Information Sciences 322 (2015) 150–160

Contents lists available at ScienceDirect

Information Sciences

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

Kernel Penalized K-means: A feature selection method based on Kernel K-means

Sebastián Maldonado [a,*], Emilio Carrizosa [b], Richard Weber [c]

[a] Universidad de los Andes, Mons. Álvaro del Portillo 12455, Las Condes, Santiago, Chile
[b] Facultad de Matemáticas, Universidad de Sevilla, Sevilla, Spain
[c] Departamento de Ingeniería Industrial, FCFM, Universidad de Chile, Santiago, Chile

ARTICLE INFO

Article history:
Received 11 June 2014
Received in revised form 23 October 2014
Accepted 11 June 2015
Available online 19 June 2015

Keywords:
Feature selection
Kernel K-means
Clustering

ABSTRACT

We present an unsupervised method that selects the most relevant features using an embedded strategy while maintaining the cluster structure found with the initial feature set. It is based on the idea of simultaneously minimizing the violation of the initial cluster structure and penalizing the use of features via scaling factors. As the base method we use Kernel K-means which works similarly to K-means, one of the most popular clustering algorithms, but it provides more flexibility due to the use of kernel functions for distance calculation, thus allowing the detection of more complex cluster structures. We present an algorithm to solve the respective minimization problem iteratively, and perform experiments with several data sets demonstrating the superior performance of the proposed method compared to alternative approaches.

1. Introduction

Clustering aims at discovering the internal organization of a dataset by finding structure within the data in the form of clusters [31]. In this paper we focus on partitioning as opposed to hierarchical cluster methods. Intuitively, the resulting partitions should be characterized by within-cluster similarity and between-cluster dissimilarity.

Assuming that a *good* partition can be obtained by a particular clustering method (e.g. K-means [22] or Kernel K-means [11]) using all available variables, we propose to select the most important features by reducing the dimensionality while *maintaining the initial cluster structure*.

We use Kernel K-means as clustering technique which allows recognizing more complex cluster shapes than traditional K-means because of its non-linear distance function based on kernels. The main goal of this work is to reduce the dimensionality of the feature space while adjusting the respective kernel shape. In this particular case we use an anisotropic Gaussian kernel.

We propose a backward elimination procedure based on an iterative algorithm that updates the kernel variables via scaling factors in order to adjust the shape of the kernel. A concave penalty function that approximates the cardinality of the scaling factors is also included in order to encourage feature selection.

In Section 2 we present the approaches that are necessary to introduce our proposed method, such as feature selection for clustering, alternative techniques for feature ranking, and Kernel K-means which is the base method we build on. Section 3

* Corresponding author. Tel.: +56 2 26181874.
E-mail address: smaldonado@uandes.cl (S. Maldonado).

http://dx.doi.org/10.1016/j.ins.2015.06.008
0020-0255/

introduces the proposed Kernel-Penalized K-means. Experimental results are provided in Section 4. Finally, Section 5 concludes this paper and indicates possible future developments.

2. Feature selection for clustering

Section 2.1 provides a brief introduction regarding the use of feature selection for clustering. Subsequently, in Sections 2.2, 2.3, and 2.4 we present four approaches for feature selection that will be used in the experimental part of this paper. Finally, Section 2.5 presents Kernel K-means, the clustering method on which we built our proposed feature selection approach.

2.1. Introduction to feature selection for clustering

Feature selection is an important data mining topic, especially in high-dimensional applications. It addresses the dimensionality reduction problem by finding a subset of available features with which to build a good predictive model. Feature selection has several advantages: first, a low-dimensional representation of the data reduces the risk of *overfitting* caused by the *curse of dimensionality*, improving the model's generalization ability by decreasing its complexity [14,23]. Additionally, appropriate selection of the most relevant features allows better interpretation of the predictive model. This is particularly important in application areas such as business analytics and biotechnology since practitioners often consider data mining methods to be *black boxes*, and are therefore hesitant to use them [7]. The understanding of the process that generates the data is also of crucial importance in business analytics, for example, to identify the relevant customer attributes that lead to a better understanding of their behavior.

Most distance-based clustering techniques assume that all variables are equally important when computing the similarity between data points. This assumption does not necessarily hold true since attributes have a different impact on clustering: while some of them may be relevant for determining the structure of the problem, others may have no impact on the clustering process or may adversely affect the model, *blurring* the clusters [1]. As a consequence, clustering algorithms may fail at identifying the underlying structure of the data in high-dimensional applications with a great number of irrelevant and noisy features. In those cases, feature selection could be useful for detecting and removing those attributes, providing a more efficient clustering process.

Although a plethora of feature selection techniques has been proposed in the literature for supervised learning, only relatively few methods are available for unsupervised learning. One such approach was presented by Dyer et al. [13] who provide theoretic results for exact feature selection in subspace clustering. Recently, techniques for feature weighting in unsupervised learning have been proposed [17], but they use the concept of probabilistic clustering. Furthermore, approaches that attempt to detect irrelevant variables by measuring the correlation between features and class labels, are not suitable for unsupervised learning since such labels are not available.

Feature selection algorithms can be categorized into supervised, unsupervised, or semi-supervised algorithms depending on the utilization of the respective class label information [1]. Furthermore, four main categories of methods for feature selection for clustering can be identified [1,14]: filter, wrapper, hybrid, and embedded approaches.

Filter methods remove features independently from the clustering algorithm, usually before applying any technique. Relevance is assessed using a predefined criterion, which could be redundancy [34] or data entropy [9], for instance.

Wrapper methods interact with the respective clustering technique and explore the entire set of variables to identify good feature subsets according to their influence in the clustering, which is computationally more demanding, but often provides better results than filter methods [1,10]. A common wrapper strategy is Sequential Backward Elimination (SBE) [19]. SBE starts with all available variables and tests them one by one, deleting any variable that is not relevant. Wrapper methods for clustering can consider k-means [18], EM (Expectation–Maximization) [12], and entropy measures [10], among others.

Hybrid approaches combine both filter and wrapper strategies by selecting candidate subsets efficiently via filter methods, while assessing them according to their cluster quality via wrapper approaches. Using such a procedure alleviates the computational costs incurred by wrapper models, while improving the quality of the clusters by providing knowledge of the clustering method in advance.

Embedded methods select features simultaneously with model construction. Although several of such approaches have been proposed for supervised learning (see e.g. [23,25,26]), there are very few embedded methods in the clustering literature. The proposed methodology introduced in Section 3 of this work belongs to this category.

2.2. Feature ranking via PCA and GHA

Extracting principal components from a set of attributes that represent observed instances can be used as a filter technique for unsupervised feature selection. Various methods have been proposed in the literature to perform such extraction of principal components, including: Principal Component Analysis (PCA), Kernel PCA, Singular Value Decomposition (SVD), non-linear PCA, and the Generalized Hebbian Algorithm (GHA); see Lee and Verleysen [20] for an overview on the respective approaches. The main drawbacks of using the extracted components instead of the original attributes are the difficulty of understanding the clusters based on the new features, and the fact that all the available information is actually used to

construct the components [1,10]. These methods can be adapted to rank the original attributes in terms of their influence on the components. In particular, we use the weights associated with each attribute that construct the first component to rank the variables in terms of relevance. As a consequence, the focus of these methods is on removing redundancy rather than detecting those attributes that are important for generating the structure of the data.

In our work we use Principal Component Analysis (PCA) and Generalized Hebbian Algorithm (GHA) for feature ranking. We do not present a detailed description of PCA here but refer the reader to Lee and Verleysen [20]. GHA offers certain advantages over PCA since it determines the principal components in an iterative way rather than calculating the covariance matrix as is the case in PCA. This is particularly appealing in the case of high-dimensional data sets which are of special interest in our study.

GHA was presented in Sanger [29] and generalizes the learning rule proposed in Hebb [16]. Inspired by insights on synaptic plasticity in neuroscience, this rule basically says that the weight between two neurons increases if both neurons are active at the same time. Applying this idea to the extraction of principal components can be formalized as follows:

$$y = \sum_{i=1}^{m} w_i x_i \tag{1}$$

where x_i represents input neuron (attribute) i and w_i its weight, $i = 1, \ldots, m$. y is the single output neuron of a one-layer feedforward neural network and represents the first principal component as will be shown below. GHA uses the following iterative updating rule to determine the attribute weights:

$$w_i(n+1) := w_i(n) + ay(n)[x_i(n) - y(n)w_i(n)] \tag{2}$$

where n is the iteration counter and a is a positive learning rate.

It can be shown that y is the first principal component [28]. By using a single-layer feedforward neural network with k output neurons the first k principal components are determined in an analogous way.

2.3. Spectral feature selection

Spectral feature selection (SPEC) [34] is a unifying framework for supervised and unsupervised feature selection based on spectral graph theory [8]. To bridge the gap between both approaches SPEC focuses on the so-called *target concept* rather than on specific information based on class labels (in the case of supervised learning) or innate cluster structures (in the case of unsupervised learning).

Since pairwise instance similarity is widely used in both supervised and unsupervised learning to describe the relationships among instances, SPEC starts with a matrix $\mathbf{S}$ of pairwise similarities of instances. By using, for example, the Radial Basis Function as the similarity measure between two instances i and j with their respective feature vectors x_i and x_j ($i.j \in \{1, \ldots, N\}$), we get:

$$\mathbf{S}_{ij} := e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}} \tag{3}$$

The main idea behind SPEC is that instances that are close to each other in the feature space (i.e. that are similar to each other) behave similarly and therefore should belong to the same *target concept*, here cluster. Spectral graph theory reveals the structure in the related graph induced by the similarity matrix $\mathbf{S}$ and assigns the corresponding weights to each feature using the Laplacian $\mathbf{L}$ and its normalized version $\mathcal{L}$.

Using the similarity matrix $\mathbf{S}$, an undirected graph $\mathbf{G} = (\mathrm{V}, \mathrm{E})$ is constructed where each vertex $v_i \in \mathrm{V}$ represents an instance i and between any two instances i and j there is an edge $e_{ij} \in \mathrm{E}$ with weight $w_{ij} = \mathbf{S}_{ij}$. For this graph an adjacency matrix $\mathbf{W}$ and a degree matrix $\mathbf{D}$ are computed as follows:

$$\mathbf{W}_{ij} := \mathbf{S}_{ij}$$

$$\mathbf{D}_{ij} := \mathbf{d}_i \text{ if } i = j \text{ and 0 otherwise; with } \mathbf{d}_i = \sum_{k=1}^{N} w_{ik}$$

The Laplacian matrix $\mathbf{L}$ and the normalized Laplacian $\mathcal{L}$ are calculated as follows:

$$\mathbf{L} = \mathbf{D} - \mathbf{W} \quad \text{and} \quad \mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$$

Feature weights are determined in SPEC based on the following three functions (see [34]):

$$\varphi_1(F_i) = \hat{f}_i^T \mathcal{L} \hat{f}_i = \sum_{j=1}^{n-1} \alpha_j^2 * \lambda_j, \tag{4}$$

where F_i represents feature i, $f_i \in \mathbb{R}^n$ is the vector associated with feature F_i, λ_j is the jth eigenvalue of the Laplacian matrix $\mathcal{L}$, (λ_j, ξ_j) is the eigensystem of matrix $\mathcal{L}$, and $\alpha_j = \cos\theta_j$ with θ_j the angle between f_i and ξ_j. Dividing $\varphi_1(F_i)$ by $\sum_{j=1}^{n-1} \alpha_j^2$, we obtain its normalized version as follows:

$$\varphi_2(F_i) = \frac{\sum_{j=1}^{n-1} \alpha_j^2 * \lambda_j}{\sum_{j=1}^{n-1} \alpha_j^2} \tag{5}$$

For situations where the number of clusters K is known, the following function has been proposed:

$$\varphi_3(F_i) = \sum_{j=1}^{K-1} (2 - \lambda_j) * \alpha_j^2 \tag{6}$$

For each one of these functions, extensions for feature ranking have been suggested. The experimental part of the present paper contains only clustering applications with a few classes, including a case in which the authors of the SPEC method propose to use the extension of $\varphi_1(F_i)$ [34].

$$\hat{\varphi}_1(F_i) = \hat{f}_i^T \gamma(\mathcal{L}) \hat{f}_i, \quad i = 1, \ldots, m \tag{7}$$

These weights are used for feature ranking and feature selection in Section 4.

2.4. Multi-cluster feature selection

Similar to SPEC, as presented in Section 2.3, the method for Multi-Cluster Feature Selection (MCFS) starts with an undirected graph $\mathbf{G} = (\mathrm{V}, \mathrm{E})$, for which the Laplacian matrix $\mathbf{L}$ is calculated; see [6]. For this calculation, different weighting schemes $W_{i,j}$ for nodes $i, j \in \mathrm{V}$ have been proposed. Solving the following generalized eigenproblem:

$$\mathbf{L}y = \lambda D y \tag{8}$$

leads to eigenvectors y_k that are used in the error minimization problem:

$$\min_{a_k} \|y_k - X^T a_k\|^2 + \beta \|a_k\|_1 \tag{9}$$

which has the following equivalent formulation:

$$\begin{aligned} &\min_{a_k} \ \|y_k - X^T a_k\|^2 \\ &s.t. \quad \|a_k\|_1 \leqslant \gamma \end{aligned} \tag{10}$$

where $\|\cdot\|$ denotes the Euclidean norm, while $\|\cdot\|_1$ denotes the Manhattan norm. This optimization problem can be solved by using the Least Angle Regression (LAR) algorithm, as suggested in Cai et al. [6].

The previously described procedure provides K coefficient vectors $a_k \in \mathbb{R}^M, k = 1, \ldots, K$ where K is the number of clusters and M is the number of feature candidates. For these vectors the MCFS score of feature j is defined as follows:

$$\mathrm{MCFS}(j) = \max_k |a_{k,j}| \tag{11}$$

Features are now ordered according to their MCFS score and the first d features are selected.

2.5. Kernel K-means

In this subsection we briefly present the method Kernel K-means. For a more detailed presentation, see Dhillon et al. [11].

The Kernel K-means algorithm assigns each training observation $i \in \{1, \ldots, N\}$ to one (and only one) of the K clusters available. These assignments can be characterized by an encoder C where $C(i) = k$ means that the ith instance is assigned to the kth cluster; $k \in \{1, \ldots, K\}$ [15]. This is performed by minimizing the total within cluster variance (also known as *energy*) in a greedy fashion, based on a two-step procedure. For a given encoder C, the total cluster variance is minimized with respect to a set of decision variables $\{m_1, \ldots, m_K\}$, which are the means of the current clusters (i.e. the cluster centroids). The Kernel K-means algorithm follows:

Algorithm 1. Kernel K-means algorithm

Random initialization of cluster centroids.
repeat
1. Given a current set of centroids $\{m_1, \ldots, m_K\}$, the total cluster variance is minimized by assigning each observation to the closest (current) cluster mean. That is, the cluster to which a data point i should be assigned is given by:

$$C(i) = argmin_{1 \leqslant k \leqslant K} \|\phi(\mathbf{x}_i) - m_k\|^2 \tag{12}$$

2. For a given cluster assignment, i.e. for a given encoder C, update the cluster centroids in the same way as in traditional K-means.

until Convergence is reached (assignments do not change)

Eq. (12) can be rewritten using Kernel functions:

$$\|\phi(\mathbf{x}_i) - m_k\|^2 = \|\phi(\mathbf{x}_i)\|^2 - 2\langle\phi(\mathbf{x}_i), m_k\rangle + \|m_k\|^2 = K(\mathbf{x}_i,\mathbf{x}_i) - \frac{2}{N_k}\sum_{i':C(i')=k} K(\mathbf{x}_i,\mathbf{x}_{i'}) + \frac{1}{N_k^2}\sum_{i:C(i)=k}\sum_{i':C(i')=k} K(\mathbf{x}_i,\mathbf{x}_{i'}) \tag{13}$$

where $K(\mathbf{x}_i,\mathbf{x}_{i'})$ is the kernel matrix of the training set for two examples $i, i' \in \{1,\ldots,N\}$. Following the notation used by Hastie et al. [15], the sum over $\{i : C(i) = k\}$ indicates all the instances i that are assigned to cluster k.

3. Proposed method for Kernel-Penalized feature selection

Kernel-Penalized K-means (KP-Kmeans), an embedded method for feature selection using Kernel K-means, is proposed in this section. The reasoning behind our approach is that we can construct a partition similar to the one obtained by using all features but selecting just the most relevant ones by penalizing their use via a concave approximation of the zero norm. Specifically, this is achieved by modifying the shape of an anisotropic Gaussian kernel used for Kernel K-means.

In this work we use the anisotropic Gaussian Kernel [23,26]:

$$K(\mathbf{x}_i,\mathbf{x}_{i'},\boldsymbol{v}) = \exp\left(-\frac{||\boldsymbol{v}*\mathbf{x}_i - \boldsymbol{v}*\mathbf{x}_{i'}||^2}{2}\right) \tag{14}$$

where $*$ denotes the componentwise vector product operator, which is defined as $\mathbf{a}*\mathbf{b} = (a_1b_1,\ldots,a_nb_n)$.

The idea is to modify the Gaussian kernel that generates the original structure via scaling factors of an anisotropic Gaussian kernel. The number of the scaling factors is penalized in the objective function (via a zero norm approximation), maintaining approximately the original clustering solution (i.e. the solution obtained using all features).

The following feature penalization function is proposed, in which the approximation parameter β is also considered.

$$f(\boldsymbol{v}) = \mathbf{e}^T(\mathbf{e} - \exp(-\beta\boldsymbol{v}) = \sum_{j=1}^{n}[1 - \exp(-\beta v_j)] \tag{15}$$

Bradley and Mangasarian suggest setting $\beta = 5$ since this value gave useful results in various settings [5]. Assuming that all elements are already assigned to a cluster k (for instance, using Kernel K-means over all variables), i.e. considering an encoder C with $C(i) = k$ for all observations $i \in \{1,\ldots,N\}$, the following family of inequalities must hold in order to respect this structure completely:

$$\|\phi(\mathbf{x}_i) - m_k\|^2 \leqslant \|\phi(\mathbf{x}_i) - m_{k'}\|^2, \quad 1 \leqslant i \leqslant N,\ 1 \leqslant k' \leqslant K,\ k' \neq k \tag{16}$$

which essentially means that the distance of an element i represented by its feature vector $\mathbf{x}_i$ to its centroid m_k should be smaller than its distance to all other clusters k' represented by their centroids m'_k.

Since our goal is to ensure that all elements should be labeled maintaining the original cluster structure (a problem that could also be computationally prohibitive, and would not allow us to improve the clustering by removing noisy attributes), we propose the following metric that we want to minimize by introducing scaling factors:

$$\mathrm{ER}(C) = \sum_{k=1}^{K}\sum_{i:C(i)=k}\sum_{k'\neq k}\frac{\|\phi(\mathbf{x}_i) - m_k\|^2}{\|\phi(\mathbf{x}_i) - m_{k'}\|^2} \tag{17}$$

We refer to (17) as *energy ratio* (ER) for a given encoder C. Minimizing ER is a relaxation of (16) since it allows violations of some of the respective inequalities but minimizes the sum of their violations.

To incorporate the concept of the anisotropic Gaussian kernel in our proposal, we rewrite measure (17) according to the derivation presented in Eq. (13):

$$\mathrm{ER}(C) = \sum_{k=1}^{K}\sum_{i:C(i)=k}\sum_{k'\neq k}\frac{H_{\boldsymbol{v},k}}{H_{\boldsymbol{v},k'}}, \tag{18}$$

where

$$H_{\boldsymbol{v},k} = K(\mathbf{x}_i,\mathbf{x}_i,\boldsymbol{v}) - \frac{2}{N_k}\sum_{i':C(i')=k} K(\mathbf{x}_i,\mathbf{x}_{i'},\boldsymbol{v}) + \frac{1}{N_k^2}\sum_{i:C(i)=k}\sum_{i':C(i')=k} K(\mathbf{x}_i,\mathbf{x}_{i'},\boldsymbol{v}). \tag{19}$$

The minimization problem we propose to solve includes both objectives simultaneously: the minimization of the energy ratio (to maintain the original cluster structure) and the penalization of feature usage (to select features), as follows:

$$\begin{aligned}\min_{\boldsymbol{v}}\quad & F(\boldsymbol{v},C) = \sum_{k=1}^{K}\sum_{i:C(i)=k}\sum_{k'\neq k}\frac{H_{\boldsymbol{v},k}}{H_{\boldsymbol{v},k'}} + \lambda f(\boldsymbol{v}) \\ & v_i \geqslant 0\ \forall i \in \{1,\ldots,N\}\end{aligned} \tag{20}$$

where λ is a predefined parameter that controls the tradeoff between the feature penalization given by Eq. (15) and the minimization of the energy ratio.

Instead of solving problem (20), we propose the following iterative algorithm, which extends the ideas of KP-SVM presented in Maldonado et al. [23]. The algorithm updates the kernel width $\boldsymbol{v}$ iteratively via successive gradient descent steps, eliminating those features that fall below a threshold ϵ, as follows:

Algorithm 2. Kernel width updating and feature elimination

1. Start with $\boldsymbol{v} = v_0\mathbf{e}$;
2. **repeat**
3. $C \leftarrow$ Kernel k-means clustering
4. **repeat**
5. $\boldsymbol{v} \leftarrow \boldsymbol{v} - \gamma\nabla F(\boldsymbol{v}, C)$
6. **for all**$(v_j < \epsilon)$ **do**
7. $v_j = 0$;
8. **end for**
9. **until** $\exists j \mid v_j = 0$
10. **until** Convergence is reached

The algorithm initializes with an isotropic kernel width obtained from a predefined value v_0, and then performs Kernel K-means clustering iteratively to obtain the encoder function C and updates the kernel widths until one or more elements of vector $\boldsymbol{v}$ fall below the threshold ϵ. The algorithm stops when convergence is reached, i. e. when kernel widths do not change between iterations or no features are eliminated after a predefined number of iterations.

In line 5. the algorithm adjusts the kernel variables by using the gradient descent procedure, incorporating a gradient parameter γ. In this step the algorithm computes the gradient of the objective function in formulation (20) for a given encoder C, obtained by training Kernel K-means clustering using Algorithm 1. For feature j, the gradient of $F(\boldsymbol{v}, C)$ is:

$$\nabla_j F(\boldsymbol{v}, C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{k' \neq k} \nabla_j \frac{H_{\boldsymbol{v},k}}{H_{\boldsymbol{v},k'}} + \lambda \nabla_j f(\boldsymbol{v}) \tag{21}$$

where

$$\nabla_j \frac{H_{\boldsymbol{v},k}}{H_{\boldsymbol{v},k'}} = \frac{\nabla_j H_{\boldsymbol{v},k} \cdot H_{\boldsymbol{v},k'} - \nabla_j H_{\boldsymbol{v},k'} \cdot H_{\boldsymbol{v},k}}{H^2_{\boldsymbol{v},k'}} \tag{22}$$

and

$$\nabla_j H_{\boldsymbol{v},k} = \nabla_j K(\mathbf{x}_i, \mathbf{x}_i, \boldsymbol{v}) - \frac{2}{N_k} \sum_{i':C(i')=k} \nabla_j K(\mathbf{x}_i, \mathbf{x}_{i'}, \boldsymbol{v}) + \frac{1}{N_k^2} \sum_{i:C(i)=k} \sum_{C(i')=k} \nabla_j K(\mathbf{x}_i, \mathbf{x}_{i'}, \boldsymbol{v}) \tag{23}$$

For a Gaussian kernel, we have

$$\nabla_j H_{\boldsymbol{v},k} = -\frac{2}{N_k} \sum_{i':C(i')=k} (v_j x_{ij} - v_j x_{i'j})^2 K(\mathbf{x}_i, \mathbf{x}_{i'}, \boldsymbol{v}) + \frac{1}{N_k^2} \sum_{i:C(i)=k} \sum_{i':C(i')=k} (v_j x_{ij} - v_j x_{i'j})^2 K(\mathbf{x}_i, \mathbf{x}_{i'}, \boldsymbol{v}) \tag{24}$$

Finally, for the penalty function we have

$$\nabla_j f(\boldsymbol{v}) = \beta exp(-\beta v_j) \tag{25}$$

4. Experimental results

The experiments we report in this section use artificially generated as well as real-world datasets. We first compared different feature selection approaches presented in this work using six artificially generated data sets, available in Zelnik-Manor and Perona [33]. Subsequently, we performed experiments using eight publicly available benchmark datasets.

4.1. Experiments using artificially generated data sets

We used six artificially generated datasets called toy1, ..., toy6. Each one of these sets contains between 238 and 622 instances, which are described by two variables. Data points are arranged in relatively complex shapes and assigned to between 3 and 5 clusters. For example, Fig. 1 illustrates the shape of dataset toy3. More details about these datasets can be found in Zelnik-Manor and Perona [33].

To make these datasets suitable for feature selection, we generated four sets of irrelevant attributes for each one: 10 independently generated attributes, 10 correlated attributes, 100 independently generated attributes, and 100 correlated attributes. All variables were created using a Gaussian distribution with zero mean and unit variance. For correlated variables, we used a correlation of 0.3 for all pairs of variables. The relevant variables were also scaled to have zero mean and unit variance.

The goal of these experiments is to assess whether the feature selection methods are able to identify the relevant features used to construct the initial clusters. For the proposed method, we trained the model with the following parameters for all cases: $v_0 = 1$, which was suggested in Rifkin and Klautau [27] as a good guess for the kernel width with normalized data, $\epsilon = 0.0001$ and $\beta = 5$, as suggested in Bradley and Mangasarian [5]. We set λ and γ as a function of the energy ratio at each iteration of the algorithm, to achieve an adequate balance between clustering performance and feature selection, and to obtain updated steps of proper size. When our method reaches convergence, we identify whether both relevant variables belong to the subset of selected attributes (denoted by ✓), or only one relevant variable belongs to the subset of selected attributes (denoted by ①), or all selected variables are irrelevant (denoted by ✗).

For filter approaches, we use PCA, GHA, MCFS, and SPEC methods to rank all variables, and we identify if both, one or none of the relevant variables belong to a subset of n ranked features, where n is the number of attributes selected by KP-Kmeans. Since filter approaches do not provide a criterion for finding an optimal number of ranked features, comparing the performance of all methods for a similar number of attributes is a fair strategy for assessing which approaches succeed at detecting the attributes used to create the underlying structure of the data. The comparisons between all approaches are presented in Table 1 where each line represents a dataset; for example, "toy1 10u" signifies the dataset based on toy1 but with 10 additional variables that are uncorrelated.

In Table 1 we observe that KP-Kmeans is able to find those attributes that were used to arrange the shape of the datasets in all cases. The model also selects between one and three noisy attributes, leading to a total of three to five variables in the clustering model. Using the number of selected variables as an input for the alternative models makes their performance completely different: in most cases these methods rank the relevant variables as the most irrelevant ones. For SPEC, MCFS, and PCA, the two relevant variables were ranked as the least important ones for 10-variable datasets, and among the 10 least relevant for 100-variable datasets. The GHA method performed slightly better, especially for independently generated 10-variable datasets.

Notice that the performance of our proposed approach, given its embedded structure, relies heavily on the initial clustering. Complex data structures may require advanced techniques for model selection, and, in particular, the right choice of the kernel width σ. Automatic techniques for the definition of this parameter, such as the one presented in Zelnik-Manor and Perona [33], can be very useful in defining the method's initial kernel width.

4.2. Benchmark datasets

In this section we first briefly describe the datasets used for benchmark. Then we present the results we obtained with different methods.

We studied four datasets from the UCI Machine Learning Repository [3]: Iris, Wine, Glass, and Waveform; and one dataset from the Statlog Project Databases, also available from UCI Repository: Segment dataset. Additionally, we studied four high-dimensional microarray datasets for multiclass classification: SRBCT [21], Lung [4], MLL [2], and Glioma [24]. These four microarray datasets for cancer diagnosis have been used previously for machine learning purposes as was reported in [32]. All these datasets are used for supervised learning. Here we use the respective class labels exclusively to determine the cluster number. Microarray datasets have also been used for unsupervised learning in Hastie et al. [15].

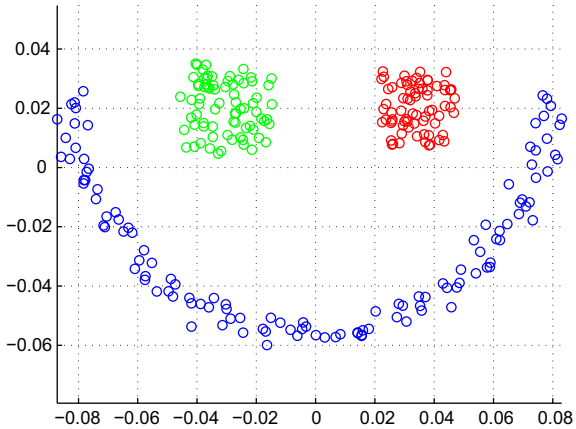

Fig. 1. Illustrative example: dataset toy3.

Table 1
Feature selection performance for all toy data sets.

	n	KP-Kmeans	PCA	GHA	SPEC	MCFS
toy1 10u	4	✓	✗	①	✗	✗
toy1 10c	4	✓	✗	✓	✗	✗
toy1 100u	3	✓	✗	✗	✗	✗
toy1 100c	3	✓	✗	✗	✗	✗
toy2 10u	4	✓	✗	✓	✗	✗
toy2 10c	3	✓	✗	✗	✗	✗
toy2 100u	4	✓	✗	✗	✗	✗
toy2 100c	3	✓	✗	✗	✗	✗
toy3 10u	3	✓	✗	✗	✗	✗
toy3 10c	3	✓	✗	✗	✗	✗
toy3 100u	4	✓	✗	✗	✗	✗
toy3 100c	4	✓	✗	✗	✗	✗
toy4 10u	4	✓	✗	✓	✗	✗
toy4 10c	3	✓	✗	✗	✗	✗
toy4 100u	5	✓	✗	✗	✗	✗
toy4 100c	3	✓	✗	✗	✗	✗
toy5 10u	4	✓	✗	✓	✗	✗
toy5 10c	4	✓	✗	✗	✗	✗
toy5 100u	5	✓	✗	①	✗	✗
toy5 100c	3	✓	✗	✗	✗	✗
toy6 10u	4	✓	✗	✓	✗	✗
toy6 10c	4	✓	✗	✗	✗	✗
toy6 100u	4	✓	✗	✗	✗	✗
toy6 100c	3	✓	✗	✗	✗	✗

Table 2 summarizes the relevant information for each benchmark dataset.

In order to study the performance of KP-Kmeans we compare the results for a given number of features n (determined by the stopping criterion of our approach) with different feature selection algorithms for clustering presented in Section 2 of this paper (PCA, GHA, MCFS, and SPEC). For all the approaches, we define the number of clusters as the number of different class labels of the respective datasets. For filter methods, a ranking is performed and then Kernel K-means is trained using n variables with $\sigma = 1$. The clustering accuracy of all methods is computed as the percentage of correct assignments of all examples, and reported in Table 3. The best performance among all methods in terms of accuracy is highlighted in bold type. Notice that the labels are not used for training, only for computing the clustering accuracy.

In Table 3 we observe that the proposed method performs better on six out of nine datasets, and, in particular, on three out of four high dimensional microarray datasets. The gain is important in most cases, while the dimensionality reduction achieved with the proposed method (from 47.4% to 99.8%) allows the practitioner to concentrate on those few attributes that are relevant for generating the structure of the data.

For the next experiment we compared all approaches in terms of accuracy for an increasing number of features. We compute the accuracy at each removal step of the algorithm (steps 6 to 8 of Algorithm 2), while training Kernel K-means for an increasing number of ranked features using filter methods. As an illustrative example, we report the results for the Lung dataset (Fig. 2), which is the only high dimensional dataset on which we have been able to reconstruct the whole accuracy curve, given the small number of attributes selected when the stopping criterion is reached.

In Fig. 2 we observe very stable results for the proposed method along the curve, reaching their peak at 1,000 attributes, and with only a slight decrease in accuracy until the convergence is finally reached with seven attributes. SPEC behaves similarly, with a peak at 2000 features (best overall performance), but with a steeper decrease in accuracy. Both methods outperformed MCFS, PCA, and GHA on this dataset. These experiments allowed us to determine the right compromise between performance and dimensionality reduction.

Table 2
Number of examples, number of variables, and number of classes for all nine data sets.

Dataset	#examples	#variables	#classes
IRIS	150	4	3
WINE	178	13	3
GLASS	214	13	6
WAVEFORM	5000	21	3
SEGMENT	2310	19	7
SRBCT	83	2308	4
LUNG	203	3312	5
GLIOMA	50	4433	4
MLL	72	5848	3

Table 3
Feature selection performance for all benchmark datasets.

	n	KP-Kmeans	PCA	GHA	SPEC	MCFS
Iris	2	**94.7**	88.0	88.0	66.7	56.0
Wine	2	**75.3**	48.9	65.7	70.2	72.4
Glass	5	**54.7**	47.2	50.5	51.4	44.9
Segment	10	53.9	51.1	**59.2**	53.8	57.2
Waveform	5	51.5	**55.8**	51.8	53.7	52.0
SRBCT	301	49.4	44.6	47.0	37.3	**66.3**
Lung	7	**65.0**	62.1	47.8	60.1	45.8
Glioma	901	**76.0**	46.0	36.0	54.0	66.0
MLL	108	**87.5**	70.8	72.2	70.8	84.7

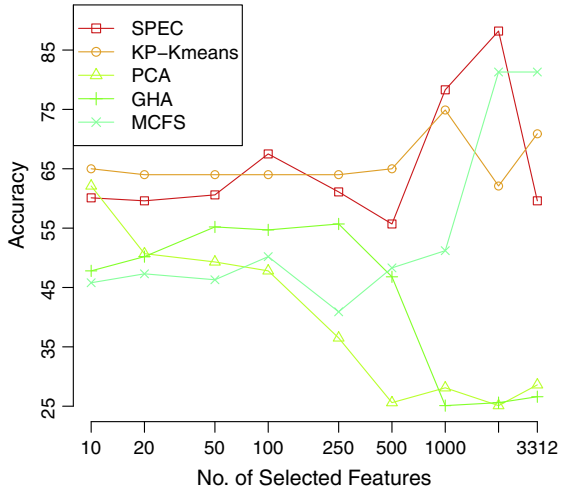

Fig. 2. Performance versus the number of selected variables for various feature selection approaches. Lung dataset.

4.3. Influence of the parameters and discussion

One advantage of the proposed approach in terms of computational effort is that it automatically obtains an optimal feature subset, avoiding a validation step to determine how many ranked features will be used for clustering. However, several parameters should be tuned in order to obtain the final solution. In this section we study the performance of KP-Kmeans by varying one parameter at a time, showing its influence on the final solution.

To illustrate the influence of parameters λ and γ in the clustering model, Table 4 presents the performance in terms of classification accuracy and number of selected features for the MLL data set. We vary the parameters according to the following values: $\gamma,\ \lambda \in \{2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, 2^{4}, 2^{5}\}$.

Table 4
Feature selection performance for MLL dataset.

	λ	n	γ	n
2^{-5}	41.7	1774	84.7	911
2^{-4}	81.9	253	84.7	215
2^{-3}	84.7	115	84.7	115
2^{-2}	87.5	108	84.7	115
2^{-1}	87.5	108	84.7	115
2^{0}	84.7	115	87.5	108
2^{1}	84.7	115	87.5	108
2^{2}	48.6	0	87.5	108
2^{3}	48.6	0	87.5	108
2^{4}	48.6	0	87.5	108
2^{5}	48.6	0	87.5	108

In Table 4 we observe that performance is relatively stable for the different values of λ between 2^{-4} and 2^{1}, while for higher values the proposed method tends to remove all attributes. For parameter γ, results remain relatively stable for all the values studied, using $\lambda = 2^{-1}$. This experiment demonstrates the method's stability with respect to these parameters, although it is still important to calibrate them using clustering-based metrics.

5. Conclusions and future work

Feature selection is an important task for knowledge discovery, especially as more and more data becomes available. While very many feature selection methods have been developed for supervised learning (classification as well as regression) this is not the case for unsupervised learning.

In this paper we propose Kernel Penalized K-means (KPKM), an unsupervised learning technique for embedded feature selection in combination with Kernel K-means. Its main idea is to minimize the violation compared to the initial cluster structure while simultaneously penalizing the use of features, thus achieving a compromise between a good cluster solution with few features. Advantages are a better understanding of the analyzed phenomena, better knowledge for the user on which attributes to concentrate on, and reduced computational complexity for future investigation. We have been able to show that KPKM outperforms alternative approaches on all the artificially generated datasets we used as well as on most benchmark datasets in use.

It is important to mention that wrapper or embedded feature selection approaches for clustering are only useful if there is prior knowledge that using all the candidate attributes leads to adequate clustering, while the feature selection strategy is used to polish this partition by removing those attributes that are unrelated to the inherent structure of the data. A discussion on this topic can be found in Alelyani et al. [1], in which the authors questioned the usefulness of wrapper or embedded methods since they depend on clustering inputs, and therefore few benefits can be gained from such processes. We believe that embedded feature selection can be very useful in domains such as business analytics (e.g. in customer segmentation tasks), where there are important collection costs linked to each attribute, while the understanding of the underlying behavior of the customers is of primary interest [30]. If no good clustering can be obtained with the candidate variables, a hybrid two-step clustering strategy is suggested, in which features are first filtered out via approaches such as SPEC until a satisfactory partition is found, while KPKM can be used to further polish this clustering.

Future work needs to be done along the following lines. The base algorithm, K-means in the case of this paper, could be replaced by alternative clustering techniques. Support Vector Clustering especially seems to be an interesting avenue for such research since it incorporates the use of kernels for clustering. Instead of working with a completely unsupervised approach, it would be interesting to apply semi-supervised clustering to maintain the cluster structure for "most" of the instances to be clustered. Fuzzy clustering could also be interesting for balancing the goal of maintaining the cluster structure in a fuzzy rather than a crisp notion. Finally, it could be interesting to observe and detect changing feature importance over time, thus improving current approaches to dynamic clustering.

Acknowledgements

The first and third authors were supported by FONDECYT project 11121196 and 1140831, respectively. The work reported in this paper has been partially funded by the Complex Engineering Systems Institute (ICM: P-05-004-F, CONICYT: FB016). The second author is partially supported by MTM2012-36163-C06-03 (Ministerio de Economía y Competitividad) and P11-FQM-7603, FQM 329 (Junta de Andalucía).

References

[1] S. Alelyani, J. Tang, H. Liu, Data Clustering: Algorithms and Applications, Data Clustering: Algorithms and Applications, CRC Press, 2013.
[2] S. Armstrong, J. Staunton, L. Silverman, R. Pieters, M. den Boer, M. Minden, S. Sallan, E. Lander, T. Golub, S. Korsmeyer, MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia, Nat. Genet. 30 (2002) 41–47.
[3] A. Asuncion, D. Newman, UCI machine learning repository, 2007.
[4] A. Bhattacharjee, W. Richards, J. Staunton, C. Li, S. Monti, V.P., C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. Mark, E. Lander, W. Wong, B. Johnson, T. Golub, D. Sugarbaker, M. Meyerson, Classification of human lung carcinomas by mrna expression profiling reveals distinct adenocarcinoma subclasses, in: Proceedings of the National Academy of Sciences of USA 98, pp. 13790–13795.
[5] P. Bradley, O. Mangasarian, Feature selection via concave minimization and support vector machines, in: Machine Learning proceedings of the fifteenth International Conference (ICML'98) 82–90, San Francisco, California, Morgan Kaufmann.
[6] D. Cai, C. Zhang, X. He, Unsupervised feature selection for multi-cluster data, in: KDD, July 25–28, 2010, Washington DC, USA.
[7] E. Carrizosa, B. Martín-Barragán, D. Romero-Morales, Detecting relevant variables and interactions in supervised classification, Eur. J. Oper. Res. 213 (2011) 260–269.
[8] D.M. Cvetković, M. Doob, H. Sachs, Spectra of Graphs: Theory and Application, Pure and Applied Mathematics, Academic Press, 1980.
[9] M. Dash, K. Choi, P. Scheuermann, H. Liu, Feature selection for clustering – a filter solution, in: Proceedings of the Second International Conference on Data Mining, pp. 115–122.
[10] M. Dash, H. Liu, Feature selection for clustering, in: PADKK '00 Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications, Springer-Verlag, 2000, pp. 110–121.
[11] I.S. Dhillon, Y. Guan, B. Kulis, Kernel k-means: spectral clustering and normalized cuts, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 551–556.
[12] J.G. Dy, C.E. Brodley, Visualization and interactive feature selection for unsupervised data, in: Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD), pp. 360–364.

[13] E.L. Dyer, A.C. Sankaranarayanan, R.G. Baraniuk, Greedy feature selection for subspace clustering, J. Mach. Learn. Res. 14 (2013) 2487–2517.
[14] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction, Foundations and Applications, Springer, Berlin, 2006.
[15] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, second ed., Springer-Verlag, 2009.
[16] D.E. Hebb, The Organization of Behavior, Wiley, New York, 1949.
[17] M.M.B. Ismail, H. Frigui, Unsupervised clustering and feature weighting based on generalized Dirichlet mixture modeling, Information Sciences 274 (2014) 35–54.
[18] Y.S. Kim, W.N. Street, F. Menczer., Feature selection in unsupervised learning via evolutionary search, in: Proceedings of ACM SIGKDD International Conference on Knowledge and Discovery, pp. 365–369.
[19] J. Kittler, Pattern recognition and signal processing, Pattern Recognition and Signal Processing, Sijthoff and Noordhoff, Netherlands, 1978.
[20] J.A. Lee, M. Verleysen, Nonlinear Dimensionality Reduction, Information Science and Statistics, Springer, Berlin, Heidelberg, 2007.
[21] T. Lin, R. Liu, C. Chen, Y. Chao, S. Chen, Pattern classification in DNA microarray data of multiple tumor types, Pattern Recogn. 39 (2006) 2426–2438.
[22] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, Berkeley, Calif., pp. 281–297.
[23] S. Maldonado, R. Weber, J. Basak, Kernel-penalized SVM for feature selection, Inf. Sci. 181 (2011) 115–128.
[24] C. Nutt, D. Mani, R. Betensky, P. Tamayo, J. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M. McLaughlin, T. Batchelor, P. Black, A. von Deimling, S. Pomeroy, T. Golub, D. Louis, Gene expression-based classification of malignant gliomas correlates better with survival than histological classification, Cancer Res. 63 (2003) 1602–1607.
[25] B. Peralta, A. Soto, Embedded local feature selection within mixture of experts, Inf. Sci. 269 (2014) 176–187.
[26] A. Rakotomamonjy, Variable selection using SVM-based criteria, J. Mach. Learn. Res. 3 (2003) 1357–1370.
[27] R. Rifkin, A. Klautau, In defense of one-vs-all classification, J. Mach. Learn. Res. 5 (2004) 101–141.
[28] M.R.M. Rizk, E. Koosha, A comparison of principal component analysis and generalized Hebbian algorithm for image compression and face recognition, in: The 2006 International Conference on Computer Engineering and Systems, Cairo, Egypt, pp. 214–219.
[29] T.D. Sanger, Optimal unsupervised learning in a single-layer linear feedforward neural network, Neural Netw. 2 (1989) 459–473.
[30] A. Seret, S. vanden Broucke, B. Baesens, J. Vanthienen, A dynamic understanding of customer behavior processes based on clustering and sequence mining, Expert Syst. Appl. 41 (2014) 4648–4657.
[31] R. Xu, D. Wunsch, Clustering, John Wiley and Sons, Hoboken, NJ, 2008.
[32] K. Yang, Z. Cai, J. Li, G. Lin, A stable gene selection in microarray data analysis, BMC Bioinform. 7 (2006) 228.
[33] L. Zelnik-Manor, P. Perona, Self-tuning spectral clustering, in: Y.W.L.K. Saul, L. Bottou (Eds.), Advances in Neural Information Processing Systems, vol. 17, MIT Press, Cambridge, MA, 2005, pp. 1601–1608.
[34] Z. Zhao, H. Liu, Spectral feature selection for supervised and unsupervised learning, in: ICML '07: Proceedings of the 24th International Conference on Machine Learning, ACM, New York, NY, 2007, pp. 1151–1157.