INF 553(Spring 2016)

Assignment 4: K-Means Clustering

In this assignment you will implement the k-means clustering algorithm. You will use it to cluster a data set. For this assignment, we will use a data set from the UC Irvine Machine Learning Repository.

Write your own code!

For this assignment to be an effective learning experience, you must write your own code! I emphasize this point because you will be able to find Python implementations of most or perhaps even all of the required functions on the web. Please do not look for or at any such code! **Also do not share code with other students in the class!!**

Format of data file

The data file that you are clustering is a database related to iris plants. A complete description can be found here: https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.names.

- You will use the file **iris.data** as your input file.
- Each line of the csv file looks something like this: 5.1,3.5,1.4,0.2, Iris-setosa
- It consists of four floating point values and a text label for the type of iris plant.
- The four floating point **attributes** correspond to:
    1. sepal length in cm
    2. sepal width in cm
    3. petal length in cm
    4. petal width in cm
- The string attribute is the iris **class**, one of the following:
    1. Iris Setosa
    2. Iris Versicolour
    3. Iris Virginica

Submission Details

You will turn in your code names as: **lastname_firstname_clustering.py**

Program Description

1. Read the data from the file. Use only the floating point values for the clustering. Don't discard the class information. While you can't use it for clustering, you will need it later for assigning names to the clusters and for checking the accuracy of the clusters.
2. **Apply the k-means algorithm to find clusters.** There are 3 natural clusters in the case of the iris data. (See below for more information on k-means).
3. Use **Euclidean distance** as your distance measure.
4. **Assign each final cluster a name** by choosing the most frequently occurring class label of the examples in the cluster.
5. **Find the number of data points that were put in clusters in which they didn't belong.** (based on having a different class label than the cluster name).

k-means algorithm:
Given k initial points that will act as the centroids of the clusters for the first iteration, you will run the standard k-means clustering algorithm that we discussed in class.

- For each point, place it in the cluster whose current centroid it is nearest to.
- After all points are assigned, update the locations of centroids of the k clusters.
- Repeat for the specified number of iterations.

Running your code

python lastname_firstname_clustering.py dataFileName k iter initialPoints

- **dataFileName:** It is the name of the data file to be clustered.
- **k**: An integer representing the number of clusters (three in the case of the iris data set).
- **Iter**: It is the number of iterations for the k-means clustering to run.
- **initialPoints**: It is the name of a file that contains a list of data points that are to be used as the starting centroids for each cluster.

Output of your program

The program will produce output of the form:

Cluster <clustername1>
(List of points in that cluster, one per line)

Cluster <clustername2>
(List of points in that cluster, one per line)

Cluster <clustername3>
(List of points in that cluster, one per line)

Number of points assigned to wrong cluster:
<number of points>

Note: Please follow the output format specified above. Few points will be deducted if it is not the same as shown above.

Testing your code

The command to execute is:
python <lastname>_<firstname>_clustering.py iris.data 3 10 initialPoints
Find the sample output in:
sampleOutput.txt