This document is the Accepted Manuscript version of the following paper: Cordeiro de Amorim, R., and Mirkin, B., 'A clustering based approach to reduce feature redundancy', in Proceedings, Andrzej M. J. Skulimowski and Janusz Kacprzyk, eds., Knowledge, Information and Creativity Support Systems: Recent Trends, Advances and Solutions, Selected papers from KICSS'2013 - 8th International Conference on Knowledge, Information, and Creativity Support Systems, Kraków, Poland, 7-9 November 2013. ISBN: 978-3-319-19089-1, e-ISBN: 978-3-319-19090-7.

Available online at doi: 10.1007/978-3-319-19090-7.

A Clustering Based Approach to Reduce Feature Redundancy

Renato Cordeiro de Amorim[1] and Boris Mirkin[2]

[1] School of Computer Science, University of Hertfordshire, College Lane Campus, Hatfield AL10 9AB, UK.
[2] Department of Computer Science and Information Systems, Birkbeck University of London, Malet Street, London WC1E 7HX
r.amorim@herts.ac.uk, mirkin@dcs.bbk.ac.uk

Abstract. Research effort has recently focused on designing feature weighting clustering algorithms. These algorithms automatically calculate the weight of each feature, representing their degree of relevance, in a data set. However, since most of these evaluate one feature at a time they may have difficulties to cluster data sets containing features with similar information. If a group of features contain the same relevant information, these clustering algorithms set high weights to each feature in this group, instead of removing some because of their redundant nature. This paper introduces an unsupervised feature selection method that can be used in the data pre-processing step to reduce the number of redundant features in a data set. This method clusters similar features together and then selects a subset of representative features for each cluster. This selection is based on the maximum information compression index between each feature and its respective cluster centroid. We present an empirical validation for our method by comparing it with a popular unsupervised feature selection on three EEG data sets. We find that our method selects features that produce better cluster recovery, without the need for an extra user-defined parameter.

Keywords: Unsupervised feature selection, feature weighting, redundant features, clustering, mental task separation.

1 Introduction

Reducing the cardinality of a data set, with out losing information, has been a constant object of research effort for a considerable amount of time [19], [29]. The general aim of feature selection is to reduce such cardinality by removing any feature in a data set that are not of interest. For instance, those that are redundant containing the same information, those which are composed of noise, or simply those that have no relevant to the particular task being performed. There are indeed a various benefits that may originate from the use of a feature selection algorithm, for instance (i) the amount an algorithm, being this in classification or clustering, takes to process a data set tends to decrease as the cardinality of a data set decreases; (ii) it may reduce the probability of a model

describing an error or noise present in a given data set, instead of the underlying data structure, a problem commonly known as overfitting; (iii) it may provide a general improvement to an algorithm used afterwards, being this in terms of accuracy or cluster recovery.

Feature selection either selects or deselects a particular feature $v \in V$ of a data set Y. It does so by providing each feature $v \in V$ with a weight of either one, or zero. The generalization of this method, in which each feature $v \in V$, receives a weight in the interval $[0, 1]$ is known as feature weighting. Clearly, the weight w_v of a particular feature v aims to reflect the degree of relevance of v to the particular task at hand. The use of feature weighting does not conflict with feature selection as the former still allows a feature v to be deselected by setting $w_v = 0$. The major difference between feature selection and feature weighting is that the latter recognizes that even among relevant features that may be different degrees of relevance.

Feature weighting algorithms have been applied to hierarchical, as well as partitional clustering [4], [8], [10], [14], [15], [16], [34]. Hierarchical clustering algorithms aim to produce a set of clusters which have a tree-like relationship, demonstrable through a dendogram, between them. These algorithms allow a given entity $y_i \in Y$ to be assigned to more than one cluster, as long as these clusters are related and the assignment happens at different levels of the tree. Partitional clustering algorithms take a different approach, originally they produce disjoint clusters, allowing a given entity $y_i \in Y$ to be assigned to a single cluster. Exceptions to this rule were introduced with Fuzzy C-Means [2] which allows a given entity to belong to all clusters with a degree of membership $u_{ik} \in [0, 1]$.

Here we expand our previous work [11] with a particular interested in improving Weighted K-Means (WK-Means) [4], and its L_p-based generalization, the intelligent Minkowski Weighted K-Means (iMWK-Means) [10] in terms of cluster recovery. Both algorithms apply cluster based feature weight, allowing a feature v to have different weights at different clusters $k = \{1, 2, ..., K\}$, where K is the total number of clusters. This weight w_{kv} is calculated following the intuitive assumption that if a feature v has a high relative dispersion in a particular cluster S_k then its degree of relevance, and by consequence its weight w_{kv}, should be low. More details related to both algorithms can be found in Section 2.

The WK-Means the iMWK-Means have been successfully applied in various scenarios [4], [9], [10], [20], [21]. However, these algorithms do introduce a new drawback. They both set w_{kv} by evaluating one feature $v \in V$ in each cluster $k = \{1, 2, ..., K\}$ at a time. Therefore, should a subset of features in V contain the same relevant information, none will be excluded by receiving a weight of zero. Quite the contrary, since they will all have similar small dispersions, each of their weights will be equally high.

In this paper we introduce a clustering-based method for feature selection that aims to reduce the number of redundant features in V. Our method, the intelligent K-Means for Feature Selection (iKFS), can be used to address the

problem described above. The iKFS algorithm is used to cluster $v \in V$, rather than $y_i \in Y$. The number of clusters in V is found by using an anomalous pattern approach based on intelligent K-Means [28], as well as the maximum compression index (MIC) [29].

We evaluate the use of iKFS as a pre-processing step for both WK-Means and the iMWK-Means by clustering three data sets containing Electroencephalography (EEG) signals. These are high-dimensional data sets with 5,680 features each, with patterns that are difficult to discern. For comparison we run similar experiments using the features selected using the popular feature selection using feature similarity (FSFS) [29]. We find that iKFS tends to select a smaller amount of features that are in fact more relevant than those selected by FSFS, with the added benefit that iKFS does not require an extra user-defined parameter.

2 Background

Redundant features are those that contain similar information, by consequence when clustering a data set most such features are unnecessary. In order to reduce the number of redundant features in a data set our proposed method, iKFS, first aims to find clusters containing similar features. Clustering is the non-trivial task of creating K groups of entities so that those within the same group are similar and those between groups are dissimilar. Clustering algorithms have been used to solve problems in various fields of research, such as data mining, computer vision, bioinformatics, text mining, etc [22], [28], [31], [36].

K-Means [1], [23] is among the most popular clustering algorithms. It performs partitional clustering, dividing a data set Y into K disjoint clusters $S = \{S_1, S_2, ..., S_K\}$. K-Means represents a given cluster S_k by its centre of gravity, the centroid c_k. Each feature $v \in V$ of a centroid, represented by c_{kv}, is equivalent to the average of v over each entity in S_k, $c_{kv} = \frac{\sum_{y_i \in S_k} y_{iv}}{|S_k|}$, assuming Euclidean distance. K-Means iteratively minimizes the the sum of the distances between each entity $y_i \in Y$ and its respective centroid.

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

where $C = \{c_1, c_2, ..., c_K\}$. There are a number of reasons for the popularity of K-Means, among these: it is a easy to implement, relatively fast algorithm, which is also intuitively easy to understand. In fact, the minimization of the K-Means criterion (1) has only the three steps below.

1. Assign the values of K random entities from Y to the initial centroids $c_1, c_2, ..., c_K$;
2. Assign each entity $y_i \in Y$ to the cluster represented by its closest centroid;
3. Update each centroid to the centre of gravity of its cluster, and go back to Step 2. Iterations cease when the algorithm converges.

The complexity of K-Means is of $\mathcal{O}(nKt)$, where t is the number of iterations K-Means takes to converge, and n the number of entities in Y. Although it is difficult to determine the value of t beforehand, we have shown that this tends to be small, particularly when K-Means is initialized with relevant, rather than random, centroids [7]. Other clustering algorithms can be much slower, for instance hierarchical algorithms have a complexity of at least $\mathcal{O}(n^2)$. Implementations of K-Means can be frequently found in software packages, such as MATLAB, R, SPSS, etc.

Although popular, K-Means does have drawbacks. Some of which have been target of research effort for a long time. For instance, K-Means requires K (the number of clusters in Y) to be known beforehand, and the clustering produced by K-Means can be heavily affected by the initial centroids used in its first step [3], [5], [24], [30], [33], [35].

Among the many algorithms addressing these two interrelated issues, intelligent K-Means (iK-Means) seems quite successful [5], [7], [28]. This algorithm finds the clusters in a data set by extracting one anomalous pattern at a time, as per below.

1. Set c_c, the centre of the data set Y.
2. Set a tentative centroid c_t, the entity $y_i \in Y$ that is the farthest from c_c.
3. Run K-Means on Y, using c_c and c_t as initial centroids. Do not allow c_c to move during the clustering.
4. If the $|S_{c_t}| \geq \theta$, add c_t to C_{init}, otherwise discard c_t. In any case, remove the entities in S_{c_t} from Y.
5. If there are entities in Y, go to Step 2.
6. Run K-Means, initialized with the centroids in C_{init} and $K = |C_{init}|$.

Recently, research effort has focused on the fact that K-Means treats all features $vinV$ equally, instead of taking into account that different features may have different degrees of relevance [17], [25]. Weighted K-Means (WK-Means) and the intelligent Minkowski Weighted K-Means (iMWK-Means), are of particular interest to us, because of their excellent ability to recover clusters [4], [9], [10], [20], [21].

The main difference between WK-Means and iMWK-Means is the distance measure in use. While the former applies the squared Euclidean distance, the latter makes a generalization of this by using the p^{th} root of the Minkowski (L_p) distance. Both algorithms add a cluster dependent weight, w_{kv} to the distance in use. To avoid linearity, this weight is put to the power of an user-defined exponent. The distance measure between an entity $y_i \in Y$ and a centroid $c_k \in C$ in WK-Means is then $d(y_i, c_k) = \sum_{v \in V} w_{kv}^{\beta} |y_{iv} - c_{kv}|^2$. In iMWK-Means, the distance exponent is the same as the weight exponent, making it possible to interpret w_{kv} as a feature re-scaling factor, for any exponent. The distance used by iMWK-Means follows.

$$d_p(y_i, c_k) = \sum_{v \in V} w_{kv}^{p} |y_{iv} - c_{kv}|^p. \tag{2}$$

Substituting the distance in the K-Means criterion (1) by the adjusted weighted distance (2) we obtain the iMWK-Means criterion.

$$W_p(S, C, w) = \sum_{k=1}^{K} \sum_{y_i \in S_K} \sum_{v \in V} w_{kv}^p |y_{iv} - c_{kv}|^p. \tag{3}$$

The exponent p is a user-defined parameter that affects equally the distance and the weights in 3. The calculation of w_{kv} for $k = 1, 2, ..., K$ and each $v \in V$ follows the intuitive idea that if a feature v has a smaller relative dispersion in cluster S_k than a different feature $u \in V$, then v should have a higher weight in S_k than u. We formalize this with the equation below.

$$w_{kv} = \frac{1}{\sum_{u \in V} [D_{kvp}/D_{kup}]^{1/(p-1)}}, \tag{4}$$

where the dispersion of v at cluster S_k and specific p is given by $D_{kvp} = \sum_{y_i \in S_k} |y_{iv} - c_{kv}|^p$. For a given cluster S_k, the weights are subject to $\sum_{v \in V} w_{kv} = 1$, and a crisp clustering, in which an entity $y_i \in Y$ can only be assigned to a single cluster S_k. The equations for WK-Means are similar to all the above, but with a distance exponent always equal to two. We formalise the iMWK-Means algorithm below.

1. Obtain the initial centroids $C = \{c_1, c_2, ..., c_K\}$ by applying the iK-Means algorithm, using the distance in Equation 2. Set each cluster $S_k \leftarrow \emptyset$.
2. Assign each entity $y_i \in Y$ to the cluster S_k represented by the closest c_k, using (2). Should there be no change in S, stop.
3. Update each centroid in C to the Minkowski centre of their respective clusters.
4. Update each weight w_{kv}, using Equation (4). Go to step 2.

The Minkowski centre of a given feature v for a cluster S_k can be found over each entity $y_{iv} \in S_k$ by using a steepest descent algorithm [10]. The WK-Means algorithm applies the squared Euclidean distance. In this we still have a user-defined exponent to set, but this is solely a weight exponent, the distance exponent is always two. The iMWK-Means is initialized with a modified version of the intelligent K-Means algorithm [28], which uses the weighted Minkowski distance (2).

The feature weighting procedure used by both WK-Means and iMWK-Means would not deal properly with a subset of V in which features contain relevant, but redundant information. Such features would have similarly low dispersions. Since a single weight w_{kv} is calculated at a time these features would have their weights set to a similarly high value. We believe that this issue makes the removal of redundant features prior to the use of either WK-Means or iMWK-Means, beneficial.

Feature selection using feature similarity (FSFS) [29] is one of the most popular unsupervised algorithms that fits to our needs. In this the authors identify

features containing similar information by using the maximum information compression index (MIC). This index is defined below, for the variables x and y.

$$2\lambda_2(x,y) = \\ var(x) + var(y) - \sqrt{(var(x)+var(y))^2 - 4var(x)var(y)(1-p(x,y)^2)}, \quad (5)$$

where $p(x,y)$ is the correlation coefficient given by $\frac{cov(x,y)}{\sqrt{var(x)var(y)}}$. MIC has various interesting properties, such as being invariant to the rotation of the variables and to the translation of the data set [29]. FSFS applies MIC as applied as follows.

1. Choose an initial value for k, following the constrain $k \leq |V|-1$. Set $R \leftarrow V$, standardise the features rather than entities (for details see Sections 3 and 4).
2. For each feature $F_i \in R$, calculate r_i^k, the dissimilarity between F_i and its kth nearest neighbour feature in R, using Equation: 5.
3. Find the feature $F_{i'}$ for which r_i^k is minimum. Retain $F_{i'}$ and discard its k nearest features. Set $\varepsilon = r_{i'}^k$.
4. Adjust k in relation to the number of features. If $k < |R|-1$, then $k = |R|-1$.
5. If $k = 1$ stop and output R.
6. Adjust k in relation to the similarity. While $r_i^k > \varepsilon$
   (a) $k = k - 1$
   (b) $r_i^k = inf_{F_i \in R} r_i^k$
   (c) if $k = 1$ go to Step 5.
7. Go to Step 2.

The above is a popular and useful algorithm. However, we see two issues that deserve to be addressed: (i) FSFS does not take into account the structure of the data set while selecting features; (ii) FSFS requires the user to define a parameter, k, beforehand. This parameter may increase the algorithm's flexibility, but unfortunately we see no clear method to estimate it. These issues, added to the inability of WK-Means and iMWK-Means to deal with redundant features made us analyse the possibility of a clustering-based solution for feature selection that could be used as a pre-processing step. We present our method in the next section.

3 Algorithm

In this section we present our feature selection method, intelligent K-Means for feature selection (iKFS). Our aim is to cluster the features in V that are similar, rather than the entities. By assigning similar features to the same cluster we can identify and remove those that are redundant. Clearly there are issues to address under this framework, for instance: (i) how many clusters of feature there would be in a given data set Y; (ii) given a cluster S_k of features, how many features should be selected from it.

Regarding issue (ii), it can be very tempting to keep a single feature from a cluster S_k of features, say the closest to the centroid c_k. However, we do not feel that each cluster should be treated the same, irrespective of its cardinality. With this in mind, we have decided to keep a subset of features of S_k, this subset cardinality is given by F_k.

$$F_k = \left\lceil \frac{|S_k|}{|Y|} * K \right\rceil, \tag{6}$$

where $|S_k|$ and $|Y|$ represent the cardinality of a given cluster of features S_k and the cardinality of the data set Y, respectively. One should note that since we are clustering features, the original data set has to be transposed, so the cardinality of Y is in fact the original number of features (Section 4 described experiments with 5,680 features). Equation (6) requires the number of clusters K to be known, taking us back to issue (i), its estimation. In our method we find K by using iK-Means, which can also be used to find good initial centroids for K-Means. The choice of iK-Means was based on its previous success as a clustering algorithm in different scenarios [28], [5], [9]. We introduce our method in full below.

1. Transpose the data set Y so that the original features become entities and then standardise the data set.
2. Apply the iK-Means algorithm setting $\theta = 0$.
3. For each cluster S_k, find F_k (Equation 6) features that have the highest maximum information compression (Equation 5) in relation to c_k. Put such features in R.
4. Output the features in R.

We are interested in selecting features that are dissimilar to all others, such features will most likely become singletons during the clustering process. In order to avoid disregarding such features we set $\theta = 0$.

4 Experiments

Electroencephalography (EEG) is the recording of high-dimensional noise-prone signals that can be captured from a brain via a non-invasive procedure. There is considerable research supporting the belief that these signals contain information about the current state or intention of a subject's mind [6], [12], [13], [18], [26].

We have recorded data from three healthy subjects (A, B and C) for our experiments, using five bipolar electrodes (five channels), and a sampling frequency of 250Hz. These five electrodes were placed on the subjects head following the standard positions in the extended 10-20 system, using fc3 to pc3, fc1 to pc1, cz to pz, fc2 to pc2, and fc4 to pc4.

Our aim is to perform mental task separation, in other words, given a set of possible tasks we would like to know what particular task a subject is thinking about. We have three possible tasks: (i) movement of the left hand; (ii) movement of the right hand; (iii) movement of the feet. After visually suggesting what

task the subject should be thinking about, we recorded the EEG data for eight seconds, constituting a trial. Here we intend to cluster trials into the right tasks. Hence, the number of clusters is known to be three. We have gathered data from 240, 120 and 350 trials for each subject, respectively. The difference in the number of trials relates solely to the availability of subjects and staff.

We have pre-processed our data sets in two steps. First, we transformed the data into its power spectrum density (PSD). EEG patterns are normally found in the frequency space rather than amplitude, and PSD helps us to identify periodicities in the data. This transformation has been successfully applied in previous research [6], [12], [13], [18], [26].

Second, having a trial represented by 71 time-related samples each with 80 PSD-features, we generated a data matrix for each subject containing the respective number of trials (240, 120 and 350) over 5,680 features (71 x 80). We then standardised the data numerically.

$$y_{iv} = \frac{x_{iv} - \bar{x_v}}{0.5 * (max(x_v) - min(x_v))}, \tag{7}$$

where x_{iv} represents the PSD value of trial i in feature v, and $\bar{x_v}$ the average of feature v over all trials in the data set. The standard deviation is surely more popular in the standardization of data sets than the range. However, we have opted for the latter as the former favours unimodal distributions [27], [32].

The FSFS algorithm requires an user-defined parameter k. We have performed experiments with such parameter from 4,800 to 5,600 in steps of 100 for WK-Means and iMWK-Means independently. With this interval it is possible for FSFS to select a quantity of features close to that selected by iKFS. Note that the optimal k for WK-Means may not be the same as for iMWK-Means. Our method, iKFS, does not require any extra parameter, so the features used in WK-Means and iMWK-Means when the data is pre-processed with iKFS are exactly the same. Regarding the parameters required by the clustering algorithms themselves, WK-Means and iMWK-Means, we have run experiments from 1.0 to 5.0 in steps of 0.1. In this paper we do not deal with their estimation.

Since we have the labels for each trial in the data sets, we present the best possible results for each of these two algorithms in terms of their cluster recovery. This is calculated by using a confusion matrix.

We show the results of our experiments in Table 1. We are happy to see that in both algorithms, WK-Means and iMWK-Means, iKFS presents features that are more representative. This is visible thanks to the differences in cluster recovery when using FSFS and iKFS in both WK-Means and iMWK-Means. Table 1 also shows us that a much higher number of features does not necessarily means features that are more representative, nor better final accuracy.

Table 1 does not present average or standard deviation values for iMWK-Means because this is a deterministic algorithm, unlike WK-Means. This happens because iMWK-Means applies a version of iK-Means in its initialization, making it output the same clustering for a given data set irrespective of how many times it is run.

Table 1: Cluster recovery of WK-Means and iMWK-Means using the features selected by iKFS and FSFS. The number of features is in the parenthesis.

	Feature selection method	Exponent		Accuracy		
		Distance	Weight	Mean	Std	Max
Subject A						
WK-Means	FSFS (25)	2.0	3.9	48.2	2.5	52.1
WK-Means	iKFS (20)	2.0	1.5	54.1	1.4	56.2
iMWK-Means	FSFS (28)	3.2	3.2	-	-	51.2
iMWK-Means	iKFS (20)	4.5	4.5	-	-	59.2
Subject B						
WK-Means	FSFS (472)	2.0	4.7	59.2	4.6	73.3
WK-Means	iKFS (9)	2.0	3.6	68.5	4.7	76.7
iMWK-Means	FSFS (393)	2.0	2.0	-	-	65.0
iMWK-Means	iKFS (9)	2.5	2.5	-	-	66.7
Subject C						
WK-Means	FSFS (33)	2.0	4.7	39.6	1.8	42.3
WK-Means	iKFS (11)	2.0	4.5	56.5	0.3	56.9
iMWK-Means	FSFS (33)	4.6	4.6	-	-	42.3
iMWK-Means	iKFS (11)	1.8	1.8	-	-	58.6

5 Conclusions

In this paper we introduced intelligent K-Means for feature selection (iKFS). This algorithm reduces the number of features that contain similar information, redundant features, in a given data set. It does so by generating clusters of features, instead of entities, using the anomalous pattern method of iK-Means. This anomalous pattern method is rather useful in finding the number of clusters of features in the data set, together with good initial centroids for such clusters. Features within the same cluster are then said to be similar, and by consequence redundant to some degree. Our method selects a representative features from each cluster, with the exact quantity calculated based on the cardinality of each cluster and the features maximum information compression.

After explain the details of our method in Section 3 we go to an empirical validation in Section 4. In this validation we perform a number of experiments using data sets comprised of Electroencephalography (EEG) signals originated from three healthy subjects. EEG data tends to be high-dimensional (our data sets have 5,680 features) and noisy, hence our choice to use this type of data in our experiments. We have experimented two feature weighting clustering algorithms, WK-Means [4] and iMWK-Means [10], as well s two feature selection algorithms used in the data pre-processing stage, our iKFS and feature selection using feature similarity (FSFS) method [29].

The WK-Means and iMWK-Means algorithms perform feature weighting which allows them to set different degrees of relevance to each feature, as well as simply remove features from a data set. However, the feature weights are set by analysing one feature at a time which means that if two features have the

exact same relevant information none will be removed. In our experiments we have found that these feature weighting algorithms benefit from an extra data pre-processing step to reduce the quantity of redundant features in a data set. This is particularly true when iKFS is used in this extra pre-processing step since unlike FSFS, iKFS takes the data structure into account when removing features.

We do find the results we show here to be rather promising and our future research will aim to further optimise the features selected by our method, as well as experiment with data sets from other scenarios. We also intend to experiment with feature weighting algorithms using fuzzy logic that have been recently introduced [34].

References

1. G. H. Ball and D. J. Hall. A clustering technique for summarizing multivariate data. *Behavioral Science*, 12(2):153–155, 1967.
2. J. C. Bezdek. *Pattern recognition with fuzzy objective function algorithms*. Norwell MA: Kluwer Academic Publishers, 1981.
3. M. E. Celebi, H. A. Kingravi, and P. A. Vela. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, 40(1):200–210, 2013.
4. E. Y. Chan, W. K. Ching, M. K. Ng, and J. Z. Huang. An optimization algorithm for clustering using weighted dissimilarity measures. *Pattern recognition*, 37(5):943–952, 2004.
5. M. M. T. Chiang and 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):3–40, 2010.
6. S. Chiappa and S. Bengio. HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems. In *European Symposium on Artificial Neural Networks, ESANN*, pages 193–204, 2004.
7. R. C. de Amorim. An empirical evaluation of different initializations on the number of k-means iterations. *Lecture Notes in Computer Science*, 7629:15–26, 2013.
8. R. C. de Amorim. Feature relevance in Ward's hierarchical clustering using the lp norm. *Journal of Classification*, (to appear).
9. R. C. de Amorim and P. Komisarczuk. On initializations for the Minkowski weighted k-means. *Lecture Notes in Computer Science*, 7619:45–55, 2012.
10. R. C. de Amorim and B. Mirkin. Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering. *Pattern Recognition*, 45(3):1061–1075, 2012.
11. R. C. de Amorim and B. Mirkin. Removing redundant features via clustering: preliminary results in mental task separation. In *Proceedings of the 8th International Conference on Knowledge, Information and Creativity Support Systems (KICSS)*. November, 7-9. Krakow, Poland, 2013.
12. R. C. de Amorim, B. Mirkin, and J. Q. Gan. A method for classifying mental tasks in the space of EEG transforms. Technical report, Technical Report BBKS-10-01, Birkbeck University of London, London, 2010.
13. R. C. de Amorim, B. Mirkin, and J. Q. Gan. Anomalous pattern based clustering of mental tasks with subject independent learning–some preliminary results. *Artificial Intelligence Research*, 1(1):46–54, 2012.

14. G. De Soete. Optimal variable weighting for ultrametric and additive tree clustering. *Quality and Quantity*, 20(2-3):169–180, 1986.
15. G. De Soete. OVWTRE: A program for optimal variable weighting for ultrametric and additive tree fitting. *Journal of Classification*, 5(1):101–104, 1988.
16. W. S. DeSarbo, J. D. Carroll, C. A. Linda, and P. E. Green. Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables. *Psychometrika*, 49(1):57–78, 1984.
17. H. Frigui and O. Nasraoui. Unsupervised learning of prototypes and attribute weights. *Pattern recognition*, 37(3):567–581, 2004.
18. J. Q. Gan. Self-adapting BCI based on unsupervised learning. In *3rd International Workshop on Brain-Computer Interfaces*, pages 50–51, 2006.
19. I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.
20. J. Z. Huang, M. K. Ng, H. Rong, and Z. Li. Automated variable weighting in k-means type clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):657–668, 2005.
21. J. Z. Huang, J. Xu, M. Ng, and Y. Ye. Weighting method for feature selection in k-means. *Computational Methods of Feature Selection*, pages 193–209, 2008.
22. A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010.
23. J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. California, USA, 1967.
24. R. Maitra, A. D. Peterson, and A. P. Ghosh. A systematic evaluation of different methods for initializing the k-means clustering algorithm. *Transactions on Knowledge and Data Engineering*, pages 522–537, 2010.
25. V. Makarenkov and P. Legendre. Optimal variable weighting for ultrametric and additive trees and k-means partitioning: Methods and software. *Journal of Classification*, 18(2):245–271, 2001.
26. J. Millan and J. Mouriño. Asynchronous BCI and local neural classifiers: an overview of the adaptive brain interface project. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 11(2):159–161, 2003.
27. G. W. Milligan and M. C. Cooper. A study of standardization of variables in cluster analysis. *Journal of Classification*, 5(2):181–204, 1988.
28. B. Mirkin. *Clustering for data mining: a data recovery approach*, volume 3. CRC Press, 2005.
29. P. Mitra, C. A. Murthy, and S. K. Pal. Unsupervised feature selection using feature similarity. *IEEE transactions on pattern analysis and machine intelligence*, 24(3):301–312, 2002.
30. J. M. Pena, J. A. Lozano, and P. Larranaga. An empirical comparison of four initialization methods for the k-means algorithm. *Pattern recognition letters*, 20(10):1027–1040, 1999.
31. M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD workshop on text mining*, pages 525–526. Boston, 2000.
32. D. Steinley. Standardizing variables in k-means clustering. In *Classification, clustering, and data mining applications*, pages 53–60. Springer, 2004.
33. D. Steinley and M. J. Brusco. Initializing k-means batch clustering: A critical evaluation of several techniques. *Journal of Classification*, 24(1):99–121, 2007.
34. L. Svetlova, B. Mirkin, and H. Lei. MFWK-Means: Minkowski metric fuzzy weighted k-means for high dimensional data clustering. In *IEEE 14th Interna-*

tional Conference on Information Reuse and Integration (IRI), pages 692–699. IEEE, 2013.
35. R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.
36. R. Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.