IE 529: Stats of Big Data and Clustering
Computational Assignment 2
Due: December 15, 2016

This assignment is worth **50 points**, so can be considered a combination computational assignment and final.

Please submit your report as a pdf file, with additional attachments as necessary included in the pdf (e.g. use a listings environment to include code in latex). **If your name is not included in the pdf, you will not receive credit.** This is a coding assignment; so code up the algorithms yourself.

I. Write a basic implementation of Lloyd's algorithm for a large set of data in $\mathbf{R}^d$ (i.e., to find a Voronoi partition and a set of K centroids). Your algorithm should attempt to solve the classic *K-means* problem, for any user-selected positive integer value K.

- Assume the input data is given to you in a matrix $X \in \mathbf{R}^{N\times d}$, where each row in X corresponds to an observation of a d-dimensional point. That is, your inputs will be a user-provided matrix X and the number of clusters K.
- Your outputs should be (i) a matrix $Y \in \mathbf{R}^{K\times d}$, where row j contains the centroid of the j^{th} partition; (ii) a cluster index vector $C \in \{1, 2, \dots K\}^N$, where $C(i) = j$ indicates that the i^{th} row of X (or the i^{th} observation x_i) belongs to cluster j; and (iii) the final objective function value, i.e., the best distortion, or averaged distance value, D obtained.
- Convergence may be based on a norm-based comparison of the iterates of Y, i.e., $\|Y_{p+1} - Y_p\| < tol$, OR on a norm-based comparison of the distortion achieved $\|D_{p+1} - D_p\| < tol$. Choose *tol* to be (1) 1×10^{-5}, and (2) a different value of your choice, with your reasoning provided. Please state clearly what convergence test you used.

II. Write a basic implementation of the "GreedyKCenters" algorithm (described in the reading by S. Har-Peled, and discussed in class). Your algorithm should attempt to solve the classic *K-centers* problem, for any user-selected positive integer value K. The underlying distance function used in your algorithm should be the Euclidean distance, and your objective should be to *minimize* the *maximum* distance between any observation $x_i \in X$ and it's closest center $c_j \in Q$, i.e., to find Q giving

$$\min_{Q\subset X,\, |Q|=K} \left(\max_{x_i\in X} (\min_{c_j\in Q} \|x_i - c_j\|_2) \right) \tag{1}$$

- You can again assume the input data is given to you as a matrix $X \in \mathbf{R}^{N\times d}$, and a positive integer K, as in **I.**
- Your output should be a matrix $Q \in \mathbf{R}^{K\times d}$ containing the final K d-dimensional centers, and the objective function value, i.e., the final $\max_{x_i\in X}(\min_{c_j\in Q} \|x_i - c_j\|_2)$ obtained.
- You do not need a convergence criteria for this algorithm.

III. Write a basic implementation of the single-swap heuristic for which you try to improve the solution to the *K-centers* problem in **II** by a implementing a series of "swaps". If Q is your current set of centers, and you make a single swap, giving $Q_{new} = Q - \{c_j\} \cup \{o\}$, then you should replace Q with Q_{new} whenever the new objective value, that is the computed value for (1), is reduced by

a factor of $(1-\tau)$. When there is no swap that improves the solution by this factor, the local search stops. Let $\tau = 0.05$.

IV. Write an implementation of the Spectral Clustering algorithm, using either basic unnormalized clustering, or normalized clustering (see the reading by Luxborg for details). Assume you are given a matrix of data X, and you would like to identify some number of clusters, K, which may be user provided. Your outputs should be

1. a similarity or weighted adjacency matrix, W;
2. a matrix U containing the first K eigenvectors of the Laplacian, L (or the generalized eigenvectors for the normalized case);
3. a cluster index vector $C \in \{1, 2, ...K\}^N$, where $C(i) = j$ indicates that the ith row of U, y_i belongs to cluster j.

You should construct the similarity scores/weighting matrix, W, using the Euclidean distance between points, pairwise.

V. Write a basic implementation of the Expectation Maximization algorithm, to be used for basic clustering. Please use a mixture of k Gaussian distributions. Recall you cannot use a built-in implementation of the algorithm such as *gmdistribution.fit()* (but you could check you answer with this).

For your convenience here is some psuedo code (you don't have to follow this exactly).

EM Algorithm:

Randomly select k data points to serve as the initial means.

Initialize the variances for each k using the covariance of the data.

Initialize a matrix W to hold the probabilities that each data point belongs to each cluster.

Assign equal prior probabilities to each cluster in a vector ϕ.

Repeat until convergence:

Expectation Step:

Using the estimated means and variances for each cluster, evaluate the Gaussian distributions for all data points.

Multiply each evaluated Gaussian dataset by the corresponding prior probability (ϕ).

Divide the weighted probabilities by the sum of weighted probabilities for each cluster to update W.

Maximization Step:

Calculate the prior probability for cluster, that is, update ϕ using the mean of the corresponding column of W.

Update the mean for each cluster by taking the weighted averages of all the data points.

Update the covariance matrix for each cluster by taking the weighted average of the covariance for each data point.

The following should be submitted in the report, implementing the above algorithms on **both** provided datasets, bigClusteringData.txt (or .mat) and clustering.csv (or .mat).

1. For each of the algorithms, present the output (clustering) results for the test data for each value of $K = 3, 4, ..., 10$ for the larger dataset and $K = 2, 3, 4, 5$ for the smaller dataset. Please only use the swap algorithm on the smaller dataset.

   (a) a plot of D (distance objective) versus K,

   (b) a scatter plot of the data with the centroids highlighted in a different color, size and shape with the corresponding points colored similarly, and

   (c) the cluster index set C (as it makes sense). If the dataset is too big, please give at least an excerpt of C (e.g. first 20 entries or so).

   For (b), please turn in plots for $K = 3, 5, 7, 9$ and $K = 2, 4$ for the larger and smaller datasets, respectively.

   For the K-means algorithm (and the Spectral Clustering Algorithm since it should call your K-means algorithm) you will need to consider a series of initializations (with a minimum of 10), and then select the 'best' results to report, for each K. State why these are 'best', and note how many different initializations you tried.

2. Discuss how many 'natural' clusters you think the test data contains, and why.

3. For the K-means algorithm, describe how you tested for convergence and why you choose this as a test.

4. Provide a short analysis of the computational effort you found was required for each of the algorithms. Clearly state what measure you used to evaluate computational effort, and how you observed this. Briefly discuss how these computational efforts compare to your expectations.

5. Discuss whether your 'Swap' algorithm improved the original solution found with your K-Centers algorithm (Since you were only asked to do swap on the smaller dataset, please ignore this question for the larger dataset). For each current set of centers Q, how many points in your data set did you evaluate to determine if there was a swap that would improve the outcome from your existing solution? If not all points, describe how/why you selected the points you tested. Discuss the computational effort needed to run your 'Swap' algorithm and how you could further improve the performance of the algorithm.

6. Write a brief summary paragraph of your findings for the test data set.

7. Include your source code, namely .m files for Matlab routines or .py for Python scripts in your pdf (e.g. using a listings or coding package).