Problem Set 1: Cluster Analysis

Due February 16

Assignment:

1. Undergraduates and graduate students (non-math): Do any four problems except that at least one must be from 1, 2,3, and 4.

2. Math graduate students: Do any four problems.

You are encouraged to work other students but turn in your own paper. Please—no electronic copies. Use the `Python` scripts posted on the Moodle page in answering the questions. We'll post the code for problems 1-4 on the github site.

1. One aspect of cluster analysis that was not addressed in detail in class is the assessment of a particular clustering algorithm. While the objective function is a helpful statistic, a measure of intra-cluster variability *for each cluster* provides more information.

   One example of a measure of intra-cluster variability is the mean Euclidean distance between members of a cluster (averaged over all pairs that may be formed from the cluster members). The mathematical definition is

   $$f_1(C) = \binom{n}{2}^{-1} \sum_{\mathbf{x}_i, \mathbf{x}_j \in C} d_E(\mathbf{x}_i, \mathbf{x}_j), \tag{1}$$

   where n is the number of members belonging to cluster C and $\binom{n}{2}$ is the number of unique sets that may be constructed when choosing 2 elements from n. Note that if there are k clusters, then one may compute $f_1(C_1), f_1(C_2), \ldots, f_1(C_k)$.

   Another measure, call it f_2, can be defined by replacing $d_E(\mathbf{x}_i, \mathbf{x}_j)$ with the cosine distance between $\mathbf{x}_i$ and $\mathbf{x}_j$ (defined in equation 3 below).

   A third measure of intra-cluster variability is the mean distance between each member and the cluster centroid $\overline{\mathbf{x}}$:

   $$f_3(C) = n^{-1} \sum_{\mathbf{x}_i \in C} d_E(\mathbf{x}_i, \overline{\mathbf{x}}). \tag{2}$$

   The objective of this exercise is to evaluate intra-cluster variability by computing and summarizing f_1, f_2, or f_3.

   (a) Write `Python` code (preferably as a `Python` function) that computes one of f_1, f_2, or f_3. Modify `clusterAnaysis.py` so that your function is called at the completion of the k-means algorithm. Paste the code into your homework document and identify it.

(b) Carry out clustering for some configuration of number of clusters and number of stocks of your choice. Compute the sum of the intra-cluster variability measure over all clusters as a single measure of clustering effectiveness, say $\sum_C f_k(C)$, where k is the number of cluster. Make a table tabulating the cluster label $(1, 2, \ldots, k)$, the number of members in the cluster, and intra-cluster variability for the cluster. Report the number of stocks and the series length (number of observations for each stock).

(c) Apply the intra-cluster variability measure when forming 3, 4, and 5 clusters[1]. Use the same starting configuration for each execution of the k-means algorithm (except for number of clusters). Report the sum of the intra-cluster variability measures for 3, 4, and 5 clusters.

(d) Theoretically, how should the mean distances from each member to the cluster centroid vary with number of clusters? If the aim of cluster analysis is to find a set of stocks each of which is to be used to forecast itself (that is, to predict future values), and the other stocks in the set, then is small intra-cluster variability desirable? Are there (potentially) any undesirable aspects to forcing intra-cluster variability to be small? Explain your reasoning.

2. Change the objective function in `clusterAnaysis.py` so that there is an option to minimize the sum of the cosine distances between the series of stock values and the centroids. Cosine distance can be computed as

$$d(\mathbf{x}, \overline{\mathbf{x}}) = \frac{1 - r(\mathbf{x}, \overline{\mathbf{x}})}{2}, \tag{3}$$

where $r(\mathbf{x}, \overline{\mathbf{x}})$ is the Pearson's correlation between the vector $\mathbf{x}$ and the cluster mean vector $\overline{\mathbf{x}}$. The `Numpy` function `corrcoef` will compute the correlation between two vectors (or the rows of a matrix), thereby making the calculation of $d(\mathbf{x}, \overline{\mathbf{x}})$ very easy to implement.

Two changes must be made: the cosine distance function must be used to determine observation to centroid distance and the objective function must be changed from Euclidean distance between the sum of the cluster centroid-to-member distances to the sum of all possible cosine distances.

Using cosine distance, form 3, 4, and then 5 clusters. Use the same starting configuration for each execution of the k-means algorithm. For one choice of number of clusters, provide two plots showing the k cluster centroids plotted against day (one for Euclidean distance and the second for cosine distance).

3. Change the objective function in `clusterAnaysis.py` so that the k-means algorithm minimizes the sum of the Hamming distances between the series of inter-day price *changes* and the cluster centroid.

First, replace the standardized prices with the binary outcomes

$$b(y_i) = \begin{cases} 1, & \texttt{if } x_{i+1} > x_i \\ 0, & \texttt{if } x_{i+1} \leq x_i \end{cases}, i = 1, 2, \ldots, n-1. \tag{4}$$

[1] You may set some other number of clusters if you like.

Note that there will be one fewer binary outcomes than standardized prices for a particular stock. It's probably best to replace the elements in the matrix $\mathbf{X}$ immediately after it is computed in the `getData` function since the observations are arranged in chronological order. Do not compute Z. The function ought to return the matrix of binary outcomes instead of Z.

Compute the cluster centroid as a series of Hamming distances means (averaged across the cluster members). Assess the results of the change by by producing two plots showing the cluster centroids plotted against day, one generated using for Hamming distance and the second generated using the usual Euclidean (or cosine) distance.

4. Change the distance function from Euclidean distance to a weighted Euclidean distance that places the most weight on the most recent observations. Use an exponential weighting function

$$w_i = \alpha(1-\alpha)^{n-i}, i = 1, 2, \ldots, n, \tag{5}$$

where nth observation is the most recent observation and $0 < \alpha < 1$. A choice of α between .05 and .2 seems reasonable.

   (a) Create plots of the cluster centroids after initialization of the centroids (and before) executing the k-means algorithm and then again after completion of the algorithm. Note: unless the objective function is modified to reflect the new distance, don't expect the objective function to decrease monotonically with iteration.

   (b) Verify computationally that $\sum_{i=1}^{n} w_i \approx 1$. In other words, compute the sum of the w_i's. Provide the code that computes the sum and report the sum.

   (c) (Math students) Verify mathematically that $\sum_{i=0}^{\infty} \alpha(1-\alpha)^i = 1$.

5. Consider two vectors $\mathbf{x}$ and $\mathbf{y} = c\mathbf{x}$, for $c \neq 0$. Show that the cosine of the angle between the two vectors is 1, and hence, the cosine distance between the vectors is 0.

6. Assume that $\mathbf{x}_i$, $\mathbf{y}$, and $\overline{\mathbf{x}}_i$ are $m \times 1$ vectors and that $\overline{\mathbf{x}}_i$ is the vector mean of $\mathbf{x}_1, \ldots, \mathbf{x}_n$. Show that

$$\sum_{i=1}^{n} ||\mathbf{x}_i - \mathbf{y}||^2 = \sum_{i=1}^{n} ||\mathbf{x}_i - \overline{\mathbf{x}}||^2 + n||\mathbf{y} - \overline{\mathbf{x}}||^2, \tag{6}$$

where $||\mathbf{x}|| = (\mathbf{x}^T\mathbf{x})^{1/2}$ is the L_2 norm of the $m \times 1$ vector $\mathbf{x}$.

7. Let

$$\underset{n\times 1}{\mathbf{j}} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

denote the *summation* vector of length n.

   (a) Show that for any n-vector $\mathbf{x}$, $n^{-1}\mathbf{j}^T\mathbf{x} = \overline{x}$ (the sample of the observations composing $\mathbf{x}$).

(b) Suppose that the p-vector of sample means $\overline{\mathbf{x}}$ is to be computed from the data matrix $\underset{n\times p}{\mathbf{X}}$. Show that

$$\overline{\mathbf{x}}^T = (\mathbf{j}^T\mathbf{j})^{-1}\underset{1\times n}{\mathbf{j}^T}\underset{n\times p}{\mathbf{X}}. \tag{7}$$

(c) Extra credit: show that $\overline{\mathbf{x}}$ minimizes $(\mathbf{X}-\mathbf{j}\mu^T)^T(\mathbf{X}-\mathbf{j}\mu^T)$ with respect to μ.