IE406 Applied Machine Learning:
Spring 2022

Assignment I
(Due Date: 11:59 PM on April 12, Tuesday)

The objective of this assignment is to implement the *K*-means algorithm on digit data. A template has been prepared (*kmeans_template.py*). Your job is to fill in the relevant functions as described below.

1. The Data

The data are held in two files: one for the inputs and one for the output. The file *digit.txt* contains the inputs: namely, 1000 observations of 157 pixels (a randomly chosen subset of the original 784) from images containing handwritten digits. The file *labels.txt* contains the true digit label (either 1, 3, 5, or 7). The Python template handles reading in the data; just make sure that all the files live in the same directory as your code. The labels correspond to the digit file, so the first line of *labels.txt* is the label for the digit in the first line of *digit.txt*.

2. The algorithm

Your algorithm will be implemented as follows:

(I) Initialize *k* starting centers by randomly choosing *k* points from your data set. You should implement this in the *initialize* function.

(II) Assign each data point to the cluster associated with the nearest of the *k* center points. Implement this by filling in the *assign* function.

(III) Re-calculate the centers as the mean vector of each cluster from (II). Implement this by filling in the *update* function.

(IV) Repeat steps (II) and (III) until convergence. This wrapping is already done for you in the loop function, which lives inside the *cluster* function.

3. Within group sum of squares

The goal of clustering can be thought of as minimizing the variation within groups and consequently maximizing the variation between groups. A good model has low sum of squares within each group. We define sum of squares in the traditional way. Let μ_k be the mean of the observations x_i in k^{th} cluster. Then the within group sum of squares for k^{th} cluster is defined as:

$$SS(k) = \sum_{i \in k} |x_i - \mu_k|^2$$

Please note that the term $|x_i - \mu_k|$ is the Euclidean distance between x_i and μ_k, and therefore should be calculated as $|x_i - \mu_k| = \sqrt{\sum_{j=1}^{d}(x_{ij} - \mu_{kj})^2}$, where *d* is the number of dimensions. Please note that that term is squared in SS(*k*). If there are *K* clusters total, then the "sum of within group sum of squares" is just the sum of all *K* of these individual SS(*k*) terms. In the code, the within-group sum of squares is referred to (for brevity) as the distortion. Your job is to fill in the function *compute_distortion*.

4. Mistake Rate

Given that we know the actual assignment labels for each data point, we can attempt to analyze how well the clustering recovered the true labels. For k^{th} cluster, let its assignment be whatever the majority vote is for that cluster. If there is a tie, just choose the digit that is smaller numerically as the majority vote.

For example, if for one cluster we had 270 observations labeled one, 50 labeled three, 9 labeled five, and 0 labeled seven then that cluster will be assigned value one and had 50 + 9 + 0 = 59 mistakes. If we add up the total number of "mistakes" for each cluster and divide by the total number of observations (1000) we will get our total mistake rate, between 0 and 1.

Your job here is to fill in the function *label_clusters*, which implements the label assignment. Once you have implemented this, the function *compute_mistake_rate* (already written) will compute the mistake rate for you.

5. Putting it all together

Once you have filled in all the functions, you can run the Python script from the command line with Python *kmeans_template.py*. The script will loop over the values k = 1, 2, …,10. For each value of k, it will randomly initialize 5 times using your initialization scheme, and keep the results from the run that gave the lowest within-group sum of squares. For the best run for each k, it will compute the mistake rate using your cluster labeling function.

The code will output a figure named *kmeans.png*. The figure will have two plots. The top one shows within-group sum of squares as a function of k. The bottom one shows the mistake rate, also as a function of k.

Write a few sentences with your thoughts on these results. Are they what you expected? Did you think that within-group sum of squares and mistake rate would go up or decrease as k increased? Did the plots confirm your expectations?