INF 553 – Homework #4

Hierarchical Clustering with CURE

Due: April/17/2019, Wednesday, 11:59pm, to Blackboard

In this homework, you are asked to implement the CURE algorithm in Python, named "cure.py", for finding k clusters via hierarchical clustering. Your algorithm should be executed as follows:

```
cure.py <k> <sample> <data-set> <n> <alpha>
```

where <k> is the number of clusters (obtained from the dendrogram), the <sample> file contains a set of samples for running the hierarchical clustering, <data-set> contains the entire dataset, n is the number of representatives in each cluster and alpha is the fraction of distance the representatives are moved towards the centroid, as described in class

Note: In CURE, you should find n representatives which are the points that are as far as possible for each cluster. If the number of points in one cluster is less than n, then all the points in this cluster should be considered as representatives. Your algorithm should assign each point in the <data-set> to the cluster with the closest representatives to the point.

Evaluation: For the evaluation purpose, the data set input to your algorithm also contains cluster label for each data point. These cluster labels form a gold standard for clustering the data set to produce a specific number of clusters. Note that you can not use the label in the clustering process. Instead, use them in evaluating the performance of your algorithm. The performance is measured by precision and recall of correct pairs. A pair of two data points x and y is correct if they belong to the same cluster according to the cluster label. Recall that precision and recall are discussed in LSH lectures. Precision is the percentage of pairs discovered by the algorithm that are correct, while recall is the percentage of correct pairs that are discovered by the algorithm.

As an example, consider five data points: 1, 2, 3, 4, 5. Suppose there are 2 clusters: {1, 2, 3} and {4, 5} according to the algorithm, while there are different 2 clusters: {1, 2} and {3, 4, 5} according to the gold standard. In this case, the algorithm discovers four pairs: (1, 2), (1, 3), (2, 3), and (4, 5), while the gold standard has the pairs: (1, 2), (3, 4), (3, 5), and (4, 5). As such, the precision is 2/4 since only (1, 2) and (4, 5) discovered by the algorithm are correct, among the 4 discovered. Recall is also 2/4, since only 2 correct pairs were discovered among the total 4 in the gold standard.

Input data format:

The data set contains a list of data points, one point per line. For each data point, it gives the value of the point at each dimension, followed by the cluster label of the point. You can assume that the sample will contain a subset of the data. The data contains 150 iris plants (https://archive.ics.uci.edu/ml/datasets/Iris). For each plant, it lists its sepal and petal length and width in centimeter, and also its type (e.g., setosa, see below).

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
…
```

Output format:

Print the results. Your algorithm should output k clusters, with each cluster contains a set of data points. Data point are numbered by their positions they appear in the data file: the first data point is numbered 0, second 1, and so on. The data points in the clusters are output in the ascending order of their numbers.

For example, here is an example print output.

```
Cluster 1: [3, 10, 13, …]
Cluster 2: [8, 52, 87, 88, …]
Cluster 3: [100, 105, …]
```

Your algorithm also prints the accuracy of the discovery. For example,

```
Precision = 0.8, recall = 0.5
```