Noname manuscript No.
(will be inserted by the editor)

Selecting the Minkowski exponent for intelligent K-Means with feature weighting

Renato Cordeiro de Amorim ·
Boris Mirkin

Accepted for publication as a chapter of: Clusters, orders, trees: methods and applications. (2013)

Abstract Recently, a three-stage version of K-Means has been introduced, at which not only clusters and their centers, but also feature weights are adjusted to minimize the summary p-th power of the Minkowski p-distance between entities and centroids of their clusters. The value of the Minkowski exponent p appears to be instrumental in the ability of the method to recover clusters hidden in data. This paper advances into the problem of finding the best p for a Minkowski metric based version of K-Means, in each of the following two settings: semi-supervised and unsupervised. This paper presents experimental evidence that solutions found with the proposed approaches are sufficiently close to the optimum.

Keywords Clustering · Minkowski metric · Feature weighting · K-Means.

1 Motivation and background

Clustering is one of the key tools in data analysis. It is used particularly in the creation of taxonomies when there are no accurate labels available identifying any taxon, or not enough such labels to train a supervised algorithm. K-Means is arguably the most popular clustering algorithm being actively used by practitioners in data mining, marketing research, gene expression analysis, etc. For example, a Google search made on the 15th of March 2013 returned 473 mln

R.C. de Amorim
Department of Computing, Glyndŵr University, Wrexham, LL11 2AW, UK.
Tel.: +44 01978 293218
E-mail: r.amorim@glyndwr.ac.uk

B. Mirkin
Department of Data Analysis and Machine Intelligence, National Research University Higher School of Economics, Moscow RF.
Tel.: +7 495 316 4641
E-mail: bmirkin@hse.ru

pages to the query "k-means" and only 198 mln pages to the query "cluster", even in spite of the latter being more general. Thanks to its popularity, K-Means can be found in a number of software packages used in data analysis, including MATLAB, R and SPSS.

K-Means aims to partition a dataset represented by a matrix $Y = (y_{iv})$, where y_{iv} represents the value of feature v, $v = 1,..., V$, on entity $i \in I$, into K homogeneous clusters S_k, together with their centroids $c_k (k = 1, 2, ..., K)$.Given a measure of distance $d(y_i, c_k)$, where y_i denotes i-th row $y_i = (y_{i1}, y_{i2}, ..., y_{iV})$ of Y, K-Means iteratively minimizes the summary distance between the entities y_i and centroids c_k of their clusters

$$W(S,C) = \sum_{k=1}^{K} \sum_{i \in S_k} d(y_i, c_k) \tag{1}$$

The minimization process works according to the alternating optimization scheme. Given the centroids c_k, the optimal clusters S_k are found to minimize criterion (1) over clusters. Given the found clusters S_k, the optimal centroids c_k are found by minimizing criterion (1) over centroids. This is especially simple when the scoring function $d(y_i, c_k)$ is the squared Euclidean distance $d(y_i, c_k) = \sum_v (y_{iv} - c_{kv})^2$. In this case, the optimal centroids are the clusters means, and the optimal clusters consist of entities that are nearest to their centroids, clearly favouring spherical clusters. The iterations stop when the centroids stabilize; the convergence is warranted by the fact that the criterion decreases at every step, whereas the number of possible partitions is finite.

Being much intuitive and simple computationally, K-Means is known to have a number of drawbacks; among them are the following: (i) the method requires the number of clusters to be known beforehand; (ii) the final clustering is highly dependent on the initial centroids it is fed; (iii) the results highly depend on feature scaling.

Building on the work by Makarenkov and Legendre [11], Huang et al. [6,9, 10], Mirkin [12], and Chiang and Mirkin [7], Amorim and Mirkin have introduced what they call the intelligent Minkowski Weighted K-Means (iMWK-Means) algorithm which mitigates the mentioned drawbacks [3]. This approach extends the K-Means criterion by distinguishing in it the feature weighting component, while using the Minkowski metric and initializing the process with anomalous clusters.

The superiority of iMWK-Means in relation to other feature-weight maintaining algorithms was experimentally demonstrated on medium-sized datasets [3,2] including a record minimum number of misclassified entities, 5, on the celebrated Iris dataset [5]. Yet choosing the "right" exponent remains of an issue. This has been addressed by Huang et al. [6,9,10] (weight exponent), Amorim and Mirkin [3] (Minkowski exponent) by exploring the similarity between the found clustering and the "right" partition on a range of possible values of the exponent and choosing the exponent value corresponding to the best match between the clustering and the partition.

In both cases the accuracy of cluster recovery appears to be highly dependent on the exponent value, which may drastically differ at different datasets. Yet in real-world problems the clusters in data are not known beforehand so this approach would not work. There can be two types of real-world scenarios:

1. semi-supervised clustering in which right cluster labels can be supplied for a small part of the data before the process of clustering;
2. unsupervised clustering in which no cluster labels are supplied beforehand at all.

This paper addresses the problem of choosing the Minkowski exponent in both scenarios. We experimentally investigate how our semi-supervised algorithm in scenario (i) works at different proportions of labelled data. We empirically demonstrate that it is possible to recover a good Minkowski exponent with as low as 5% of data being labelled, and that large increases in this proportion of labelled data tend to have a small effect on cluster recovery. In scenario (ii), we look at applying various characteristics of the cluster structure as potential indexes for choosing the right Minkowski exponent. Among them we introduce an index based on the iMWK-Means criterion and, also, indexes related to the so-called silhouette width [13]. It appears our approaches show rather satisfactory results.

The remainder is structured as follows. Section 2 describes the generic iMWK-Means in a greater detail. Section 3 describes adaptations of iMWK-Means to the supervised and unsupervised clustering situations. Section 4 presents our experimental setting, with both real-world benchmark data and synthetic datasets, and the experimental results. Section 5 concludes the paper.

2 Minkowski Weighted K-Means and iMWK-Means

Weighted K-Means (WK-Means) automatically calculates the weight of each feature conforming to the intuitive idea that features with low within-cluster variances are more relevant for the clustering than those with high within-cluster variances. Each weight can be computed both for the entire dataset, w_v, or within-clusters, w_{kv}. We utilize the latter approach involving cluster specific feature weights. The WK-Means criterion by Huang et al. [6] is as follows:

$$W(S,C,w) = \sum_{k=1}^{K} \sum_{i \in S_k} \sum_{v=1}^{V} w_{kv}^{p} |y_{iv} - c_{kv}|^2 \tag{2}$$

where V is the number of features in Y; $w = (w_{kv})$, the set of non-negative within-cluster feature weights such that $\sum_{v=1}^{V} w_{kv} = 1$ for each $k = 1$, 2,..., K; and p, the adjustable weight exponent. This criterion is subjective to a crisp clustering, in which a given entity y_i can only be assigned to a single cluster. The MWK-Means [3] is a further extension of the criterion, in which the squared Euclidean distance is changed for the p-th power of the Minkowski

p-distance. The Minkowski p-distance between a given entity y_i and centroid c_k is defined below:

$$d_p(y_i, c_k) = (\sum_{v=1}^{V} |y_{iv} - c_{kv}|^p)^{1/p} \tag{3}$$

By separating positive measurement scales w_{kv} of features v in the coordinates of y_i and c_k, the p-th power of the Minkowski p-distance, hence without the $1/p$ exponent, can be expressed as:

$$d_{wp}(y_i, c_k) = \sum_{v=1}^{V} w_{kv}^p |y_{iv} - c_{kv}|^p \tag{4}$$

Putting (4) into criterion (2), one arrives at the Minkowski Weighted K-Means criterion:

$$W(S, C, w) = \sum_{k=1}^{K} \sum_{i \in S_k} \sum_{v=1}^{V} w_{kv}^p |y_{iv} - c_{kv}|^p \tag{5}$$

Because of the additivity of the criterion, the weights can be set to be cluster-specific, so that any feature v may have different weights at different clusters k as reflected in 5. Given the partition and centroids; the weights are computed according to equations derived from the first-order optimality condition for criterion 5:

$$w_{kv} = \frac{1}{\sum_{u \in V} [D_{kv}/D_{ku}]^{1/(p-1)}} \tag{6}$$

where $D_{kv} = \sum_{i \in S_k} |y_{iv} - c_{kv}|^p$ and k is an arbitrary cluster [3]. In our experiments we have added a very small constant to the dispersions, avoiding any issue related to a dispersion being equal to zero. Equation (6) at $p = 0$ may seem problematic at first; however, such case is in fact of simple resolution. At $p = 0$, Equation (6) is equivalent to a weight of one for the feature v with the smallest dispersion, and weights of zero in the others, all cluster specific [3,6]. The MWK-Means iterative minimization of criterion (5) is similar to the K-Means algorithm, but each iteration here consists of three stages, rather than the two stages of the generic K-Means, to take the computation of weights into account.

MWK-Means algorithm

1. Initialization. Define a value for the Minkowski exponent p. Select K centroids from the dataset at random. Set $v_{ik} = 1/V$.
2. Cluster update. Assign each entity to its closest centroid applying the Minkowski weighted distance (4).
3. Centroid update. Calculate the cluster centroids as the within-cluster Minkowski centers. Should the centroids remain unchanged, stop the process and output the results.
4. Weight update. Given clusters and centroids, compute the feature weights using Equation (6). Go back to Step 2 for the next iteration.

The original K-Means algorithm makes use of the squared Euclidean distance, which favours spherical clusters. The MWK-Means criterion (5) favours any interpolation between diamond and square shapes, depending on the value of p. This property of MWK-Means makes the selection of p rather important for cluster recovery.

The Minkowski centers can be computed using a steepest descent algorithm from Amorim and Mirkin [3]. More precisely, given a series of reals, y_1, ..., y_n, its Minkowski p-center is defined as c minimizing the summary value

$$d(c) = \sum_{i=1}^{n} |y_i - c|^p \tag{7}$$

The algorithm is based on the property that $d(c)$ in (7) is convex for $p > 1$, and it uses the first derivative of $d(c)$ equal to $'d(c) = p(\sum_{i \in I^+}(c - y_i)^{p-1} - \sum_{i \in I^-}(y_i - c)^{p-1}$, where I^+ is the set of indices i at which $c > y_i$, and I^- is the set of indices i at which $c \leq y_i, i = 1, ..., n$.

Minkowski center algorithm

1. Sort given reals in the ascending order so that $y_1 \leq y_2 \leq ... \leq y_n$.
2. Initialize with $c_0 = y_{i^*}$, the minimizer of $d(c)$ on the set y_i and a positive learning rate λ that can be taken, say, as 10% of the range $y_n - y_1$.
3. Compute $c_0 - \lambda d'(c_0)$ and take it as c_1 if it falls within the minimal interval $(y_{i'}, y_{i''})$ containing y_{i^*} and such that $d(y_{i'}) > d(y_{i^*}), d(y_{i''}) > d(y_{i^*})$. Otherwise, decrease λ a bit, say, by 10%, and repeat the step.
4. Test whether c_1 and c_0 coincide up to a pre-specified precision threshold. If yes, halt the process and output c_1 as the optimal value of c. If not, move on.
5. Test whether $d(c_1) \leq d(c_0)$. If yes, set $c_0 = c_1$ and $d(c_0) = d(c_1)$, and go to step 2. If not, decrease λ a bit, say by 10%, and go to step 3 without changing c_0.

Similarly to K-Means, the MWK-Means results highly depend on the choice of the initial centroids. This problem has been addressed for K-Means by using initialization algorithms that provide K-Means with a set of good centroids. Taking this into account, we have adapted the Anomalous Pattern algorithm from Mirkin [12] to supply MWK-Means with initial centroids. A combination of K-Means with the preceding Anomalous Pattern algorithm is referred to as the intelligent K-Means, iK-Means, in Mirkin [12]. The iK-Means has proved superior in cluster recovery over several popular criteria in experiments reported by Chiang and Mirkin [7]. The modified Anomalous Pattern algorithm involves the weighted Minkowski metric (4) with the weights computed according to Equation (6). Together with the Anomalous Pattern initialization, MWK-Means forms what is referred to as the intelligent Minkowski Weighted K-Means algorithm (iMWK-Means), ceasing to be a non-deterministic algorithm. Its formulation is as follows.

iMWK-Means algorithm

1. Sort all the entities according to their Minkowski weighted distance (4) to the Minkowski center of the dataset, c_c, using the Minkowski weighted metric and $1/V$ for each weight.
2. Select the farthest entity from the Minkowski centre, c_t, as a tentative anomalous centroid.
3. Assign each of the entities to its nearest centroid of the pair, c_c and c_t, according to the Minkowski weighted metric.
4. Compute c_t as the Minkowski center of its cluster.
5. Update the weights according to equation (6). If c_t has moved on step 4, return to step 3 for the next iteration.
6. Set all the entities assigned to c_t as an anomalous cluster and remove it from the dataset. If there are still unclustered entities remaining in the dataset, return to step 2 to find the next anomalous cluster.
7. Run MWK-Means using the centroids of the K anomalous clusters with the largest cardinality.

3 Selection of the Minkowski Exponent in the semi- and unsupervised settings

In this section we present methods for selecting the Minkowski exponent in each of the two settings, semi-supervised and unsupervised. The first utilizes a small portion of data that have been labelled; the second, a cluster scoring function over partitions.

3.1 Choosing the Minkowski exponent in the semi-supervised setting

The semi-supervised setting relates to the scenario in which the cluster labels are known not for all, but only for a relatively small proportion q of the dataset being clustered. Then either of two options can be taken: (a) first, cluster only those labelled entities to learn the best value for the Minkowski exponent p as that leading to the partition best matching the pre-specified labels, then using the learnt p, cluster the entire dataset, or (b) to cluster all the entities and learn the best p at the labelled part of the dataset. In Amorim and Mirkin [3] the option (a) has been disapproved rather convincingly. Therefore, only option (b) is tested in this paper at differing values of q.

The algorithm comprising both learning and testing p runs at a dataset, at which all the pre-specified clustering labels are known, as follows:

Run R=50 times:

1. Get a random sample of labelled entities of size q: cluster labels on this set are assumed to be known.
2. Run iMWK-Means over the whole dataset with p taken in the interval from 1 to 5 in steps of 0.1.
3. Select the Minkowski exponent with the highest accuracy achieved on the entities whose labels are known as p^*.

4. Calculate the accuracy using p^* and the whole dataset by comparing the found partition and that one pre-specified.

Calculate the average accuracy of the R runs and standard deviation.

Our search for a good p occurs in the interval [1, 5]. We have chosen the lower bound of one because this is the minimum for which we can calculate a Minkowski center (median), since Equation (7) is convex for $p > 1$. We have chosen the upper bound of five following our previous experiments [3].

Of course, in a real-life clustering scenario one would not normally have access to the labels of the whole dataset. Here we utilize it only for evaluation purposes.

3.2 Choosing the Minkowski exponent in an unsupervised setting

When no prior information of the hidden partition is available, a reasonable idea would be to find such a scoring function over the iMWK-Means partitions, that reaches its extreme value, that is, the maximum or minimum, at a resulting partition that is most similar to the hidden partition.

Under the original K-Means framework, one of the most popular scoring functions is the output of the K-Means criterion itself. One simply runs K-Means a number of times and sets the optimal partition to be that with the smallest sum of distances between the entities and their respective centroids. Unfortunately the iMWK-Means criterion (5) is not comparable at different ps, making its raw value inappropriate for a scoring function. However, we can normalize (5) in such a way that its dependence on p can be disregarded, we called it the Minkowski clustering index (MCI). For comparison, we also experiment with the so-called silhouette width [13] which has been reported as a good index in various empirical studies, one of the latest being by Arbelaitz et al. [4].

Let us define MCI, an index based on the minimized value of the MWK-Means criterion. We normalize the criterion over what may be called the Minkowski data p-scatter according to the feature weights found as an output of a run of the iMWK-Means:

$$MCI = \frac{W_p(S, C, w)}{\sum_{k=1}^{K} \sum_{i \in S_k} \sum_{v=1}^{V} |w_{kv} y_{iv}|^p} \tag{8}$$

This index is comparable at clusterings with different ps. We choose that p at which the MCI is at its minimum.

Let us turn now to the silhouette width based indexes. Given a set of clusters and an entity-to-entity dissimilarity measure, the silhouette width of an entity is defined as the relative difference between the average dissimilarity of that entity from the other entities in its cluster compared to its average dissimilarities to other clusters according to formula:

$$S(y_i) = \frac{b(y_i) - a(y_i)}{max\{a(y_i), b(y_i)\}} \tag{9}$$

where $a(y_i)$ is the average dissimilarity of $y_i \in S_k$ from all other entities in its cluster S_k, and $b(y_i)$ the lowest average dissimilarity of the y_i from another cluster S_l at $l \neq k$. The larger the $S(y_i)$, the better the entity y_i sits in its cluster. The silhouette width of the partition is the sum of all $S(y_i)$ over all $i \in I$.

We apply the concept of silhouette width over five different measures of dissimilarity between vectors $x = (x_v)$ and $y = (y_v)$:

1. Squared Euclidean distance $d(x, y) = \sum_v (x_v - y_v)^2$;
2. Cosine $1 - c(x, y)$ where $c(x, y) = \sum_v x_v y_v / \sqrt{\sum_v x_v^2}\sqrt{\sum_v y_v^2}$;
3. Correlation $1-r(x, y)$ where $r(x, y) = \sum_v (x_v-\bar{x})(y_v-\bar{y}) / \sqrt{\sum_v (x_v - \bar{x})^2}\sqrt{\sum_v (y_v - \bar{y})^2}$;
4. Power p of Minkowski p-distance $d_p^p(x, y) = \sum_v |x_v - y_v|^p$ where p is the same as in the tested run of iMWK-Means; and
5. A p-related analogue to the cosine dissimilarity defined as

$$c_p(x, y) = \sum_v |\frac{x_v}{\sqrt[p]{\sum_{v=1}^V |x_v|^p}} \frac{y_v}{\sqrt[p]{\sum_{v=1}^V |y_v|^p}}|^p \qquad (10)$$

This definition is based on an analogue to the well-known equation relating the squared Euclidean distance and cosine, $d(x', y') = 2 - 2c(x', y')$, where $x' = \frac{x}{||x||}$, $y' = \frac{y}{||y||}$ are normed versions of the vectors. We select the p with the highest sum of silhouette widths.

4 Experiments

To validate the Minkowski exponent selection methods we experiment with, we use both real-world and synthetic datasets. The six real-world datasets taken from the UCI Irvine repository [5] are those that have been used by Huang et al. [6,9] as well as by Amorim and Mirkin [3]. Also, versions of these datasets obtained by adding uniformly random features are used to see the impact of the noise on the recovery of the Minkowski exponent.

The datasets are:

1. Iris. This dataset contains 150 flower specimens over four numerical features; it is partitioned in three groups. We devised two more Iris dataset versions by adding, respectively, extra two and extra four noise features. These are uniformly random.
2. Wine. This dataset contains 178 wine specimens partitioned in three groups and characterized by 13 numerical features that are chemical analysis results. We also use two more datasets by adding 7 and 13 noise features, respectively.
3. Hepatitis. This dataset contains 155 cases over 19 features, some of them categorical, partitioned in two groups. Two more versions of this dataset have been obtained by adding, in respect, 10 and 20 noise features.
4. Pima Indians Diabetes. This dataset contains 768 cases over 8 numerical features, partitioned in two groups. Two more versions of this dataset have been obtained by adding, in respect, 4 and 8 noise features.

5. Australian Credit Card Approval. This dataset contains 690 cases partitioned in two groups, originally with 15 features, some of them categorical. After a pre-processing step, described later in this section, we had a total of 42 numerical features.
6. Heart Disease. This dataset contains 270 cases partitioned in two groups referring to the presence or absence of a heart disease. This dataset has originally 14 features, including categorical ones. After the pre-processing step, there are 32 features in total.

Our synthetic data sets are of three formats:(F1) 1000x8-5: 1000 entities over 8 features consisting of 5 Gaussian clusters; (F2) 1000x15-7: 1000 entities over 15 features consisting of 7 Gaussian clusters; (F3) 1000x60-7: 1000 entities over 60 features consisting of 7 Gaussian clusters.

All the generated Gaussian clusters are spherical so that the covariance matrices are diagonal with the same diagonal value σ^2 generated at each cluster randomly between 0.5 and 1.5, and all centroid components independently generated from the Gaussian distribution with zero mean and unity variance. Cluster cardinalities are generated uniformly random, with a constraint that each generated cluster has to have at least 20 entities.

We standardize all datasets by subtracting the feature average from all its values, and dividing the result by half the feature's range. The standardization of categorical features follows a process described by Mirkin [12] to allow us to remain within the original K-Means framework. In this, each category is represented by a new binary feature, by assigning 1 to each entity which falls in the category and zero, otherwise. We then standardize these binary features by subtracting their grand mean, that is, the category's frequency. By adopting this method the centroids are represented by the proportions and conditional proportions rather than modal values.

Since all class pre-specified labels are known to us, we are able to map the clusters generated by iMWK-Means using a confusion matrix. We calculate the accuracy as the proportion of entities correctly clustered by each algorithm.

4.1 Results for the semi-supervised settings

Table 1 presents the results of our experiments for the semi-supervised setting at the real-world data. Different proportions of the labelled data are assumed to be known: $q = 5$, 10, 15, 20 and 25 percent. For the purposes of comparison, the table also contains the results of a fully supervised experiment at which $q = 100\%$ - the maximum accuracy. We have obtained these by running iMWK-Means for every p from the range of 1 to 5 in steps of 0.1 and checking which one had the highest accuracy using all the class labels at each dataset. The results using the semi-supervised selection of p show that at $q = 5\%$ and $q = 10\%$ an increase of 5% in the size of the learning data amounts to an increase of about 1% in the accuracy, with the accuracy reaching, at $q = 15\%$, to about 1-2% within the maximum. Further increases of q to 20% and 25% bring almost no increase in the accuracy, except for the case of the noisy Hepatitis data.

The results of our experiments using the semi-supervised algorithm on the synthetic datasets, shown in Table 2, present the same pattern as in Table 1. An increase of 5% in the learning data produces an increase of around 1% in the final accuracy of the algorithm till $q = 15\%$. In these, even with only 5% of the data having been labelled, the algorithm still can reach accuracy of within 1-2% of the maximum possible. Moreover, the exponent values stabilize from the very beginning at $q = 5\%$. This is an improvement over the real-world datasets, probably, because of a more regular structure of the synthetic datasets.

4.2 Results at the unsupervised setting

In this set of experiments there is no learning stage. Because of this we found it reasonable to take p in the interval from 1.1 to 5 rather than 1 to 5, still in steps of 0.1. At $p = 1$ iMWK-Means selects a single feature from the dataset [3] putting a weight of zero in all others. In our view, there are only a few scenarios in which the optimal p would be 1 and these would be very hard to find without learning data.

Table 3 presents the results of our experiments with the silhouette width for five dissimilarity measures; three Euclidean: squared distance, cosine, correlation, and two Minkowski's: distance and cosine. When using the latter two, the exponent p is the same as the one used in the iMWK-Means clustering.

For the sake of comparison, the Table 3 also presents the maximum accuracy for each dataset. The table shows that it is indeed possible to select a good p without having labels for a given dataset. For example, silhouette width based on Minkowski distance works well on Iris and Heart, based on Euclidean squared distance, on Wine, Pima, and Heart, and MCI works well on Hepatitis. Overall, the best performance has shown the Minkowski clustering index MCI as it has the best worst case scenario among all the indexes under consideration. Its highest difference between its accuracy and the maximum possible is equal to -12.24% (on Pima), whereas the other indexes lead to the highest difference between 24% to 38.71%. Yet none of the indexes is reliable enough to be used with no reservations. Taking into account the fact that different indexes lead to different solutions, one may suggest using a consensus rule.

Specifically, let us take the MCI index and two silhouette width indexes, that based on the Euclidean squared distance (SWE) and that based on Minkowski distance (SWM), and, when they are in disagreement, use the value of p that is the average of those two that are in agreement (see Table 5).When iMWK-Means is unable to find the required number of clusters using the p agreed between two indexes, as with the Hepatitis + 20 at $p = 1.3$, we use the p from the remaining index in our experiments The results of experiments in the unsupervised setting at the synthetic data sets are presented in Table 5. At the synthetic datasets we can observe a different pattern. The Minkowski clustering index is no longer the most promising algorithm to select p. In these the

Table 1 Results of semi- and fully- supervised experiments with the real-world datasets and their noisy versions. The accuracy and exponent shown are the averages and, after slash, standard deviations, over R=50 runs.

	Semi-supervised at different q										Supervised	
	5%		10%		15%		20%		25%		100%	
	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p
Iris	93.09/6.07	1.14/0.53	94.60/1.96	1.07/0.09	95.15/1.94	1.23/0.62	95.63/1.69	1.10/0.08	95.60/1.75	1.16/0.44	96.67	1.1
Iris+2	93.55/2.19	1.150.52	94.28/1.94	1.05/0.07	94.95/1.97	1.13/0.28	95.12/1.93	1.14/0.28	95.81/1.53	1.10/0.06	96.67	1.1
Iris+4	93.39/3.89	1.20/0.73	94.53/1.59	1.07/0.06	94.76/1.46	1.10/0.09	94.36/1.79	1.14/0.37	95.19/1.21	1.13/0.11	96.00	1.1
Wine	87.64/6.30	1.48/0.84	90.87/3.85	1.60/0.87	91.88/2.28	1.88/1.06	92.53/1.04	1.98/1.04	92.45/1.09	2.13/1.19	93.82	1.6
Wine+7	89.22/4.51	1.13/0.22	91.43/1.76	1.32/0.45	91.81/1.77	1.44/0.50	92.33/1.80	1.61/0.56	92.85/1.46	1.55/0.48	94.38	2.2
Wine+13	90.18/7.40	1.21/0.53	92.74/2.09	1.09/0.08	93.24/1.73	1.13/0.10	93.44/1.61	1.13/0.10	93.63/1.44	1.13/0.09	94.38	1.1
Hepatitis	65.12/8.78	1.69/0.75	71.63/5.75	2.26/0.43	72.65/3.56	2.44/0.73	73.37/2.70	2.27/0.39	73.32/2.17	2.61/0.91	74.84	2.1
Hepatitis+10	69.16/9.13	2.31/1.52	75.33/7.78	2.86/1.51	76.59/7.74	3.46/1.24	78.82/5.28	3.76/1.07	80.10/4.09	3.90/1.04	82.58	4.3
Hepatitis+20	74.85/9.14	2.2/1.24	80.94/3.67	2.81/1.13	81.44/4.29	2.80/1.08	82.70/3.65	2.85/0.93	83.25/3.17	2.99/0.80	85.81	3.1
Pima	64.09/4.60	3.2/1.29	66.25/3.07	3.74/1.11	67.67/2.33	4.28/0.89	67.97/1.99	4.38/0.80	68.09/1.73	4.48/0.57	69.14	4.9
Pima+4	65.86/1.44	2.51/1.14	66.42/1.43	2.34/0.94	66.47/1.13	2.61/1.14	66.51/1.09	2.64/1.07	66.70/0.97	2.45/0.92	67.71	1.8
Pima+8	66.51/2.90	2.28/0.95	67.05/2.62	1.96/0.66	68.06/1.39	2.11/0.72	68.84/1.16	1.92/0.63	68.74/1.14	1.93/0.41	69.66	1.8
Austral CC	83.81/3.36	1.66/0.62	84.71/1.20	1.60/0.50	84.76/1.14	1.59/0.50	85.19/0.67	1.38/0.39	85.07/1.06	1.41/0.42	85.51	1.2
Heart	80.00/5.09	2.27/0.72	82.11/2.94	2.45/0.51	82.46/2.58	2.52/0.43	82.29/2.71	2.52/0.44	82.61/2.43	2.51/0.43	83.70	2.7

Table 2 Semi-supervised setting experiments with synthetic datasets. The accuracy and exponent shown are the averages, accompanied by the standard deviations, over 50 runs for each of 10 Gaussian Model generated datasets.

	Semi-supervised at different q										MIC	
	5%		10%		15%		20%		25%		100%	
	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p
1000x8-5	80.45/5.34	3.79/0.63	81.76/4.45	3.81/0.59	82.40/3.91	3.81/0.64	82.57/3.80	3.86/0.65	82.75/3.67	3.82/0.63	82.94	3.83
1000x15-7	92.60/7.11	2.37/0.66	94.21/5.41	2.44/5.93	94.56/4.98	2.40/0.49	94.61/5.07	2.42/0.48	94.69/4.96	2.44/0.48	94.88	2.45
1000x60-7	99.19/3.09	1.56/0.41	99.89/0.89	1.51/0.33	99.87/1.43	1.48/0.22	100.0/0.00	1.47/0.19	100.0/0.00	1.47/0.19	100.0	1.48

Table 3 The values of Minkowski exponent p and the accuracy obtained at the maximum of the silhouette width and minimum of MCI at the unsupervised experiments with the real-world datasets and their noisy versions; the maximum achievable accuracies are in the column on the left.

	Max		Silhouette width										**MIC**	
			Sq. Euclid.		Cos		Corr.		**Mink**		Cos$_p$			
	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p
Iris	96.67	1.1	93.33	**3.7**	90.67	5.0	94.00	3.4	96.00	**1.3**	90.67	2.3	96.67	**1.1**
Iris+2	96.67	1.1	84.00	**4.4**	87.33	3.7	90.00	1.8	96.67	**1.1**	90.00	1.8	96.67	**1.1**
Iris+4	96.00	1.1	72.00	**4.7**	72.00	4.7	72.00	4.7	96.00	**1.1**	93.33	1.9	95.33	**1.4**
Wine	93.82	1.6	93.82	**1.6**	93.82	1.6	92.13	2.2	92.70	**1.4**	92.13	2.2	90.45	**1.2**
Wine+7	94.38	2.2	93.26	**2.0**	91.57	1.9	90.45	2.5	92.70	**1.3**	92.13	2.3	89.89	**1.2**
Wine+13	94.38	1.1	93.82	**1.3**	93.26	1.6	89.89	2.2	94.38	**1.1**	92.13	1.9	93.82	**1.3**
Hepatitis	74.84	2.1	70.32	**2.2**	70.97	4.8	70.97	4.8	47.10	**1.1**	47.10	2.9	74.19	**2.4**
Hepatitis+10	82.58	4.3	81.94	**5.0**	63.23	3.6	63.23	3.6	76.13	**1.4**	62.58	1.8	52.9	**1.9**
Hepatitis+20	85.81	3.1	80.65	**5.0**	75.48	4.2	75.48	4.2	74.84	**1.1**	47.10	2.7	79.35	**1.5**
Pima	69.14	4.9	67.58	**4.3**	65.49	2.8	60.81	3.6	67.58	**4.3**	65.76	2.3	57.55	**1.4**
Pima+4	67.71	1.8	64.06	**4.5**	66.28	2.8	64.45	2.0	66.02	**1.9**	66.02	1.9	60.94	**1.4**
Pima+8	69.66	1.8	63.93	**4.8**	65.76	1.9	65.76	1.9	65.76	**1.9**	65.10	4.5	68.49	**1.5**
Aust CC	85.51	1.2	85.51	**1.2**	85.55	1.2	85.55	1.2	85.51	**1.2**	73.33	3.8	78.84	**2.4**
Heart	83.70	2.7	83.33	**2.6**	83.33	2.6	83.33	2.6	75.19	**1.1**	83.33	2.6	75.19	**1.9**

cosine and correlation are those recovering the adequate ps and, thus, having considerably better results.

5 Conclusion

The use of weights in K-Means clustering has shown good results [11,6,9,10, 8], in particular when utilizing the Minkowski distance metric [3,2]. Its version oriented at determining the number of clusters and the initial centroids, the intelligent Minkowski Weighted K-Means showed considerably better accuracy results, at an appropriate Minkowski exponent p, than a number of other algorithms [3]. However, finding an appropriate p remained an open issue. This paper presents a study regarding the amount of labelled data necessary for a good recovery of p under a semi-supervised approach, as well as an unsupervised method based on indexes of correspondence between the found partition and the dataset structure.

We have found that in most datasets it is possible to recover a good p with as low as 5% of the data being labelled, and that reasonable results can be obtained by using individual indexes over the clustering, the Minkowski clustering index or silhouette width indexes, or a combined "consensus" rule. It is quite likely that these findings can be relevant for the Minkowski partition around medoids algorithm [1].

However, the iMWK-Means algorithm may have difficulties finding appropriate weights for datasets containing informative but redundant features. In this case the algorithm sets the weights of all such features to high values instead of removing some features by setting some weights to zero. This is an issue that we intend to address in future research.

References

1. de Amorim, R.C., Fenner, T.: Weighting features for partition around medoids using the minkowski metric. Lecture Notes in Computer Science **7619**, 35–44 (2012)
2. de Amorim, R.C., Komisarczuk, P.: On initializations for the minkowski weighted k-means. Lecture Notes in Computer Science **7619**, 45–55 (2012)
3. de Amorim, R.C., Mirkin, B.: Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. Pattern Recognition **45**(3), 1061–1075 (2012)
4. Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J.M., Perona, I.: An extensive comparative study of cluster validity indices. Pattern Recognition **46**, 243–256 (2012)
5. Bache, K., Lichman, M.: UCI machine learning repository (2013). URL http://archive.ics.uci.edu/ml
6. Chan, E.Y., Ching, W.K., Ng, M.K., Huang, J.Z.: An optimization algorithm for clustering using weighted dissimilarity measures. Pattern recognition **37**(5), 943–952 (2004)
7. Chiang, M.M.T., Mirkin, B.: Intelligent choice of the number of clusters in k-means clustering: an experimental study with different cluster spreads. Journal of classification **27**(1), 3–40 (2010)
8. Frigui, H., Nasraoui, O.: Unsupervised learning of prototypes and attribute weights. Pattern recognition **37**(3), 567–581 (2004)
9. Huang, J.Z., Ng, M.K., Rong, H., Li, Z.: Automated variable weighting in k-means type clustering. Pattern Analysis and Machine Intelligence, IEEE Transactions on **27**(5), 657–668 (2005)

10. Huang, J.Z., Xu, J., Ng, M., Ye, Y.: Weighting method for feature selection in k-means. Computational Methods of Feature Selection, Chapman & Hall/CRC pp. 193–209 (2008)
11. Makarenkov, V., Legendre, P.: Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software. Journal of Classification **18**(2), 245–271 (2001)
12. Mirkin, B.: Clustering for data mining: a data recovery approach, vol. 3. Chapman and Hall/CRC (2005)
13. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics **20**, 53–65 (1987)

Table 4 The values of Minkowski exponent p and the accuracy obtained at different rules defined above.

	Max		Sq. Euclid.		Mink		MCI		Consensus	
	Acc	p	Acc	p	Acc	p	Acc	p	p	Acc
Iris	96.67	1.1	93.33	3.7	96.00	1.3	96.67	1.1	1.20	96.00
Iris+2	96.67	1.1	84.00	4.4	96.67	1.1	96.67	1.1	1.10	96.67
Iris+4	96.00	1.1	72.00	4.7	96.00	1.1	95.33	1.4	1.25	94.67
Wine	93.82	1.6	93.82	1.6	92.70	1.4	90.45	1.2	1.30	92.13
Wine+7	94.38	2.2	93.26	2.0	92.70	1.3	89.89	1.2	1.25	89.33
Wine+13	94.38	1.1	93.82	1.3	94.38	1.1	93.82	1.3	1.30	93.83
Hepatitis	74.84	2.1	70.32	2.2	47.10	1.1	74.19	2.4	2.30	60.00
Hepatitis+10	82.58	4.3	81.94	5.0	76.13	1.4	52.90	1.9	1.65	70.97
Hepatitis+20	85.81	3.1	80.65	5.0	74.84	1.1	79.35	1.5	5.00	80.65
Pima	69.14	4.9	67.58	4.3	67.58	4.3	57.55	1.4	4.30	67.58
Pima+4	67.71	1.8	64.06	4.5	66.02	1.9	60.94	1.4	1.65	66.41
Pima+8	69.66	1.8	63.93	4.8	65.76	1.9	68.49	1.5	1.70	69.53
Aust CC	85.51	1.2	85.51	1.2	85.51	1.2	78.84	2.4	1.20	85.51
Heart	83.70	2.7	83.33	2.6	75.19	1.1	75.19	1.9	1.50	75.19

Table 5 Results of searching for the best p at the unsupervised setting. The accuracy and exponent values are the averages and standard deviations over each of the 10 GMs.

	Max		Silhouette width										MIC	
			Sq. Euclid.		Cos		Corr.		Mink		Cos$_p$			
	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p	Acc	p
1000x8-5	82.94	3.83	81.79/4.20	**3.80/0.73**	81.81/4.18	3.81/0.73	81.65/4.16	3.58/0.60	81.80/4.19	**3.82/0.73**	69.59/13.44	2.47/0.67	71.70/10.41	**5.0/0.00**
1000x15-7	94.88	2.45	92.28/8.77	**2.63/0.60**	93.47/7.42	2.46/0.37	93.47/7.42	2.46/0.37	92.00/8.91	**2.91/0.86**	93.22/8.38	2.16/0.15	76.49/7.98	**4.99/0.03**
1000x60-7	100.0	1.48	99.83/0.54	**1.80/1.10**	100.0/0.00	1.48/0.20	100.0/0.00	1.48/0.20	99.84/0.51	**2.08/0.37**	98.58/2.93	2.65/0.72	90.50/8.30	**5.0/0.00**