1 K-means Clustering

1.1 Theory

K-means is a clustering algorithm. Clustering algorithms are unsupervised techniques for sub-dividing a larger data set into smaller groups. The term "unsupervised" indicates that the data do not originate from clearly defined groups that can be used to label them a priori. For example, say that you're exploring a group of 300 hens in a barn and you've measured a bunch of parameters from each bird: height, weight, egg size, laying regularity, egg color, etc. Based on all these parameters, you want to figure out if the hens fall into a small number of distinct groups or if they constitute just a single homogeneous population. Note that this question has two features: 1) At the outset you don't know how many groups there are and therefore 2) you have no way of assigning a given hen to a given group. All hens are treated equally. For problems such as this, you can use k-means clustering to objectively assign each hen to a group. Using further techniques (described below) you can objectively test whether the assignments you've made are reasonable.

As you might imagine, if there are "unsupervised" techniques then there must also be "supervised" techniques. Supervised techniques are often also called "classification" techniques and are applied to data sets where the identity of the groups are known beforehand. We will not discuss these approaches here but you can read about them in the introductory page on clustering and classification.

For the theory of k-means clustering technique, the students can find the algorithm details in lecture slides 5. Also, you should check the Section 13.2 from the book "The Elements of Statistical Learning" of Hastie, Tibshirani and Friedman.

The k-mean clustering includes some steps:

- Step 1: Initlialize randomly centroids of clusters using points from the dataset.
- Step 2: Assign datapoints to the clusters using a distrance metric.
- Step 3: Compute new centroids using the mean of the clusters.
- Repeat step 2 and step 3 until there is no change in the clusters.

1.2 Python Exercise

In this exercise, you will implement k-means clustering algorithm to subdivide the `sklearn`'s dataset *digits*. The dataset contains images of handwritte digits from 0 to 9. Table 1.2 provides information regarding this dataset.

You are going to do the following steps.

- Step 1: In this step, you will load the dataset *Digits* with 5 classes. You can choose how much data you want to use, but the whole dataset will take you more time to process. Then, you will visualize some images in this dataset using `matplotlib`. Also, you will be provided the code to visualize the dataset using `t-SNE`, which is one of the strongest method for visualizing high dimensional data. An optional step is to try different data reduction methods for visualization such as PCA[1].

- Step 2: You have to implement function `kmeans` in this step. The function takes two inputs: the data from `Digits` and the number of clusters. The expected outputs are the centroids of clusters and the labels of the data.

- Step 3: This step will evaluate your clustering results. You will use the function `kmeans` to make clusters. Also, you will visualize some images from the clusters to make sure that members of a cluster are indeed similar.

- Step 4: In this step, you will calculate the *silhouette* score. *Silhouette* is a metric to measure the quality of clustering. The score is bounded between -1 for incorrect clustering and +1 for highly dense clustering. Scores around zero indicate overlapping clusters. The score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster. `sklearn` has an implementation for this score[2]

[1] http://scikit-learn.org/stable/auto_examples/manifold/plot_lle_digits.html
[2] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

Table 1: *Digits* dataset.

Classes	10
Samples per class	~ 180
Samples total	1797
Dimensionality	64
Features	integers $0-16$

2 Hierarchical Clustering

In this exercise, we will perform a simple clustering task using the Agglomerative clustering algorithm, a well-known hierarchicar clustering algorithm. We will explore useful built-in functions in *sklearn* for this exercise.

2.1 Theory

Agglomerative clustering is a bottom-up hierarchical clustering algorithm. The basic idea behind this algirithm is to have each sample starts in its own cluster, and gradually merge pairs of clusters and move up the hierarchy, until all data samples are merged into a single cluster.

In Agglomerative clustering, the pairs of clusters are selected for merging based on their similarity measures. Some popular similarity measures include the 'single linkage', 'complete linkage' and 'group average'.

2.2 Python Exercise

In this exercise, we will perform Agglomerative clustering algorithm on the sample digit datasets in *sklearn*. We will make full use of available functions, built-in in *sklearn* for this exercise.

Step 1: Load Digit dataset. Use the function load digits from sklearn to load the dataset. In this exercise, we will only work with samples from 5 classes instead of all the samples in this dataset.

Step 2: Perform Agglomerative Clustering in sklearn. Perform Agglomerative Clustering with 5 clusters and two different linkage options:

- complete linkage

- group average

You should call the function *AgglomerativeClustering* [3] with suitable arguments for this task.

Step 3: Calculate the mean Silhouette Coefficient. Calculate the mean Silhouette Coefficient over all data samples with respect to the clustering results in Step 2. You should call the function *silhouette_score* [4] with suitable arguments for this task.

Step 4: Compare against Kmeans. Compare the Silhouette Coefficients calculated above with that of the Kmean algorithm calculated in the previous exercise.

Step 5: Dendrogram visualization. Generate dendogram figures to visualize the Agglomerative clustering on the digit dataset. You should use the function *dendrogram* [5] for this task.

[3] http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html.

[4] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

[5] https://docs.scipy.org/doc/scipy-0.19.1/reference/generated/scipy.cluster.hierarchy.dendrogram.html