Applying subclustering and L$_p$ distance in Weighted K-Means with distributed centroid

Renato Cordeiro de Amorim[a,*], Vladimir Makarenkov[b]

[a] School of Computing, University of Hertfordshire, College Lane, AL10 9AB, UK.
[b] Département d'informatique, Université du Qubec à Montréal, C.P. 8888 succ. Centre-Ville, Montreal (QC) H3C 3P8 Canada.

Abstract

We consider the weighted K-Means algorithm with distributed centroids aimed at clustering data sets with numerical, categorical and mixed types of data. Our approach allows given features (i.e., variables) to have different weights at different clusters. Thus, it supports the intuitive idea that features may have different degrees of relevance at different clusters. We use the Minkowski metric in a way that feature weights become feature re-scaling factors for any considered exponent. Moreover, the traditional Silhouette clustering validity index was adapted to deal with both numerical and categorical types of features. Finally, we show that our new method usually outperforms traditional K-Means as well as the recently proposed WK-DC clustering algorithm.

Keywords: clustering, mixed data, feature weighting, K-Means, Minkowski metric.

1. Introduction

Clustering algorithms aim to find natural groups in a given data set so that each group is composed of similar entities (i.e., objects), whereas entities between groups are dissimilar. These data-driven algorithms are commonly used in exploratory data analysis without learning any supplementary information

[*] Corresponding author
Email addresses: r.amorim@herts.ac.uk (Renato Cordeiro de Amorim), makarenkov.vladimir@uqam.ca (Vladimir Makarenkov)

from test data. Clustering has been used to address various practical problems such as: image segmentation [1, 2], mining text data [3, 4], marketing [5], general data mining [6], bioinformatics [7, 8], etc.

Over the years there has been a considerable research effort in data clustering, generating a large number of clustering algorithms. Such a diversification of clustering algorithms can be explained by a variety of different ways in which data groups, normally referred to as clusters, may be formed. Some algorithms allow a given entity to belong to two or more clusters, sometimes even with different degrees of membership, but most of them allow it to belong to a single cluster only [9, 10]. In this paper, we are particularly interested in the latter approach, including data partitioning algorithms with a crisp membership.

Among this kind of partitioning algorithms K-Means [11, 12] is, arguably, the most popular one [6, 13]. K-Means clusters a given data set Y composed of entities y_i for $i = 1, 2, ..., N$, each described over features $v_1, v_2, ..., v_V$ into K non-overlapping partitions. Each cluster $k = 1, 2, ..., K$ has a single centroid $c_1, c_2, ..., c_K$, which represents the cluster's centre. K-Means minimizes the sum of the squared errors (E) between entities and their respective centroids, as shown below:

$$E = \sum_{k=1}^{K} \sum_{i=1}^{N} u_{ik} \cdot d(y_i, c_k), \tag{1}$$

where $u_{ik} \in \{0, 1\}$ is a binary variable, specifying whether the entity y_i is assigned to cluster k or not, and d is a function returning the distance between y_i and c_k. In most cases, the distance in use is the squared Euclidean distance in which

$$d(y_i, c_k) = \sum_{v=1}^{V} (y_{iv} - c_{kv})^2. \tag{2}$$

In this case, calculating centroids becomes a trivial exercise:

$$c_{kv} = \frac{\sum_{i=1}^{N} u_{ik} \cdot y_{iv}}{\sum_{i=1}^{N} u_{ik}}. \tag{3}$$

The minimization of (1) can be done using the iterative method below, which is carried out until convergence.

1. Select K entities from Y at random and copy their values to the initial centroids $c_1, c_2, ..., c_K$. Set each $u_{ik} = 0$.
2. Assign each entity $y_i \in Y$ to the cluster represented by its closest centroid. If entity y_i is assigned to cluster k, then set u_{ik} to 1. If there are no changes in u_{ik}, then stop the algorithm.
3. Update each centroid to the centre of its cluster. Go to Step 2.

Because of its wide use, the weaknesses of K-Means are rather well known. The main of them are as follows: (i) there is no guarantee that the algorithm will reach global optimum; (ii) K has to be known beforehand; (iii) it assumes that each feature has the same degree of relevance (i.e., the same weights); (iv) it does not support categorical features.

Aiming to address the point (iii), Makarenkov and Legendre [14] and Chan et al. [15] introduced the Weighted K-Means (WK-Means). This algorithm automatically sets a weight, s_v, for each feature v in the data set, which is then incorporated into the calculation of distances (see Section 2 for more details). The approach of Chan et al. [15] also considers a new parameter, β, which is the exponent of s_v, but does not specify a clear way of assessing it. However, WK-Means does outperform K-Means and seems to work well in general [15, 4].

Regarding the point (iv) above, K-Means, as many other algorithms, assumes that any categorical feature has to be transformed into numerical. There are indeed various methods for such a transformation [6]. However, most of them result in information loss [16].

The latter issue has motivated recent developments of clustering algorithms allowing for both numerical and categorical types of features. Huang [17] introduced the k-prototype algorithm, a popular variation of K-Means designed to deal directly with data sets containing mixed data types. This approach has been also extended to fuzzy clustering [10].

Kim et al. [9] introduced the concept of fuzzy centroids. In this concept, the centroids are no longer represented by hard-type values, but instead by fuzzy centroids. The latter authors showed that their approach works quite well in a fuzzy scenario, in which a given entity may belong to two or more clusters with different degrees of membership. Ji et al. [18], inspired by the concept of fuzzy centroids, has recently introduced the concept of distribution centroids which represent the centers of clusters for categorical features in a crisp scenario, rather than in a fuzzy one. Ji et al. [18] also incorporated into their algorithm the variable weight estimation procedure, thus addressing the above-mentioned weaknesses (iii) and (iv) of traditional K-Means.

The WK-Means algorithm with distributed centroids (WK-DC) was proved to work well on data sets with numerical, categorical and mixed data types [18]. However, as we will show here, there is still room for further improvement of this approach.

The main contribution of this paper is to further improve the WK-DC algorithm by: (i) applying subclustering, so that a given feature v may have different weights at different clusters, and thus supporting the intuitive idea that features may have different degrees of relevance at different clusters; (ii) using the L_p metric with the same exponent p in the weight and distance calculation so that feature weights could be seen as feature re-scaling factors for any exponent; (iii) calculating the centroids by using the minimization of the L_p metric; (iv) showing how the Silhouette clustering validity index can be adapted to deal with both numerical and categorical types of features.

2. Background and related work

The Weighted K-Means algorithm (WK-Means) aims to estimate the degree of relevance s of each feature v, taking it into account in the clustering process [15, 14]. The concept of feature weighting is closely related to that of distance, requiring an update in the latter. WK-Means incorporates feature weighting in the squared Euclidean distance, by defining the distance between entity y_i and centroid c_k as $\sum_{v=1}^{V} s_v^{\beta} \cdot (y_{iv} - c_{kv})^2$, where s_v is the weight of feature v, and β is a user-selected exponent.

The partitioning of a data set Y, containing N entities y_i, $i = 1, 2, ..., N$, each described over V features, $v_1, v_2, ..., v_V$, into K clusters is given by minimizing the WK-Means criterion, which is as follows:

$$E = \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{v=1}^{V} u_{ik} \cdot s_v^{\beta} \cdot d(y_{iv}, c_{kv}), \tag{4}$$

where c_{kv} is the feature v of the centroid of cluster k. The WK-Means criterion is subject to $\sum_{v=1}^{V} s_v = 1$ and $0 \leq s_v \leq 1$ (for $v = 1, \ldots, V$), and a crisp clustering in which a given entity y_i is assigned to a single cluster. The binary variable u_{ik} indicates whether y_i is assigned to cluster k:

$$u_{ik} = \begin{cases} 1, \text{ if } y_i \text{ belongs to cluster } k, \\ 0, \text{ if } y_i \text{ does not being to cluster } k. \end{cases} \tag{5}$$

The algorithm to minimize WK-Means is very similar to that of K-Means shown in Section 1, with a couple of extra considerations only. First, each s_v is initialized to a non-negative random value, but still subject to $\sum_{v=1}^{V} s_v = 1$. Second, after updating each centroid, the algorithm updates each feature weight as well, and then goes back to Step 2. This weight update occurs in a similar fashion to that of WK-Means with distributed centroids (explained in Section 3). The WK-Means algorithm was shown to provide steady results [15, 4], but it does not deal with categorical features directly. A data set containing categorical features would need to be pre-processed to transform such features, or a k-prototype algorithm should be applied.

To address this important limitation, Ji et al. [18] proposed to use distributed centroids in WK-Means (WK-DC), showing that WK-DC outperforms other popular partitioning algorithms such as k-prototype, SBAC and KL-FCM-GM [19, 10]. The WK-DC algorithm deals with data sets containing both numerical and categorical features, but still represents each partition using a single centroid $C = \{c_1, c_2, ..., c_K\}$. The centroid of a cluster k is represented by $c_k = \{c_{k1}, c_{k2}, ..., c_{kV}\}$. This representation is straightforward for a numerical variable v. If v is a categorical variable containing T categories $a \in v$, then $c_{kv} = \{\{a_v^1, w_{kv}^1\}, \{a_v^2, w_{kv}^2\}, ..., \{a_v^t, w_{kv}^t\}, ..., \{a_v^T, w_{kv}^T\}\}$.

The above representation of the centroid of a categorical feature v allows each category $a \in v$ to have a weight directly related to its frequency in the data set Y, where the value of w_{kv}^t is calculated using the following equation:

$$w_{kv}^t = \sum_{i=1}^{N} \eta_{tk}(y_{iv}), \tag{6}$$

and,

$$\eta_{tk}(y_{iv}) = \begin{cases} \left(\sum_{j=1}^{N} u_{jk}\right)^{-1}, \text{ if } y_{iv} = a_v^t, \\ 0, \text{ if } y_{iv} \neq a_v^t. \end{cases} \tag{7}$$

Unlike WK-Means, the distance $d(y_{iv}, c_{kv})$ for a numerical v in WK-DC is set to be the Manhattan distance, defined by $|y_{iv} - c_{kv}|$. Obviously, the Manhattan distance cannot be applied if v is categorical. In this case, $d(y_{iv}, c_{kv})$ is equivalent to $\varphi(y_{iv}, c_{kv})$, where:

$$\varphi(y_{iv}, c_{kv}) = \sum_{t=1}^{T} \vartheta(y_{iv}, a_v^t), \tag{8}$$

and,

$$\vartheta(y_{iv}, a_v^t) = \begin{cases} 0, \text{ if } y_{iv} = a_v^t, \\ w_{kv}^t, \text{ if } y_{iv} \neq a_v^t. \end{cases} \tag{9}$$

Moreover, the WK-DC algorithm incorporates the feature weighting component of WK-Means. Thus, each feature $v \in V$ has a weight s_v representing its degree of relevance, which is assumed to be inversely proportional to its dispersion. This is an intuitive concept in which features that are compact, and by consequence have a smaller dispersion, are thought to be more relevant than features that are sparser, and by consequence have a higher dispersion. The dispersion of v is given by D_v, as shown below:

$$D_v = \sum_{k=1}^{K} \sum_{i=1}^{N} u_{ik} \cdot d(y_{iv}, c_{kv}). \tag{10}$$

Using D_v, it becomes possible to calculate s_v:

$$s_v = \begin{cases} 0, \text{ if } D_v = 0, \\ \left(\sum_{\tau=1}^{h} [\frac{D_v}{D_\tau}]^{\frac{1}{(\beta-1)}}\right)^{-1}, \text{ if } D_v \neq 0. \end{cases} \tag{11}$$

where h is the number of features for which $D_v \neq 0$. Note that Equation (11) leads to a division by zero at $\beta = 1$. In this case, the minimization of s results in finding the feature with the highest D_v for which $s_v = 1$, while the weight of all other features is set to zero [15].

There has been limited indication for what values of the exponent β WK-Means provides the best clustering results, or how β could be successfully estimated for a particular data set. On the other hand, WK-DC was introduced using solely, and rather successfully, $\beta = 8$ [18]. Based on this success, we decided to extend WK-DC by addressing the following three major points. First, the WK-DC makes use of a single weight per feature s_v. We believe that it would be more intuitive to apply subclustering, in which a given feature v could have different weights at different clusters (i.e., effectively using s_{kv} rather than s_v). Obviously, s_{kv} will be subject to the following constraints: $\sum_{v=1}^{V} s_{kv} = 1$ and $0 \leq s_{kv} \leq 1$ (for $v = 1, \ldots, V$), for each fixed value of $k = 1, 2, ..., K$.

Second, the difference of one between the exponent β and the distance exponent in WK-DC [18] does not allow the feature weights to be seen as feature re-scaling factors. If the two exponents were identical, the weights could be used to re-scale the given data set as a part of data pre-processing step. This is not only a matter of setting β to one, as in this case the minimization of (10) would set the weight s_v of a single feature v to one and all other feature weights to zero [15]. We propose to change the distance exponent instead, considering the L$_p$ metric instead of the Manhattan distance.

Third, the WK-DC algorithm proposed by Ji and colleagues [18] uses the Manhattan distance to assign entities to clusters. It relies on the mean to determine the cluster centroids. It is well known, however, that the median instead of the mean should be used with the Manhattan distance. In our algorithm, we ensure that each cluster centroid will have the smallest sum of distances to all of the cluster's entities by aligning the distance used to assign entities to centroids with the minimization used to find each centroid. The distance we consider here is the Minkowski distance and its minimization is given by the steepest descent algorithm [20] discussed below.

3. The Minkowski Weighted K-Means with distributed centroid

We have recently considered the use of the L$_p$ distance in clustering with Minkowski Weighted K-Means [20]. Unfortunately, our algorithm [20], as the original WK-Means algorithm, was not designed to deal with categorical features directly. The L$_p$ distance between an entity y_i and a centroid c_k is given by $\sqrt[p]{\sum_{v=1}^{V} |y_{iv} - c_{kv}|^p}$. Here, we propose to remove the p^{th} root, similarly to the popular use of the squared Euclidean distance in clustering, as well as consider the feature weights.

In our new algorithm, called Minkowski Weighted K-Means with distribution centroid (MWK-DC), we will apply subclustering allowing a feature v to have different degrees of relevance at different clusters. This will have an impact on the distance calculation as we will show below:

$$d_p(y_i, c_k) = \sum_{v=1}^{V} s_{kv}^{p} \cdot |y_{iv} - c_{kv}|^p, \tag{12}$$

where p, an exponent of both the distance and the feature weight, is a user-defined parameter. As we will show in the next section, an optimal value of p can be selected using a modified version of the Silhouette clustering index which will be adapted to deal with both numerical and categorical types of features. Any distance measure creates a bias in terms of the shape of clusters which the partitioning algorithm should be able to recover. WK-DC uses the Manhattan distance, meaning that the bias is towards diamond shapes. WK-Means uses the squared Euclidean distance, setting the bias towards spherical clusters. When the L$_p$ distance is used, the resulting bias depends on p. If $p = 1$, our function will be the Manhattan distance; if $p = 2$, our function will be the squared Euclidean distance; and if $p \to \infty$, the bias will be towards squares. In summary, by using the L_p distance we can set the bias given by the distance to any interpolation between a diamond and a square. With the weighted L_p distance in (12), we can rewrite (4) as follows:

$$E_p = \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{v=1}^{V} u_{ik} \cdot s_{kv}^{p} \cdot |y_{iv} - c_{kv}|^p. \tag{13}$$

The calculation of s_{kv} in both WK-DC and MWK-DC is based on the dispersion of v. However, the use in MWK-DC of an equation similar to Equation (10) in WK-DC would be problematic. Indeed, consider a scenario in which a feature v has K different values which do not change within the same cluster $k = 1, 2, ..., K$. If we were to simply update (10) to calculate dispersions per cluster, it would lead to a weight of zero as per (11), even so such a feature seems a perfect feature to separate clusters. We solve this problem by adding a small constant to our dispersion. We define the dispersion, D_{kvp}, of feature v in cluster k at a given exponent p as follows:

$$D_{kvp} = \sum_{i=1}^{N} u_{ik} \cdot |y_{iv} - c_{kv}|^p + 0.01, \tag{14}$$

where the small constant of 0.01 is used to avoid the case of $D_{kvp} = 0$. We think that the features whose value is constant over all entities $y_i \in Y$ should be addressed in the data pre-processing stage, rather than by the clustering algorithm in question. Using (14), we can calculate the weight, s_{kv}, of feature v at cluster k as follows:

$$s_{kv} = \left(\sum_{u=1}^{V} \left[\frac{D_{kvp}}{D_{kup}} \right]^{\frac{1}{(p-1)}} \right)^{-1}, \tag{15}$$

where s_{kv} is subject to the following constraints: $\sum_{v=1}^{V} s_{kv} = 1$ and $0 \leq s_{kv} \leq 1$ (for $v = 1, \ldots, V$), for each $k = 1, 2, ..., K$. The only division by zero in (15) would happen if $p = 1$. In this case, the minimization of our objective function implies that for a given cluster k, only a single feature, that with the highest D_{kv}, has the weight of one, whereas all other features have the weight of zero.

The algorithm used to iteratively minimize our objective function, given by Equation (13), is presented below:

1. Randomly select K entities $y_i \in Y$ and set their values as initial centroids $c_1, c_2, ..., c_K$. Set each weight s_{kv} randomly, but subject to $\sum_{v=1}^{V} s_{kv} = 1$ and $0 \leq s_{kv} \leq 1$ (for $v = 1, \ldots, V$), for each $k = 1, 2, ..., K$.
2. For each entity $y_i \in Y$ and cluster $k = 1, 2, ..., K$, set $u_{ik} = 1$ iff c_k is the closest centroid to y_i as per Equation (12), otherwise set $u_{ik} = 0$.
3. Update each centroid c_k for $k = 1, 2, ..., K$ to the L$_p$ centre of its cluster. If there are no changes, then stop the algorithm and output u, $c_1, c_2, ..., c_K$, and s.
4. Update each feature weight s_{kv}, applying Equation (15). Go back to Step 2.

At $p = 1$, $p = 2$ and $p = \infty$, the L$_p$ center of a cluster is given by the component-wise median, mean and midrange, respectively. For other values of p the center can be obtained using a steepest descent algorithm [20], since we can assume $p \geq 1$.

4. Setting of the experiments

The original paper introducing the use of the distributed centroid in WK-Means [18] does not deal with data normalization. Its authors probably assume that the normalization is carried out during the feature weighting process, which may not be true in all cases. The distance between y_{iv} and a c_{kv} for a categorical v, given by $\varphi(y_{iv}, c_{kv})$ in (8), has a maximum of one. However, if v is numerical, the maximum distance, given by $\sum_{v=1}^{V} |y_{iv} - c_{kv}|$, will tend to the range of v.

We can solve this problem by normalizing each numerical feature in such a way that the maximum distance between entities and centroids is the same, whether v is numerical or categorical. Often data sets are normalized using the popular z-score standardization. In this, $y_{iv} = \frac{y_{iv} - \bar{y_v}}{stdev(y_v)}$, where $stdev(y_v)$

represents the standard deviation of feature v over $y_i \in Y$. In this study, we chose a different normalization, which is as follows:

$$y_{iv} = \frac{y_{iv} - \bar{y_v}}{range(y_v)}, \tag{16}$$

where $\bar{y_v}$ represents the average value for v over each entity $y_i \in Y$. The above equation guarantees that any numerical feature $v \in V$ will have a range of one. This means that the maximum distance between any entity and its corresponding centroid will tend to one.

The choice of the range, rather than of the standard deviation in (16) has yet another interesting characteristic: unlike the latter, the former is not biased towards unimodal distributions [21, 6]. Consider the following example of two features, a being unimodal and b being bimodal. The standard deviation of b will be higher than that of a, which means that after the standardization the values of b will be smaller than those of a, thus having a smaller contribution to any distance-based clustering process. However, b is clearly a feature that can be used to discriminate between natural groups sought by any clustering algorithm.

We have applied the above normalization to the numerical features of all 12 real-life data sets we analyzed in this study. Table 1 describes each data set in terms of its number of entities, numerical features, categorical features and clusters. These data sets are freely available at the popular UCI machine learning repository [22].

Our experiments involved four different partitioning algorithms, which were the following: (i) K-Means; (ii) WK-DC; (iii) WK-DC S, a version of WK-DC using cluster dependant feature weights; and (iv) MWK-DC.

1. K-Means

   Traditional K-Means algorithm was carried out on the data sets with numerical features only. We have run 100 experiments per data set.

2. WK-DC

   As originally introduced in [18]. We have run 100 experiments per data

Table 1: 12 real-life data sets from the UCI machine learning repository [22] used in our experiments

		Numerical	Categorical	
	Entities	Features	Features	Clusters
Australian credit approval	690	6	8	2
Balance	625	0	4	3
Breast cancer	699	9	0	2
Car Evaluation	1728	0	6	4
Ecoli	336	7	0	8
Glass	214	9	0	6
Heart Disease	270	6	7	2
Ionosphere	351	33	0	2
Iris	150	4	0	3
Soya	47	0	35	4
Teaching Assistant	151	1	4	3
Tic Tac Toe	958	0	9	2

set.

3. WK-DC S

   A version of WK-DC in which a given feature v has K weights. We have added a small constant to each feature dispersion, according to Equation (14), to avoid issues related to features having the same value within a cluster. We have also run 100 experiments per data set.

4. MWK-DC

   As described in Section 3. We have run 100 experiments per value of $p = \{1.0, 1.1, 1.2, ..., 5.0\}$.

Several previous works, including our own, measured the quality of algorithms by estimating the proportion of correctly classified entities [20, 18]. However, we acknowledge that in some cases the estimated rate might be a poor measure for assessing the quality of a clustering solution [23]. Thus, in this study, we use the adjusted Rand index (ARI)[24] to measure the quality of clustering. We have also decided to discard any experiment in which we did not obtain the expected number of clusters.

Note that all the four algorithms we consider here are non-deterministic. This means that we not only have to run them many times, but also that we have to find what the best run was. Two sets of experiments were carried out. In the first, we were interested in determining which algorithm was the best in terms of cluster recovery. We ran each algorithm and compared its clustering solution to the known reference clustering by using the ARI index. Of course this approach is not feasible in real life. Thus, we also needed a method to identify optimal cluster distribution without any prior knowledge. This was investigated in our second set of experiments, in which we attempted to recover the correct clustering without considering any reference solution. We used the Silhouette width [25] to find this solution in unsupervised way:

$$si = \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{b(y_i) - a(y_i)}{max\{a(y_i), b(y_i)\}}, \tag{17}$$

where $a(y_i)$ is the average distance between y_i and all the entities in its cluster, and $b(y_i)$ is the lowest average distance between y_i and any cluster not including y_i. Note that $-1 < si < 1$, and the closer si is to one, the better the obtained clustering solution is. Here, the distance between y_{iv}, and y_{jv} where $y_i, y_j \in Y$, for a numerical feature v is given by the squared Euclidean distance $d(y_{iv}, y_{jv}) = (y_{iv} - y_{jv})^2$. Clearly, calculating si in a data set containing categorical features requires a distance measure that also supports categorical features. For this reason, we calculated the dissimilarity between categorical features using Equation (8) and initializing each w_{kv}^t using Equation (6).

5. Results and discussion

We conducted our simulations with the WK-DC, WK-DC S and MWK-DC algorithms which were applied to analyze the experimental data sets in Table 1. Moreover, we also carried out experiments using the traditional K-Means algorithm on the data sets containing solely numerical features. We did not include in our simulations other popular clustering algorithms capable of dealing with data sets containing categorical data, such as k-prototype, SBAC and KL-FCM-GM [19, 10, 17], because WK-DC was already shown to outperform all of them [18].

Table 2 shows the results of our experiments in terms of the adjusted Rand index (ARI) and the number of completed iterations necessary for convergence (Itr). We were looking for an algorithm that produces high values of ARI with a small number of iterations. We limited the number of iterations in each algorithm to a maximum of 100. In the cases where more than one optimum clustering solution was obtained (i.e., where different clustering solutions corresponded to the same optimum value of SI), we reported their average ARI and average number of iterations the algorithm took to converge. Table 2 has two main columns: (i) Optimal run, in which the optimal clustering is that with the highest ARI; and (ii) Optimal SI run, a totally unsupervised experiment in which the optimal clustering is that with the highest Silhouette width (SI).

Some interesting patterns can be observed when considering the results of Optimal run. WK-DC S produced equivalent or higher values of ARI than WK-DC for nine of the twelve data sets we considered. Our MWK-DC algorithm did even better, providing identical or higher values of ARI than WK-DC for eleven of the twelve data sets. WK-DC outperformed MWK-DC in a single data set, the Heart disease, with a difference of 0.045 in ARI. Table 2 also shows that WK-DC was unable to generate a clustering solution with the expected number of clusters for the Ecoli data set. This is possibly related to the fact that WK-DC does not minimize its distance when calculating centroids.

In terms of the number of iterations, WK-DC S provided identical or better results than WK-DC for nine data sets, whereas MWK-DC produced competitive or better results than WK-DC for ten of them. We believe that our alignment between the distance used to assign entities and the minimization used to find centroids was the key factor to achieve the reduction in the number of iterations. We can also see that WK-DC was the only algorithm to reach an optimum solution without converging when applied to the Teaching Assistant data set.

Our second set of experiments included a totally unsupervised clustering approach based on our modified Silhouette index capable of dealing with data sets containing categorical features. In these experiments WK-DC S generated mixed results, being equivalent or better than WK-DC in terms of ARI for six data sets only. However, our MWK-DC algorithm was equivalent or better than WK-DC in terms of ARI for nine data sets. In terms of the number of iterations the algorithms took to complete, WK-DC S provided equivalent or better results than WK-DC for eight data sets, whereas MWK-DC was equivalent or better than WK-DC for seven of them.

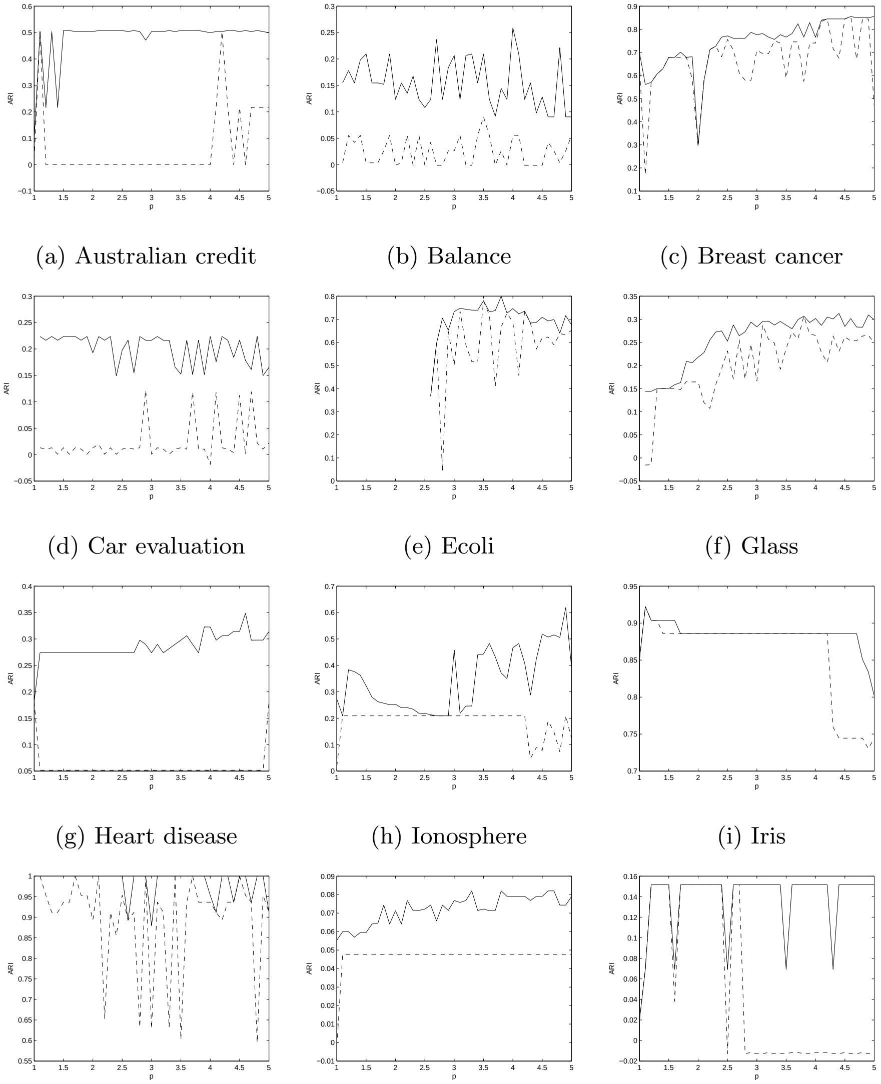
Figure 1 illustrates the relationship between the ARI index corresponding to the clustering solution provided by our MWK-DC algorithm and the exponent p. Here we present the maximum ARI per value of p (solid line), analogous to our experiments in Table 2 under Optimal run, as well as the ARI estimated using the Silhouette width for each value of p, analogous to our experiments under Optimal SI run. Note that sometimes the estimated ARI in Figure 1 appears to be higher than in Table 2. This is due to the fact that the former shows the ARI corresponding to the highest SI at each p, while the latter reports a single ARI related to the highest SI value obtained over all possible values of p. In all experiments, clusterings that did not contain the expected number of clusters were disregarded. We can see that this was a particular issue for the Ecoli data set at $p \leq 2.5$.

6. Conclusion

Most of the clustering algorithms have been designed to deal with data sets containing numerical features only. In this paper, we introduce the Minkowski Weighted K-Means with distributed centroids (MWK-DC), a feature weighting algorithm capable of dealing with both numerical and categorical types of features. Our algorithm extends the approach of Weighted K-Means with distributed centroids (WK-DC) [18] in three different ways: (i) it allows for an intuitive idea that one feature v may have different degrees of relevance at dif-

Table 2: Experimental results. Section Optimal run: the best results provided by MWK-DC over all considered values of p = 1.0, 1.1, ... , 5.0 are shown. Section Optimal SI run: SI (Silhouette) values were calculated over all considered values of p in MWK-DC (unsupervised clustering); the results corresponding to the clustering solution that maximized SI are shown.

		Optimal run			Optimal SI run			
		ARI	Itr	p	ARI*	Itr	p	Si
Australian	K-Means	-	-	-	-	-	-	-
	WK-DC	0.2976	8	-	0.0048	3	-	0.9444
	WK-DC S	0.5119	6	-	0.0001	2	-	0.9999
	MWK-DC	0.5077	3	1.5,1.6,2.1-2.4,2.6	0.1679	2	1.1,1.2	1.0000
Balance	K-Means	-	-	-	-	-	-	-
	WK-DC	0.1545	2	-	-0.0011	1	-	0.1467
	WK-DC S	0.1545	3	-	0.0553	2	-	0.9299
	MWK-DC	0.2588	1	4.0	0.0553	2	5.0	0.9594
Breast C.	K-Means	0.8823	4	-	0.8823	4	-	0.9107
	WK-DC	0.7400	6	-	0.7350	8	-	0.6510
	WK-DC S	0.6195	3	-	0.2015	3	-	0.9462
	MWK-DC	0.8552	5	5.0	0.5747	6	3.8	0.9595
Car Ev.	K-Means	-	-	-	-	-	-	-
	WK-DC	0.1323	2	-	0.0407	3	-	0.3142
	WK-DC S	0.1601	2	-	0.0842	4	-	0.9865
	MWK-DC	0.2236	3	13 ps	0.0081	4	1.1-1.3	1.0000
Glass	K-Means	0.2583	3	-	0.1735	8	-	-0.1150
	WK-DC	0.2895	13	-	0.2386	13	-	0.5058
	WK-DC S	0.2611	5	-	0.2069	13	-	0.1703
	MWK-DC	0.3126	11	4.4	0.2639	6	4.3	0.7976
Ecoli	K-Means	0.6357	13	-	0.4537	8	-	0.1968
	WK-DC	NaN	NaN	-	NaN	NaN	-	NaN
	WK-DC S	0.0318	2	-	0.0318	2	-	-0.0352
	MWK-DC	0.7989	22	3.8	0.6522	11	5.0	0.6845
Heart	K-Means	-	-	-	-	-	-	-
	WK-DC	0.3938	5	-	0.0280	1	-	0.8656
	WK-DC S	0.4323	10	-	0.1807	2	-	0.9996
	MWK-DC	0.3485	5	4.6	0.0932	3	1.1,1.2	1.0000
Ionosphere	K-Means	0.2038	4	-	0.2038	4	-	0.4125
	WK-DC	0.2092	3	-	0.2092	4	-	1.0000
	WK-DC S	0.3833	6	-	0.2092	6	-	0.9978
	MWK-DC	0.6178	14	4.9	0.2092	3	1.1	1.0000
Iris	K-Means	0.9222	2	-	0.8668	4	-	0.8640
	WK-DC	0.8512	9	-	0.7445	4	-	0.5846
	WK-DC S	0.8680	4	-	0.5685	2	-	0.6354
	MWK-DC	0.9222	4	1.1	0.8857	8	3.5	0.8663
Soya	K-Means	-	-	-	-	-	-	-
	WK-DC	0.9533	6	-	0.9366	5	-	0.7855
	WK-DC S	0.8178	3	-	0.3211	3	-	0.9134
	MWK-DC	1.0000	2	33 ps	0.9533	2	4.9	0.9778
Teaching A.	K-Means	-	-	-	-	-	-	–
	WK-DC	0.0320	100	-	0.0248	100	-	0.4611
	WK-DC S	0.0638	3	-	0.0477	3	-	0.9875
	MWK-DC	0.0820	5	3.3,3.8,4.6,4.7	0.0477	3	4.6	0.9949
TicTacToe	K-Means	-	-	-	-	-	-	-
	WK-DC	0.1515	5	-	0.0692	9	-	0.3071
	WK-DC S	0.1515	2	-	-0.0184	2	-	0.9283
	MWK-DC	0.1515	3	35 ps	-0.0128	2	5	0.9858



(a) Australian credit

(b) Balance

(c) Breast cancer

(d) Car evaluation

(e) Ecoli

(f) Glass

(g) Heart disease

(h) Ionosphere

(i) Iris

(j) Soya

(k) Teaching assistant

(l) Tic tac toe

Figure 1: Maximum possible ARI per data set at each value of p (solid line) and estimated ARI (using the Silhouette) per data set at each value of p (dashed line).

ferent clusters; (ii) it uses the L_p metric with the same exponent p in the weight and distance calculation. Thus, each feature weight can now be seen as a feature re-scaling factor for any considered exponent p; and (iii) it calculates the centroids by minimizing the metric used in the distance calculation. In our case, we proceed by minimizing the L_p metric.

Taking into account that both MWK-DC and WK-DC are non-deterministic, we ran them 100 times for each experimental data set and then selected the solutions maximizing the popular Silhouette width index. Using the concept of distributed centroids, we extended the classical application of Silhouette to support both numerical and categorical features.

Our results suggest that MWK-DC generally outperforms WK-DC as well as its subclustering version, WK-DC S, in terms of both cluster recovery, measured using the adjusted Rand index and the number of completed iterations required for convergence. Furthermore, we still see potential for extension of MWK-DC, considering its version allowing different distance biases at different clusters as well as its version including the fuzzy Minkowski Weighted K-Means procedure [26].

References

[1] S. Chen, D. Zhang, Robust image segmentation using fcm with spatial constraints based on new kernel-induced distance measure, Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 34 (4) (2004) 1907–1916. doi:10.1109/TSMCB.2004.831165.

[2] A. K. Jain, F. Farrokhnia, Unsupervised texture segmentation using gabor filters, Pattern recognition 24 (12) (1991) 1167–1186. doi:10.1016/0031-3203(91)90143-S.

[3] C. C. Aggarwal, C. Zhai, A survey of text clustering algorithms, in: C. C. Aggarwal, C. Zhai (Eds.), Mining Text Data, Springer, 2012, pp. 77–128. doi:10.1007/978-1-4614-3223-4_4.

[4] J. Z. Huang, J. Xu, M. Ng, Y. Ye, Weighting method for feature selection in k-means, in: H. Liu, H. Motoda (Eds.), Computational Methods of Feature Selection, Data Mining & Knowledge Discovery, Chapman & Hall/CRC, 2008, pp. 193–209. doi:10.1007/978-3-642-37807-2_2.

[5] P. Arabie, L. Hubert, Advanced methods in marketing research, Blackwell, 1994, chapter: Cluster Analysis in Marketing Research.

[6] B. Mirkin, Clustering: A Data Recovery Approach, Vol. 19, Chapman and Hall/CRC, Boca Raton, FL, USA, 2012.

[7] P. Baldi, G. W. Hatfield, DNA microarrays and gene expression: from experiments to data analysis and modeling, Cambridge University Press, 2002.

[8] J. D. MacCuish, N. E. MacCuish, Clustering in bioinformatics and drug discovery, CRC Press, 2011.

[9] D.-W. Kim, K. H. Lee, D. Lee, Fuzzy clustering of categorical data using fuzzy centroids, Pattern Recognition Letters 25 (11) (2004) 1263–1271. doi:10.1016/j.patrec.2004.04.004.

[10] J. Ji, W. Pang, C. Zhou, X. Han, Z. Wang, A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data, Knowledge-Based Systems 30 (2012) 129–135. doi:10.1016/j.knosys.2012.01.006.

[11] 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, University of California Press, Bekerley, CA, USA, 1967, pp. 281–297.

[12] G. Ball, D. Hall, A clustering technique for summarizing multivariate data, Behavioral Science 12 (2) (1967) 153–155. doi:10.1002/bs.3830120210.

[13] A. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (8) (2010) 651–666. doi:10.1016/j.patrec.2009.09.011.

[14] V. Makarenkov, P. Legendre, Optimal Variable Weighting for Ultrametric and Additive Trees and K-means Partitioning: Methods and Software., Journal of Classification 18 (2) (2001) 245–271. doi:10.1007/s00357-001-0018-x.

[15] E. Y. Chan, W. K. Ching, M. K. Ng, J. Z. Huang, An optimization algorithm for clustering using weighted dissimilarity measures, Pattern recognition 37 (5) (2004) 943–952. doi:10.1016/j.patcog.2003.11.003.

[16] C.-C. Hsu, S.-H. Lin, W.-S. Tai, Apply extended self-organizing map to cluster and classify mixed-type data, Neurocomputing 74 (18) (2011) 3832–3842. doi:10.1016/j.neucom.2011.07.014.

[17] Z. Huang, Clustering large data sets with mixed numeric and categorical values, in: Proceedings of the 1st Pacific-Asia Conference on Knowledge Discovery and Data Mining,(PAKDD), Singapore, 1997, pp. 21–34.

[18] J. Ji, T. Bai, C. Zhou, C. Ma, Z. Wang, An improved k-prototypes clustering algorithm for mixed numeric and categorical data, Neurocomputing 120 (2013) 590–596. doi:10.1016/j.neucom.2013.04.011.

[19] S. P. Chatzis, A fuzzy c-means-type algorithm for clustering of data with mixed numeric and categorical attributes employing a probabilistic dissimilarity functional, Expert systems with applications 38 (7) (2011) 8684–8689. doi:10.1016/j.eswa.2011.01.074.

[20] R. Cordeiro de Amorim, B. Mirkin, Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering, Pattern Recognition 45 (3) (2012) 1061–1075. doi:10.1016/j.patcog.2011.08.012.

[21] M. M.-T. Chiang, B. Mirkin, Intelligent choice of the number of clusters in k-means clustering: an experimental study with different cluster spreads, Journal of classification 27 (1) (2010) 3–40. doi:10.1007/s00357-010-9049-5.

[22] K. Bache, M. Lichman, UCI machine learning repository (2013).
URL http://archive.ics.uci.edu/ml

[23] D. Steinley, Properties of the hubert-arable adjusted rand index., Psychological methods 9 (3) (2004) 386. doi:10.1037/1082-989X.9.3.386.

[24] L. Hubert, P. Arabie, Comparing partitions, Journal of classification 2 (1) (1985) 193–218. doi:10.1007/BF01908075.

[25] P. J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, Journal of computational and applied mathematics 20 (1987) 53–65. doi:10.1016/0377-0427(87)90125-7.

[26] L. Svetlova, B. Mirkin, H. Lei, Mfwk-means: Minkowski metric fuzzy weighted k-means for high dimensional data clustering, in: Proceedings of the 14th International Conference on Information Reuse and Integration (IRI), IEEE, San Francisco, CA, USA., 2013, pp. 692–699.