

AMSC 607 / CMSC 878o Fall 2003
Homework 1 Due September 30, 2003

Many projects in engineering and science require the classification of data based on different heuristics. For example,

- Designers classify automobile engine performance as acceptable or unacceptable based on a combination of efficiency, emissions, noise levels, and other criteria.
- Researchers routinely classify documents as “relevant to the current project” or “irrelevant”.
- Genome decoding divides chromosomes into genes, regulatory regions, signals, etc.
- Pathologists identify cells as cancerous or benign.

We can classify data into different groups by *clustering* data that are close with respect to some distance measure. In this project, we investigate the design, use, and pitfalls of a popular clustering algorithm, the k -means algorithm.

The Problem

For concreteness, we will cluster the pixels in the image shown in Figure 1. Suppose our original image is of size $m \times p$, with the color for each of the mp pixels recorded by b bits. Then the total storage requirement is mpb bits. We will choose k pixel values (colors) as *cluster centers* and map each pixel to one of these. This will form k clusters of pixels. This saves space (since we store the cluster index for each pixel instead of the pixel value) and, in addition, filters out noise in the image.

The data for our sample problem is a 500×500 pixel image in jpeg format. For jpeg, the b bits for a pixel store $q = 3$ values (red, green, blue), each ranging between 0 and 255. We begin our investigation in Problem 1 by seeing how clustering reduces data storage.

Problem 1 (5 points). Compare the number of bits required to store the original image and the clustered image.

We can state our clustering problem this way. Given n data points $\mathbf{x}_i \in \mathcal{R}^q$, $i = 1, \dots, n$, and given a value of k , we want to find k cluster centers $\mathbf{c}_j \in \mathcal{R}^q$, $j = 1, \dots, k$, that are in some sense optimal and then assign each data point to a cluster. We assign \mathbf{x}_i to cluster \mathcal{C}_j if it is closer to that cluster’s center than it is to any other center. (Break ties in an arbitrary way.) The distance from data point i to its cluster’s center is thus

$$d_i = \min_{j=1, \dots, k} \|\mathbf{x}_i - \mathbf{c}_j\|,$$

and we define the radius of cluster \mathcal{C}_j as

$$r_j = \max_{i: \mathbf{x}_i \in \mathcal{C}_j} d_i.$$



Figure 1: Use clustering algorithms to group the pixels of this image of Charlie, photographed by Timothy O'Leary.

For good clustering, we want each point to be close to one cluster's center. Therefore, we might want to minimize either

$$R = \sum_{j=1}^k r_j^\ell$$

or

$$D = \sum_{i=1}^n d_i^\ell,$$

where $\ell = 1$ or 2 . The variables in the minimization problem are the cluster centers.

Why the Problem Is Hard

In Problem 2, we consider some properties of the functions R and D .

Problem 2 (8 points). For this problem, use the Euclidean norm with $q = 1$ and $\ell = 2$.

(a) If a function is *convex* and bounded below, then any local minimizer is a global minimizer. If not, then an algorithm for minimization might report a local minimizer (a point as good or better than any in its neighborhood) rather than a global one (a point as good or better than any other). Consider the problem with $n = 2$ points and $k = 2$ clusters. Are D and R convex functions?

(b) Are D and R differentiable functions when $n = 2$ and $k = 2$?

(c) Derive a formula for the minimizer of D when $k = 1$ and n is arbitrary.

(d) Suppose we move one of our data points \mathbf{x}_i very far away from the other points, making it an *outlier*. As that point moves further away from the others, what will happen to the cluster centers?

From Problem 2, we know that the minimization problem has some difficult properties, but let's try to compute the cluster centers using a standard optimization algorithm. To get the solution process started, we provide an array of k approximate centers. A common method for finding initial centers is to select k distinct points randomly among the data values, or perhaps to use k extreme values. When comparing algorithms, each should use the same initial centers. Problem 3 investigates our clustering criteria and the behavior of optimization algorithms.

Problem 3 (10 points). Use your favorite optimization function to minimize R with $\ell = 2$ and the Euclidean norm. Use the data of Figure 1, and provide a function to evaluate R . Try $k = 3, 4, 5$.

Also minimize D with the same parameters.

- How does the number of variables increase with k ?
- How does the running time increase with k ?
- Evaluate the results of the four clusterings and justify the criteria that you choose. As one criterion, discuss how the clustered images look in comparison to the original image.
- How might a good value of k be determined experimentally?

Write a function `map_to_cluster` that takes the data values and cluster centers as input and returns the cluster number for each data value and the counts of the number assigned to each cluster. Use this function to generate the clustered image.

To keep the computation time reasonable, determine the cluster centers based on a sample of points in the image rather than using all 250,000 pixels. Choose the 1000 points in columns 210 and 211 in the sample image. Then experiment with other choices of points to study the algorithm's sensitivity to this choice.

The k -Means Algorithm

A general purpose minimization routine is a good tool to have, because it is useful for a wide variety of problems. But sometimes we can develop a better algorithm by taking advantage of special structure in the problem. Here is an algorithm known as the *k-Means Algorithm*. It minimizes neither D nor R , but it iterates by clustering based on the current centers and then moving each center to the centroid of the points in the cluster.

Choose initial centers $\mathbf{c}_1, \dots, \mathbf{c}_k$. Repeat until the centers stop changing:

- For $i = 1, \dots, n$, assign each data point \mathbf{x}_i to the cluster \mathcal{C}_j whose center \mathbf{c}_j is closest to it, breaking ties in an arbitrary way.
- For $j = 1, \dots, k$, recompute the center \mathbf{c}_j to be the mean (centroid) of the points in the cluster:

$$\mathbf{c}_j = \frac{1}{n_j} \sum_{i: \mathbf{x}_i \in \mathcal{C}_j} \mathbf{x}_i,$$

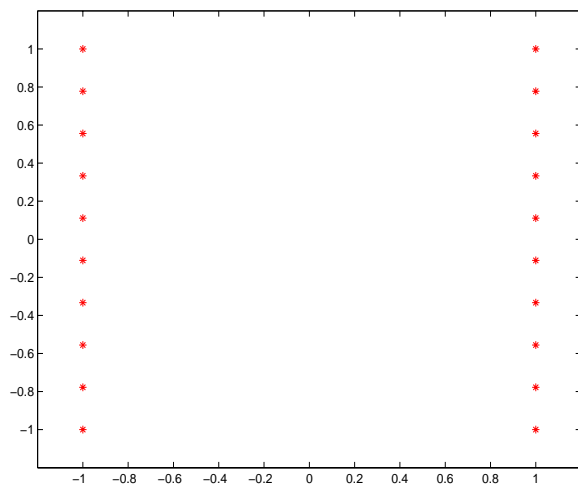


Figure 2: This data set illustrates some of the pitfalls of clustering.

where n_j is the number of datapoints in \mathcal{C}_j .

In Problem 4, we implement this algorithm.

Problem 4 (10 points). Implement the k -Means algorithm and run it with the same data and values of k as Problem 3. Compare its performance to that of the algorithm in Problem 3.

Your implementation for this problem should be rather general: write a function `mycluster` that takes as input the n data values, an initial guess for the k cluster centers, a convergence tolerance, and a maximum number of iterations. The output will be assignments of each datapoint to a cluster, the set of k cluster centers, the number of data values in each cluster, and the radius of each cluster. Use another function to evaluate R or D , given the k current cluster centers.

Pitfalls in Data Clustering

This form of data clustering is quite useful, and the k -means algorithm is very successful in practice. Nevertheless, there are many pitfalls associated with its use. We investigate two of these, dependence of the answer on the starting data and the number of clusters, in Problem 5.

Problem 5 (3 points). Consider the dataset of $n = 20$ datapoints with $q = 2$, shown in Figure 2:

$$(1, -1 + 2j/9), (-1, -1 + 2j/9),$$

for $j = 0, \dots, 9$. Run the k -means algorithm with $k = 2, 3, 4$. Initialize the centers to the first k points in the list

$$(-1, -1), (1, 1), (-1, 1), (1, -1).$$

Display the clustered data. Discuss the effects of choosing the “wrong” value for k .

Then repeat the experiment, initializing the centers to

$$(0, -1 + 2j/(k - 1)),$$

$j = 0, \dots, k - 1$. Note that although the answer is different, it is also a local minimizer of the (nonconvex) function R . Compare with the first set of answers and discuss the difficulty it illustrates with this kind of clustering.

Sensitivities of the clustering to the initial choice of centers and the number of clusters are serious pitfalls. As we see in Problem 6, another serious pitfall arises from the sensitivity of the clustering to variable transformations.

Problem 6 (4 points). Consider the dataset from Problem 5, but multiply the second component of each datapoint by 100. Repeat the clustering experiments, applying the same transformation to the initial centers. Discuss why coordinate scaling is important in clustering algorithms.

Tools:

In Problem 1, note that it takes $\log_2 256 = 8$ bits to store a number that ranges between 0 and 255.

For Problem 2, a real analysis text will provide information on convexity, differentiability, and minimization. A function $f(\mathbf{z})$ is convex over some convex domain if every one of its secants (line segments connecting two points on its graph) is above the graph.

Problem 3 relies on a program for unconstrained minimization, such as Matlab's `fminunc`. Nash and Sofer¹ discuss various algorithms for solving minimization problems.

Many sources present the k -means algorithm used in Problem 4. as well as other approaches to clustering^{2,3,4,5}.

Davidson⁶ gives a nice discussion of the pitfalls we illustrate in Problems 5 and 6 as well as many others.

There are many codes that implement variants of the k -means algorithm; see, for example, the software of Guan⁷.

References:

1. Stephen G. Nash and Ariela Sofer, *Linear and Nonlinear Programming*, McGraw-Hill, 1996.
2. A. K. Jain, M. N. Murty, and P. J. Flynn, "Data Clustering: A Review," *ACM Computing Surveys* 31 (1999) 264-323.
3. Julius T. Tou and Rafael C. Gonzales, *Pattern Recognition Principles*, Addison-Wesley Publishing Co., 1974.
4. Anil Jain and Richard C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.
5. Keinosuke Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd Edition, Academic Press, 1990.
6. Ian Davidson, "Understanding K-Means Non-hierarchical Clustering," Computer Science Department, State University of New York, Albany, Technical Report 02-2. <http://www.cs.albany.edu/~davidson/courses/CSI635/UnderstandingK-MeansClustering.pdf>
7. Yuqian Guan, "Gmeans- Clustering with First Variation and Splitting," <http://www.cs.utexas.edu/users/yguan/datamining/gmeans.html>