Constrained K-Means Clustering

P. S. Bradley
Microsoft Research
One Microsoft Way
Redmond, WA 98052

bradley@microsoft.com

K. P. Bennett **A. Demiriz**
Dept. of Mathematical Sciences
Dept. of Decision Sciences and Eng. Sys.
Renselaer Polytechnic Institute
Troy, NY 12180

{bennek,demira}@rpi.edu

MSR-TR-2000-65

May, 2000

Abstract

We consider practical methods for adding constraints to the K-Means clustering algorithm in order to avoid local solutions with empty clusters or clusters having very few points. We often observe this phenomena when applying K-Means to datasets where the number of dimensions is $n \geq 10$ and the number of desired clusters is $k \geq 20$. We propose explicitly adding k constraints to the underlying clustering optimization problem requiring that each cluster have at least a minimum number of points in it. We then investigate the resulting cluster assignment step. Preliminary numerical tests on real datasets indicate the constrained approach is less prone to poor local solutions, producing a better summary of the underlying data.

1 Introduction

The K-Means clustering algorithm [5] has become a workhorse for the data analyst in many diverse fields. One drawback to the algorithm occurs when it is applied to datasets with m data points in $n \geq 10$ dimensional real space R^n and the number of desired clusters is $k \geq 20$. In this situation, the K-Means algorithm often converges with one or more clusters which are either empty or summarize very few data points (i.e. one data point). Preliminary tests on clustering sparse 300-dimensional web-browsing data indicate that K-Means frequently converges with truly empty clusters. For $k = 50$ and $k = 100$, on average 4.1 and 12.1 clusters are empty.

We propose explicitly adding k constraints to the underlying clustering optimization problem requiring that cluster h contain at least τ_h points. We focus on the resulting changes to the K-Means algorithm and compare the results of standard K-Means and the proposed Constrained K-Means algorithms. Empirically, for modest values of τ_h, solutions are obtained that better summarize the underlying data.

Since clusters with very few or no data points may be artifacts of poor local minima, approaches to handling them include re-running the algorithm with new initial cluster centers or checking the cluster model at algorithm termination, resetting empty clusters, and re-running the algorithm. Our approach avoids the additional computation of these heuristics which may still produce clusters with too few points. In addition to providing a well-posed, mathematical way to avoid small clusters, this work can generalize to other constraints ensuring desirable clustering solutions (e.g. outlier removal or specified groupings) and to Expectation-Maximization probabilistic clustering.

Alternatively, empty clusters can be regarded as desirable "natural" regularizers of the cluster model. This heuristic argument states that if the data do not "support" k clusters, then allowing clusters to go empty, and hence reducing the value of k, is a desirable side effect. But there are applications in which, given a value of k, one desires to have a cluster model with k non-empty clusters. These include the situation when the value of k is know *a priori* and applications in which the cluster model is utilized as a compressed version of a specific dataset [1, 8].

The remaining portion of the paper is organized as follows. Section 2 formalizes the constrained clustering optimization problem and outlines the algorithm computing a locally optimal solution. The sub-problem of computing cluster assignments so that cluster h contains at least τ_h points is discussed in Section 3. Section 4 presents numerical evaluation of the algorithm in comparison with the standard K-Means implementation on real datasets and Section 5 concludes the paper.

2 Constrained Clustering Problem and Algorithm

Given a dataset $D = \{x^i\}_{i=1}^m$ of m points in R^n and a number k of desired clusters, the K-Means clustering problem is as follows. Find cluster centers $C^1, C^2, \ldots, C^k$ in R^n such that the sum of the 2-norm distance squared between each point x^i and its *nearest* cluster center C^h is minimized. Specifically:

$$\min_{C^1,\ldots,C^k} \sum_{i=1}^{m} \min_{h=1,\ldots,k} \left(\frac{1}{2} \|x^i - C^h\|_2^2 \right). \tag{1}$$

By [4, Lemma 2.1], (1) is equivalent to the following problem where the min operation in the summation is removed by introducing "selection" variables $T_{i,h}$.

$$\begin{array}{ll} \underset{C,T}{\text{minimize}} & \sum_{i=1}^{m} \sum_{h=1}^{k} T_{i,h} \cdot \left(\frac{1}{2} \|x^i - C^h\|_2^2 \right) \\ \text{subject to} & \sum_{h=1}^{k} T_{i,h} = 1,\ i = 1, \ldots, m, \\ & T_{i,h} \geq 0,\ i = 1, \ldots, m,\ h = 1, \ldots, k. \end{array} \tag{2}$$

Note that $T_{i,h} = 1$ if data point x^i is closest to center C^h and zero otherwise.

Problem (2) or, equivalently (1), is solved by the K-Means algorithm iteratively. In each iteration, Problem (2) is solved first for $T_{i,h}$ with the cluster centers C^h fixed. Then, (2) is solved for C^h with the assignment variables $T_{i,h}$ fixed. The stationary point computed satisfies the Karush-Kuhn-Tucker (KKT) conditions [6] for Problem (2), which are necessary for optimality.

Algorithm 2.1 K-Means Clustering Algorithm *Given a database D of m points in R^n and cluster centers $C^{1,t}, C^{2,t}, \ldots, C^{k,t}$ at iteration t, compute $C^{1,t+1}, C^{2,t+1}, \ldots, C^{k,t+1}$ at iteration $t+1$ in the following 2 steps:*

1. **Cluster Assignment.** *For each data record $x^i \in D$, assign x^i to cluster $h(i)$ such that center $C^{h(i),t}$ is nearest to x^i in the 2-norm.*
2. **Cluster Update.** *Compute $C^{h,t+1}$ as the mean of all points assigned to cluster h.*

Stop when $C^{h,t+1} = C^{h,t}$, $h = 1, \ldots, k$, else increment t by 1 and go to step 1.

Suppose cluster h is empty when Algorithm 2.1 terminates, i.e. $\sum_{i=1}^{m} T_{i,h} = 0$. The solution computed by Algorithm 2.1 in this case satisfies the KKT conditions for (2). Hence, it is plausible that the standard K-Means algorithm may converge with empty clusters. In practice, we observe this phenomenon when clustering high-dimensional datasets with a large number of clusters.

To avoid clustering solutions with empty clusters, we propose explicitly adding constraints to Problem (2) requiring that cluster h contain at least τ_h

data points, where $\sum_{h=1}^{k} \tau_h \leq m$. This yields the following Constrained K-Means problem:

$$\begin{array}{ll} \underset{C,T}{\text{minimize}} & \sum_{i=1}^{m} \sum_{h=1}^{k} T_{i,h} \cdot \left(\frac{1}{2}\|x^i - C^h\|_2^2\right) \\ & \sum_{i=1}^{m} T_{i,h} \geq \tau_h,\ h = 1, \ldots, k \\ \text{subject to} & \sum_{h=1}^{k} T_{i,h} = 1,\ i = 1, \ldots, m, \\ & T_{i,h} \geq 0,\ i = 1, \ldots, m,\ h = 1, \ldots, k. \end{array} \tag{3}$$

Like the classic K-Means algorithm, we propose an iterative algorithm to solve (3).

Algorithm 2.2 Constrained K-Means Clustering Algorithm *Given a database D of m points in R^n, minimum cluster membership values $\tau_h \geq 0$, $h = 1, \ldots, k$ and cluster centers $C^{1,t}, C^{2,t}, \ldots, C^{k,t}$ at iteration t, compute $C^{1,t+1}, C^{2,t+1}, \ldots, C^{k,t+1}$ at iteration $t+1$ in the following 2 steps:*

1. **Cluster Assignment.** *Let $T_{i,h}^t$ be a solution to the following linear program with $C^{h,t}$ fixed:*

$$\begin{array}{ll} \underset{T}{\text{minimize}} & \sum_{i=1}^{m} \sum_{h=1}^{k} T_{i,h} \cdot \left(\frac{1}{2}\|x^i - C^{h,t}\|_2^2\right) \\ & \sum_{i=1}^{m} T_{i,h} \geq \tau_h,\ h = 1, \ldots, k \\ \text{subject to} & \sum_{h=1}^{k} T_{i,h} = 1,\ i = 1, \ldots, m, \\ & T_{i,h} \geq 0,\ i = 1, \ldots, m,\ h = 1, \ldots, k. \end{array} \tag{4}$$

2. **Cluster Update.** *Update $C^{h,t+1}$ as follows:*

$$C^{h,t+1} = \begin{cases} \frac{\sum_{i=1}^{m} T_{i,h}^t x^i}{\sum_{i=1}^{m} T_{i,h}^t} & \text{if } \sum_{i=1}^{m} T_{i,h}^t > 0, \\ C^{h,t} & \text{otherwise.} \end{cases} \tag{5}$$

Stop when $C^{h,t+1} = C^{h,t}$, $h = 1, \ldots, k$, else increment t by 1 and go to step 1.

Like the traditional K-Means approach, the Constrained K-Means algorithm iterates between solving (3) in $T_{i,h}$ for fixed C^h, then solving (3) in C^h for fixed $T_{i,h}$.

We end this section by with a finite termination result similar to [3, Theorem 7].

Proposition 2.3 *The Constrained K-Means Algorithm 2.2 terminates in a finite number of iterations at a cluster assignment that is locally optimal. Specifically, the objective function of (3) cannot be decreased by either reassignment of a point to a different cluster, while maintaining $\sum_{i=1}^{m} T_{i,h} \geq \tau_h$, $h = 1, \ldots, k$, or by defining a new cluster center for any of the clusters.*

Proof: At each iteration, the cluster assignment step cannot increase the objective function of (3). The cluster update step will either strictly decrease the value of the objective function of (3) or the algorithm will terminate since

$$C^{h,t+1} = \arg\min_{C} \sum_{i=1}^{m} \sum_{h=1}^{k} T_{i,h}^{t} \cdot \left(\frac{1}{2} \| x^i - C^h \|_2^2 \right) \tag{6}$$

is a strictly convex optimization problem with a unique global solution. Since there are a finite number of ways to assign m points to k clusters so that cluster h has at least τ_h points, since Algorithm 2.2 does not permit repeated assignments, and since the objective of (3) is strictly nonincreasing and bounded below by zero, the algorithm must terminate at some cluster assignment that is locally optimal. □

In the next section we discuss solving the linear program sub-problem in the cluster assignment step of Algorithm 2.2 as a minimum cost network flow problem.

3 Cluster Assignment Sub-problem

The form of the constraints in the cluster assignment sub-problem (4) make it equivalent to a Minimum Cost Flow (MCF) linear network optimization problem [2]. This is used to show that the optimal cluster assignment will place each point in exactly one cluster and can be found using fast network simplex algorithms. In general, a MCF problem has an underlying graph structure. Let $\mathcal{N}$ be the set of nodes. Each node $i \in \mathcal{N}$ has associated with it a value b_i indicating whether it is a supply node ($b_i > 0$), a demand node ($b_i < 0$), or a transshipment node ($b_i = 0$). If $\sum_{i \in \mathcal{N}} b_i = 0$, the problem is feasible (i.e. the sum of the supplies equals the sum of the demands). Let $\mathcal{A}$ be the set of directed arcs. For each arc $(i,j) \in \mathcal{A}$, the variable $y_{i,j}$ indicates amount of flow on the arc. Additionally, for each arc (i,j), the constant $c_{i,j}$ indicates the cost of shipping one unit flow on the arc. The MCF problem is to minimize $\sum_{(i,j) \in \mathcal{A}} c_{i,j} \cdot y_{i,j}$ subject to the sum of the flow leaving node i minus the sum of flow incoming is equal to b_i. Specifically, the general MCF is:

$$\begin{array}{ll} \underset{y}{\text{minimize}} & \sum_{(i,j) \in \mathcal{A}} c_{i,h} \cdot y_{i,j} \\ \text{subject to} & \sum_j y_{i,j} - \sum_j y_{j,i} = b_i, \forall i \in \mathcal{N} \\ & 0 \leq y_{i,j} \leq u_{i,j}, \forall (i,j) \in \mathcal{A}. \end{array} \tag{7}$$

Let each data point x^i correspond to a supply node with supply $= 1$ ($b_{x^i} = 1$). Let each cluster C^h correspond to a demand node with demand $b_{C^h} = -\tau_h$. Let there be an arc in $\mathcal{A}$ for each (x^i, C^h) pair. The cost on arc (x^i, C^h) is $\| x^i - C^h \|_2^2$. To satisfy the constraint that the sum of the supplies equals the sum of the demands, we need to add an artificial demand node a with demand

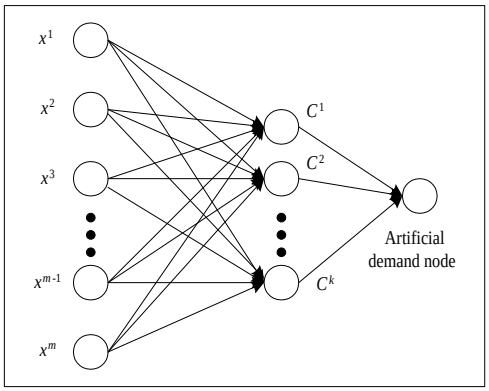


Figure 1: Equivalent Minimum Cost Flow formulation of (4).

$b_a = -m + \sum_{h=1}^{k} \tau_h$. There are arcs from each cluster node C_h to a with zero cost. There are no arcs to or from the data point nodes x^i to the artificial node a. See Figure 1. Specifically, let $\mathcal{N} = \{x^i,\, i = 1, \ldots, m\} \cup \{C^h,\, h = 1, \ldots, k\} \cup \{a\}$. Let $\mathcal{A} = \{(x^i, C^h),\, x^i, C^h \in \mathcal{N}\} \cup \{(C^h, a),\, C^h \in \mathcal{N}\}$. With these identifications and the costs, supplies and demands above, (4) has an equivalent MCF formulation. This equivalence allows us to state the following proposition that integer values of $T_{i,h}$ are optimal for (4).

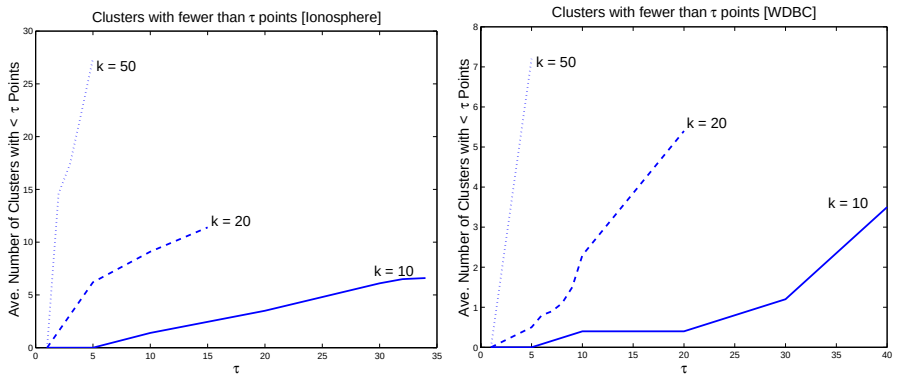
Proposition 3.1 *If each τ_h, $h = 1, \ldots, k$ is an integer, then there exists an optimal solution of (4) such that $T_{i,h} \in \{0, 1\}$.*

Proof: Consider the equivalent MCF formulation of (4). Since $b_{x^i} = 1, \forall x^i \in \mathcal{N}$, $b_{C^h} = -\tau_h$, and $b_a = -m + \sum_{h=1}^{k} \tau_h$ are all integers, it follows from [2, Proposition 2.3] that an optimal flow vector y is integer-valued. The optimal cluster assignment values $T_{i,h}$ correspond to y_{x^i, C^h} and, since each node x^i has 1 unit of supply, the maximum value of $T_{i,h}$ is 1. □

Hence, we are able to obtain optimal $\{0, 1\}$ assignments without having to solve a much more difficult integer programming problem. In addition to deriving the integrality result of Proposition 3.1, the MCF formulation allows one to solve (4) via codes specifically tailored to network optimization [2]. These codes usually run 1 or 2 orders of magnitude faster than general linear programming (LP) codes.

4 Numerical Evaluation

We report results using two real datasets: the Johns Hopkins Ionosphere dataset and the Wisconsin Diagnostic Breast Cancer dataset (WDBC) [7]. The Ionosphere dataset contains 351 data points in R^{33} and values along each dimension

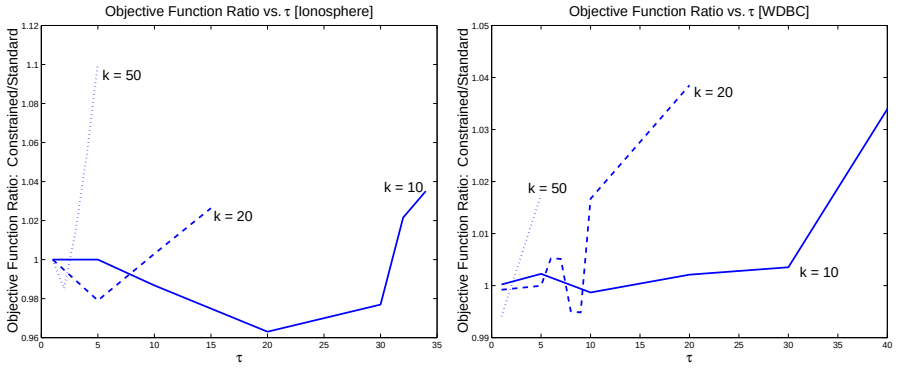


(a) Ionosphere, $K = 10, 20, 50$ (b) WDBC, $K = 10, 20, 50$

Figure 2: Average number of clusters with fewer than τ data points computed by the standard K-Means Algorithm 2.1

were normalized to have mean 0 and standard deviation 1. The WDBC data subset consists of 683 normalized data points in R^9. The values of τ_h (denoted by τ) were set equally across all clusters. The ILOG CPLEX 6.0 LP solver was used for cluster assignment. For initial cluster centers sampled uniformly on the range of the data, K-Means produced at least 1 empty cluster in 10 random trials on WDBC for $k \geq 30$ and on Ion for $k \geq 20$. Figures 2 and 3 give results for initial clusters chosen randomly from the dataset. This simple technique can eliminate many truly empty clusters. Figure 2 shows the frequency with which the standard K-Means algorithm converges to clusters having fewer than τ points.

The effect on the quality of the clustering by the constraints imposed by the Constrained K-Means Algorithm 2.2 is quantified by the ratio of the average objective function of (1) computed at the Constrained K-Means solution over that of the standard K-Means solution. Adding constraints to any minimization problem can never decrease the **globally** optimal objective value. Thus we would expect this ratio to be greater than 1. Surprisingly the Constrained K-Means algorithm frequently found better local minima (ratios less than 1) than did the standard K-Means approach. Note that the same starting points were used for both algorithms. Results are summarized in Figure 3. Notice that for a fixed k, solutions computed by Constrained K-Means are generally equivalent to standard K-Means for small τ-values. For large τ-values, the



(a) Ionosphere, $K = 10, 20, 50$ (b) WDBC, $K = 10, 20, 50$

Figure 3: Average ratio of objective function (1) computed at the Constrained K-Means solution over that of the standard K-Means solution versus τ.

Constrained K-Means solution is often inferior to those of standard K-Means. In this case, to satisfy the τ-constraints, the algorithm must group together points which are far apart resulting in a higher objective value. Superior clustering solutions are computed by the Constrained K-Means algorithm when τ is chosen in conjunction with k. For small values of k (e.g. $k = 5$) we observe ratios $<$ 1 up to $\tau = 50$ (maximum tested) on Ionosphere. For $k = 20$, we begin to see ratios > 1 for $\tau = 10$. Similar results are observed on WDBC.

5 Conclusion

K-Means can be extended to insure that every cluster contains at least a given number of points. Using a cluster assignment step with constraints, solvable by linear programming or network simplex methods, can guarantee a sufficient population within each cluster. A surprising result was that Constrained K-Means was less prone to local minima than traditional K-Means. Thus adding constraints may be beneficial to avoid local minima even when empty clusters are permissible. Constrained clustering suggests many research directions. Robust clustering can be done by simply adding an "outlier" cluster with high fixed distance that gathers "outliers" far from true clusters. Constraints forcing selected data into the same cluster could be used to incorporate domain knowledge or to enforce consistency of successive cluster solutions on related

data.

References

[1] K. P. Bennett, U. M. Fayyad, and D. Geiger. Density-based indexing for approximate nearest neighbor queries. In *Proc. 5th Intl. Conf. on Knowledge Discovery and Data Mining (KDD99)*, pages 233–243, New York, 1999. ACM Press.

[2] D. P. Bertsekas. *Linear Network Optimization*. MIT Press, Cambridge, MA, 1991.

[3] P. S. Bradley and O. L. Mangasarian. k-Plane clustering. (98-08), August 1998. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-08.ps*Journal of Global Optimization*, 16(1),2000.

[4] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Clustering via concave minimization. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems -9-*, pages 368–374, Cambridge, MA, 1997. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/96-03.ps.

[5] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, 1973.

[6] O. L. Mangasarian. *Nonlinear Programming*. McGraw–Hill, New York, 1969. Reprint: SIAM Classic in Applied Mathematics 10, 1994, Philadelphia.

[7] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases, 1992. www.ics.uci.edu/∼mlearn/MLRepository.html.

[8] J. Shanmugusundaram, U. M. Fayyad, and P. S. Bradley. Compressed data cubes for olap aggregate query approximation on continuous dimensions. In *Proc. 5th Intl. Conf. on Knowledge Discovery and Data Mining (KDD99)*, pages 223–232, New York, 1999. ACM Press.