Clustering with Quantitative User Preferences on Attributes

Adnan El Moussawi *†, Ahmed Cheriat†, Arnaud Giacometti†, Nicolas Labroche† and Arnaud Soulet†
*Kalidea Group, France, email: {aelmoussawi, acheriat}@kalidea.com
†LI - University of Tours, France, email: firstname.lastname@univ-tours.fr

***Abstract*—In recent years, many semi-supervised clustering methods have integrated constraints at instance-level so that the final partition is consistent with the users' needs. However, it is also necessary that the user can express his/her preferences at the attribute level when several relevant subsets of attributes can co-exist. This paper proposes such a clustering framework to represent and integrate quantitative preferences on attributes that will guide the learning of the distance. We derive a new algorithm from this framework that achieves a compromise clustering between a data-driven and a user-driven solution and converges with a good complexity. We observe experimentally that the addition of preferences may be essential to achieve a better clustering. We also show that our approach performs better than the state-of-the art algorithms.**

***Index Terms*—Preference on the attributes, clustering, metric learning.**

I. Introduction

Data clustering, one of the most important unsupervised learning problem, is widely used in the field of Customer Relationship Management (CRM). For example, it is commonly used for customer segmentation, i.e. the process of dividing customers into homogeneous groups or profiles on the basis of common values of attributes. Nevertheless, our recent experiments show that many problems remain to be solved.

First, because the number of possible descriptive attributes can be very large, it is important to help the user to find subspaces where interesting clusterings exist. Feature selection techniques [1], [2] can be used to solve this problem. However, using these techniques, important features can be removed. Moreover, in the context of subspace clustering [3], [4], it has been shown that a large number of interesting clusterings can exist (in different subspaces) and that it is difficult to automatically select one particular solution.

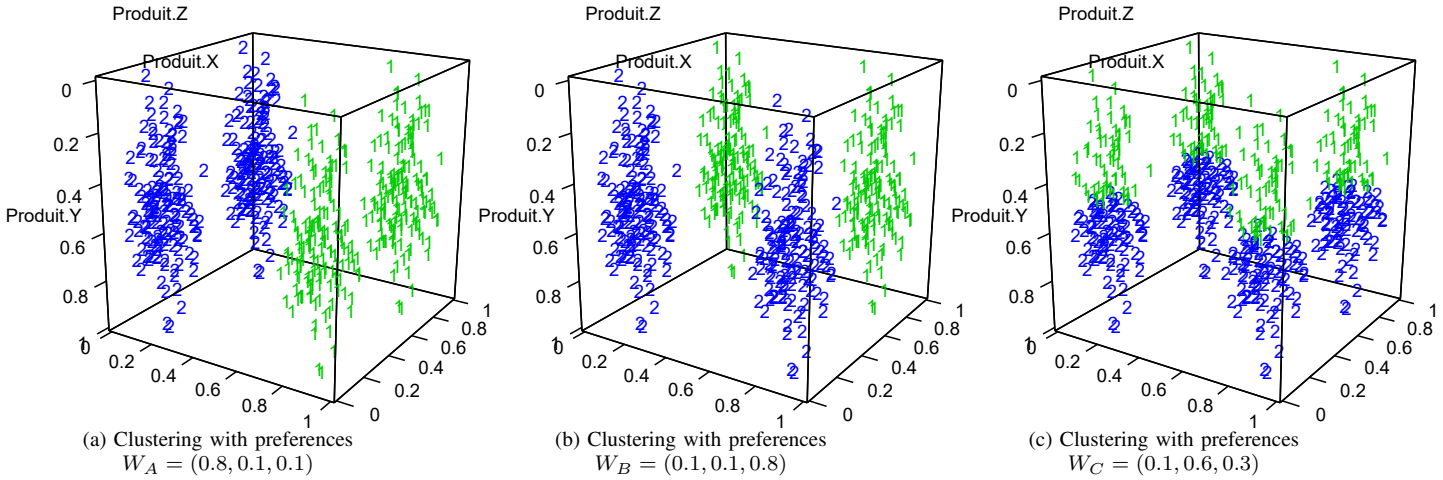
Second, because different users may have different center of interest or preferences, it is important to propose a clustering system that can integrate these preferences when a clustering is built and selected (among all the possible solutions). In that context, the main objective of our work is to show how to take into account the knowledge and preferences of an expert to build a clustering that is a good compromise between a data-driven and a user-driven solution. Different experts with different preferences will obtain different views of the data among all the possible views.

Motivating example. In order to illustrate the objective of our work, let us consider the following toy example. Different experts from a marketing agency want to build a customer segmentation based on their purchase of various categories of products named X, Y and Z. As these experts have different professional experiences, we consider that their degree of interest or preferences on categories of product is not the same.

First, consider an expert A that is more interested by purchases of product X, than purchases of products Y and Z. In this paper, we use a quantitative model to represent the preferences of A. More precisely, we assume that each expert assigns to each descriptive attribute a weight proportional to his/her interest for this attribute in clustering analysis. Thus, the preferences of expert A will be represented by the preference vector $W_A = (0.8, 0.1, 0.1)$. If this expert wants to build a customer segmentation with two clusters, using a K-means algorithm with a weighted distance, he/she will obtain the clustering result presented Figure 1a. From another point of view, an expert B with a preference vector $W_B = (0.1, 0.1, 0.8)$ will obtain the clustering presented Figure 1b. It is important to note that the two clusterings obtained by experts A and B are two interesting views of the same data set. Only the preferences of the experts allow to select one of these two possible clusterings.

Consider now an expert C with a preference vector $W_C = (0.1, 0.6, 0.3)$. Using a simple K-means with a weighted distance, this expert will obtain the segmentation given by Figure 1c, which is not satisfactory. Indeed, the expert C formulates a high degree of preference on product Y, whereas this attribute does not separate well the set of customers. In this paper, to avoid such a problem, we propose a new approach that can take into account the confidence level of an expert in his/her preferences. The confidence level κ of an expert is represented by a real value in $[0, 1]$. Thus, if the expert C has a very high confidence in his/her preferences ($\kappa = 1.0$), he/she will still obtain the segmentation depicted by Figure 1c. However, with a lower confidence level in his/her preferences ($\kappa = 0.6$), he/she will obtain the segmentation presented by Figure 1b. Indeed, as the attribute Y does not separate well clusters, our method will tend to favor the other attributes (especially the attribute Z whose preference is higher than that of X).

In order to tackle the problems and challenges illustrated by our motivating example, we propose in this paper a new semi-supervised clustering algorithm that integrates user preferences on attributes during the construction of clusters. More precisely, the main contributions of this paper are the



(a) Clustering with preferences $W_A = (0.8, 0.1, 0.1)$

(b) Clustering with preferences $W_B = (0.1, 0.1, 0.8)$

(c) Clustering with preferences $W_C = (0.1, 0.6, 0.3)$

Fig. 1: Customer segmentation using their purchase of products X, Y and Z.

following:

- We show how to integrate user preferences *on descriptive attributes* in a new clustering objective function defined mainly on set of instances. Lots of papers have adapted clustering objective functions in order to take into account user knowledge represented by constraints. However, these constraints mainly specify if two instances should be in the same cluster or not [5]–[9]. By contrast, the approach presented in this paper takes into account user preferences (soft constraints) on descriptive attributes. To the best of our knowledge, only the works in [10] addresses the same problem.
- We propose to use a *quantitative* model of preferences to represent the user preferences on attributes. The *qualitative* model proposed in [10] is more expressive. However, the authors underline that their model of preferences can be vey difficult to set. By comparison, our *quantitative* model is easier to use by an expert. Moreover, we show that it guarantees a lower time complexity when we build the clusters, in particular when the user preferences define a total pre-order on the set of descriptive attributes.
- We present the new clustering algorithm MAPK-means (*Metric Attribute Preferential K-means*) that minimizes our clustering objective function. The complexity of this algorithm is analyzed and we present a large set of experiments to prove the efficiency and interest of our approach. In particular, these experiments show how important it is to consider user preferences on attributes to improve the quality of clustering.

The rest of the paper is organized as follows. In Section II, we discuss some related works. Then, we formulate the problem and show in Section III how user preferences on attributes can be integrated in a clustering objective function. The new clustering algorithm MAPK-means that minimizes our new clustering objective function is derived, explained and analyzed in Section IV. Then, in Section V, we evaluate and compare our algorithm to other approaches using several data sets. Finally, we conclude the paper and discuss some perspectives in Section VI.

II. Related work

Numerous studies [1], [2] have tackled the problem of feature selection or weighting in the classification or clustering contexts, but all these methods are only data-driven. Based on some internal criteria, they select the features that might improve clustering accuracy or interpretability without any user interaction. Subspace clustering [3], [4] also proposes an exploration of different subsets of features where clusters are relevant according to a criterion, for example density as in CLIQUE [11]. As a consequence, there exists a very large number of possible clusterings and the difficulty is to choose from all these potential solutions. Moreover, as with the previous feature weighting methods, these methods are only data-driven and do not rely on user preferences for some feature subset.

Our proposal is part of the family of semi-supervised clustering algorithms that can improve performances and stability from expert knowledge. This knowledge is generally provided as label or pairwise instances constraints that indicate if two objects should be in the same cluster (Must-Link or ML constraints) or not (Cannot Link or CL constraints) and has been adapted to numerous clustering approaches [5]–[9]. Other kinds of constraints have been proposed at the cluster level, mainly to avoid contradictions at the instance level [12]–[15] or as relative distance constraints [16] that are more adapted to ranking and instance order preferences. Semi-supervised clustering methods can be categorized in three main families depending if they impose a strict [6] or a soft enforcement of constraints with a penalization term in the objective function [17], or a soft enforcement via the learning of a metric space that minimizes the number of violated constraints [16], [18]–[20]. [20] is the first to motivate the need for space-level constraints with some direct modifications to the distance matrix depending on constraints in a hierarchical clustering. In [18], [19] the authors learn a Mahalanobis distance based

on ML and CL constraints with regularization terms in the objective function to avoid trivial solutions where only the feature with the lowest variance gets all the weight. More recently, [16] learns a Mahalanobis distance from distance comparisons constraints before applying a K-means, while previous works introduce the metric learning directly in the alternate optimization of K-means.

Our approach falls into the category that achieves a soft enforcement via the learning of a metric space, by taking into account user preferences on attributes. Following our previous example in Figure 1, we propose to use quantitative preferences on attributes as expressed by the feature weights vector $W_A = (0.8, 0.1, 0.1)$ defined for expert A. An alternative approach [10], [21] consists in expressing attributes preferences by mean of a triple $(s; t; d)$ which indicates that attribute t is preferred over s with a degree d. However, when a total pre-order is expected over the set of all attributes as in our example, the expert would have to set a quadratic number of preferences with the attribute number, setting 6 constraints even in this case with only 3 attributes, such as $(X, Y, 0.7)$, $(Y, X, -0.7)$ $\ldots (Y, Z, 0.001)$ and $(Z, Y, -0.001)$. Contrary to this approach, our model of quantitative preferences only requires a linear number of preferences, i.e. 1 per attribute. Moreover, in addition to simplifying the interaction with the user, our model leads to a better complexity in metric learning.

III. Problem Statement

Our objective is to propose a new semi-supervised clustering algorithm that can handle quantitative user preferences on attributes. To this aim, we introduce a K-means like algorithm that learns the attribute weights that are the best compromise between the weights provided by the user preferences and the attribute weights that would best fit the natural distribution of data.

Notations: In the following, a data set $\mathcal{X}$ is a set of N data objects described by M attributes. Clustering analysis aims at finding a partition of K clusters, denoted by $\{\mathcal{X}_j\}_{j=1}^{K}$. The centroid of cluster $\mathcal{X}_j$ is denoted by c_j.

A. Quantitative user preferences

The originality of our approach is to incorporate user preferences on attributes to construct the right partition. More precisely, we model user preferences with a quantitative model of preferences in which the end-user assigns to each attribute a weight proportional to his/her interest for this attribute in clustering analysis. More formally, we use a *preference vector* $\mathbf{W}^*$ to model preferences where each weight w_i^* represents the weight expressed by the user on the ith attribute. Without loss of generality, we consider that $w_i^* \geq 0$ for all $i \in \{1, \ldots, M\}$ such that $\sum_{i=1}^{M} w_i^* = 1$. The set of all preference vectors is denoted by $\mathcal{P}$.

It is often necessary to measure the dissimilarity between two preference vectors of $\mathcal{P}$. To do this, we propose to use the Kullback-Leibler divergence which measures the dissimilarity between two distributions. In our case, the two distributions correspond to the learned vector $\mathbf{W} \in \mathcal{P}$ and a reference vector $\mathbf{P} \in \mathcal{P}$: $D_{KL}(\mathbf{P}\|\mathbf{W}) = \sum_{i=1}^{M} p_i \log\left(\frac{p_i}{w_i}\right)$. In the following, we manipulate two reference vectors $\mathbf{P}$ to express our objective function: the user preferences $\mathbf{W}^*$ and the uniform vector $\mathbf{U} = (1/M, \ldots, 1/M)$.

B. Attribute preferential clustering objective function

Our clustering objective function consists of three terms that are detailed in the following paragraphs.

a) Intra-cluster distance: First, as with any K-means like algorithm, we want to minimize the intra-cluster distance of the clusters $\{\mathcal{X}_j\}_{j=1}^{K}$. A naive solution could be to directly input the preference vector $\mathbf{W}^*$ to parameter Euclidean distance as follows: $\|x - c_j\|_{\mathbf{W}^*} = \sqrt{\sum_{i=1}^{M} w_i^*(x[i] - c_j[i])^2}$. Unfortunately, in this case our solution would only rely on the user expertise and would not take into account the natural distribution of the data. As a side effect, we could output a poor clustering if the user preference vector does not sufficiently discriminate between the data objects (see Figure 1c as a typical example). Consequently, we propose to learn a vector $\mathbf{W} \in \mathcal{P}$ that performs a projection of the initial data space so that the clusters are more compact and well separated in the new space. Thus, we want to minimize: $\sum_{j=1}^{K} \sum_{x \in \mathcal{X}_j} \|x - c_j\|_{\mathbf{W}}^2$.

b) Deviation from attribute preferences: Second, we want that the learned vector $\mathbf{W}$ deviates as less as possible from $\mathbf{W}^*$ in order to respect user preferences. Thus, it is necessary to introduce a penalty term to reduce the dissimilarity of $\mathbf{W}$ with $\mathbf{W}^*$. Using the Kullback-Leibler divergence, we want to minimize: $D_{KL}(\mathbf{W}^*\|\mathbf{W})$.

c) Regularization term: Third, we want to prevent overfitting. Indeed, one trivial solution, while learning Mahalanobis distance, consists in assigning the maximum weight to the attribute on which the intra-cluster distances are minimal. Of course, this statistical optimal solution is of no interest in real use cases. Consequently, we add a regularization term that prevents the vector $\mathbf{W}$ to deviate too much from a traditional K-means where all attributes have equal weights. This idea can be formulated as the divergence between the vector to learn and a uniform vector $\mathbf{U} = (1/M, \ldots, 1/M)$: $D_{KL}(\mathbf{U}\|\mathbf{W})$.

By combining these three terms, it is possible to define an attribute preferential clustering objective function that expresses a compromise:

$$\begin{aligned} \mathcal{I}_{map} = \; & \alpha\Big(\mathcal{Z}\sum_{j=1}^{K}\sum_{x\in\mathcal{X}_j}\|x-c_j\|_{\mathbf{W}}^2\Big) \\ & + (1-\alpha)\Big(\kappa D_{KL}(\mathbf{W}^*\|\mathbf{W}) + (1-\kappa)D_{KL}(\mathbf{U}\|\mathbf{W})\Big) \end{aligned} \tag{1}$$

where $\mathcal{Z}$ is a normalizing constant greater than 0, the parameters α and κ are between 0 and 1. Note that $\mathcal{Z}$ is a normalizing constant between intra-cluster distance and other terms because the parameterized Euclidean distance and the Kullback-Leibler divergence have really different ranges. Section IV will discuss how to set this constant such that a

median value of α corresponds to a default value guaranteeing a trade-off between the two parts.

- **Intra-cluster distance weight α:** This parameter controls the importance of data compared to that of user preferences. Clearly, it seems difficult for the end-user to set this technical parameter. Fortunately, we will see in the experimental section that it is possible to set it to an appropriate default value.
- **Confidence level κ:** the user-specified parameter κ gives the importance of his/her preferences. When $\kappa = 1$, the regularization term is not used. The user forces the method to meet his/her preferences. When $\kappa = 0$, user preferences are ignored and minimizing Equation 1 is equivalent to minimize the objective function of MPCK-means without constraints [19].

Given a set of data points $\mathcal{X}$, a number of clusters $K \geq 1$, a preference vector $\mathbf{W}^* \in \mathcal{P}$, $\alpha \in [0,1]$ and $\kappa \in [0,1]$, our goal is to find a K-partition $\{\mathcal{X}_j\}_{j=1}^K$ of data minimizing the objective function $\mathcal{I}_{map}$ while learning a vector $\mathbf{W} \in \mathcal{P}$.

IV. MAPK-means Algorithm

To find a K-partition that minimizes our objective function, we use the method of Lagrange multiplier as a strategy (Section IV-A) to add a metric learning step to the K-means algorithm (Section IV-B).

A. Reformulation with a Lagrange multiplier

As mentioned in Section III-A, all preference vectors of $\mathcal{P}$ are such that each weight is positive and the sum of weights equals to 1. In particular, the learned vector $\mathbf{W}$ in objective function $\mathcal{I}_{map}$ has to satisfy these constraints:

$$\min_{\mathbf{W}} \ \mathcal{I}_{map} \qquad \text{subjectto} \ \sum_{i=1}^{M} w_i - 1 = 0;\ w_i > 0; \quad \text{for all } i \in \{1,\dots,M\} \tag{2}$$

This constrained minimization problem can be solved using the method of Lagrange multiplier as strategy (because $\mathcal{I}_{map}$ and $\sum_{i=1}^{M} w_i - 1$ have continuous first partial derivatives). We introduce a Lagrange multiplier λ and consider the following function: $\mathcal{I}'_{map} = \mathcal{I}_{map} + \lambda\Big(\sum_{i=1}^{M} w_i - 1\Big)$. If $\mathbf{W}$ minimizes $\mathcal{I}_{map}$, then there exists a value of λ such that $\mathbf{W}$ is a stationary point for $\mathcal{I}'_{map}$. The stationary point is the point where the partial derivatives of $\mathcal{I}'_{map}$ is zero:

$$\frac{\partial \mathcal{I}'_{map}}{\partial w_i} = \alpha\mathcal{Z}\overbrace{\sum_{j=1}^{K}\sum_{x\in\mathcal{X}_j}\|x[i]-c_j[i]\|^2}^{S_i} - (1-\alpha)\big(\kappa\frac{w_i^*}{w_i} + (1-\kappa)\frac{1}{Mw_i}\big) + \lambda = 0$$

Algorithm 1 MAPK-means

input a data set $\mathcal{X}$, a number of clusters K, a preference vector $\mathbf{W}^*$, κ, α
output a partition $\{\mathcal{X}_j\}_{j=1}^K$ and a learned vector $\mathbf{W}$

1: Get K center $\{c_j\}_{j=1}^K$ with K-means++
2: Initialize $\mathbf{W} := (1/M,\dots,1/M)$
3: Initialize $\mathcal{Z} := \sum_{i=1}^{M} \frac{\kappa w_i^* + (1-\kappa)/M}{S_i}$
4: **repeat**
5: *// update the partition* $\{\mathcal{X}_j\}_{j=1}^K$
6: $\mathcal{X}_j := \Big\{x \in \mathcal{X} : \arg\min_{l\in\{1,\dots,K\}} \|x - c_l\|^2_{\mathbf{W}} = j\Big\}$ for $j \in \{1,\dots,K\}$
7: $c_j[i] := \frac{\sum_{x\in\mathcal{X}_j} x[i]}{|\mathcal{X}_j|}$ for $i \in \{1,\dots,M\}$ and $j \in \{1,\dots,K\}$
8: *// update the vector* $\mathbf{W}$
9: Compute λ using a dichotomic search
10: $w_i := \frac{(1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)}{\alpha\mathcal{Z}S_i + \lambda}$ for $i \in \{1,\dots,M\}$
11: **until** $\{\mathcal{X}_j\}_{j=1}^K$ remains unchanged
12: **return** $\{\mathcal{X}_j\}_{j=1}^K$ and $\mathbf{W}$

Assuming that $S_i = \sum_{j=1}^{K}\sum_{x\in\mathcal{X}_j}\|x[i]-c_j[i]\|^2$ is the total intra-cluster distance on the i-th attribute. We rewrite the above equation for obtaining the update of weight w_i:

$$w_i = \frac{(1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)}{\alpha\mathcal{Z}S_i + \lambda} \tag{3}$$

The update of weight w_i is central in our algorithm depicted by the next section for learning the metric. It is easy to see that the lower the variance S_i, the higher the weight of the attribute w_i. Moreover, when κ is set to 1, only the preferences are used. Conversely, when κ is zero, user preferences are not considered.

B. Algorithm derivation

Our algorithm follows the optimization scheme introduced by MPCK-means [19] which consists in three phases after initialization: 1) points assignment, 2) centroid re-estimation and 3) metric learning. More specifically, for a given data set $\mathcal{X}$, a number of clusters $K \geq 1$, a preference vector $\mathbf{W}^* \in \mathcal{P}$, a confidence level $\kappa \in [0,1]$ and an intra-cluster distance weight $\alpha \in [0,1]$, the algorithm MAPK-means (*Metric Attribute Preferential K-means*, provided by Algorithm 1) returns a K-partition $\{\mathcal{X}_j\}_{j=1}^K$ minimizing the objective function $\mathcal{I}_{map}$ by learning a vector $\mathbf{W}$.

a) Algorithm initialization: We use the same initialization as K-means++ [22] (line 1). The first center is randomly selected from the data objects. Then, each of the $K-1$ other initial centers is randomly selected with a probability proportional to the sum of the distances to the centers that have already been chosen. Besides, the weights of attributes for $\mathbf{W}$ are initially equally distributed (line 2): $w_i = \frac{1}{M}$ for $i \in \{1,\dots,M\}$. Finally, the constant $\mathcal{Z}$ is initialized such that

the intra-cluster distance and the other terms have a similar impact during the update of a weight w_i (see Equation 3) when $\alpha = 0.5$. For this, we choose a $\mathcal{Z}$ value as our update that is identical to that of MPCK-means [19] when $\alpha = 0.5$ (line 3).

b) Cluster assignment: The assignment step is the same as K-means (line 5-6), with the only difference that the distances between points and centroid are parameterized with a vector $\mathbf{W}$. Each point is assigned to the nearest cluster (line 6). This assignment reduces the intra-cluster distance and it also minimizes the objective function $\mathcal{I}_{map}$.

c) Centroid re-estimation: Once all points are assigned to a cluster, we update the center of each cluster by calculating the centroid for each attribute i (line 7). Unlike some approaches (e.g., MPCK-means), the calculation of the centers is insensible to the order of the assignment of points in the previous step.

d) Metric learning: In this step, MAPK-means learns the right metric by updating the vector $\mathbf{W}$ that minimizes the objective function $\mathcal{I}_{map}$ (line 8-10). As explained in Section IV-A, the update of $\mathbf{W}$ is obtained by taking the derivative $\frac{\partial \mathcal{I}_{map}}{\partial w_i}$ equal to 0. In order to get the exact update of $\mathbf{W}$, we have to compute the Lagrange multiplier λ (see Equation 3). We introduce $p_i = (1-\alpha)\big(\kappa w_i^* + (1-\kappa)/M\big)$ as numerator part and $q_i = \alpha \mathcal{Z} S_i$ as denominator part (excluding λ). Then, Equation 3 becomes: $w_i = \frac{p_i}{q_i+\lambda}$ and the calculation of the λ consists in solving the following equation: $\sum_{i=1}^{M} \frac{p_i}{q_i+\lambda} = 1$.

We use a dichotomic search to determine an approximate solution to this equation (line 9). Consequently, it is necessary to bound the λ to initialize this search:

Property 1 (Bounds)**.** *The Lagrange multiplier λ is bounded as follows:*

$$\underbrace{-\min_i(q_i)}_{\inf_\lambda} \leq \lambda \leq \underbrace{\sum_{i=1}^{M} \min_i(p_i) - \max_i(q_i)}_{\sup_\lambda}$$

Proof. Using the bound $\sum_{i=1}^{M} w_i \leq \sum_{i=1}^{M} \max_i(w_i)$ and we get $\sum_{i=1}^{M} \min_i(p_i) - \max_i(q_i)$ as an upper bound of λ. Furthermore, the positive constraint on $\mathbf{W}$ induces that $w_i > 0$. Thus, $\min_i(w_i) = \min_i(q_i) + \lambda > 0$ and we obtain that $-\min_i(q_i)$ is a lower bound of λ. □

Properties 2 and 3 show that the algorithm MAPK-means converges and that its time complexity is reasonable in comparison with the state-of-the-art methods:

Property 2 (Termination)**.** *MAPK-means converges to a locally optimal solution in a finite number of steps.*

Proof. Cluster assignment and centroid re-estimation decreases the intra-cluster distance without changing preferential and regularization terms. Besides, the metric learning minimizes the objective function $\mathcal{I}_{map}$. As the three steps of MAPK-means decrease $\mathcal{I}_{map}$ (which is bounded by 0), it converges to a locally optimal solution in a finite number of steps. □

TABLE I: Data sets summary.

	Iris	*Wine*	*Ionosphere*	*Optdigits*	*Pendigits*	*Pgblocks*	*Vowel*	*Wdbc*
N	150	178	351	5620	10992	5473	990	569
M	4	13	34	64	16	10	10	30
K	3	3	2	10	10	5	11	2

Property 3 (Complexity)**.** *The time complexity of MAPK-means is $O(i(NKM + NM + jM))$ where i is the number of iterations and j the number of dichotomic search iterations.*

Proof. The time complexity of cluster assignment and centroid re-estimation steps are respectively $O(NKM)$ and $O(NM)$. The time complexity of metric learning mainly relies on a Lagrange multiplier resolution benefiting from a dichotomic search. Its time complexity is $O(jM)$. □

The time complexity of MAPK-means is less than the complexity of [10] where the computation of weights optimization is quadratic with $P+M$ (where P is the number of preferences which is upper bounded by M^2).

V. Experiments

We first present in this section results related to the setting of the parameter α that show that its value can be set by default. Second, we show that in some cases, it is important to be able to express quantitative constraints on attributes to reach the best possible clustering solution as a trade-off between data and user-driven exploration. Finally, we compare our new MAPK-means to the method introduced in [10] and show that we achieve slightly better results, but solved more efficiently and depending on a single parameter that is very easy to be understood and set by a user.

A. Data sets and evaluation measures

We performed experiments on multivariate attributes data sets from UCI repository[1] for the ease of reproducibility and comparison with other approaches like [10]. Table I details for each data set its number of data points N, its number of attributes M[2], and its number of clusters K.

We use mainly two methods to evaluate our experiments. The first evaluates the quality of the attribute weights learned by MAPK-means. To this aim, we consider the extensive work on features selection detailed in [23] as a ground-truth reference. According to them, the relevant feature dimensions for Iris, Wine and Ionosphere are respectively the attributes {3, 4}, {7, 12, 13} and {1, 3}. During the experiments we refer to several settings on the initial preferences ($\mathbf{W}^*$) depending on those identified attributes as follows: (1) random weights on attributes, (2) uniform distribution of weights, (3)

[1] archive.ics.uci.edu/ml/datasets.html

[2] The attributes 1 and 2 for Ionosphere, 1 and 40 for Optdigits are constant for all data points, and therefore have been discarded from our analysis.

important weights on *relevant* attributes, (4) important weights on *irrelevant* attributes.

The second is the Normalized Mutual Information (NMI) [10] which is a quality measure that measures the agreement between two partitions based on the amount of knowledge we gain about one partition knowing the other. Its value ranges from 0 to 1: 0 indicates that the two partitions are completely independent and 1 means that they are identical.

B. Tuning α parameter

The aim of this first experiment is to show that for a specific range of α, and particularly for $\alpha \approx 0.5$, it is always possible to achieve good clustering results on our benchmark data sets.

a) Experimental setting: For different initialization scenarios of preferences ($\mathbf{W}^*$), we vary the values of α in (0, 1). For each value of α, we consider all the values of $\kappa \in [0, 1]$ (0 ignores and 1 enforces completely the user preferences), and for each of these values of κ we run several tests to ensure the significance of the results. Here the number of tests is set to $0.2 \times N$. From these tests, we only keep the best clustering, that is to say the one that minimizes the intra-cluster distance. Indeed, our objective function expresses a trade-off between user preferences and the natural distribution of data. If the objective is to give more weight to the data part (while not ignoring the user preferences) it is more efficient to consider the intra-cluster distance as an evaluation measure.

At this point, for each α value, we have a best clustering for all $\kappa \in [0, 1]$. Then, it is possible to keep the overall best clustering called $\max_\alpha(NMI)$, based on the NMI score as we want to evaluate in the end the ability to produce the expected clustering.

b) Results: Figure 2 shows the variation of the learned vector $\mathbf{W}$ and NMI scores for values of α ranging between 0 and 1 (exclusive). The results are only shown here in worst case for Iris data set, i.e. with *irrelevant* initial user preferences $\mathbf{W}^*$ as the results are similar if not better with *relevant* initial user preferences. Similar results have also been observed for Wine and Ionosphere. For all data sets, setting more weight to the preference and regularization terms by decreasing the value of α leads to a decrease in the weights of the *relevant* attributes and consequently in the NMI score. Conversely, the closer α gets to 1, the more weight is given to the intra-cluster term which in turns conducts to converge to a solution with only one relevant attribute, which is not desirable.

It can be seen on the example of Figure 2 that for $\alpha \approx 0.5$ the *relevant* attributes (w_3 and w_4) are discovered and that NMI is nearly maximal. Section V-C provides results obtained with $\alpha = 0.5$ and also illustrates on several experiments that this choice for α is appropriate.

C. Impact of user preferences

Our aim is to motivate the importance of quantitative user preferences on attributes for clustering. Indeed, we illustrate in this section that the best clustering solution using metric learning is not always obtained by a data-driven exploration (i.e. $\kappa = 0$), neither it is the case when the exploration is only guided by the user (i.e. $\kappa = 1$), but when a compromise is established as reflected by MAPK-means with $\kappa \in (0, 1)$.

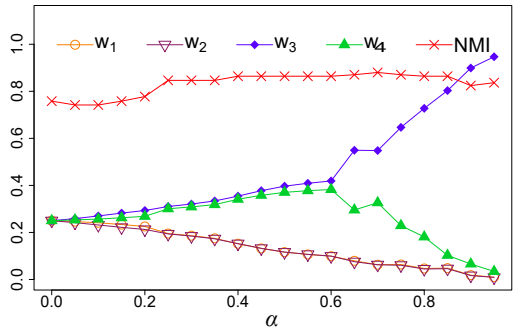


Fig. 2: Variation of w_i and the NMI with respect to α values for *Iris* data set with initial weights on *irrelevant* attributes.

a) Experimental setting: We consider the 4 initial settings of $\mathbf{W}^*$ described in Section V-A. Then, similarly to what is done in Section V-B, for each of the setting, we observe the best NMI scores and the attribute weights $\mathbf{W}$ that have been learned for values of κ ranging from 0 to 1.

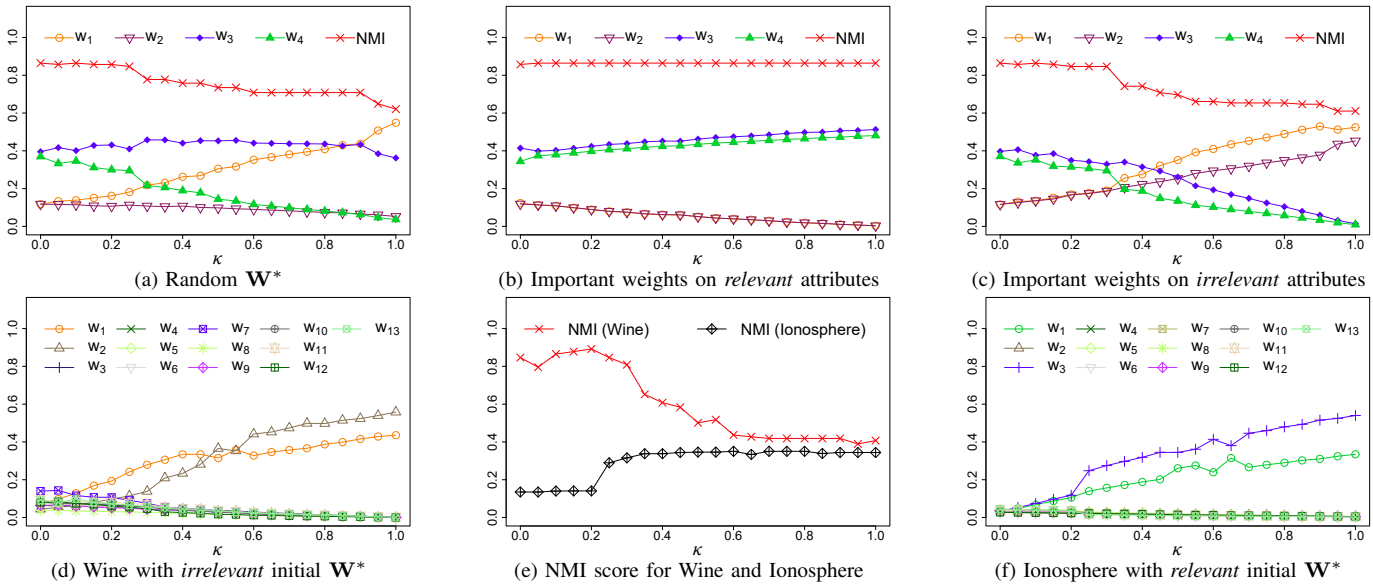
In the following, we present an extensive study of the results obtained on Iris. Then we focus on the most interesting cases for Wine and Ionosphere, for which we present the significant results.

1) Iris data set: The first experiment concerns an initialization with random weights on the different attributes ($w_1^* = 0.6, w_2^* = 0.1, w_3^* = 0.25, w_4^* = 0.05$). Figure 3a shows the evolution of learned weights (w_i) with respect to κ values. We can observe that user preferences ($\mathbf{W}^*$) are respected for $\kappa \approx 1$. These weights change when the value of κ decreases in a way to associate the important weights to *relevant* attributes (3 and 4), and less to others. The NMI increases by reducing κ which means that the obtained clustering comes closer to ground truth clustering. The instability of curves at some points is related to the centers initialization.

The second experiment is initiated with higher weights on *relevant* attributes 3 and 4 ($w_3^* = w_4^* = 0.49$) and lower (0.01) for the others. Figure 3b shows the coherence between user preferences and obtained results where the important weights remains on the *relevant* attributes when the value of κ changes. The NMI is also stable and close to 1, which means that the result is close to the ground truth.

Last experiment considers the case when the higher weights are on the *irrelevant* attributes - i.e. the user preferences are not pertinent. In this case, we set the weight equal to 0.4 for *irrelevant* attributes 1 and 2, and 0.1 to others. We show in Figure 3c that the NMI is lower when $\kappa \approx 1$, which is coherent with the *irrelevant* initial preferences. However the NMI increases when the weights of *relevant* attributes 3 and 4 learned by MAPK-means increase.

The experiments on Iris data set described before illustrate



(a) Random $\mathbf{W}^*$ (b) Important weights on *relevant* attributes (c) Important weights on *irrelevant* attributes

(d) Wine with *irrelevant* initial $\mathbf{W}^*$ (e) NMI score for Wine and Ionosphere (f) Ionosphere with *relevant* initial $\mathbf{W}^*$

Fig. 3: Variation of the learned weights of attributes ($\mathbf{W}$) and NMI with respect to κ values, using three scenarios for user preferences initialization for *Iris* data set, *irrelevant* initial $\mathbf{W}^*$ for *Wine* and *relevant* initial $\mathbf{W}^*$ for *Ionosphere*.

the efficiency of our approach in case of data sets with few attributes: *(i)* the preferences of user represented by $\mathbf{W}^*$ are always respected when κ is close to 1, *(ii)* the learning process without user preferences (i.e. $\kappa \approx 0$) associates important weights to the *relevant* attributes, *(iii)* the stability of the NMI for some intervals of variation of κ shows that the number of potential target clustering, to be suggested to the user, are limited.

2) Wine data set: In this experiment, we consider the case where the user sets MAPK-means with *irrelevant* initial preferences: weights are 0.495 for the first two attributes, and 0.001 for others. Figure 3d shows that the weights w_1 and w_2 of *irrelevant* attributes decrease with the value of κ, while the corresponding NMI increases (see Figure 3e). The NMI is maximized at $\kappa = 0.16$, and the weights of *relevant* attributes are greater than uniform weight distribution $> 1/M$. This experiment shows that the trade-off between data and user-driven exploration allows for a better clustering solution.

3) Ionosphere data set: In this experiment, we consider the case where the user preferences are pertinent, i.e. where the important weights are set to *relevant* attributes. In this case, $\mathbf{W}^*$ is initialized as follows: 0.401 for *relevant* attributes 1 and 3, and $w_i = 0.0066$ (for $i \neq 1, 3$). As shown in Figure 3f, the weights of w_1 and w_3 decrease with κ, until all the weights are approximately equals at $\kappa = 0$[3]. The values of NMI are maximal for κ between $(0.25, 1]$, then they decrease when $\kappa \in [0, 0.25]$ where the weights on *irrelevant* attributes represent more than 70% of the sum of all weights. This decrease is due to the consideration of a large number of *irrelevant* attributes in the learning process, while initially the results are computed based on *relevant* weights (the weights of *irrelevant* attributes initially represents less than 15% of the sum of all weights).

The results obtained on Wine and Ionosphere show that the best clustering quality can be achieved using at the same time the user preferences and the regularization term, that is to say by using a weight $0 < \kappa < 1$. More generally, these experiments illustrate how important it is to consider user preferences to improve the quality of clustering, in particular in case of relatively high-dimensional data sets as Ionosphere.

D. Comparative experiments

Finally, we compare the performances of MAPK-means with CFP algorithm introduced in [10]. Compared with our approach, note that CFP uses a *qualitative* model of preferences on attributes rather than a *quantitative* model using weights on attributes. However, it is important to note that, similarly to our proposal, [10] learns a metric parameterized by an attribute feature weight vector, i.e. the most appropriate weight vector with respect to the data set and the user preferences.

a) Experimental setting: For the purpose of this experiment, we replicate the same protocol as [10]. Similarly to [10], we first define the *natural* most interesting attributes by computing a weight vector $\tilde{W}$ using the inverse *intra-cluster distortion* Γ_i computed for each attribute. More precisely, the weight of each attribute is defined as follows: $\tilde{w}_i = \frac{\Gamma_i}{\sum_{d=1}^{M} \Gamma_d}$. In our approach, this weight vector $\tilde{W}$ is used to initialize our preference vector $\mathbf{W}^*$, i.e. $\mathbf{W}^* = \tilde{W}$. Then, similarly to [10], in order to select the best clustering over different runs and different values of $\kappa \in [0, 1]$, we consider the one that minimizes the value of our objective function (see Equation

[3]We are limited in Figure 3f to the weights of the first 13 attributes for readability reasons. Note that the weights of other attributes have the same behavior as the *irrelevant* attributes and that their values vary in $(0, 0.04)$ for all $\kappa \in [0, 1]$

TABLE II: NMI values for clustering results on K-means, K-means with a weighted distance, CFP [10] and our algorithm MAPK-means. The results of MAPK-means are obtained with $\kappa = 0$, $\kappa = 1$ and $\kappa \in [0, 1]$ which maximizes the NMI.

	K-means	*WK-means*	*CFP*	*MAPK-means*			
				$\kappa = 0$	$\kappa = 1$	$\kappa \in [0, 1]$	
Iris	0.742	0.758	**0.864**	0.778	**0.864**	**0.864**	1
Optdigits	**0.756**	0.743	0.715	0.655	**0.720**	**0.720**	1
Pendigits	0.682	0.710	0.707	0.698	**0.718**	**0.735**	0.95
Pgblocks	0.150	0.149	**0.204**	0.107	0.202	**0.204**	0.68
Vowel	0.415	0.397	0.424	0.387	**0.453**	**0.473**	0.83
Wdbc	0.623	0.613	0.628	0.605	**0.665**	**0.677**	0.78
	NMI					$\max(NMI)$	κ

1). Finally, we set $\alpha = 0.5$ and run 100 tests to ensure the significance of the results.

b) Results: The clustering results on all the data sets are shown in Table II. This table compares the clustering results in terms of NMI of the algorithms K-means, K-means with a weighted distance, CFP, for which we present only the best result obtained in [10] (using different values of their parameters m) and finally MAPK-means for which we provide several results, obtained respectively when:

- $\kappa = 0$: this result is equivalent to the result obtained using MPCK-means [19] with metric learning but without ML and CL constraints.
- $\kappa = 1$: the NMI value is obtained when we enforce the user preferences.
- $\kappa \in [0, 1]$: we show the value of NMI of the clustering that maximizes the objective function (see Equation 1). We also give the associated value of parameter κ.

As can be seen in Table II, the best NMI values obtained with CFP and MAPK-means are very similar on *Iris* and *Pgblocks* data set. With the *Optdigits* data set, K-means gives the best NMI value; however, our algorithm MAPK-means outperforms CFP. For all other data sets, the quality of the clusters produced by MAPK-means is better than the quality of clusters produced by CFP, MPCK-means without instance constraints (i.e. MAPK-means with $\kappa = 0$) and a basic K-means. These results also show that even a K-means whose metric is set with the *relevant* weights cannot compete with MAPK-means. Finally, these experiments confirm that the best clustering quality can be achieved with a compromise ($\kappa \in (0, 1)$) between data and user-driven exploration.

VI. Conclusion

We propose in this paper a clustering method that allows the user to express preferences on attributes, relying on the learning of a distance metric. User preferences are formulated as a simple vector which is taken into account by the objective function. We demonstrate that this quantitative model of preferences leads to an efficient metric learning step iterated by our algorithm MAPK-means. Furthermore, experimental results illustrate the positive impact of user preferences on clustering quality and on helping the method finding the right subspace. We also observe that the best clustering result is not achieved by the fully data-driven approach, nor with the fully user-driven one. Finally we show that MAPK-means generally performs better than other algorithms of the literature.

We intend to integrate this framework in an exploratory system using OLAP operations to navigate between sets of attributes. More precisely, we would like to deduce the preference vector from the operations performed by the user. For now, the learned vector is used in a *descriptive* way to explain what attributes are significant to construct the K-partition. This vector could also be used in a *prescriptive* way to recommend the user to explore other subspaces.

References

[1] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering: Algorithms and Applications*. Chapman and Hall/CRC, 2013, pp. 29–60.

[2] V. Kumar and S. Minz, "Feature selection: A literature review," *Smart CR*, vol. 4, no. 3, pp. 211–229, 2014.

[3] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," *SIGKDD Explor. Newsl.*, vol. 6, no. 1, pp. 90–105, Jun. 2004.

[4] H.-P. Kriegel and A. Zimek, "Subspace clustering, ensemble clustering, alternative clustering, multiview clustering: what can we learn from each other," *In Proc. ACM SIGKDD Workshop MultiClust*, 2010.

[5] I. I. Davidson and S. Basu, "A survey of clustering with instance level constraints," *ACM Transactions on Knowledge Discovery from data*, pp. 1–41, 2007.

[6] S. Basu, A. Banerjee, and R. J. Mooney., "Semi-supervised clustering by seeding," in *Proc. of the 19th ICML*, 2002, pp. 27–34.

[7] T. F. C. oes, E. R. Hruschka, and J. Ghosh, "A study of k-means-based algorithms for constrained clustering," *Intelligent Data Analysis*, vol. 17, no. 3, pp. 485–505, 2013.

[8] X. Wang and I. Davidson, "Flexible constrained spectral clustering," in *Proc. of KDD*, 2010, pp. 563–572.

[9] C. Ruiz, M. Spiliopoulou, and E. Menasalvas, "Density-based semi-supervised clustering," *Data Mining and Knowledge Discovery*, vol. 21, no. 3, pp. 345–370, 2010.

[10] J. Sun, W. Zhao, J. Xue, Z. Shen, and Y. Shen, "Clustering with feature order preferences," *Intelligent Data Analysis*, vol. 14, pp. 479–495, 2010.

[11] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, "Automatic subspace clustering of high dimensional data for data mining applications," *SIGMOD Rec.*, vol. 27, no. 2, pp. 94–105, 1998.

[12] M. H. C. Law, A. Topchy, and A. K. Jain, "Clustering with soft and group constraints," in *Proc. of Joint IAPR International Workshops, SSPR and SPR 2004*, 2004, pp. 662–670.

[13] A. Dubey, I. Bhattacharya, and S. Godbole, "A cluster-level semi-supervision model for interactive clustering," in *ECML PKDD*, Berlin, Heidelberg, 2010, pp. 409–424.

[14] B. M. Nogueira, A. M. Jorge, and S. O. Rezende, "Hcac: Semi-supervised hierarchical clustering using confidence-based active learning," in *Proc. of 15th Int. Conf. Discovery Science*, 2012, pp. 139–153.

[15] H. Liu and Y. Fu, "Clustering with partition level side information," in *IEEE ICDM*, Nov 2015, pp. 877–882.

[16] E. Y. Liu, Z. Guo, X. Zhang, V. Jojic, and W. Wang, "Metric learning from relative comparisons by minimizing squared residual," in *Proc. IEEE 12th ICDM*, 2012, pp. 978–983.

[17] A. Bouchachia and W. Pedrycz, "A semi-supervised clustering algorithm for data exploration," in *Proc. of the 10th IFSA*, 2003, pp. 328–337.

[18] E. P. Xing, A. Y. Ng, M. Jordan, and S. Russel, "Distance metric learning, with application to clustering with side-information," in *Proc. of NIPS*, 2002, pp. 505–512.

[19] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *In Proc. of the 21st ICML*. ACM, 2004, p. 11.

[20] D. Klein, S. Kamvar, and C. Manning, "From instance-level constraints to space-level constraints: making the most of prior knowledge in data clustering," in *Proc. of the 19th ICML*, 2002, pp. 307–314.

[21] J. Wang, S. Wu, and G. Li, "Clustering with instance and attribute level side information," *Int. Journal of Computational Intelligence Systems*, vol. 3, no. 6, pp. 770—785, 2010.

[22] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proc. Symp. Discrete Algorithms*, 2007, pp. 1027–1035.

[23] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *J. Mach. Learn. Res.*, vol. 5, pp. 845–889, 2004.