \lambda(X_i,X_j)$ and $f(R(X_i,X_j)) = 1 - R(X_i,X_j)$.

Clustering problems are usually classified as NP-hard [12] and attempting to find a globally optimum solution is usually not computationally feasible [1]. This difficulty has stimulated the search for efficient approximate algorithms. This work follows this trend, employing a heuristic procedure, which is based on the simplified silhouette criterion [16,15], for finding the number of clusters and the corresponding feature partitions.

Before defining the simplified silhouette [16,15], let us introduce the silhouette proposed in [19]. Let $d(X_i,C_j)$ be the average dissimilarity of a feature X_i to all other features of the cluster C_j according to a correlation measure $c(X_i,X_j)$, i.e., $d(X_i, C_j) = \frac{1}{|C_j - \{X_i\}|} \sum_{X_u \in C_j - \{X_i\}} f(c(X_i, X_u))$. Consider that X_i belongs to cluster C_a. The average dissimilarity of X_i to all other features of C_a is denoted by $a(i)$, i.e., $a(i) = d(X_i,C_a)$. After computing $d(X_i,C_j)$ for all clusters $C_j \neq C_a$, the smallest one is selected, i.e. $b(i) = \min d(X_i,C_j)$, $C_j \neq C_a$. This value represents the dissimilarity of X_i to its neighbor cluster, and $s(i)$ is given by [19]:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}. \tag{10}$$

The higher $s(i)$, the better the assignment of feature X_i to a given cluster. In addition, if $s(i) = 0$ then it is not clear whether the feature should have been assigned to its current cluster or to a neighboring one [11]. Finally, if C_a is a singleton, then $s(i)$ is not defined and the most neutral choice is to set $s(i) = 0$ [19]. The average of $s(i)$, $i = 1, 2, \ldots, M$, can be used as a criterion to assess the quality of a given feature partition. Doing so, the best clustering is achieved when the silhouette value is maximized.

The silhouette proposed in [19] depends on the computation of all distances between features, leading to an estimated computational cost of $O(M^2N)$. To alleviate this, a simplified silhouette [16,15] can be used. The simplified silhouette (SS) [16,15] is based on the computation of distances between features and cluster centroids (in the present work medoids are used instead of centroids), i.e., $d(X_i,C_j)$ is calculated as $d(X_i,C_j) = f(c(X_i,\eta_j))$, being η_j the medoid of the cluster[2]:

$$\eta_j = \arg\min_{X_v \in C_j} \sum_{X_u \in C_j} f(c(X_v, X_u)). \tag{11}$$

In other words, the term $a(i)$ of Eq. (10) becomes the dissimilarity of feature X_i to its corresponding cluster medoid (η_a). Similarly, instead of computing $d(X_i,C_j)$ as the average dissimilarity of X_i to all features of $C_j \neq C_a$, distances between X_i and the medoid of C_j are computed. These modifications reduce the computational cost from $O(M^2N)$ to $O(kMN)$, where k is usually significantly less than M.

The computation of the original silhouette [19], as well as of its simplified version [16,15], depends only on the achieved partition and not on the adopted clustering algorithm. Thus, these silhouettes can be applied to assess partitions (taking into account the number of clusters) obtained by several clustering algorithms. We adopt the k-medoids algorithm (Fig. 1) to obtain partitions to be evaluated by the simplified silhouette (SS). In brief, k-medoids is designed to minimize the sum of distances between features and nearest medoids. From the SS criterion viewpoint, good partitions are also obtained when this minimization is suitably performed, as well as when clusters are well separated. In other words, although other clustering algorithms could be used, our approach favors a synergy between k-medoids and the SS criterion.

[2] Note that when the *interdependence redundancy measure* (R) is used, Eqs. (11) and (7) are equivalent.

1. Choose the minimum and the maximum number of clusters in a set of values for k, respectively k_{min} and k_{max}, as well as a number of different initial partitions (n_p) for k-medoids (Fig. 1);
2. SSV← -∞; / SSV = Simplified Silhouette Value /
3. For each $k \in \{k_{min}, k_{min}+1, \ldots, k_{max}-1, k_{max}\}$ do:
   3.1 Generate n_p random initial partitions of features into k nonempty clusters;
   3.2 Run k-medoids for each initial partition generated in Step 3.1 and compute its corresponding simplified silhouette. Let the best obtained value be BOV;
   3.3 If (BOV > SSV) then {
       SSV←BOV;
       $k^* \leftarrow k$;
       Hold the corresponding partition for k^*. }
4. Return SSV and its corresponding feature clusters for k^*.

Fig. 2. Sampling strategy for k-medoids.

The SS is a numeric criterion that allows estimating the number of clusters automatically. Thus, it can provide a way of circumventing an important limitation of k-medoids (Fig. 1), namely: the number of clusters k must be determined *a priori*. In this sense, one can perform multiple runs of k-medoids (for different k values) and then choose the best available partition, which corresponds to the maximum achieved value for the silhouette. It is also well-known that k-medoids may get stuck at suboptimal solutions for a given k [44]. To alleviate this problem, one can perform multiple runs of k-medoids for a fixed k. The sampling strategy depicted in Fig. 2 summarizes the method used to estimate the number of clusters k^* according to the SS, as well as to find the corresponding clusters of features. The parameters k_{min} and k_{max} define the range of possible values for k, i.e., k is varied over the range $[k_{min}, k_{max}]$. The parameter n_p refers to the number of different random initial partitions for each k, and it is usually chosen in an ad hoc fashion, depending both on the expected complexity of the clustering problem to be solved and on the computational resources available.

We propose two alternatives for selecting features from each cluster. The first one involves selecting only the feature most correlated to the other features in the same cluster according to (11). In this case, the selected feature is the medoid of the cluster and it can be viewed as its representative feature. After selecting the medoids, a subset of k^* features is achieved. This approach is particularly useful if the clusters are well separated. In the second approach, besides selecting the medoid we select the feature least correlated (least redundant) with the medoid. Thus, two features from each cluster are chosen, resulting in the selection of $2 \cdot k^*$ features. Contrarily to ACA [3], which selects the r features most correlated to the others in the same cluster, our approach tends to avoid the selection of undesirable redundant features, being more interesting when one faces with overlapping clusters. In this case, our approach can be more suitable than ACA [3] for avoiding information loss, in the sense that the feature least correlated to the medoid would not be discarded by our filter. This feature may contain important information not encoded into the medoid, but it would be discarded in most of the cases by ACA.

Provided that our filter is based on the Simplified Silhouette (SS) [16,15] addressed above, we will call it *Simplified Silhouette Filter* (SSF). Both ACA and SSF are based on the k-medoids algorithm. Therefore, their overall computational complexities are similar. Specifically, the overall computational complexity of SSF is estimated as:

$$O((k_{\min} + k_{\max})(k_{\max} - k_{\min}) \cdot M + M^2 \cdot \ln(k_{\max} - k_{\min})), \tag{12}$$

where $k_{min}$and k_{max} are the minimum and maximum number of clusters from the set $\{k_{min}, k_{min}+1, \ldots, k_{max}-1, k_{max}\}$, respectively. Following the development presented in Section 3.2, if we run SSF for every $k \in \{2, \ldots, M-1\}$, i.e., if no domain knowledge is available, then its overall computational cost is estimated as $O(M^3)$. Two particular cases deserve further attention. The first one involves scenarios for which the user somehow suspects that the number of clusters is low - e.g., $k_{max} \approx 2$. In this case, it is easy to see that the time complexity of SSF is estimated as $O(M^2)$. The second case involves scenarios in which domain knowledge suggests that the number of clusters of features is high (e.g., $k_{max} \approx M^{1/2}$). In this case, the user would run SSF for the number of clusters around $k_{max} = M^{1/2}$. This would also make the computational cost of SSF be estimated as $O(M^2 \ln M)$. In what concerns the computational complexity of SSF in terms of the number of instances (N), it will depend upon the adopted correlation measure. As discussed previously in Sections 3.1 and 3.2, the correlation measures used in this work have a computational complexity of $O(N)$.

4.1. Supervised Simplified Silhouette Filter (S^3F)

The Simplified Silhouette Filter (SSF) originally only takes into consideration correlations between *independent variables*. In classification problems, information on the class (a *dependent variable*) is available and, as so, correlations between attributes and the class can be useful for the feature selection process. Under this perspective, we propose three modifications on

SSF in order to incorporate attribute-class correlations. As the result of these modifications, we obtain a supervised variant of SSF, here named S^3F (from Supervised Simplified Silhouette Filter).

The first modification involves incorporating attribute-class correlations in the distance measure used to assess dissimilarities between features. As in Section 3.2, we will assume that all features of the dataset are discrete. Before describing the adopted correlation measure, we need to introduce three concepts. Firstly, the concept of the *entropy* of a feature X_i – captured by Eq. (13) – which measures the uncertainty associated with a random variable. Secondly, the conditional entropy, which is the remaining entropy of a random variable X_i given that the value of another random variable X_j is known – Eq. (14). Finally, the amount by which the entropy of X_i decreases after observing X_j reflects additional information about X_i, provided by X_j, and it is called the *information gain* [35] – Eq. (15).

$$H(X_i) = -\sum_{k=1}^{m_i} P(X_i = v_{ik}) \log P(X_i = v_{ik}), \tag{13}$$

$$H(X_i|X_j) = -\sum_{l=1}^{m_j} P(X_j = v_{jl}) \sum_{k=1}^{m_i} P(X_i = v_{ik}|X_j = v_{jl}) \log P(X_i = v_{ik}|X_j = v_{jl}), \tag{14}$$

$$IG(X_i|X_j) = H(X_i) - H(X_i|X_j). \tag{15}$$

Given these definitions, the correlation measure used by S^3F is known as *Symmetrical Uncertainty* (SU) [32] – Eq. (16). Such a correlation measure is widely used by feature selection algorithms (e.g., [14,43]). It can be viewed as a normalized version of the information gain that compensates its bias towards features with more values. The value of SU is 0 (zero) when the features are independent and 1 (one) if they are completely correlated.

$$SU(X_i, X_j) = 2\left[\frac{IG(X_i|X_j)}{H(X_i) + H(X_j)}\right]. \tag{16}$$

The dissimilarity measure between two features, considering their correlations with the class, can then be defined as:

$$SU_s(X_i, X_j) = \frac{1 - SU(X_i, X_j) + |SU(X_i, C) - SU(X_j, C)|}{2}. \tag{17}$$

The measure in Eq. (17) is normalized in the range $[0, 1]$, being 0 (zero) when the features are completely correlated and have the same information (correlation) with respect to the class, and 1 (one) when the features are independent and, at the same time, one of them is independent to the class and the other is completely correlated with the class.

The second modification is related to the medoid concept. SSF has been adapted so that the medoid is now the feature most correlated with the other features of the cluster and the most correlated with the class as well – see Eq. (18).

$$\eta_r = \arg\max_{X_i \in C_r} \left\{ \frac{1}{2} \left[\frac{\sum_{X_j \in C_r} SU(X_i, X_j)}{|C_r| - 1} + SU(X_i, C) \right] \right\}. \tag{18}$$

Finally, the third modification is in the approach used to select the second feature for each cluster, now taking the class information into account. More precisely, the feature least correlated with the medoid and most correlated with the class is selected – according to Equation (19).

$$\arg\max_{X_i \in C_r} \left\{ \frac{1 - SU(X_i, \eta_r) + SU(X_i, C)}{2} \right\}. \tag{19}$$

5. Empirical evaluation

Ten datasets were used to assess the performance of the proposed filters. Six of them are bioinformatics datasets used by Yeung et al. [42], who created five types of synthetic datasets with error distributions derived from real-world bioinformatics data. These datasets are available in [41] (low noise data) and are here called Bio1, Bio2, Bio3, Bio4, and Bio5. They are composed of 400 genes (instances) described by 20 measurements (features). There are six approximately equal-sized classes in each dataset. In addition, we used a real-world dataset (Yeast Galactose data [41,42]) composed of 20 measurements and 205 genes. In this dataset, the expression patterns reflect four functional categories. All these bioinformatics datasets consider four repeated measurements, what may yield more accurate and stable classes [42]. The repeated measurements are taken into account by averaging their expression levels. We also employed the real-world *Ovarian Cancer* dataset [31], which has 15,154 features and 253 instances, whose classes correspond to either cancer or normal samples. The other real-world datasets used in our study are widely known and available at the UCI Machine Learning Repository [2], namely:

- Ionosphere: This classic dataset is related to a problem of determining whether a signal received by radar is "good", which means that the signal contains potentially useful information about the ionosphere [34].
- Wisconsin Breast Cancer: This breast cancer database was obtained from the University of Wisconsin Hospitals, where samples arrive periodically as doctors report clinical cases. The database therefore reflects this chronological grouping of the data [2].

Table 1
Summary of the datasets used in the experiments (N and M are the number of instances and features, respectively).

Dataset	Identifier	N	M	# Classes (distributions – %)
Bio{1,...,5}	Bio{1,...,5}	400	20	6 (≈ equally distributed)
Yeast	Yeast	205	20	4 (40.5 – 7.3 – 45.4 – 6.8)
Ionosphere	Iono	351	34	2 (35.9 – 64.1)
Breast Cancer	Wisc	683	9	2 (65.0 – 35.0)
Spambase	Spam	4,601	57	2 (39.4 – 60.6)
Ovarian Cancer	Ova	253	15,154	2 (64.0 – 36.0)

- Spambase: In this problem domain the task is to determine whether a particular e-mail message is an advertisement that the receiver would never want to read [34].

A summary of these datasets is provided in Table 1. The column "identifier" describes the name by which each dataset will be referred to, N is the number of instances, and M is the number of features. We set $k_{min} = 2$ and $n_p = 20$ for clustering features by ACA (Section 3.2), SSF (Section 4), and S^{3}F (Section 4.1). The parameter k_{max} was set according to the criteria for selecting features from each cluster. More specifically, $k_{max} = M - 1$ has been employed for selecting one feature per cluster and $k_{max} = M/2$ for selecting two features per cluster (as discussed in Section 4). From a practical viewpoint, one can consider that these values determine the size of the search space to be assessed and, accordingly, the computational effort to find the corresponding solution. Domain knowledge is not necessary to set such values but, when available, it can be incorporated into this approach in order to set those parameters – like, for instance, when the user knows the number of clusters *a priori*. Finally, continuous features were discretized by the PKID algorithm [40] before computing the *Interdependence Redundancy Measure* – Eq. (4) – and the *Symmetrical Uncertainty*– Eq. (16).

The quality of each feature subset found by a given filter has been assessed by (i) the generalization capability of two classifiers widely used in practice [38] – k-Nearest Neighbors (NN) and Naïve Bayes (NB)[3] – and (ii) by the amount of redundancy in the subset. For analyzing the amount of redundancy of feature subsets, we used the *representation entropy measure* [8]:

$$H = -\sum_{i=1}^{d} \omega_i \log \omega_i, \tag{20}$$

where ω_i, $i = 1, \ldots, d$, is the ith normalized eigenvalue of the covariance matrix of a feature subset of size d. In Eq. (20), $H = 0$ is obtained when all the eigenvalues, except one, are zero. The maximum value is achieved when the eigenvalues are equal, i.e., when the information is equally distributed among all features [29].

We have followed an established methodology [34] to compare feature selection algorithms. This methodology is based on a cross-validation process (as usual, we have chosen 10 folds), where feature selection is performed by using only the training folds and the classification accuracy is estimated in the test folds. The same training/test folds were used for all algorithms. In order to provide some reassurance about the validity and non-randomness of the obtained results, we present the results of statistical tests following the study of Demšar [9], which includes comparing multiple algorithms on multiple datasets by using the Friedman test with a corresponding post-hoc test. The Friedman test is a non-parametric statistic test equivalent to the repeated-measures ANOVA. If the null hypothesis, which states that the algorithms under study have similar performances, is rejected, then we proceed with the Nemenyi post-hoc test for pairwise comparisons between algorithms. For comparing two algorithms on multiple datasets, Demšar [9] recommends the Wilcoxon Signed-Rank test.

The analyses of the obtained results were divided into three sections. In Section 5.1 we compare SSF (Section 4) with MMP (Section 3.1), whereas in Section 5.2 SSF is compared to ACA (Section 3.2). The correlation measures originally proposed by the authors of MMP and ACA – namely, the *Maximal Information Compression Index* (λ) in Eq. (1) and the *Interdependence Redundancy Measure* (R) in Eq. (4), respectively – have been used. In Section 5.3, we compare SSF to S^{3}F (Section 4.1).

Finally, considering computational efficiency issues, we are interested in empirically evaluating the magnitude of the constant terms - neglected by the asymptotic time complexity analysis performed in Sections 3.1, 3.2 and 4. To that end, all algorithms under study were implemented in Java[4], using only the necessary commands. This way, more uniform efficiency comparisons can be performed. The same computer (Opteron, 2.0 GHz, 8 Gb RAM), running only the operational system, was used for all the controlled experiments. Running times of the assessed algorithms are presented in Table 13.

5.1. Comparisons between SSF and MMP

As observed in Section 3.1, the performance of MMP is highly dependent upon a parameter, k_{NN}, chosen by the user and that controls the cardinality of the subset of selected features. To make this point clearer, we run MMP [29] by varying k_{NN} over the range of all possible values – $\{1, \ldots, M-1\}$. Due to space restrictions, we report only the best and the average results obtained by MMP [29], but we shall note that such results are particularly useful to assess the difficulty faced by the user in

[3] Available in the Weka System [37]. In our experiments, their default parameters have been used.
[4] A package for the WEKA System [37] with the implemetation of SSF and S^{3}F is available at http://www.icmc.usp.br/~tcovoes/SSF_Package.zip.

Table 2
Summary of the classification error rates – E(%) – obtained by NN and NB classifiers.

Dataset	SSF-λ-1		SSF-λ-2		MMP-B		MMP-A		All	
	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB
Bio1	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	1.4 (0.4)	1.0 (0.4)	0.0 (0.0)	0.0 (0.0)
Bio2	6.0 (3.0)	5.5 (3.6)	1.0 (1.6)	1.2 (1.2)	0.2 (0.7)	0.0 (0.0)	5.7 (1.4)	4.9 (1.1)	1.0 (1.6)	1.2 (1.2)
Bio3	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	2.8 (0.6)	2.2 (0.4)	0.0 (0.0)	0.0 (0.0)
Bio4	5.5 (3.5)	5.2 (3.6)	4.0 (2.7)	4.0 (3.7)	0.5 (1.0)	0.7 (1.6)	6.2 (1.7)	5.1 (1.6)	2.5 (2.5)	2.5 (2.9)
Bio5	0.0 (0.0)	0.2 (0.7)	0.0 (0.0)	0.2 (0.7)	0.0 (0.0)	0.0 (0.0)	2.1 (0.5)	1.8 (0.6)	0.0 (0.0)	0.2 (0.7)
Yeast	10.7 (8.2)	9.2 (8.4)	8.8 (5.2)	7.2 (7.2)	1.4 (2.2)	1.4 (2.2)	8.5 (2.9)	7.1 (3.7)	1.9 (2.3)	2.4 (3.3)
Iono	30.5 (6.3)	23.0 (5.1)	22.7 (9.2)	24.4 (4.2)	7.1 (4.4)	13.6 (2.7)	13.5 (4.5)	22.3 (4.0)	13.6 (6.7)	16.7 (3.8)
Wisc	7.3 (1.7)	6.1 (2.4)	4.3 (2.5)	3.9 (1.7)	2.6 (1.7)	2.4 (1.9)	6.9 (1.7)	6.9 (1.2)	4.2 (1.6)	3.6 (1.7)
Spam	37.1 (1.0)	53.4 (1.2)	32.7 (3.6)	51.6 (1.6)	8.3 (0.9)	19.8 (1.6)	16.7 (0.8)	33.3 (1.5)	9.1 (1.2)	20.2 (1.8)

Table 3
Summary of the number of selected features (#FS) and representation entropy values (H).

Dataset	SSF-λ-1		SSF-λ-2		MMP-B (NN)		MMP-B (NB)	
	#FS	H	#FS	H	#FS	H	#FS	H
Bio1	9.5 (1.1)	1.0 (0.0)	17.2 (4.1)	1.1 (0.0)	2.7 (0.4)	0.8 (0.1)	2.5 (0.5)	0.8 (0.1)
Bio2	9.8 (0.4)	1.0 (0.0)	19.6 (0.8)	1.1 (0.0)	3.1 (1.0)	0.8 (0.2)	3.7 (2.1)	0.9 (0.1)
Bio3	8.6 (0.4)	1.0 (0.0)	17.2 (0.9)	1.1 (0.0)	2.2 (0.4)	0.7 (0.1)	2.2 (0.4)	0.7 (0.0)
Bio4	5.4 (2.1)	1.1 (0.1)	10.8 (4.3)	1.2 (0.0)	6.9 (3.4)	1.0 (0.1)	5.8 (3.5)	1.0 (0.1)
Bio5	8.0 (0.0)	1.0 (0.0)	16.0 (0.0)	1.0 (0.0)	2.4 (0.8)	0.6 (0.1)	2.4 (0.8)	0.6 (0.1)
Yeast	2.0 (0.0)	0.3 (0.1)	3.0 (0.0)	0.6 (0.2)	6.4 (2.9)	0.6 (0.2)	5.0 (1.7)	0.6 (0.3)
Iono	2.0 (0.0)	0.6 (0.0)	3.4 (0.4)	1.0 (0.1)	11.2 (6.1)	2.0 (0.4)	13.7 (12.63)	1.7 (0.8)
Wisc	2.6 (0.4)	0.6 (0.1)	5.1 (1.1)	1.0 (0.1)	4.6 (0.6)	1.0 (0.1)	4.2 (1.4)	0.9 (0.2)
Spam	2.0 (0.0)	0.0 (0.0)	3.0 (0.0)	0.4 (0.1)	52.9 (3.2)	0.3 (0.0)	53.5 (3.5)	0.3 (0.0)

setting the k_{NN} value. In addition, such results favor the illustration of practical scenarios of particular interest, in which either MMP [29] or SSF may be preferred.

Table 2 reports the average classification error rates obtained in a 10-fold cross-validation process (standard deviations appear within parentheses). The acronyms SSF-λ, MMP, and *All* refer to the *Simplified Silhouette Filter* (SSF) using the *Maximal Information Compression Index*(Eq. (1)), Mitra et al.'s algorithm [29] (for which B and A stand for the best and average results, respectively), and finally the results found by using all features. A third parameter can also be found for SSF, making reference to the number of selected features from each cluster. For instance, SSF-λ-1 stands for SSF-λ selecting one feature from each cluster. Also, E(%)-NN and E(%)-NB refer to the average error rates for the k-Nearest Neighbor and Naïve Bayes classifiers, respectively. Table 3 reports the average number of features selected, denoted by #FS, and the corresponding values of the *representation entropy measure* – H in Eq. (20). The Ovarian Cancer dataset was not used in these experiments due to the computational restrictions to run MMP in such a large dataset.

Considering the average error rates – E(%) – obtained for every scenario formed by a pair of <dataset, classifier> in Table 2, SSF has shown better or equal results by selecting two features in 95% of the scenarios when compared to the selection of a single feature from each cluster. MMP has in general obtained the best results when the k_{NN} parameter was optimized by running MMP over all $k_{NN} \in \{1,\ldots,M-1\}$ and testing all the resulting feature subsets by the classifiers employed in our study (see results for MMP-B). Obviously this approach typically incurs in a significant computational burden. Considering MMP-A, i.e., the MMP average performance over $k_{NN} \in \{1,\ldots,M-1\}$, SSF-$\lambda$-2 presented better accuracies in 67% of the scenarios. The adopted statistical procedure (Friedman test) suggests that we can reject the null hypothesis (at $\alpha = 10\%$) of equal error rates – considering the results obtained by MMP-B, SSF-λ-*, and all features (for both NB and NN). Significant differences in pairwise comparisons have not been observed only between MMP-B and SSF-λ-2 for NN. Therefore, by taking into account the computational efficiency (Table 13), the accuracies achieved by SSF can be considered very good, in the sense that they are competitive with a much more computationally demanding exploratory data analysis approach performed by MMP, which was significantly slower ($\alpha = 10\%$) than SSF selecting two features from each cluster when NN is the baseline classifier. We have not found significant statistical difference (at $\alpha = 10\%$) for the number of selected features by each filter (Table 3), except that SSF-λ-1 has selected less features than SSF-λ-2 (as expected). By taking the *representation entropy measure*– H in Eq. (20) – into account, our results (Table 3) allow concluding that SSF-λ-2 has shown significantly better results than MMP-B (NB) ($\alpha = 10\%$). Considering all these experimental evidences, SSF λ-2 can be preferred over its counterpart when the k_{NN} value is unknown and when classifiers significantly more computationally demanding than NB are employed.

The comparisons just reported are interesting under the perspective of the use of SSF in practical applications, especially when the number of features to be selected is *a priori* unknown. However, it can be of interest to assess SSF and MMP when they select (approximately) the same number of features. Under this perspective, an additional evaluation was performed in order to compare SSF-λ-2 (the SSF variant that obtained the best results) to MMP. To that end, for every fold of the cross-

Table 4
Classification error rates (E(%)) and representation entropy values (H) obtained by SSF and MMP selecting (approximately) the same number of features.

Dataset	SSF-λ-2			MMP		
	E(%)-NN	E(%)-NB	H	E(%)-NN	E(%)-NB	H
Bio1	0.0 (0.0)	0.0 (0.0)	1.1 (0.0)	0.0 (0.0)	0.0 (0.0)	1.1 (0.0)
Bio2	1.0 (1.6)	1.2 (1.2)	1.1 (0.0)	0.8 (1.2)	1.0 (1.3)	1.1 (0.0)
Bio3	0.0 (0.0)	0.0 (0.0)	1.1 (0.0)	0.0 (0.0)	0.0 (0.0)	1.1 (0.0)
Bio4	4.0 (2.7)	4.0 (3.7)	1.2 (0.0)	1.8 (1.7)	2.8 (2.8)	1.1 (0.0)
Bio5	0.0 (0.0)	0.2 (0.7)	1.0 (0.0)	0.0 (0.0)	0.3 (0.8)	1.0 (0.0)
Yeast	8.8 (5.2)	7.2 (7.2)	0.6 (0.2)	9.4 (5.1)	7.8 (4.2)	0.3 (0.1)
Iono	22.7 (9.2)	24.4 (4.2)	1.0 (0.1)	20.5 (7.3)	20.2 (6.7)	1.1 (0.2)
Wisc	4.3 (2.5)	3.9 (1.7)	1.0 (0.1)	4.5 (2.5)	4.1 (2.1)	1.0 (0.2)
Spam	32.7 (3.6)	51.6 (1.6)	0.4 (0.1)	31.3 (2.8)	51.3 (8.5)	0.5 (0.2)

Table 5
Classification error rates (E(%)) and representation entropy values (H) obtained by SSF and MMP for different proportions of selected features.

Algorithms	%F	Criteria	Datasets								
			Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	Iono	Wisc	Spam
SSF-λ-2	25%	E(%)-NN	0.0 (0.0)	3.5 (3.6)	0.0 (0.0)	4.0 (2.7)	0.0 (0.0)	6.4 (4.7)	15.1 (6.9)	4.1 (2.4)	17.0 (1.8)
		E(%)-NB	0.0 (0.0)	2.3 (1.8)	0.0 (0.0)	7.0 (4.5)	0.0 (0.0)	7.3 (4.7)	26.2 (8.1)	3.4 (1.7)	34.4 (10.9)
		H	1.0 (0.0)	1.0 (0.0)	1.1 (0.1)	1.1 (0.0)	1.0 (0.0)	0.8 (0.1)	1.6 (0.1)	0.8 (0.0)	0.7 (0.2)
	50%	E(%)-NN	0.0 (0.0)	3.0 (2.3)	0.0 (0.0)	3.0 (2.3)	0.0 (0.0)	4.5 (6.0)	11.1 (6.1)	4.2 (2.5)	13.2 (2.5)
		E(%)-NB	0.0 (0.0)	1.5 (1.3)	0.0 (0.0)	3.5 (3.2)	0.3 (0.8)	5.0 (5.7)	23.1 (10.0)	4.7 (1.9)	41.9 (7.3)
		H	1.1 (0.0)	1.0 (0.0)	1.0 (0.0)	1.2 (0.0)	1.1 (0.0)	0.6 (0.1)	1.9 (0.2)	1.0 (0.1)	0.4 (0.8)
	75%	E(%)-NN	0.0 (0.0)	2.3 (1.8)	0.0 (0.0)	3.8 (2.4)	0.0 (0.0)	1.9 (2.5)	11.7 (6.5)	4.3 (2.2)	11.4 (2.0)
		E(%)-NB	0.0 (0.0)	1.3 (1.8)	0.0 (0.0)	3.0 (3.1)	0.3 (0.8)	3.5 (4.1)	21.7 (8.3)	4.0 (2.5)	33.8 (3.1)
		H	1.1 (0.0)	1.1 (0.0)	1.1 (0.0)	1.2 (0.0)	1.0 (0.0)	0.8 (0.1)	2.2 (0.1)	1.1 (0.0)	0.7 (1.2)
MMP	25%	E(%)-NN	5.3 (2.5)	1.5 (1.3)	1.5 (1.7)	5.8 (3.1)	1.5 (1.7)	5.4 (4.9)	12.3 (5.7)	6.0 (3.1)	20.2 (2.4)
		E(%)-NB	3.0 (3.3)	1.3 (1.8)	1.3 (1.3)	4.0 (2.9)	0.8 (1.7)	4.4 (5.0)	22.2 (4.6)	4.5 (2.4)	41.7 (3.8)
		H	0.7 (0.0)	0.8 (0.0)	0.9 (0.1)	0.9 (0.0)	0.6 (0.0)	0.5 (0.1)	1.9 (0.1)	0.7 (0.0)	1.2 (0.9)
	50%	E(%)-NN	0.3 (0.8)	5.0 (5.0)	0.0 (0.0)	8.5 (4.6)	0.3 (0.8)	4.4 (4.4)	12.2 (5.4)	3.1 (1.8)	14.3 (1.3)
		E(%)-NB	0.5 (1.1)	4.0 (3.8)	0.0 (0.0)	7.8 (3.8)	0.5 (1.1)	3.5 (4.7)	25.1 (5.3)	3.9 (2.4)	33.1 (3.1)
		H	0.9 (0.0)	0.8 (0.0)	1.1 (0.0)	1.0 (0.0)	1.0 (0.0)	0.7 (0.1)	2.3 (0.0)	1.0 (0.0)	0.1 (0.1)
	75%	E(%)-NN	0.0 (0.0)	3.5 (3.4)	0.0 (0.0)	3.5 (2.7)	0.0 (0.0)	1.5 (2.3)	11.7 (5.1)	4.7 (2.4)	12.2 (1.2)
		E(%)-NB	0.0 (0.0)	3.3 (2.9)	0.0 (0.0)	2.5 (2.0)	0.0 (0.0)	2.5 (3.5)	22.8 (6.4)	3.5 (1.8)	22.5 (2.3)
		H	1.2 (0.0)	1.1 (0.0)	1.1 (0.0)	1.2 (0.0)	1.2 (0.0)	0.9 (0.0)	2.6 (0.0)	1.2 (0.0)	0.3 (0.0)

validation process, we selected, among the corresponding feature subsets provided by MMP (please note that there is one subset for each k_{NN} value), the one with the most similar number of features when compared to the subset selected by SSF-λ-2. Table 4 presents the obtained average error rates. The Wilcoxon Signed-Rank Test [9] shows that significant statistical differences ($\alpha = 10\%$) were not observed, i.e., both filters have shown equal accuracies. The same holds with respect to the representation entropy measure.

We have also analyzed the performance of each algorithm when the number of features to be selected is fixed in advance. To that end, three different proportions (25%, 50%, and 75%) of the number of features have been considered. Then, for MMP, we have used an approximation, $d + k_{NN} \approx M$, where d is the number of selected features and M is the number of features in the dataset [29]. Thus, the value for the parameter k_{NN} is equal to $M - \lceil M^* pct \rceil$, where $\lceil\ \rceil$ denotes the *ceiling* operation, and $pct \in \{0.25, 0.5, 0.75\}$. For SSF-$\lambda$-2, the value of k (number of clusters) is $\lceil (\lceil M^* pct \rceil)/2 \rceil$ and $k_{min} = k_{max} = k$. Table 5 shows the obtained results, where %F denotes the percentage of selected features. Considering the NN classifier, SSF-λ-2 has shown better or equal results than MMP in 55% of the cases. By taking the NB classifier and the representation entropy into account, SSF-λ-2 has shown better or equal results than MMP in 65% of the cases. The Wilcoxon Signed-Rank test suggests that there is a significant difference favoring SSF-λ-2 w.r.t. the NN accuracy when approximately 50% of the features are selected ($\alpha = 10\%$). Similarly, MMP has shown better results when the representation entropy measure is considered and 75% of the features are selected.

5.2. Comparisons between SSF and ACA

For the filter named ACA [3] (Section 3.2), the user has to choose the number of selected features (r). Aimed at performing interesting comparisons with SSF, we show the obtained results for both one ($r = 1$) and two ($r = 2$) selected features from each cluster. The average classification error rates obtained in a 10-fold cross-validation process are reported in Table 6. The acronyms SSF-R and ACA refer to the *Simplified Silhouette Filter* (SSF) using the *Interdependence Redundancy Measure* – Eq. (4) – and the *Attribute Clustering Algorithm* [3], respectively. An additional parameter can also be found for both algo-

Table 6
Summary of the classification error rates – E(%) – obtained by the classifiers NN and NB.

Dataset	SSF-R-1		SSF-R-2		ACA-1		ACA-2		All	
	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB
Bio1	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.2 (0.7)	0.7 (1.1)	0.2 (0.7)	0.2 (0.7)	0.0 (0.0)	0.0 (0.0)
Bio2	3.0 (1.8)	2.5 (1.9)	3.0 (1.8)	2.7 (2.0)	3.0 (1.8)	2.2 (2.0)	1.7 (1.1)	1.5 (2.0)	1.0 (1.6)	1.2 (1.2)
Bio3	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)	2.7 (2.6)	2.2 (2.6)	0.5 (1.0)	1.0 (1.2)	0.0 (0.0)	0.0 (0.0)
Bio4	4.0 (2.0)	5.0 (2.2)	4.5 (2.6)	5.0 (3.3)	9.2 (2.2)	5.7 (1.9)	4.2 (2.2)	4.0 (1.6)	2.5 (2.5)	2.5 (2.9)
Bio5	0.5 (1.0)	0.0 (0.0)	0.7 (1.1)	0.5 (1.0)	0.5 (1.0)	0.0 (0.0)	0.2 (0.7)	0.0 (0.0)	0.0 (0.0)	0.2 (0.7)
Yeast	2.4 (3.2)	1.4 (2.2)	2.4 (3.2)	1.9 (2.3)	8.8 (7.6)	8.8 (6.2)	4.4 (3.4)	7.4 (7.1)	1.9 (2.3)	2.4 (3.3)
Iono	11.6 (2.3)	14.5 (5.6)	11.3 (5.2)	14.2 (3.9)	16.8 (5.6)	23.0 (7.8)	12.8 (5.3)	32.4 (4.4)	4.2 (1.6)	3.6 (1.7)
Wisc	6.4 (2.2)	6.0 (2.3)	6.0 (1.67)	5.2 (2.1)	6.7 (1.4)	6.6 (2.5)	6.8 (1.7)	5.8 (1.7)	30.2 (5.1)	24.3 (4.8)
Spam	12.2 (1.4)	29.9 (3.6)	10.5 (1.2)	22.3 (2.1)	16.3 (1.4)	39.6 (1.4)	13.3 (1.0)	34.1 (1.8)	13.6 (6.7)	16.7 (3.8)
Ova	5.5 (4.3)	7.9 (3.0)	6.3 (3.6)	6.3 (4.4)	5.9 (4.0)	7.9 (4.0)	6.3 (4.0)	8.3 (4.1)	4.7 (4.6)	7.1 (5.2)

Table 7
Summary of the number of selected features (#FS) and representation entropy values (H).

Dataset	SSF-R-1		SSF-R-2		ACA-1		ACA-2	
	#FS	H	#FS	H	#FS	H	#FS	H
Bio1	3.9 (3.1)	0.7 (0.2)	4.4 (2.8)	0.9 (0.1)	2.0 (0.0)	0.5 (0.1)	3.0 (0.0)	0.8 (0.1)
Bio2	2.3 (0.9)	0.5 (0.1)	3.6 (1.8)	0.8 (0.0)	2.0 (0.0)	0.5 (0.1)	3.0 (0.0)	0.6 (0.0)
Bio3	8.5 (1.1)	0.9 (0.1)	12.9 (1.3)	1.0 (0.1)	2.0 (0.0)	0.5 (0.0)	3.0 (0.0)	0.9 (0.1)
Bio4	5.3 (3.1)	0.7 (0.2)	7.2 (3.7)	1.0 (0.0)	2.0 (0.0)	0.6 (0.0)	2.0 (0.0)	0.7 (0.0)
Bio5	2.0 (0.0)	0.6 (0.1)	3.0 (0.0)	0.8 (0.0)	2.0 (0.0)	0.6 (0.1)	3.0 (0.0)	0.7 (0.1)
Yeast	8.0 (0.7)	0.6 (0.1)	13.7 (1.1)	0.8 (0.0)	2.0 (0.0)	0.2 (0.1)	4.0 (0.0)	0.3 (0.1)
Iono	14.2 (1.6)	2.2 (0.1)	25.1 (1.4)	2.6 (0.0)	3.2 (0.4)	1.1 (0.1)	6.4 (0.8)	1.5 (0.1)
Wisc	2.4 (1.2)	0.5 (0.2)	3.9 (0.3)	0.9 (0.0)	2.0 (0.0)	0.5 (0.0)	3.0 (0.0)	0.5 (0.0)
Spam	25.4 (3.2)	0.0 (0.0)	25.4 (3.2)	0.3 (0.0)	6.0 (0.0)	0.0 (0.0)	12.0 (0.0)	0.1 (0.0)
Ova	116.5 (5.3)	2.2 (0.0)	232.9 (10.9)	2.7 (0.0)	121.8 (0.9)	2.2 (0.0)	243.6 (1.9)	2.2 (0.0)

rithms, making reference to the number of selected features from each cluster. For instance, ACA-1 stands for ACA selecting one feature from each cluster. Table 7 presents the number of selected features (#FS) and the values for the *representation entropy measure* (H).

Consider now the average error rates obtained for every scenario formed by a pair of <dataset, classifier> in Table 6. For the same number of selected features per cluster, SSF has shown better or equal results in 82.5% of those scenarios (i.e., comparing SSF-R-1 with ACA-1 and SSF-R-2 with ACA-2). The adopted statistical procedure indicates that the hypothesis of equal error rates, considering the results obtained by SSF-R-*, ACA-*, and the use of all features, can be rejected (at $\alpha = 10\%$). For both classifiers, statistically significant differences favor SSF-R-1 when compared to ACA-1, which, by its turn, has shown worse results than using all features. In what concerns the number of selected features, ACA has shown better or equal results in 90% of the scenarios. However, significant differences (at $\alpha = 10\%$) were only observed between SSF-R-2 and ACA-1. For the representation entropy measure (Table 7), SSF-R-2 showed statistically significant better results than SSF-R-1, ACA-1, and ACA-2 ($\alpha = 5\%$). ACA-2, by its turn, obtained better results than ACA-1 ($\alpha = 10\%$). Considering the computational efficiency (Table 13), SSF-R-2 and ACA-2 were significant faster (at $\alpha = 10\%$) than the respective variants that select one feature per cluster (as expected). All these results suggest that SSF can be preferred over ACA when classification accuracy is more important than reducing the number of features.

As stated in Section 5.1, it can be of interest to assess two filters (here SSF and ACA) when they select (approximately) the same number of features. To do so, we consider the best version of ACA according to the classification results reported in Table 6, namely: ACA-2. Accordingly, we compare it to SSF-R-2 by means of two related approaches. In the first approach the number of clusters (k) is defined by SSF-R-2, whereas in the second approach k is defined by ACA-2. More precisely, for every fold of the cross-validation process, we provide the number of clusters automatically estimated by SSF-R-2 to ACA-2 (first approach) and, analogously, in the second approach the number of clusters found by ACA-2 is provided as a parameter to SSF-R-2. Table 8 and Table 9 report the achieved classification error rates (and their standard deviations). In both approaches, the Wilcoxon Signed-Rank test [9] has not found statistical significant differences w.r.t. the accuracy (at $\alpha = 10\%$). However, it is interesting to observe that, when k was determined by SSF, this algorithm has obtained better or equal results than ACA in 70% of the scenarios, whereas when ACA defined k, the algorithms presented similar results (40% wins for SSF, 55% wins for ACA, and 5% ties). For the representation entropy measure, the statistical tests suggest that SSF-R-2 is better than ACA-2 for both scenarios ($\alpha = 5\%$). From all these results, one can conclude that SSF gives better or equal performance compared to ACA.

Following the same approach adopted in Section 5.1, we have also analyzed the performance of each algorithm when the number of features to be selected is fixed in advance. Again, three different proportions (25%, 50%, and 75%) of the number of

Table 8
Classification error rates (E(%)) and representation entropy values (H) obtained by SSF and ACA – number of clusters (k) estimated by SSF-R-2.

Dataset	SSF-R-2			ACA-2		
	E(%)-NN	E(%)-NB	H	E(%)-NN	E(%)-NB	H
Bio1	0.0 (0.0)	0.0 (0.0)	0.9 (0.1)	0.2 (0.7)	0.0 (0.0)	0.9 (0.2)
Bio2	3.0 (1.8)	2.7 (2.0)	0.8 (0.1)	2.0 (1.5)	1.7 (1.9)	0.7 (0.1)
Bio3	0.0 (0.0)	0.0 (0.0)	1.0 (0.1)	0.0 (0.0)	0.0 (0.0)	1.0 (0.1)
Bio4	4.5 (2.6)	5.0 (3.3)	1.0 (0.1)	4.7 (2.6)	3.5 (2.0)	0.9 (0.2)
Bio5	0.7 (1.1)	0.5 (1.0)	0.8 (0.0)	0.2 (0.7)	0.0 (0.0)	0.7 (0.1)
Yeast	2.4 (3.2)	1.9 (2.3)	0.8 (0.0)	2.9 (2.4)	4.4 (5.2)	0.7 (0.1)
Iono	11.3 (5.2)	14.2 (3.9)	2.6 (0.0)	11.9 (7.7)	15.0 (6.9)	2.5 (0.0)
Wisc	6.0 (1.6)	5.28 (2.1)	0.9 (0.0)	6.8 (1.7)	5.8 (1.7)	0.5 (0.0)
Spam	10.5 (1.2)	22.3 (2.1)	0.3 (0.0)	10.1 (1.8)	31.0 (3.1)	0.2 (0.1)
Ova	6.3 (3.6)	6.3 (4.4)	2.7 (0.0)	6.3 (4.3)	7.8 (3.9)	2.2 (0.0)

Table 9
Classification error rates (E(%)) and representation entropy values (H) obtained by SSF and ACA – number of clusters (k) estimated by ACA-2.

Dataset	SSF-R-2			ACA-2		
	E(%)-NN	E(%)-NB	H	E(%)-NN	E(%)-NB	H
Bio1	0.0 (0.0)	0.2 (0.7)	0.8 (0.1)	0.2 (0.7)	0.2 (0.7)	0.8 (0.1)
Bio2	2.7 (2.0)	2.5 (2.2)	0.7 (0.0)	1.7 (1.1)	1.5 (2.0)	0.6 (0.0)
Bio3	1.7 (2.5)	2.5 (1.9)	0.7 (0.2)	0.5 (1)	1.0 (1.2)	0.9 (0.1)
Bio4	8.0 (3.6)	6.0 (1.6)	0.8 (0.1)	4.2 (2.2)	4.0 (1.6)	0.7 (0.0)
Bio5	0.7 (1.1)	0.5 (1.0)	0.8 (0.0)	0.2 (0.7)	0.0 (0.0)	0.7 (0.1)
Yeast	10.3 (7.8)	6.3 (5.0)	0.9 (0.1)	4.4 (3.4)	7.4 (7.1)	0.3 (0.1)
Iono	10.8 (4.7)	12.2 (2.5)	1.6 (0.1)	12.8 (5.3)	32.4 (4.4)	1.5 (0.1)
Wisc	6.0 (1.6)	5.2 (2.1)	0.9 (0.0)	6.8 (1.7)	5.8 (1.7)	0.5 (0.0)
Spam	19.8 (2.8)	43.7 (6.0)	1.5 (0.7)	13.3 (1.0)	34.1 (1.8)	0.1 (0.0)
Ova	4.7 (3.9)	6.3 (3.6)	2.7 (0.0)	6.3 (4.0)	8.3 (4.1)	2.2 (0.0)

Table 10
Classification error rates (E(%)) and representation entropy values (H) obtained by SSF and ACA for different proportions of selected features.

Algorithms	%F	Criteria	Datasets									
			Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	Iono	Wisc	Spam	Ova
SSF-R-2	25%	E(%)-NN	0.0 (0.0)	1.5 (2.4)	0.3 (0.8)	4.8 (3.4)	0.3 (0.8)	4.4 (3.6)	9.7 (4.7)	6.0 (1.8)	13.8 (1.7)	5.5 (4.6)
		E(%)-NB	0.3 (0.8)	2.0 (2.6)	0.3 (0.8)	5.3 (3.0)	0.5 (1.1)	3.0 (3.5)	10.0 (4.1)	5.3 (2.2)	40.3 (5.5)	7.5 (6.0)
		H	0.8 (0.0)	0.8 (0.1)	0.9 (0.1)	1.0 (0.1)	0.9 (0.0)	0.8 (0.1)	2.0 (0.0)	0.9 (0.0)	0.2 (0.1)	2.4 (0.0)
	50%	E(%)-NN	0.0 (0.0)	1.8 (1.7)	0.0 (0.0)	3.0 (2.0)	0.0 (0.0)	1.5 (2.4)	11.1 (5.3)	4.5 (1.5)	11.5 (1.0)	5.5 (4.6)
		E(%)-NB	0.0 (0.0)	2.3 (2.8)	0.0 (0.0)	4.3 (3.1)	0.0 (0.0)	1.0 (2.1)	11.4 (3.7)	5.3 (1.6)	31.7 (3.5)	7.5 (5.5)
		H	0.9 (0.1)	1.0 (0.0)	1.0 (0.1)	1.1 (0.0)	0.9 (0.1)	0.8 (0.1)	2.4 (0.0)	1.0 (0.0)	0.2 (0.1)	2.3 (0.0)
	75%	E(%)-NN	0.0 (0.0)	1.8 (2.1)	0.0 (0.0)	3.0 (1.6)	0.0 (0.0)	2.9 (3.4)	10.5 (4.0)	5.1 (1.7)	10.8 (1.2)	4.7 (4.8)
		E(%)-NB	0.0 (0.0)	2.8 (3.0)	0.0 (0.0)	2.8 (2.5)	0.0 (0.0)	2.0 (2.5)	13.1 (5.1)	5.4 (1.6)	23.9 (3.4)	7.2 (5.9)
		H	1.0 (0.1)	1.0 (0.1)	1.0 (0.0)	1.1 (0.1)	1.0 (0.1)	0.8 (0.0)	2.6 (0.0)	1.1 (0.0)	0.3 (0.0)	2.3 (0.0)
ACA-2	25%	E(%)-NN	0.3 (0.8)	3.0 (2.6)	0.3 (0.8)	5.5 (2.6)	0.3 (0.8)	6.3 (5.1)	10.8 (5.5)	6.9 (1.9)	15.1 (1.4)	4.7 (4.8)
		E(%)-NB	0.0 (0.0)	2.0 (2.3)	0.5 (1.1)	4.0 (3.4)	0.0 (0.0)	6.9 (7.1)	25.1 (6.6)	5.9 (1.8)	34.4 (3.4)	7.5 (5.7)
		H	1.0 (0.0)	0.8 (0.0)	1.0 (0.1)	0.8 (0.0)	0.8 (0.0)	0.5 (0.1)	1.9 (0.1)	0.5 (0.0)	1.3 (1.1)	2.3 (0.0)
	50%	E(%)-NN	0.0 (0.0)	2.5 (2.9)	0.3 (0.8)	4.3 (1.7)	0.3 (0.8)	3.9 (3.9)	10.8 (6.9)	5.4 (2.2)	13.2 (1.7)	5.1 (4.6)
		E(%)-NB	0.0 (0.0)	1.8 (2.1)	0.3 (0.8)	2.8 (3.0)	0.0 (0.0)	4.0 (5.7)	15.1 (8.8)	4.8 (1.8)	28.9 (3.5)	7.1 (5.9)
			1.1 (0.0)	0.9 (0.0)	1.0 (0.0)	1.0 (0.1)	1.0 (0.1)	0.6 (0.1)	2.3 (0.0)	0.8 (0.0)	0.1 (0.0)	2.3 (0.0)
	75%	E(%)-NN	0.0 (0.0)	2.0 (2.3)	0.0 (0.0)	3.3 (1.7)	0.0 (0.0)	3.0 (2.5)	13.1 (6.0)	5.4 (1.9)	11.3 (1.4)	5.1 (4.6)
		E(%)-NB	0.0 (0.0)	1.5 (1.7)	0.3 (0.8)	3.0 (2.6)	0.3 (0.8)	4.5 (5.5)	15.1 (6.8)	5.0 (1.7)	27.0 (2.3)	7.2 (5.9)
		H	1.1 (0.0)	1.0 (0.0)	1.0 (0.1)	1.1 (0.0)	1.0 (0.1)	0.7 (0.1)	2.5 (0.0)	0.9 (0.0)	0.1 (0.0)	2.3 (0.0)

features were adopted. The parameter k has been set equal to $\lceil(\lceil M^* \, pct\rceil)/2\rceil$, with $pct \in \{0.25,0.5,0.75\}$, for both SSF-R-2 and ACA-2. Table 10 shows the obtained results. The Wilcoxon Signed-Rank test ($\alpha = 10\%$) suggests that SSF-R-2 showed better results than ACA-2 for all proportions of selected features (25%, 50%, and 75%) for the NN classifier. For the NB classifier, SSF-R-2 presented better results in 63% of the cases, but significant statistical differences have not been observed. By taking the representation entropy measure into account, statistically significant difference favoring SSF-R-2 has only been observed when approximately 50% of the features were selected ($\alpha = 5\%$). SSF-R-2 has also shown better or equal results compared to ACA-2 in at least 60% of the cases for all proportions of selected features.

Table 11
Summary of the classification error rates – E(%) – obtained by the classifiers NN and NB.

Dataset	SSF-SU-2		S^3F	
	E(%)-NN	E(%)-NB	E(%)-NN	E(%)-NB
Bio1	0.0 (0.0)	0.2 (0.7)	0.0 (0.0)	0.0 (0.0)
Bio2	2.7 (2.0)	2.5 (2.2)	1.5 (1.6)	0.7 (1.1)
Bio3	1.5 (2.5)	1.7 (2.2)	0.0 (0.0)	0.0 (0.0)
Bio4	7.0 (4.3)	4.7 (2.8)	2.5 (2.2)	2.7 (2.0)
Bio5	0.7 (1.1)	0.5 (1.0)	0.0 (0.0)	0.2 (0.7)
Yeast	2.4 (3.2)	1.9 (2.3)	1.9 (2.3)	1.9 (2.3)
Iono	11.9 (5.8)	14.2 (3.9)	12.5 (5.8)	12.8 (4.2)
Wisc	6.0 (1.6)	5.2 (2.1)	3.9 (1.4)	3.5 (1.7)
Spam	11 (1.7)	22.5 (2.3)	9.6 (1.2)	20.2 (1.5)
Ova	5.5 (3.6)	7.1 (3.4)	2.3 (2.6)	2.3 (1.9)

Table 12
Summary of the number of selected features (#FS) and representation entropy values (H).

Dataset	SSF-SU-2		S^3F	
	#FS	H	#FS	H
Bio1	3.0 (0.0)	0.8 (0.1)	18.5 (1.0)	1.1 (0.0)
Bio2	3.0 (0.0)	0.7 (0.0)	17.7 (0.6)	1.1 (0.0)
Bio3	7.3 (5.3)	0.8 (0.3)	16.6 (1.0)	1.0 (0.0)
Bio4	5.8 (4.3)	0.9 (0.1)	16.0 (1.9)	1.2 (0.0)
Bio5	3.0 (0.0)	0.8 (0.0)	16.9 (1.1)	1.0 (0.0)
Yeast	13.7 (1.1)	0.8 (0.0)	15.8 (1.5)	0.9 (0.1)
Iono	24.9 (1.7)	2.6 (0.0)	26.9 (1.7)	2.6 (0.0)
Wisc	3.9 (0.3)	0.9 (0.0)	8.0 (0.0)	1.2 (0.0)
Spam	42.2 (3.7)	0.3 (0.0)	44.1 (2.0)	0.3 (0.0)
Ova	228.9 (11.2)	2.7 (0.0)	196.0 (69.7)	2.7 (0.1)

Table 13
Average running times (in seconds) - standard deviations between parentheses.

Method	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	Ionosphere	Wisconsin	Spambase	Ovarian Cancer
SSF-λ-1	0.30 (0.2)	0.12 (0.0)	0.12 (0.0)	0.12 (0.0)	0.12 (0.0)	0.10 (0.0)	0.30 (0.0)	0.30 (0.0)	5.55 (0.0)	66187.8 (45911.0)
SSF-λ-2	0.06 (0.0)	0.05 (0.0)	0.05 (0.0)	0.05 (0.0)	0.05 (0.0)	0.04 (0.0)	0.12 (0.0)	0.02 (0.0)	1.80 (0.0)	66330.6 (45908.5)
SSF-R-1	0.17 (0.0)	0.13 (0.0)	0.14 (0.0)	0.13 (0.0)	0.13 (0.0)	0.10 (0.0)	0.36 (0.0)	0.04 (0.0)	7.19 (0.1)	18813.4 (514.2)
SSF-R-2	0.07 (0.0)	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.05 (0.0)	0.14 (0.0)	0.02 (0.0)	2.92 (0.1)	19372.4 (1033.0)
SSF-SU-1	0.13 (0.0)	0.13 (0.0)	0.13 (0.0)	0.13 (0.0)	0.13 (0.0)	0.10 (0.0)	0.35 (0.0)	0.04 (0.0)	7.16 (0.0)	17831.1 (605.2)
SSF-SU-2	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.13 (0.0)	0.06 (0.0)	0.05 (0.0)	0.13 (0.0)	0.02 (0.0)	2.80 (0.0)	17522.6 (741.3)
S^3F	0.06 (0.0)	0.13 (0.0)	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.05 (0.0)	0.14 (0.0)	0.02 (0.0)	3.15 (0.1)	38014.7 (4620.2)
ACA-1	0.15 (0.0)	0.13 (0.0)	0.13 (0.0)	0.13 (0.0)	0.13 (0.0)	0.10 (0.0)	0.34 (0.0)	0.04 (0.0)	7.13 (0.1)	19412.0 (750.0)
ACA-2	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.06 (0.0)	0.05 (0.0)	0.14 (0.0)	0.02 (0.0)	2.88 (0.0)	19559.4 (640.8)
MMP-KNN	0.21 (0.0)	0.09 (0.0)	0.09 (0.0)	0.09 (0.0)	0.09 (0.0)	0.06 (0.0)	0.17 (0.0)	0.10 (0.0)	32.62 (1.3)	—
MMP-NB	0.16 (0.0)	0.09 (0.0)	0.09 (0.0)	0.09 (0.0)	0.08 (0.0)	0.06 (0.0)	0.13 (0.0)	0.06 (0.0)	4.11 (0.0)	—

5.3. Comparisons between SSF and S^3F

This section compares the performance of the supervised and unsupervised SSF variants. The acronyms SSF-SU and S^3F refer to the SSF based on the *Symmetrical Uncertainty* (Eq. (16)) and the supervised SSF variant (Section 4.1), respectively. Table 11 reports a summary of the classification error rates obtained in a 10-fold cross validation, whereas Table 12 summarizes the number of features selected by each algorithm and the corresponding values of the *representation entropy measure*.

Considering the average error rates obtained for every scenario formed by a pair of <dataset, classifier> , S^3F presented better or equal results compared to SSF-SU in 95% of the scenarios. Indeed, the Wilcoxon Signed-Rank Test [9] allows concluding that S^3F obtained significantly better results compared to SSF-SU, while selecting more features than the latter (at $\alpha = 10\%$). S^3F has also shown better results compared to SSF-SU-2 for the representation entropy measure (at $\alpha = 5\%$). Considering the computational efficiency, significant differences were not observed. All these results suggest that S^3F represents a significant improvement over SSF for applications in which obtaining better classification accuracy is more important than reducing the number of selected features.

Finally, for the sake of illustration, we have run a Naïve Bayes Wrapper (NBW) and compared its selected features to those found by S^3F. This comparison is aimed at assessing the relevance of the features selected by S^3F. To that end, only the (com-

plete) training datasets were used. The obtained results show that in three datasets (Bio5, Wisc, and Spam) only one of the features selected by the NBW has not been chosen by S^3F. For the datasets Bio1-Bio4, Yeast, and Iono, all features selected by NBW have also been selected by S^3F. In what concerns the Ova dataset, S^3F has not selected the features chosen by NBW (with *best first search*). However, the difference in terms of the accuracy of the resulting classifiers, using a 10-fold cross validation process, was approximately 2% (this was also the average difference between S^3F and NBW for all the other datasets). As observed in Section 1, in general filters are less computationally expensive than wrappers[5], but these may be superior in relation to the quality of the feature subsets. From this point of view, although S^3F has selected more features, the relevant ones identified by the more computationally demanding NBW have also been selected by S^3F (except for the Ovarian Cancer dataset).

6. Conclusions

A filter for feature selection based on the partitioning of a set of features into clusters was introduced. The proposed algorithm, called Simplified Silhouette Filter (SSF), relies on selecting representative features from the obtained clusters, thus allowing the elimination of redundant features. We have shown that SSF presents improvements in relation to the state of art methods described by Mitra et al. [29] and by Au et al. [3], which also remove redundant features from clustering them. Therefore, SSF becomes eligible to join a pool of feature selection algorithms to be used in practice. As discussed in the paper, although the algorithms described in [29,3] are useful, in general there is no clear guidance on how to choose their user-defined parameter values, which can be critical for some applications. In this context, more automatic tools may be preferred. In the approach proposed in our work, a clustering algorithm induces a set of k clusters (k is estimated directly from data) from which a number of representative features is automatically selected. From this perspective, SSF selects features in a more *automatic* fashion when compared to their counterparts. Finally, a variant of SSF – named S^3F, from Supervised Simplified Silhouette Filter – that incorporates attribute-class correlations has also been introduced. As an additional contribution of our work, we have elaborated on a theoretical framework to formally analyze some properties of feature selection methods that rely on finding clusters of features.

We have illustrated the performance of SSF (and its variants) by reporting empirical results obtained in ten datasets, and comparing them to MMP [29] and ACA [3]. The relative quality of each feature subset was assessed by means of the generalization capability of two widely used classifiers (k-Nearest Neighbors and Naïve Bayes). For SSF, we assessed two alternatives for selecting features from each cluster. The first one involves selecting only the feature most correlated to the other features in the same cluster. This selected feature is the medoid of the cluster. In the second approach, along with the medoid, we select the feature least correlated (least redundant) with the medoid. Thus, two features from each cluster are chosen. Considering the average classification error rates, SSF has shown better results by selecting two features in most of the experiments. From this observation, we recommend the selection of two features from each cluster. Also, the accuracies achieved by such a version of SSF are competitive with the more computationally demanding exploratory data analysis approach performed by MMP. Considering the accuracy provided by the features selected by SSF in comparison to ACA, SSF has also shown better results in most of the investigated scenarios (for which significant differences in terms of computational efficiency have not been observed). We have not found significant differences with regard to the number of features selected by SSF when compared to the number of features selected by MMP and ACA. In addition, we have compared the algorithms by selecting feature subsets with (approximately) the same cardinality, where the number of features was determined by each algorithm and provided as constraints to the others. In these cases, SSF performance has shown to be better or equal compared to MMP and ACA. Finally, we have also analyzed the performance of each algorithm when the number of features to be selected is fixed in advance. To that end, three different proportions (25%, 50%, and 75%) of the number of features were considered. In these experiments, SSF also presented better or equal results than ACA and MMP.

In what concerns the SSF supervised variant (S^3F), it has shown significantly better accuracy results than SSF, which, by its turn, selected less features than S^3F. Considering computational efficiency issues, significant differences were not observed between S^3F and SSF. These results suggest that S^3F represents a significant improvement over SSF for applications in which obtaining better classification accuracy is more important than reducing the number of features.

Although promising results have been achieved in this work, there are several issues that can be investigated in the future. For example, provided that SSF does not necessarily require the use of a particular clustering algorithm, the investigation of the suitability of clustering algorithms different from the one used in our study is interesting. Also, other correlation measures, such as the recently proposed Weighted Goodman–Kruskal [6], can be used in our algorithmic framework. Finally, using other relative validity criteria (different from the silhouette), as well as studying cluster ensembles for feature selection, are possible venues for further research.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive and detailed comments. Also, we would like to thank Estevam R. Hruschka Jr., Nelson F. F. Ebecken, and Maria C. Monard for some valuable discussions. Finally, we also acknowledge the Brazilian Research Agencies CNPq and FAPESP for their financial support to this work.

[5] For instance, the running time difference between NBW and S^3F was equal to 36 hours for the Ovarian Cancer dataset.

References

[1] P. Arabie, L.J. Hubert, An Overview of Combinatorial Data Analysis, in: P. Arabie, L.J. Hubert, G. DeSoete (Eds.), Clustering and Classification, World Scientific, 1999. Chapter 1.
[2] A. Asuncion, D.J. Newman, UCI Machine Learning Repository, Irvine, CA: University of California <http://www.ics.uci.edu/~mlearn/MLRepository.htm>.
[3] W. Au, K.C.C. Chan, A.K.C. Wong, Y. Wang, Attribute clustering for grouping, selection, and classification of gene expression data, IEEE/ACM Transactions on Computational Biology and Bioinformatics 2 (2) (2005) 83–101.
[4] B. Bonev, F. Escolano, M. e Cazorla, Feature selection, mutual information, and the classification of high-dimensional patterns, Pattern Analysis & Applications 11 (3) (2008) 309–319.
[5] G. Casella, L.R. Berger, Statistical Inference, Second ed., Duxbury, 2002.
[6] R.J.G.B. Campello, E.R. Hruschka, On comparing two sequences of numbers and its applications to clustering analysis, Information Sciences 179 (8) (2009) 1025–1039.
[7] T.F. Covões, E.R. Hruschka, L.N. de Castro, A. dos Santos, A cluster-based feature selection approach. In Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems, Lecture Notes in Artificial Intelligence, Vol. 5572, 2009, pp. 169–176.
[8] P.A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Prentice Hall, 1992.
[9] J. Demšar, Statistical Comparisons of Classifiers over Multiple Data Sets, Journal of Machine Learning Research 7 (2006) 1–30.
[10] P.A. Estevez, M. Tesmer, C.A. Perez, J.M. Zurada, Normalized mutual information feature selection, IEEE Transactions on Neural Networks 20 (2) (2009) 189–201.
[11] B.S. Everitt, S. Landau, M. Leese, Cluster Analysis, Arnold Publishers, London, 2001.
[12] E. Falkenauer, Genetic Algorithms and Grouping Problems, John Wiley & Sons, 1998.
[13] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
[14] M.A. Hall, Correlation-based Feature Selection for Machine Learning, PhD Thesis, University of Waikato, 1999.
[15] E.R. Hruschka, R.J.G.B. Campello, L.N. de Castro, Evolving clusters in gene-expression data, Information Sciences, Vol. 176, Elsevier, 2006, pp. 1898–1927. 13.
[16] E.R. Hruschka, L.N. de Castro, R.J.G.B. Campello, Evolutionary Algorithms for Clustering Gene-Expression Data, Proceedings of the 4th IEEE International Conference on Data Mining 2004, Brighton, UK, 2004, pp. 403–406.
[17] J. Hua, W.D. Tembe, E.R. Dougherty, Performance of feature-selection methods in the classification of high-dimension data, Pattern Recognition 42 (3) (2009) 409–424.
[18] G. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem, in: Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann, 1994.
[19] L. Kaufman, P.J. Rousseeuw, Finding Groups in Data – An Introduction to Cluster Analysis, Wiley Series in Probability and Mathematical Statistics (1990).
[20] B. King, Step-wise clustering procedures, Journal of the American Statistical Association (1967) 86–101.
[21] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1-2) (1997) 273–324.
[22] D. Koller, M. Sahami, Toward optimal feature selection, Proceedings of the 13th International Conference on Machine Learning (1996) 284–292.
[23] J. Liang, S. Yang, A. e Winstanley, Invariant optimal feature selection: A distance discriminant and feature ranking based solution, Pattern Recognition 41 (5) (2008) 1429–1439.
[24] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic, 1998.
[25] H. Liu, J. Sun, L. Liu, H. e Zhang, Feature selection with dynamic mutual information, Pattern Recognition 42 (7) (2009) 1330–1339.
[26] H. Liu, L. Yu, Toward integrating feature selection algorithms for classification and clustering, IEEE Transactions on Knowledge and Data Engineering 17 (3) (2005) 1–12.
[27] S. Maldonado, R. e Weber, A wrapper method for feature selection using support vector machines, Information Sciences 179 (13) (2009) 2208–2217. Special Section on High Order Fuzzy.
[28] S. Maldonado, R. Weber, J. Basak, Simultaneous feature selection and classification using kernel-penalized support vector machines, Information Sciences 181 (1) (2011) 115–128.
[29] P. Mitra, C.A. Murthy, S.K. Pal, Unsupervised feature selection using feature similarity, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (4) (2002) 301–312.
[30] L. Nanni, Cluster-based pattern discrimination: A novel technique for feature selection, Pattern Recognition Letters 27 (6) (2006) 682–687.
[31] E.F. Petricoin III, A.M. Ardekani, B.A. Hitt, P.J. Levine, V.A. Fusaro, S.M. Steinberg, G.B. Mills, C. Simone, D.A. Fishman, E.C. Kohn, L.A. Liotta, Use of proteomic patterns in serum to identify ovarian cancer, The Lancet 359 (2002) 572–577.
[32] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. e Vetterling, Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, Cambridge, 1990.
[33] P. Pudil, J. Novovicová, J. Kittler, Floating search methods in feature selection, Pattern Recognition Letters 15 (1994) 1119–1125.
[34] J. Reunanen, Overfitting in making comparisons between variable selection methods, Journal of Machine Learning Research 3 (2003) 1371–1382.
[35] J.R. Quinlan, C4.5: programs for machine learning, Morgan Kaufmann Publishers Inc., 1993.
[36] Y. Weiss, Y. Elovici, L. Rokach, The CASH Algorithm-Cost-Sensitive Attribute Selection using Histograms, Information Sciences, in: Press, Accepted Manuscript, Available online 2 February 2011.
[37] I.H. Witten, E. Frank, Data Mining – Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann Publishers, USA, 2000.
[38] X. Wu, V. Kumar, Ross, J. Ghosh, Q. Yang, H. Motoda, G. Mclachlan, A. Ng, B. Liu, P. Yu, Z.-H. Zhou, M. Steinbach, D. Hand, D. e Steinberg, Top 10 algorithms in data mining, Knowledge and Information Systems 14 (1) (2008) 1–37.
[39] Y. Yang, J. Pederson, A comparative study on feature selection in text categorization, Proc. of the Fourteenth International Conference on Machine Learning, 1997.
[40] Y. Yang, G. Webb, Proportional k-interval discretization for naive-bayes classifiers, in: Proceedings of the 12th European Conference on Machine Learning (ICML-2001), Springer-Verlag, 2001, pp. 564–575.
[41] K.Y. Yeung, M. Medvedovic, R.E. Bumgarner, <http://expression.microslu.washington.edu/expression/kayee/cluster2003/yeunggb2003.html>. (2003).
[42] K.Y. Yeung, M. Medvedovic, R.E. Bumgarner, Clustering gene-expression data with repeated measurements, Genome Biology 4 (5) (2003). article R34.
[43] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, Journal of Machine Learning Research 5 (2004) 1205–1224.
[44] T. Zhang, R. Ramakrishnan, M. Livny, BIRCH: A new data clustering algorithm and its applications, Data Mining and Knowledge Discovery, Vol. 1, Springer, 1997. pp. 141–182 (2) (42).

Thiago Ferreira Covões received his B.Sc. degree in Computer Science from the University of Santos, Brazil, in 2007, and his Master of Science degree in Computer Science from the University of São Paulo (USP) at São Carlos, Brazil, in 2010. His research interest is data mining, with emphasis on clustering algorithms, feature selection, and classification.

Eduardo Raul Hruschka received his B.Sc. degree in Civil Engineering from Federal University of Paraná, Brazil, in 1995, and his M.Sc. and Ph.D. degrees in Computational Systems from Federal University of Rio de Janeiro in 1998 and 2001, respectively. He is currently assistant professor of the Department of Computer Sciences of the University of São Paulo, Brazil. He has authored or coauthored more than 50 research publications in peer-reviewed reputed journals, book chapters, and conference proceedings. Dr. Hruschka has been a reviewer for several journals such as Information Sciences, IEEE TSMC, IEEE TKDE, IEEE TEC, IEEE TNN, Journal of Heuristics, Pattern Recognition Letters, Applied Soft Computing, ACM Transactions on Autonomous and Adaptive Systems, Computational Statistics & Data Analysis, and Bioinformatics.