CS39000-DM0 Homework 4

Due date: Wednesday March 29, 11:59pm

In this programming assignment you will implement the k-means clustering algorithm and apply it on the *Yelp* dataset. Instructions below detail how to turn in your code and assignment to Blackboard.

Note: You can use any standard python libraries available on the CS computers (e.g., numpy) but don't use methods from machine learning libraries (e.g., scikit-learn).

Dataset Details

Use the `yelp3.csv` dataset. It contains 21,091 rows and 19 attributes.

Algorithm Details

Implement the k-means algorithm and apply it to the features **X** defined below.

Features: Consider the 4 attributes, `latitude`, `longitude`, `reviewCount`, `checkins`, in `yelp3.csv` for **X**.

Distance: Use Euclidean distance unless otherwise specified.

Score function: Use within-cluster sum of squared error (where r_k is the centroid of cluster C_k):

$$wc(\mathbf{C}) = \sum_{k=1}^{K} \sum_{x(i) \in C_k} d(x(i), r_k)^2$$

Code specification

Your python script should take four arguments as input.

1. *dataFilename*: corresponds to the `yelp3.csv` dataset that should be clustered by k-means algorithm.
2. *K*: the value of k to use when clustering.
3. *clustering option*: takes one of the following five values, 1 (use the four original attributs for clustering, which corresponds to Q2(i)), 2 (apply a log transform to `reviewCount` and `checkins`, which corresponds to Q2(ii)), 3 (use the standardized four attributes for clustering, which corresponds to Q2(iii), 4 (use the four original attributes and Manhattan distance for clustering, which corresponds to Q2(iv)), and 5 (use 3% random sample of data for clustering, which corresponds to Q2(v)).
4. *plot option*: takes one of the following three values, no (no plot), 1 (plot the clusters with their centroids using `latitude` and `longitude`), and 2 (plot the clusters with their centroids using `reviewCount` and `checkins`).

Your code should read in the dataset from the csv file, cluster the examples using the specified value of k, and output the within-cluster sum of squared error and cluster centroids. For the centroid of each cluster report the values for each of the four attributes {`latitude`, `longitude`, `reviewCount`, `checkins`). In addition, if *plot option* = 1 or 2 is specified, the code should draw a plot of the clusters with their centroids as well.

Name your file kmeans.py. The input and output should look like this:

```
$ python kmeans.py yelp3.csv K 1 no
WC-SSE=15.2179
Centroid1=[49.00895,8.39655,12,3]
...
CentroidK=[33.33548605,-111.7714182,9,97]
```

Assignment

1. Implement the k-means clustering algorithm in python. (15 pts)
   We will run several tests on your code to assess it on different samples. Your python script should strictly follow the code specification that is described above.

2. K-means analysis (25 pts)

   In this problem, you will do some experiments using your k-means algorithm on the `yelp3.csv` data.

   (i) Cluster the Yelp3 data using k-means.
   - Use a random set of examples as the initial centroids.
   - Use values of $K = [3, 6, 9, 12, 24, 48]$.
   - Plot the within-cluster sum of squares (wc) as a function of K.
   - Choose an appropriate K from the plot and argue why you choose this particular K.
   - For the chosen value of K, plot the clusters with their centroids in two ways: first using `latitude` vs. `longitude` and second using `reviewCount`, `checkins`. Discuss whether any patterns are visible.

   (ii) Do a log transform of `reviewCount`, `checkins`. Describe how you expect the transformation to change the clustering results. Then repeat the analysis (i). Discuss any differences in the results.

   (iii) Transform the four original attributes so that each attribute has $mean = 0$ and $stdev = 1$. You can do this with the numpy functions, `numpy.mean()` and `numpy.std()` (i.e., subtract mean, divide by stdev). Describe how you expect the transformation to change the clustering results. Then repeat the analysis (i). Discuss any differences in the results.

   (iv) Use Manhattan distance instead of Euclidean distance in the algorithm. Describe how you expect the change in the clustering results. Then repeat the analysis (i). Discuss any differences in the results.

   (v) Take a 3% random sample of the data. Describe how you expect the downsampling to change the clustering results. Then run the analysis (i) five times and report the average performance. Specially, you should use a single random 3% sample of the data. Then run 5 trials where you start k-means from different

random choices of the initial centroids. Report the average wc when you plot wc vs. K. For your chosen K, determine which trial had performance closest to the reported average. Plot the centroids from that trial. Discuss any differences in the results and comment on the variability you observe.

3. Improve the score function. (10 pts)
   To evaluate the resulting clustering, it is not sufficient to measure only the within-cluster sum of squares (wc) that you used for Q2. It is also desired to have each cluster separate from others as much as possible. To improve the resulting clustering, define your own score function that takes into account not only the compactness of the clusters but also the separation of the clusters. Write a formal mathematical expression of your score function and explain why you think your score function is better than the within-cluster sum of squares. Also, using the same configuration as Q2(i), plot the results of your score function for $K = [3, 6, 9, 12, 24, 48]$, and compare the results to Q2(i).

Submission Instructions:

You should upload your work to Blackboard, as a ZIP file. Your submission should include the following files:

(a) The source code in python, `kmeans.py`.

(b) Solutions to Q2 and Q3 in **.pdf** format.

(c) A README file containing your name, instructions to run your code and anything you would like us to know about your program (like errors, special conditions etc.).