2009 Ninth International Conference on Intelligent Systems Design and Applications

An Experimental Study on Unsupervised Clustering-based Feature Selection Methods

Thiago F. Covões and Eduardo R. Hruschka
Computer Science Department
University of São Paulo (USP) at São Carlos
São Carlos, Brazil
{*tcovoes,erh*}*@icmc.usp.br*

***Abstract*—Feature selection is an essential task in data mining because it makes it possible not only to reduce computational times and storage requirements, but also to favor model improvement and better data understanding. In this work, we analyze three methods for unsupervised feature selection that are based on the clustering of features for redundancy removal. We report experimental results obtained in ten datasets that illustrate practical scenarios of particular interest, in which one method may be preferred over another. In order to provide some reassurance about the validity and non-randomness of the obtained results, we also present the results of statistical tests.**

***Keywords*-unsupervised feature selection; feature clustering; clustering problems;**

I. Introduction

Feature selection aims at choosing a subset of 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 [1], [2]: facilitating data visualization and understanding, reducing the measurement and storage requirements, reducing training and utilization times, and defying the curse of dimensionality to improve prediction performance. Many of these benefits can also be achieved in unsupervised learning (clustering). However, most of the existing supervised methods for feature selection rely on assessing how well some features discriminate among a set of predefined classes. These classes are not available in clustering tasks, in which one seeks to identify a finite set of categories (clusters) to describe a given dataset, both maximizing homogeneity within each cluster and heterogeneity among different clusters [3]. In this sense, it is difficult to assess the relevance of a subset of features for describing classes that are not known a priori.

We assume that clustering involves the partitioning of a set $\mathbf{X}$ of instances into a collection of mutually disjoint subsets $\mathbf{C}_i$ of $\mathbf{X}$ whose union is $\mathbf{X}$. Formally, let us consider a set of N instances $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ to be clustered, where each $\mathbf{x}_i$ is a vector consisting of M measurements. The instances must be clustered into non-overlapping groups (here called a partition) $\mathbf{C} = \{\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_c\}$ where c is the number of clusters, such that $\mathbf{C}_1 \cup \mathbf{C}_2 \cup \cdots \cup \mathbf{C}_c = \mathbf{X}$, $\mathbf{C}_i \neq \emptyset$, and $\mathbf{C}_i \cap \mathbf{C}_j = \emptyset$ for $i \neq j$.

Although many algorithms for clustering have been proposed in the literature (e.g., see [4], [5]), relatively little work has been done on feature selection for clustering [6]–[12]. Most clustering methods assume that all features are equally important [6]. However, some features may be more important than others for inducing clusters. In these cases, feature selection methods can be useful. A comprehensive survey of feature selection algorithms for classification and clustering is presented in [13]. In brief, there are two fundamentally different approaches for feature selection [14], [15]: wrapper and filter. The former evaluates the subset of selected features using criteria based on the results of clustering algorithms, i.e., the clustering method is wrapped into the feature selection procedure. The latter involves performing feature assessments based on intrinsic properties of the data. These properties are presumed to affect the ultimate performance of the clustering algorithm, but the feature set is filtered without considering the clustering algorithm that will be ultimately used.

In this work, we analyze three unsupervised feature selection methods, namely: the filter proposed by Mitra et al. [16], the *Attribute Clustering Algorithm* [17], and the *Simplified Silhouette Filter* [18]. These methods are based on the clustering of features for redundancy removal. To identify redundancy between features, correlation measures are used. Following [2], *"it is widely accepted that two features are redundant if their values are completely correlated"*. Thus, feature clustering can be defined as the partitioning of a set A of attributes $A = \{A_1, A_2, \ldots, A_M\}$, into a collection $C^A = \{C_1, C_2, \ldots, C_k\}$ of mutually disjoint subsets of correlated features C_i of A, where k is the number of cluster of features (attributes), such that $C_1 \cup C_2 \cup \cdots \cup C_k = A$, $C_i \neq \emptyset$, and $C_i \cap C_j = \emptyset$ for $i \neq j$.

The remainder of this paper is organized as follows. The next section elaborates on the feature selection methods studied in this work. Section III describes the experimental results obtained. Finally, Section IV concludes the paper and

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.79

IEEE computer society

points out some future work.

II. Analyzed Methods

A. Filter proposed by Mitra et al. [16]

The unsupervised filter proposed in [16] (here called MMP from the last names of its authors: Mitra, Murthy, and Pal) 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. In particular, it is shown in [16] that the proposed *maximal information compression index* — $\lambda(A_i, A_j)$ — 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 A_i and A_j be two random variables (here called features). $\lambda(A_i, A_j)$ is defined as:

$$2\lambda_2(A_i, A_j) = \xi - \sqrt{\xi^2 - 4s^2_{A_i} s^2_{A_j}(1 - \rho(A_i, A_j)^2)}, \quad (1)$$

where $s^2_{A_i}$ denotes the variance of A_i, $\xi = s^2_{A_i} + s^2_{A_j}$ and $\rho(A_i, A_j)$ is the correlation coefficient between the features A_i e A_j:

$$\rho(A_i, A_j) = \frac{covariance(A_i, A_j)}{\sqrt{s^2_{A_i} s^2_{A_j}}} \quad (2)$$

Clusters of features are obtained via the well-known k-nearest neighbors (k-NN) principle. 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 [16], 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. On the other hand, the choice of the value of k_{NN} may be hard to be accomplished in practice, because the user is left to estimate a critical parameter of the algorithm. The overall computational complexity of the algorithm is estimated in [16] as $O(M^2 \cdot N)$ for a given value of k_{NN}. If the k_{NN} value is unknown, we shall note that an exploratory data analysis can be performed by varying k_{NN} in the range [1, M-1], leading to a computational cost of $O(M^2 \cdot N + M^3)$, where M and N stand for the number of features and instances, respectively.

B. Filter proposed by Au et al. [17]

Au et al. [17] proposed a filter named *Attribute Clustering Algorithm* (ACA) that suggests using 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 $A = \{A_1, A_2, \cdots, A_M\}$ is such a set of discrete features and that $\forall A_i \in A, dom(A_i) = \{v_{i1}, \cdots, v_{im_i}\}$. The correlation measure used by ACA is the *interdependence redundancy measure*, $R(A_i, A_j)$, defined in [17] as:

$$R(A_i, A_j) = \frac{I(A_i, A_j)}{H(A_i, A_j)} \quad (3)$$

where $I(A_i, A_j)$ and $H(A_i, A_j)$ are the mutual information and the joint entropy between the features A_i and A_j, respectively:

$$I(A_i, A_j) = \sum_{k=1}^{m_i} \sum_{l=1}^{m_j} P(v_{ik}, v_{jl}) \log \frac{P(v_{ik}, v_{jl})}{P(v_{ik})P(v_{jl})}, \quad (4)$$

$$H(A_i, A_j) = -\sum_{k=1}^{m_i} \sum_{l=1}^{m_j} P(v_{ik}, v_{jl}) \log P(v_{ik}, v_{jl}). \quad (5)$$

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

The authors in [17] also introduced the concept of a *mode feature*, which is the feature more correlated with the other features of a given group. More precisely, the mode of the group C_r is computed as:

$$\eta_r = \underset{A_i}{\operatorname{argmax}} \sum_{A_j \in C_r} R(A_i, A_j) \quad (6)$$

For clustering features the authors [17] use a variant of the well-known k-means, denominated k-modes. The main differences between k-means and k-modes are: (i) the replacement of the centroid (cluster mean vector) by the *mode feature*; (ii) the replacement of the Euclidean distance by the *interdependence redundancy measure* — $R(A_i, A_j)$ — in Eq. (3). 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 — $R(A_i, A_j)$ — instead of the widely used Euclidean distance.

In principle, the clustering algorithm used by ACA needs the definition of the number of groups a priori, but the authors [17] suggest that the sum of the multiple significant interdependence redundancy measure in Eq. (7) can indicate the best value for *k*.

$$k = \underset{k\in\{2,\ldots,M\}}{\operatorname{argmax}} \sum_{r=1}^{k} \sum_{A_i\in\{C_r-\eta_r\}} R(A_i,\eta_r). \qquad (7)$$

After partitioning the features into distinct clusters, the authors propose to select, from each cluster, the r features with the higher correlation with the other features from 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. In what concerns computational efficiency issues, the computational complexity of ACA is estimated in [17] as $O(k \cdot N \cdot M^2 \cdot t)$, where t is the number of k-modes iterations and k is the given number of clusters. If the number of clusters is unknown and k is varied in the range [2,M-1], we can estimate the overall computational complexity of ACA as $O(M^3)$. The computational complexity in terms of computing the correlation between features is estimated as $O(N)$, when the PKID algorithm [19] is used for discretization (as done in this work).

C. Simplified Silhouette Filter [18]

The Simplified Silhouette Filter (SSF) was introduced in [18], in which it was assessed in classification problems. In our current paper, we investigate the use of SSF for clustering problems, comparing it to the state of the art feature selection methods just described in previous sections. In addition, three additional correlation measures are here used, and a promising variant of SSF is now described. In particular, such a variant selects two features from each cluster and it can provide better performance than its predecessor in some applications, as discussed in the sequel.

SSF is also based on feature clustering (as defined in the introductory section). After the partitioning of the set of features A into k mutually disjoint subsets of correlated features, 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 is often tackled indirectly, i.e. distance measures can be used to quantify dissimilarities (lack of correlation) between features. In this work, we employ four correlation measures for finding clusters of features, namely: the *maximal information compression index* in Eq. (1), the *correlation coefficient* in Eq. (2), the *interdependence redundancy measure* in Eq. (3), and the *symmetrical uncertainty* [20]:

$$SU(A_i,A_j) = 2\left[\frac{IG(A_i,A_j)}{H(A_i)+H(A_j)}\right] \qquad (8)$$

where $IG(A_i,A_j)$ and $H(A_i)$ denote the *information gain* [20] between features A_i and A_j and the entropy of feature A_i, respectively. The value of the symmetrical uncertainty is in the range [0,1], 1 (one) indicating full correlation and 0 (zero) indicating that the two features are independent.

Attempting to find a globally optimum solution for clustering problems is usually not computationally feasible [3]. 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 [21], for finding the number of clusters and the corresponding feature partitions.

To define the simplified silhouette (SS) [21], consider a feature A_i belonging to cluster C_a. The dissimilarity of A_i to the medoid of C_a is denoted by $a(i)$. Now let us take into account cluster C_j. The dissimilarity of A_i to medoid of C_j will be called $d(A_i,C_j)$. After computing $d(A_i,C_j)$ for all clusters $C_j \neq C_a$, the smallest one is selected, i.e. $b(i) = min\ d(A_i,C_j)$, $C_j \neq C_a$. This value represents the dissimilarity of A_i to its neighbor cluster, and the silhouette $s(i)$ is given by:

$$s(i) = \frac{b(i)-a(i)}{max\{a(i),b(i)\}}. \qquad (9)$$

The higher $s(i)$ the better the assignment of A_i to a given cluster. In addition, if $s(i)$ is equal to zero, then it is not clear whether the feature should have been assigned to its current cluster or to a neighboring one [4]. Finally, if C_a is a singleton, then $s(i)$ is not defined and the most neutral choice is to set $s(i) = 0$ [5]. The average of $s(i)$, i = 1, 2, ..., M, can be used as a criterion to assess the quality of a given feature partition. By doing so, the best clustering is achieved when the silhouette value is maximized.

The computation of the simplified silhouette (SS) [21] depends only on the achieved partition and not on the adopted clustering algorithm. Thus, the SS can be applied to assess partitions (taking into account the number of clusters) obtained by several clustering algorithms. We adopt the well-known k-medoids algorithm to obtain partitions to be evaluated by the SS. This algorithm is interrupted as soon as medoids from two consecutive iterations are equal. Roughly speaking, 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.

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, i.e. the number of clusters k must be determined a priori. In this sense, one can perform multiple runs of k-medoids (for different values of k) and then choose the best available partition, which corresponds to the maximum achieved value for the SS. It is also well-known that k-medoids may get

stuck at suboptimal solutions for a given k [22]. To alleviate this problem, one can perform multiple runs of k-medoids for a fixed k.

We here investigate two alternatives for selecting features from each obtained cluster. The first one involves selecting only the feature more correlated to the other features in the same cluster. The selected feature is indeed the medoid of the cluster and it can be viewed as the representative feature of that cluster. Doing so, 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 of the cluster, the feature less correlated (less redundant) with the medoid is also selected. Thus, two features from each cluster are chosen, resulting in the selection of $2k$ features. Contrarily to the ACA [17], this approach tends to avoid the selection of undesirable redundant features, being more interesting when one faces with overlapping clusters. In this case, we could roughly say that SSF is more suitable than ACA [17] for avoiding information loss, in the sense that the feature less correlated to the medoid would not be discarded by this filter. This feature may contain important information not encoded into the medoid, but it would be discarded in most of the cases by ACA, which selects the r features more correlated to the others in the same cluster.

The overall computational complexity of SSF is very similar to ACA, and it is estimated as $O(M^2)$ for a given value of k and as $O(M^3)$ when k is varied in the range of [2,M-1]. In what concerns the computational complexity of SSF in terms of the number of instances (N), it will depend upon the adopted correlation measure. In this work, all the correlation measures have a computational complexity of $O(N)$ (in the case of entropy-based measures, this holds when the algorithm PKID [19] is used for discretization, as done in this work).

III. Experimental Evaluation

Our experimental setting is based on the desire to evaluate the relative performance of the feature selection algorithms being studied under controlled conditions. With this purpose in mind, we have used ten datasets. Six of them are bioinformatics datasets used by [23]. These authors created five types of synthetic array datasets with error distributions derived from bioinformatics real data. These datasets (here called Bio1, Bio2, Bio3, Bio4, and Bio5) are composed of 400 genes (instances), described by 20 measurements (features). There are six approximately equal-sized clusters in each dataset. Four clusters represent sine waves shifted in phase relative to each other (a periodic pattern) and the two remaining ones represent linear functions (non-periodic). Error is added to the synthetic patterns (for each data point) using an experimentally derived error distribution. In addition, we tested the described feature selection algorithms in a real-world dataset (Yeast Galactose data [23]), which is composed of 20 measurements (nine single-gene deletions, one wild-type experiment with galactose and raffinose, nine deletions, and one wild-type experiment without galactose and raffinose) and 205 genes. In this dataset, the expression patterns reflect four functional categories (clusters). The datasets used in the experiments reported here take into account four repeated measurements, what may yield more accurate and more stable clusters. The repeated measurements are taken into account by averaging the expression levels over all repeated measurements. Other four datasets were artificially generated by using procedures inspired on the ideas by Milligan [24], [25]:

Table I
Summary of the employed datasets.

Dataset	N	M	# clusters (distribution - %)
Bio1, …, Bio5	400	20	6 (≈ equally distributed)
Yeast	205	20	4 (40.5 - 7.3 - 45.4 - 6.8)
10_250	250	10	5 (equally distributed)
12_200	200	12	4 (equally distributed)
20_250	250	20	5 (equally distributed)
1000_1000	1000	1000	5 (equally distributed)

- *10_250*: This dataset is composed of 250 instances (50 instances in each cluster) and 10 features (attributes). The first two features describe five clusters if combined, but individually each of them only allows recovering three clusters. The other eight features are *noisy*, having values derived from a normal distribution that is independent of the distribution of the clusters.
- *12_200*: This dataset is composed of 200 instances (50 instances in each cluster) and twelve features. The first feature is created using the data generator proposed in [25], and describes four clusters. The second feature describes two clusters. Features 3, 5, 7, 9, and 11 are correlated with the first feature having 10%, 20%, 30%, 40%, and 50% of noisy values, respectively. Features 4, 6, 8, 10, and 12 are correlated with the second feature and have 10%, 20%, 30%, 40%, and 50% of noisy values, respectively.
- *20_250*: This dataset is composed of 250 instances (50 instances in each cluster) and 20 features. The first two features describe five clusters if combined, but individually each of them only describes three clusters. Features 3, 5, 7, 9, and 11 are correlated with the first feature and have 10%, 20%, 30%, 40%, and 50% of noisy values, respectively. Features 4, 6, 8, 10, and 12 are correlated with the second feature and have 10%, 20%, 30%, 40%, and 50% of noisy values, respectively. The values for features 13,14,…, and 20 are derived from uniform probability distributions.
- *1000_1000*: This dataset consists of 1,000 instances and 1,000 features, and it was obtained by means of the data generator proposed in [25]. The first feature perfectly describes five clusters, whereas the remaining features make them overlapped.

A summary of the employed datasets is presented in Table I. In the performed experiments, feature selection by SSF/ACA was performed by running k-medoids/modes for a variable number of clusters ($k_{min}, k_{min}+1, \ldots, k_{max}-1, k_{max}$). For each value of k, 20 different partitions were generated, for that k-medoids/modes may get stuck at suboptimal solutions [22]. Thus, $20(k_{max}-k_{min}+1)$ partitions are obtained for each filter. Among them, the best one is chosen by SSF/ACA (as discussed in Sections II-B and II-C) for feature selection purposes. We set the minimum k value to 2 ($k_{min}=2$) and k_{max} was set according to the criteria for selecting features from each cluster (as discussed in Section II-C). More specifically, k_{max} for selecting one feature per cluster is M-1 ($k_{max}=M$-1), and for selecting two features per cluster it is $M/2$ ($k_{max}=M/2$). From a practical viewpoint, one can consider that these values somehow determine the size of the search space to be assessed, as well as the computational effort to find the corresponding solution. Therefore, domain knowledge, when available, can be incorporated into this approach in order to set those parameters in scenarios that present computational resources limitations.

As observed in Section II-A, 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. The authors [16] claim that it may be useful to control the representation of the data at different levels of details, performing some kind of exploratory data analysis. Despite the good results reported in [16], a number of experiments reported here illustrate that the choice of the k_{NN} value may be hard to be accomplished in practice. To make this point more clear, we run MMP by varying k_{NN} into the range of all its possible values — $[1, M\text{-}1]$ — for each dataset. For the filter named ACA , addressed in Section II-B, 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. Continuous features were discretized using the algorithm PKID [19] before running ACA and SSF when the *interdependence redundancy measure* and the *symmetrical uncertainty* are used.

The acronyms SSF-λ, SSF-ρ, SSF-R, SSF-S, MMP, ACA and All refer to the SSF method using the *maximal information compression index* (Eq. (1)), SSF using the *correlation coefficient* (Eq. (2)), SSF using the *interdependence redundancy measure* (Eq. (3)), SSF using the *symmetrical uncertainty* (Eq. (8)), Mitra et al.'s algorithm [16] (for which B, A, and W stand for the best, average, and worse results, respectively), Attribute Clustering Algorithm [17], and finally the results found by using all features of the dataset, respectively. A third parameter can also be found for both SSF and ACA, making reference to the number of selected features from each cluster. For instance, SSF-ρ-1 stand for SSF using the correlation coefficient and selecting one feature from each cluster.

The quality of the feature subsets found by the studied filters is assessed by means of the quality of the data partitions obtained by k-means, which was chosen due to its widespread use in practice [26]. Since k-means has the limitation of getting stuck at suboptimal solutions, we run it 50 (fifty) times for each dataset obtained from a feature selection algorithm. Then, the data partition that provides the most compact clusters (according to the average quadratic errors, computed from distances between instances and cluster centroids) is chosen. The quality of such partitions is evaluated by computing the well known Adjusted Rand Index (ARI) [27] and Jaccard Coefficient (JC) [28], which are external indices of partitional adequacy. In order to provide some reassurance about the validity and non-randomness of the obtained results, we present the results of statistical tests by following the approach proposed in [29]. In brief, this approach is aimed at comparing multiple algorithms on multiple datasets, and it is based on the use of the well-known 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 pair-wise comparisons between algorithms.

Table II summarizes the obtained ARI values. The last column of this table refers to the average rank obtained from performing the Friedman test[1]. It is interesting to observe from Table II that the two best ranked algorithms (MMP and SSF-ρ-1) provided better or equal results in more than 80% of the datasets when compared to the use of all features. Due to space limitations we omit the JC values, but we shall note that the obtained results are very consistent with those achieved by the ARI. Independently of the clustering external criterion used (either ARI or JC), the statistical procedure just described indicates that are significant differences only between MMP-B and SSF-λ-1/2 (at $\alpha = 10\%$). Table III presents the number of wins, ties and losses between all pairs of algorithms considering the ARI values, e.g., "1/3/6" in the 7^{th} line and 3^{rd} column indicates that SSF-ρ-1 was better than MMP-B only once, there were three ties, and it was worse six times. It can be seen that, despite the good results obtained in the best case for MMP (MMP-B), ACA and SSF showed better results in 50-60% of the datasets in relation to MMP-A (average case), which suggests that both methods can be preferred when the value of the k_{NN} parameter is unkwnown, especially if computational efficiency is of concern (as discussed in the sequel).

[1]We conservatively only considered the best results achieved by MMP (MMP-B) [16] in the statistical tests.

Table II
SUMMARY OF THE ADJUSTED RAND INDEX VALUES.

Method	10_250	20_250	12_200	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	1000_1000	Ranking (Friedman)
All	1.00	0.95	1.00	0.53	0.80	1.00	0.82	0.55	0.97	0.78	5.80
MMP-B	1.00	0.98	1.00	0.82	0.97	1.00	0.90	0.81	0.98	0.93	3.10
MMP-A	0.57	0.53	0.56	0.64	0.78	0.92	0.75	0.69	0.76	0.78	—
MMP-W	0.02	0.02	0.26	0.49	0.53	0.55	0.52	0.48	0.09	0.28	—
SSF-λ-1	0.49	0.11	0.43	0.81	0.50	1.00	0.77	0.54	0.62	0.56	9.75
SSF-λ-2	0.31	0.17	0.45	0.59	0.80	1.00	0.79	0.55	0.65	0.90	8.00
SSF-ρ-1	1.00	1.00	1.00	0.53	0.91	1.00	0.82	0.54	0.75	0.78	5.55
SSF-ρ-2	1.00	1.00	0.52	0.53	0.81	1.00	0.80	0.54	0.66	0.78	6.75
ACA-1	0.49	1.00	1.00	1.00	0.62	0.78	0.80	0.79	0.60	0.78	6.40
ACA-2	1.00	0.99	0.99	1.00	0.54	0.81	0.92	0.54	0.71	1.00	5.70
SSF-R-1	0.24	0.97	1.00	1.00	0.91	0.81	0.93	0.53	0.96	0.76	6.50
SSF-R-2	0.25	0.56	0.52	0.82	0.92	0.78	0.8	0.56	0.97	0.67	7.30
SSF-S-1	0.24	1.00	1.00	1.00	0.91	0.81	0.87	0.53	0.96	0.76	6.30
SSF-S-2	0.25	1.00	0.52	0.82	0.92	0.78	0.79	0.56	0.97	0.67	6.85

Table III
WIN/TIE/LOSS FOR METHODS IN THE 1^{st} COLUMN (IN RELATION TO ADJUSTED RAND INDEX VALUES).

Method	All	MMP-B	MMP-A	SSF-λ-1	SSF-λ-2	SSF-ρ-1	SSF-ρ-2	ACA-1	ACA-2	SSF-R-1	SSF-R-2	SSF-S-1	SSF-S-2
All	—	0/3/7	7/1/2	8/1/1	5/3/2	2/6/2	4/4/2	5/2/3	5/1/4	5/1/4	6/1/3	5/1/4	5/1/4
MMP-B	7/3/0	—	10/0/0	9/1/0	9/1/0	6/3/1	7/2/1	7/1/2	5/1/4	7/1/2	9/1/0	7/1/2	8/1/1
MMP-A	2/1/7	0/0/10	—	7/0/3	6/0/4	3/1/6	4/1/5	4/1/5	4/0/6	4/0/6	5/0/5	4/0/6	5/0/5
SSF-λ-1	1/1/8	0/1/9	3/0/7	—	2/1/7	1/2/7	1/2/7	2/1/7	1/1/8	3/0/7	2/0/8	3/0/7	2/0/8
SSF-λ-2	2/3/5	0/1/9	4/0/6	7/1/2	—	3/1/6	3/1/6	4/0/6	3/0/7	4/0/6	3/0/7	4/0/6	3/1/6
SSF-ρ-1	2/6/2	1/3/6	6/1/3	7/2/1	6/1/3	—	4/6/0	5/3/2	5/2/3	5/2/3	6/0/4	4/3/3	5/1/4
SSF-ρ-2	2/4/4	1/2/7	5/1/4	7/2/1	6/1/3	0/6/4	—	4/3/3	3/2/5	5/0/5	4/2/4	4/1/5	4/2/4
ACA-1	3/2/5	2/1/7	5/1/4	7/1/2	6/0/4	2/3/5	3/3/4	—	4/1/5	4/2/4	6/2/2	3/3/4	6/2/2
ACA-2	4/1/5	4/1/5	6/0/4	8/1/1	7/0/3	3/2/5	5/2/3	5/1/4	—	4/2/4	7/0/3	4/2/4	6/0/4
SSF-R-1	4/1/5	2/1/7	6/0/4	7/0/3	6/0/4	3/2/5	5/0/5	4/2/4	4/2/4	—	6/0/4	1/8/1	5/0/5
SSF-R-2	3/1/6	0/1/9	5/0/5	8/0/2	7/0/3	4/0/6	4/2/4	2/2/6	3/0/7	4/0/6	—	4/0/6	1/8/1
SSF-S-1	4/1/5	2/1/7	6/0/4	7/0/3	6/0/4	3/3/4	5/1/4	4/3/3	4/2/4	1/8/1	6/0/4	—	5/1/4
SSF-S-2	4/1/5	1/1/8	5/0/5	8/0/2	6/1/3	4/1/5	4/2/4	2/2/6	4/0/6	5/0/5	1/8/1	4/1/5	—

Table IV
SUMMARY OF THE NUMBER OF SELECTED FEATURES.

Method	10_250	20_250	12_200	Bio1	Bio2	Bio3	Bio4	Bio5	Yeast	1000_1000	Ranking (Friedman)
MMP-B	6	14	11	17	14	4	6	9	8	4	8.5
MMP-W	1	1	2	8	1	1	19	4	1	1	—
SSF-λ-1	2	2	2	3	10	9	4	8	2	2	3.8
SSF-λ-2	4	3	4	6	20	18	8	16	3	4	7.7
SSF-ρ-1	4	5	2	6	4	8	3	5	2	4	4.2
SSF-ρ-2	8	10	4	12	8	16	6	10	3	8	8.3
ACA-1	2	2	2	2	2	2	2	2	2	2	1.6
ACA-2	4	4	4	3	3	3	3	3	4	4	4.4
SSF-R-1	4	7	2	8	7	10	3	6	9	2	5.2
SSF-R-2	8	14	4	12	11	15	6	11	15	3	8.9
SSF-S-1	4	6	2	8	7	10	2	6	9	2	4.8
SSF-S-2	8	11	4	12	11	15	3	11	15	3	8.3

Now let us shed light on some particular results obtained for specific datasets. Considering the 10_250 dataset, only SSF-ρ, ACA-2, and MMP-B were able to select the two relevant features. In the 12_200 dataset, ACA and SSF (all versions, except SSF-λ) were able to identify the correct clusters of features (given by even and odd "feature labels"). In this particular dataset, the differences observed for the ARI values obtained by ACA-2 and SSF-2 versions can be explained by observing the nature of this dataset. More precisely, SSF-2 selects, from each cluster, its medoid and the feature less correlated with the medoid. Thus, it follows that SSF-2 selects noisy features, which, by the construction of the dataset, are the less correlated with the medoid. Such noisy features naturally tend to deteriorate the quality of the data partitions induced by k-means. For the 20_250 dataset, all algorithms (except SSF-λ) were able to select the two relevant features.

Table IV presents the number of selected features by each feature selection method[2]. The last column presents the average rank obtained from the Friedman test[1]. In summary, ACA and SSF-λ-1 provided better results than MMP in our study. Significant differences were observed between SSF-λ-1 and (SSF-ρ-2, MMP-B, SSF-R-2, and SSF-S-2) and between ACA-1 and (SSF-λ-2, SSF-ρ-2, MMP-B, SSF-R-2, and SSF-S-2) at $\alpha = 10\%$.

[2]Results for MMP-A are not included in Table IV since they do not represent a specific subset of features but the average result obtained by varying the value of the k_{NN} parameter in the range [1,M-1].

Let us now take into account computational efficiency issues. We are interested in empirically evaluating the magnitude of the constant terms neglected by the asymptotic time complexity analysis reported in Sections II-A-II-C. To do so, we implemented all the studied algorithms in Java, 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. Detailed results are not shown here due to space limitations. Instead, we focus on providing a summary of the results of the statistical tests, which suggest ($\alpha = 5\%$) that: (i) SSF-λ-2, SSF-ρ-2, ACA-2, SSF-R-2, and SSF-S-2 presented better performance than MMP-B, SSF-ρ-1, and SSF-S-1; (ii) SSF-λ-2 presented better performance than SSF-R-1 and ACA-1; (iii) SSF-λ-1 presented better performance than MMP-B.

IV. Conclusions

In this work we analyzed three unsupervised feature selection methods for clustering problems. More specifically, the filters proposed by Mitra et al. (MMP) [16], Au et al. (ACA) [17], and Covões et al. (SSF) [18] were empirically compared. Experiments in ten datasets showed that both ACA [17] and SSF [18] provide competitive results (especially if computational efficiency is under consideration) to those found by MMP [16], which is considered a state of the art method for feature selection in the statistical pattern recognition field [2]. ACA and SSF showed similar performances. However, the user needs to choose the number of features to be selected when running ACA, whereas SSF selects it automatically, and thus it can be preferred in certain applications. In what concerns the different correlation measures analyzed for SSF, significant differences have not been observed between the *interdependence redundancy measure* and the *symmetrical uncertainty*, whereas the *correlation coefficient* has shown better results than the *maximal information compression index* in most of the employed datasets, as well as it has shown competitive results with the non-linear correlation measures. In addition, the SSF variant here investigated that involves the selection of two features for each cluster (medoid and feature less correlated with the medoid) has presented good results for the bioinformatics datasets.

Although interesting results have been reported in this comparative study, there are several issues that can be investigated in the future. For example, provided that SSF in principle 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 an interesting future work. Also, a more comprehensive experimental evaluation, comprising more datasets, is in order.

Acknowledgments

We acknowledge the Brazilian Research Agencies CNPq and FAPESP for their financial support to this work. We also thank Lucas Vendramin for having provided the data generator used in some of the reported experiments.

References

[1] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[2] L. Yu and H. Liu, "Efficient feature selection via analysis of relevance and redundancy," *Journal of Machine Learning Research*, vol. 5, pp. 1205–1224, 2004.

[3] P. Arabie and L. J. Hubert, *An Overview of Combinatorial Data Analysis*. World Scientific Publishing Company, 1999, ch. 1, pp. 5–64.

[4] B. S. Everitt, *Cluster Analysis*. Edward Arnold and Halsted Press, 2001.

[5] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990.

[6] M. Dash and H. Liu, "Feature selection for clustering," in *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications (PADKK-2000)*. Springer-Verlag, 2000, pp. 110–121.

[7] M. H. C. Law, M. A. T. Figueiredo, and A. K. Jain, "Simultaneous feature selection and clustering using mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1154–1166, 2004.

[8] V. Roth and T. Lange, "Feature selection in clustering problems," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[9] K. Mao, "Identifying critical variables of principal components for unsupervised feature selection," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 35, no. 2, pp. 339–344, April 2005.

[10] A. Vellido, "Assessment of an unsupervised feature selection method for generative topographic mapping," in *ICANN (2)*, ser. Lecture Notes in Computer Science, S. D. Kollias, A. Stafylopatis, W. Duch, and E. Oja, Eds., vol. 4132. Springer, 2006, pp. 361–370.

[11] Y. Li, M. Dong, and J. Hua, "Localized feature selection for clustering," *Pattern Recognition Letters*, vol. 29, no. 1, pp. 10 – 18, 2008.

[12] Y. Hong, S. Kwong, Y. Chang, and Q. Ren, "Unsupervised feature selection using clustering ensembles and population based incremental learning algorithm," *Pattern Recognition*, vol. 41, no. 9, pp. 2742 – 2756, 2008.

[13] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[14] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.

[15] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*. Springer, 1998.

[16] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised feature selection using feature similarity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 301–312, 2002.

[17] W.-H. Au, K. C. C. Chan, A. K. C. Wong, and Y. Wang, "Attribute clustering for grouping, selection, and classification of gene expression data," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 2, no. 2, pp. 83–101, 2005.

[18] T. F. Covões, E. R. Hruschka, L. N. de Castro, and Átila Menezes dos Santos, "A cluster-based feature selection approach," in *Hybrid Artificial Intelligence Systems, 4th International Conference (HAIS-2009)*, ser. Lecture Notes in Artificial Intelligence, vol. 5572. Springer-Verlag, 2009, pp. 168–176.

[19] Y. Yang and 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.

[20] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, Washington DC, USA, 2003.

[21] E. R. Hruschka, R. J. Campello, and L. N. de Castro, "Evolving clusters in gene-expression data," *Information Sciences*, vol. 176, no. 13, pp. 1898 – 1927, 2006.

[22] T. Zhang, R. Ramakrishnan, and M. Livny, "BIRCH: a new data clustering algorithm and its applications," *Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 141–182, Jun. 1997.

[23] K. Yeung, M. Medvedovic, and R. Bumgarner, "Clustering gene-expression data with repeated measurements," *Genome Biology*, vol. 4, no. 5, p. R34, 2003.

[24] G. Milligan, "An examination of the effect of six types of error perturbation on fifteen clustering algorithms," *Psychometrika*, vol. 45, no. 3, pp. 325–342, September 1980.

[25] ——, "A monte carlo study of thirty internal criterion measures for cluster analysis," *Psychometrika*, vol. 46, no. 2, pp. 187–199, June 1981.

[26] 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, and D. Steinberg, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, January 2008.

[27] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, pp. 193–218, 1985.

[28] P. Jaccard, "Nouvelles recherches sur la distribution florale," *Bulletin de la Société Vaudoise de Sciences Naturelles*, vol. 44, pp. 223–370, 1908.

[29] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, 2006.