ASSIGNMENT 5

Assignment 5 is described in the next few pages. Before that, here are some ground rules that everyone needs to follow:

- **Do not ask questions through email**. There is a Piazza discussion board at https://piazza.com/, where you can post questions, and the instructor and TA or even other students can answer them. This will prevent us from answering the same questions individually.
Again - absolutely no emails to the instructor or TA about this homework! All questions are answered on discussion board only.
You will receive an email from Piazza with details on how to join.

- The deadline is **fixed and final.** There will be no extensions for any reason. You should plan your time accordingly. In the past, I have received many requests at the last hour for extensions through email. From now on, these will simply be ignored. If you can't finish on time, you can submit whatever you have done.

- You can code in Java or Python, **but cannot use R or any other package that has k-means built in it for part 1 of this assignment**. Again - you have to write your own code for k-means.

- There are two parts to this. Requirements of what to turn in are clearly mentioned. Keep them in separate folders titled PartI and PartII.

- The TA should be able to run your code from command line as explained in both the parts.

- Part I is worth 200 points and Part II is worth 300 points

Part I

In the first part of this assignment, you have to implement the k-means algorithm using **Euclidean distance** on a dataset with two attributes. The dataset is available for download at: http://www.utdallas.edu/~axn112530/cs6375/assignment5/test_data.txt

The algorithm was discussed in class and you have to will have to write code in Java or Python only. You cannot use any pre-built package in R language or any other API. The parameter to the program should be the number of clusters (k). The TA should be able to run your code as follows:

./k-means <numberOfClusters> <input-file-name> <output-file-name>

Initialization:

Based on the input parameter(k), you should randomly select k points as centroids.

Termination Condition:

The usual termination condition in k-means is when the centroids no longer more. In this assignment, you should also limit your update step to a maximum of 25 iterations.

Input:

The input file will be specified by the parameter <input-file-name>.

Output:

Your code should also output to a file called specified by the parameter <output-file-name> and should be in the format:

<cluster-id> <List of points ids separated by comma>

For example,
1 2, 4, 7, 10

Validation:

The usual method of evaluating the goodness of clustering will be used. It is the Sum of Squared Error function, defined as:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

where m_i is the centroid of the i^{th} cluster.

You should write a method to compute the SSE of clustering i.e. it should be separate from main method.

How many times to run your code:

You should run your code at least 5 times with different values of k and include the details of SSE in your README file.

What to Turn In for Part I :
(1) The source code to finish this task.
(2) A Readme file contains of the 5 runs and their SSE values.

* Idea and data courtesy of Brian Kulis, Ohio State University *

Part II

Tweets Clustering using k-means

Twitter provides a service for posting short messages. In practice, many of the tweets are very similar to each other and can be clustered together. By clustering similar tweets together, we can generate a more concise and organized representation of the raw tweets, which will be very useful for many Twitter-based applications (e.g., truth discovery, trend analysis, search ranking, etc.)

In this assignment, you will learn how to cluster tweets by utilizing Jaccard Distance metric and K-means clustering algorithm.

Objectives:

- Compute the similarity between tweets using the Jaccard Distance metric.
- Cluster tweets using the K-means clustering algorithm.

Introduction to Jaccard Distance:
The Jaccard distance, which measures dissimilarity between two sample sets (A and B). It is defined as the difference of the sizes of the union and the intersection of two sets divided by the size of the union of the sets.

$$Dist(A,B) = 1 - \frac{|A \bigcap B|}{|A \bigcup B|} = \frac{|A \bigcup B| - |A \bigcap B|}{|A \bigcup B|}$$

For example, consider the following tweets:

Tweet A: the long march

Tweet B: ides of march

$|A \cap B| = 1$ and $|A \cup B| = 5$, therefore the distance is $1 - (1/5)$

In this assignment, a tweet can be considered as an unordered set of words such as {a,b,c}. By "unordered", we mean that {a,b,c}={b,a,c}={a,c,b}=...
A Jaccard Distance Dist(A, B) between tweet A and B has the following properties:

- It is small if tweet A and B are similar.
- It is large if they are not similar.
- It is 0 if they are the same.
- It is 1 if they are completely different (i.e., no overlapping words).

Here is the reference for more details about Jaccard Distance: Jaccard Distance

Hint: Note that the tweets do not have the numerical coordinates in Euclidean space, you might want to think of a sensible way to compute the "centroid" of a tweet cluster. *This could be the tweet having minimum distance to all of the other tweets in a cluster.*

Exercises:
Implement the tweet clustering function using the Jaccard Distance metric and K-means clustering algorithm to cluster redundant/repeated tweets into the same cluster. You are expected to do the K-means implementation by yourself, so please do **not** use any external library that has K-means implementation in your code.

Note that while the K-means algorithm is proved to converge, the algorithm is sensitive to the k initial selected cluster centroids (i.e., seeds) and the clustering result is not necessarily optimal on a random selection of seeds. In this assignment, we provide you with a list of K initial centroids that have been tested to generate good results.

Inputs to your K-means Algorithm:

(1) The number of clusters K (default to K=25).

(2) A real world dataset sampled from Twitter during the Boston Marathon Bombing event in April 2013 that contains 251 tweets. The tweet dataset is in JSON format and can be downloaded from http://www.utdallas.edu/~axn112530/cs6375/assignment5/Tweets.json

(3) The list of initial centroids can be downloaded from:
http://www.utdallas.edu/~axn112530/cs6375/assignment5/InitialSeeds.txt

Note that each element in this list is the tweet ID (i.e., the id field in JSON format) of the tweet in the dataset.

How to run your code:

The TA should be able to run your code as follows:

./tweets-k-means <numberOfClusters> <initialSeedsFile> <TweetsDataFile> <outputFile>

<numberOfClusters> is the number of Clusters and
<initialSeedsFile> is a text file containing value of the initial seed data.
<TweetsDataFile> is the data file containing Tweets in JSON format
<outputFile> is the output file explained below.

The default value for <numberOfClusters> should be 25 and for the <initialSeeds> to be the file provided to you containing number of seeds.

Output:

Your code should also output to a file called tweets-k-means-output.txt. It should contain following:

<cluster-id> <List of tweet ids separated by comma>

For example,
1 323906397735641088, 900906397735641088, ..

Validation:

The same method as described in part I will be used. The distance metric will be Jaccard.

What to Turn In for Part II :
(1) A result file that contains the clustering results. Each line represents a cluster. It is in the form of *cluster_id: a list of tweet IDs that belongs to this cluster*
(2) The source code to finish this task.
(3) A Readme file that contains the SSE value for k = 25.

* Idea and data courtesy of Dong Wang, University of Notre Dame *