Equi-sized, Homogeneous Partitioning

Frank Klawonn[1] and Frank Höppner[2]

[1] Department of Computer Science
University of Applied Sciences Braunschweig /Wolfenbüttel
Salzdahlumer Str. 46/48
38302 Wolfenbüttel, Germany

[2] Department of Economics
University of Applied Sciences Braunschweig /Wolfenbüttel
Robert Koch Platz 10-14
38440 Wolfsburg, Germany
{f.klawonn,f.hoeppner}@fh-wolfenbuettel.de

Abstract. We consider the problem of partitioning a data set of n data objects into c homogeneous subsets (that is, data objects in the same subset should be similar to each other), such that each subset is of approximately the same size. This problem has applications wherever a population has to be distributed among a limited number of resources and the workload for each resource shall be balanced. We modify an existing clustering algorithm in this respect, present some empirical evaluation and discuss the results.

1 Introduction

Cluster analysis is a widely used technique that seeks for groups in data. The result of such an analysis is a set of groups or clusters where data in the same group are similar (homogeneous) and data from distinct groups are different (heterogeneous) [1]. In this paper, we consider a variation of the clustering problem, namely the problem of subdividing a set X of n objects into c homogeneous groups *of equal size*. In contrast to the clustering problem, we abandon the heterogeneity between groups and introduce the requirement of having equi-sized groups.

Applications for this kind of *uniform clustering* include for instance: (a) The distribution of n students into c groups of equal strength to obtain fair class sizes and with homogeneous abilities and skills to allow for teaching methods tailored to the specific needs of each group. (b) The distribution of n jobs to c machines or workers such that every machine has an identical workload and as similar jobs as possible to reduce the configuration time. (c) The placement of c sites such that goods from n locations can be transported to the c sites, while the total covered distance is minimized and queuing at the sites is avoided, that is, approximately the same number of goods should arrive at each site.

Due to the similarity of our problem with traditional clustering problems, we are going to modify an existing clustering algorithm, which will be reviewed

in section 2. This objective function-based clustering algorithm – a variant of k-means – transforms the discrete, combinatorial problem into a continuous one, such that numerical problem solving methods can be applied. We modify the objective function such that the equi-sized clusters are considered in section 3 and discuss the results in section 4.

2 The FCM Algorithm

The fuzzy c-means (FCM) clustering algorithm partitions a data set $X := \{x_1, ..., x_n\} \subset \mathbf{R}^d$ into c clusters. A cluster is represented by a prototype $p_i \in \mathbf{R}^d$, $1 \leq i \leq c$. The data-prototype relation is not binary, but a membership degree $u_{ij} \in [0,1]$ indicates the degree of belongingness of data object x_j to prototype p_i or cluster number i. All membership degrees form a membership matrix $U \in \mathbf{R}^{c \times n}$. We can interpret the membership degrees as "probabilistic memberships", since we require

$$\forall 1 \leq j \leq n : \qquad \sum_{i=1}^{c} u_{ij} = 1 \,. \tag{1}$$

The clustering process is carried out by minimizing the objective function

$$J_m = \sum_{j=1}^{n} \sum_{i=1}^{c} u_{ij}^m d_{ij} \quad \text{with} \quad d_{ij} = \|x_j - p_i\|^2 \,. \tag{2}$$

under constraint (1). If the Euclidean distance between datum x_j and prototype p_i is high, J_m is minimized by choosing a low membership degree near 0. If the distance is small, the membership degree approaches 1. J_m is effectively minimized by alternating optimisation, that is, we alternatingly minimize (2) with respect to the prototypes (assuming memberships to be constant) and then with respect to the membership degrees (assuming prototypes to be constant). In both minimization steps, we obtain closed form solutions, for the prototypes:

$$\forall 1 \leq i \leq c : \qquad p_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m} \tag{3}$$

and for the membership degrees:

$$u_{ij} = \begin{cases} \dfrac{1}{\sum_{l=1}^{c} \left(\frac{\|x_j - p_i\|^2}{\|x_j - p_l\|^2} \right)^{\frac{1}{m-1}}} & \text{in case } I_j = \emptyset \\ \dfrac{1}{|I_j|} & \text{in case } I_j \neq \emptyset, i \in I_j \\ 0 & \text{in case } I_j \neq \emptyset, i \notin I_j \end{cases} \tag{4}$$

where $I_j = \{k \in \mathbf{N}_{\leq c} \,|\, x_j = p_k\}$. The FCM algorithm is depicted in Fig. 1. For a more detailed discussion of FCM and examples we refer for instance to [2, 3].

```
choose m > 1 (typically m = 2)
choose termination threshold ε > 0
initialize prototypes p_i (randomly)
repeat
  update memberships using (4)
  update prototypes using (3)
until change in memberships drops below ε
```

Fig. 1. The FCM algorithm.

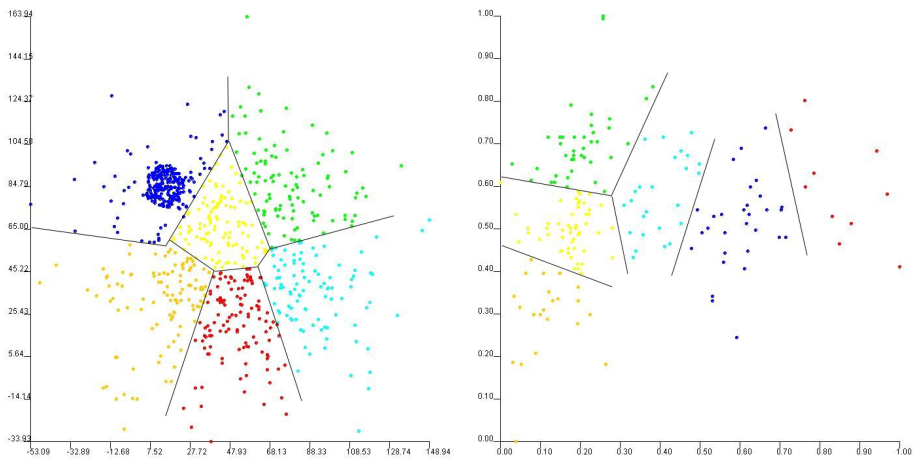

Fig. 2. Results of the FCM algorithm on two data sets.

3 Equi-sized Clusters

It is often said that the k-means (as well as the FCM) algorithm seeks for clusters of approximately the same size, but this is only true if the data density is uniform. As soon as the data density varies, a single prototype may very well cover a high-density cluster and thereby gains many more data objects than the other clusters. This leads to large differences in the size of the clusters. Examples for this phenomenon are shown in Fig. 2 for two data sets: On the left image, there is a very high density cluster in the top left corner, on the right image, the density decreases from left to right, so the rightmost cluster has only some data.

The idea of our modification is to include an additional constraint in the objective function (2) that forces the clusters to cover the same number of data objects. The size of cluster i (number of data objects) corresponds to the sum of the membership values $\sum_{j=1}^{n} u_{ij}$. In fact, since we have continuous membership degrees we may require

$$\sum_{j=1}^{n} u_{ij} \;=\; \frac{n}{c} \tag{5}$$

for all $i \in \{1, \ldots, c\}$ even if n is not a multitude of c. This additional constraint (5) is – together with the constraint (1) – integrated into the objective function (2) via Lagrange multipliers. We then solve for the cluster prototypes and Lagrange multipliers by setting the partial derivatives to zero. This turns out to be a difficult problem for the general case of an arbitrary value of m, therefore we restrict ourselves to the case of $m = 2$, which is the most frequently used value of m in FCM. Given our Lagrange function

$$L \;=\; \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^2 d_{ij} \;+\; \sum_{j=1}^{n} \alpha_j \left(1 - \sum_{i=1}^{c} u_{ij}\right) \;+\; \sum_{i=1}^{c} \beta_i \left(\frac{n}{c} - \sum_{j=1}^{n} u_{ij}\right) \tag{6}$$

we obtain as partial derivatives

$$\frac{\partial L}{\partial u_{ij}} \;=\; 2u_{ij}d_{ij} - \alpha_j - \beta_i \;=\; 0 \tag{7}$$

These equations, together with the constraints (1) and (5), lead to the following system of $(c \cdot n + c + n)$ linear equations for the variable u_{ij}, α_i and β_j ($i \in \{1, \ldots, c\}$, $j \in \{1, \ldots, n\}$). Empty entries indicate the value zero, RHS stands for the right hand side of the equation.

	$u_{1,1}$	$\ldots$	$u_{1,n}$	$\ldots$	$u_{c,1}$	$\ldots$	$u_{c,n}$	α_1	$\ldots$	α_n	β_1	$\ldots$	β_c	RHS
$\frac{\partial L}{\partial u_{1,1}}$	$2d_{1,1}$							-1			-1			
$\vdots$		$\ddots$							$\ddots$		$\vdots$			
$\frac{\partial L}{\partial u_{1,n}}$			$2d_{1,n}$							-1	-1			
$\vdots$				$\ddots$					$\ddots$			$\ddots$		
$\frac{\partial L}{\partial u_{c,1}}$					$2d_{c,1}$			-1					-1	
$\vdots$						$\ddots$			$\ddots$				$\vdots$	
$\frac{\partial L}{\partial u_{c,n}}$							$2d_{c,n}$			-1			-1	
$\sum u_{i,1}$	1			$\ldots$	1									1
$\vdots$		$\ddots$				$\ddots$								$\vdots$
$\sum u_{i,n}$			1				1							1
$\sum u_{1,j}$	1	$\ldots$	1											n/c
$\vdots$				$\ddots$										$\vdots$
$\sum u_{c,j}$					1	$\ldots$	1							n/c

In principle, this system of linear equations could be solved by a suitable numerical algorithm. Even for small data sets with 200 data objects and 5 clusters, this would mean that we have to solve a system of 1205 equations in each iteration step of the clustering algorithm, which is not acceptable in terms of computational costs. However, it is possible to solve this system of equations in a more efficient way. When multiplying the equations for $u_{k1}, \ldots, u_{kn}$ by $\frac{1}{2d_{k1}}, \ldots, \frac{1}{2d_{kn}}$,

respectively, and then subtracting the resulting equations from the equation for $\sum_j u_{kj}$, we obtain

$$\sum_{j=1}^{n} \frac{\alpha_j}{2d_{kj}} + \beta_k \sum_{j=1}^{n} \frac{1}{2d_{kj}} = \frac{n}{c}. \tag{8}$$

From equation (7), we obtain

$$u_{ij} = \frac{\alpha_j + \beta_i}{2d_{ij}}. \tag{9}$$

Taking constraint (1) into account, yields

$$1 = \sum_{i=1}^{c} u_{ij} = \frac{\alpha_j}{2} \sum_{i=1}^{c} \frac{1}{d_{ij}} + \frac{1}{2} \sum_{i=1}^{c} \frac{\beta_i}{d_{ij}},$$

leading to

$$\alpha_j = \frac{2 - \sum_{i=1}^{c} \frac{\beta_i}{d_{ij}}}{\sum_{i=1}^{c} \frac{1}{d_{ij}}}. \tag{10}$$

Inserting (10) into (8), we obtain: $\displaystyle\sum_{j=1}^{n} \frac{2 - \sum_{i=1}^{c} \frac{\beta_i}{d_{ij}}}{2\sum_{i=1}^{c} \frac{d_{kj}}{d_{ij}}} + \beta_k \sum_{j=1}^{n} \frac{1}{2d_{kj}} = \frac{n}{c}$

and thus

$$-\sum_{j=1}^{n} \frac{\sum_{i=1}^{c} \frac{\beta_i}{d_{ij}}}{2\sum_{i=1}^{c} \frac{d_{kj}}{d_{ij}}} + \beta_k \sum_{j=1}^{n} \frac{1}{2d_{kj}} = \frac{n}{c} - \sum_{j=1}^{n} \frac{1}{\sum_{i=1}^{c} \frac{d_{kj}}{d_{ij}}}. \tag{11}$$

This induces a system of c linear equations for the β_k with coefficients

$$a_{k\ell} = \begin{cases} -\sum_{j=1}^{n} \frac{\sum_{i=1}^{c} \frac{1}{d_{\ell j}}}{2\sum_{i=1}^{c} \frac{d_{kj}}{d_{ij}}} & \text{if } k \neq \ell \\ -\sum_{j=1}^{n} \frac{\sum_{i=1}^{c} \frac{1}{d_{\ell j}}}{2\sum_{i=1}^{c} \frac{d_{kj}}{d_{ij}}} + \sum_{j=1}^{n} \frac{1}{2d_{kj}} & \text{if } k = \ell. \end{cases} \tag{12}$$

This system of linear equations can be solved by a suitable numerical algorithm. The computation time is acceptable, since the number of equations is equal to the number of clusters and therefore independent of the number of data. Once the β_i have been determined, we can compute the α_j using equation (10) and finally obtain the membership degrees based on equation (9). After all, we arrive at the clustering algorithm depicted in Fig. 3. Note that the boundedness of the membership degrees $u_{ij} \in [0, 1]$ represents an additional constraint on the objective function of FCM as well as the objective function of our new algorithm. In the original FCM, however, it was not necessary to consider it explicitly, because one can easily see from the resulting membership degrees (4) that this condition is satisfied. It is not possible to conclude this boundedness for the new membership degrees (9). It is clear, however, that the influence of negative memberships will be rather small: Since the objective function (2) and (6) uses only positive weights u_{ij}^2, large negative values cannot help in the minimization. We will comment on this in the following section.

choose termination threshold $\varepsilon > 0$
initialise prototypes p_i
repeat
  solve linear equation system (11) for β
  using β, calculate α using (10), update memberships using (9)
  update prototypes using (3)
until change in memberships drops below ε

Fig. 3. The proposed algorithm.

4 Examples and Discussion

To illustrate the impact of our modified objective function, we show the results of the new algorithm for the data sets shown in Fig. 2, where the standard FCM algorithm yielded a result with high variation in the cluster size. The results are shown in the left images of Figs. 4 and 6. By comparison to Fig. 2 we see, that the high-density cluster has been split into two clusters (Fig. 4) and that the data on the left of Fig. 6 is now distributed among four rather than three clusters, such that the rightmost cluster gains more data. As expected, the sum of membership degrees for each individual cluster equals $\frac{n}{c}$.

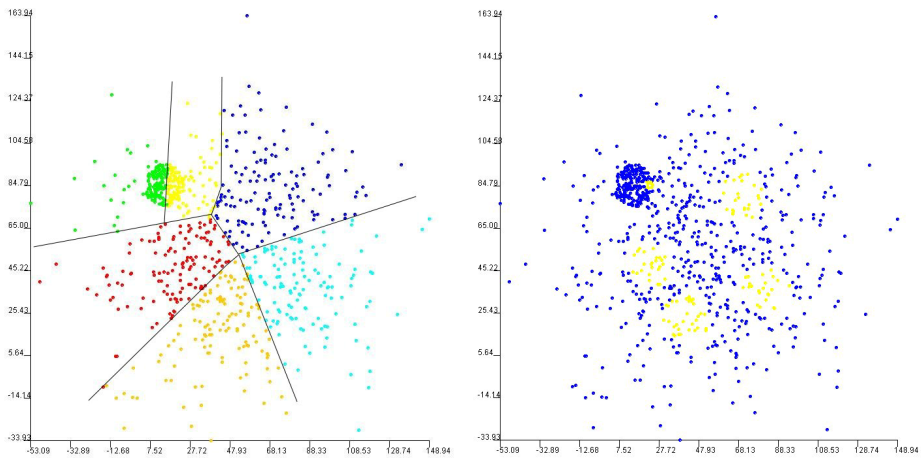

Fig. 4. Results of the new algorithm on the data set shown in Fig. 2. Left: Resulting partition. Right: Points with negative membership degrees (marked in lighter shading).

Regarding the boundedness of the membership degrees u_{ij} it turned out that they actually take negative values. This is, of course, an undesired effect, because then the interpretation of $\sum_{j=1}^{n} u_{ij}$ as the *size* or number of data objects is not quite correct. As conjectured in the previous section, it turned out on closer examination that the total sum of negative weights is rather small. In both

data sets, the sum of all negative membership degrees was below 0.5% of the total data set size n. We want to illustrate the kind of situation in which negative membership degrees occur with the help of the data set shown in Fig. 5. Consider the data set is partitioned into three clusters. Since the leftmost cluster has an additional data object x in the middle, it is not obvious how to distribute the data among all clusters in equal shares.

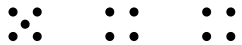

Fig. 5. A 'difficult' example data set.

Regarding the minimization of the sum of weighted distances, it would be optimal to assign high membership degrees to all five data objects. This would, however, violate the constraint that all clusters must share the same size. To get membership degrees as high as possible, the membership of all other data objects (middle and right cluster) to this cluster are chosen slightly negative. Since the sum of all membership values is constrained to be one, negative values for the middle and right data allow us to have slightly higher degrees for the leftmost data. On the other hand, having a negative membership degrees for some data object x on the right forces us to increase other membership degrees of x (to guarantee a sum of 1). This is possible almost at no cost, if x is close to the centre of another cluster, because then we have a small distance value and increasing the membership degree to this cluster does no harm in the minimization of (2). (For a detailed discussion of the influence of the membership weight u_{ij}^m see [4].)

To summarise: In a situation where an equi-sized partition is difficult to obtain while minimizing at the same time the sum of weighted distances (2), the cluster with too many data objects 'borrows' some membership from data near the centres of the other clusters. Figs. 4 and 6 show this effect for the two example data sets. The data for which negative membership values occur are shown in a lighter shading. These data objects are all close to the respective cluster prototype. And there is always one cluster without any negative membership degrees, which corresponds to the rightmost cluster in our data set in Fig. 5.

In all our experiments, the side effects of this trade off between minimizing (2) and satisfying (5) were quite small, so we do not consider this as a major drawback of our approach. We can even make use of this information: By analysing which cluster has no negative membership degrees at all, we can find out which cluster tends to be 'too big'. When breaking ties in the final assignment of data to clusters, it should be this cluster that gets more data objects than the other. It should be noted that the assignment of a data object to the cluster with the highest membership degree does not guarantee that each cluster contains exactly the number of data. This is anyway impossible, except when n/c is an integer number. The small deviations from n/c resulting from our algorithm can be easily balanced by taking the more ambiguous membership degrees into account in order to re-assign a few data to other clusters.

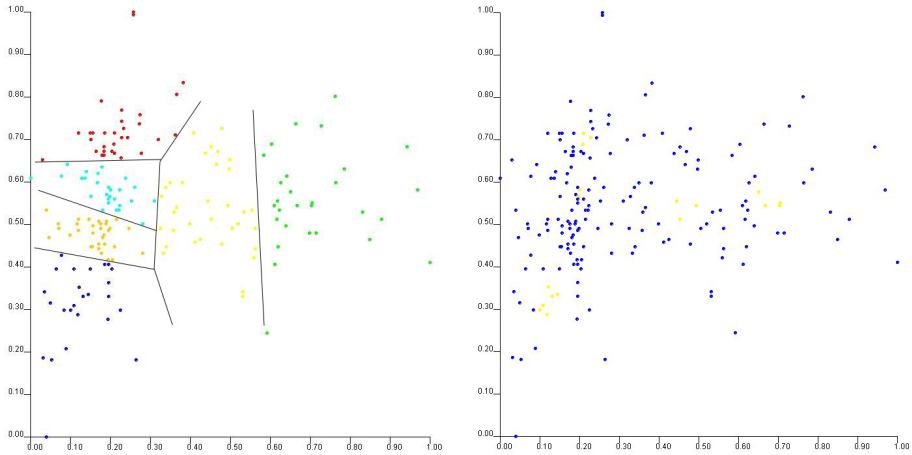

Fig. 6. Results on the wine data set shown as projections. Left: Derived partition. Right: Points with negative membership degrees (marked in lighter shading).

5 Conclusions

In this paper, we have considered the problem of subdividing a data set into homogeneous groups of equal size. Finding homogeneous groups is a typical task for clustering algorithms, however, if the data density is not uniform, such algorithms usually tend to deliver clusters of unequal size, which is inappropriate for some applications. We have proposed an algorithm that outperforms a popular variant of k-means in that respect. Although we have only discussed the case of equi-sized clusters, in principle it is also possible to subdivide the data set into groups of any predefined size, which makes our approach quite useful for a range of applications where capacity restrictions apply.

Another, slightly weaker approach to the problem of uniform clustering would be to replace the strict constraints (5) by adding a (weighted) penalty term of the form $\sum_{i=1}^{c}(\frac{n}{c} - \sum_{j=1}^{n} u_{ij})^2$ to the objective function (2). Due to the limited space here, we leave this discussion open for a subsequent paper.

References

1. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall (1988)
2. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
3. Höppner, F., Klawonn, F., Kruse, R., Runkler, T.A.: Fuzzy Cluster Analysis. John Wiley & Sons, Chichester, England (1999)
4. Klawonn, F., Höppner, F.: What is fuzzy about fuzzy clustering? – Understanding and improving the concept of the fuzzifier. In: Advances in Intelligent Data Analysis, Springer (2003) 254–264