Lecture 22 - K-Means Clustering

K-Means Algorithm

K-Means algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to **only one group**.

- K-Means Clustering is a centroid-based clustering.
- It tries to make the *inter-cluster* data points as similar as possible while also keeping the clusters as different (far) as possible.
- It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all data points that belong to that cluster) is at the minimum.
- The less variation we have within clusters, the more homogenous (similar) the data points are within the same cluster.

The **pseudo-code** is summarized as follows:

1. Specify number of clusters K
2. Initialize centroids by first shuffling the dataset amd then randomly selecting K data points for the centroids without replacement. (There are different ways of initializing the centroids)
3. Keep iterating until there is no change to the centroids, i.e., assignment of data points to clusters isn't changing.
    - Compute the sum of the squared distance between data points and all centroids
    - Assign each data pint to the closest cluster (centroid)
    - Compute the centroids for the clusters by taking the average of all data points that belong to each cluster

In [1]:
```
from IPython.display import Image
Image('figures/KMeans.png',width=800)
```

Out[1]:

Algorithm 1 K-Means Algorithm

Define number of clusters, K
Initialize cluster representatives
repeat
  for i = 1 to N **do**
    Determine the closest centroid representative, θ_k, for $\mathbf{x}_i$
    Set label of data point i to the cluster whose centroid θ_k is closest
  end for
  for j = 1 to K **do**
    Update cluster centroid representative θ_k to the mean of the points with cluster label k
  end for
until Change in cluster centers is small

The K-Means algorithm uses **Expectation-Maximization (EM) as the optimization approach**:

- The E-Step is assigning that data points to the closest cluster
- The M-step is computing the centroid of each cluster

As we learned before, optimization with EM is called **Alternating Optimization** and therefore the final solution will be **dependent** on the initialization (of cluster centroids). Therefore the final solution may not be the *optimal* (also referred to as *global*) solution.

- The objective function for the K-Means algorithm is:

$$J(\Theta, U) = \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ik} d^2 \left(x_i, \theta_k\right)$$
$$= \sum_{i=1}^{N} \sum_{k=1}^{K} u_{ik} \|x_i - \theta_k\|_2^2$$
$$\text{such that } u_{ik} \in \{0, 1\} \text{ and } \sum_{k=1}^{K} u_{ik} = 1$$

where u_{ij} are cluster assignments, θ_j is the j^{th} cluster representative and $d\left(x_i, \theta_k\right)$ is the distance between data point x_i and cluster centroid θ_k.

- In K-Means, we want to optimize:

$$\arg_{\Theta, U} \min J(\Theta, U)$$

- Does the K-means algorithm make any assumptions on cluster shape?
- Given a data set with an unknown number of clusters, can you come up with a strategy for determining the *right* number of clusters?
- Can we use other distance metrics in objective function $J(\Theta, U)$?

Optimization of K-Means

In optimizing the cost function $J(\Theta, U)$, we want to find the best set of parameters $\{\Theta, U\}$ that minimize it.

E-step

$$\frac{\partial J(\Theta, U)}{\partial u_{ik}} = 0$$
$$\sum_{i=1} \|x_i - \theta_k\|_2^2 = 0$$
$$\Rightarrow u_{ik} = \begin{cases} 1 & \text{if } k = \arg\min_k \|x_i - \theta_k\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

- In other words, assign the data point x_i to the closest cluster judged by its sum of squared distance from the cluster's centroid.

M-step

$$\frac{\partial J(\Theta, U)}{\partial \Theta} = 0$$
$$-2 \sum_{i=1}^{N} u_{ik} (x_i - \theta_k) = 0$$
$$\sum_{i=1}^{N} u_{ik} x_i = \sum_{i=1}^{N} u_{ik} \theta_k = 0$$
$$\sum_{i=1}^{N} u_{ik} x_i = \theta_k \sum_{i=1}^{N} u_{ik} = 0$$
$$\theta_k = \frac{\sum_{i=1}^{N} u_{ik} x_i}{\sum_{i=1}^{N} u_{ik}}$$
$$\theta_k = \frac{\sum_{i=1}^{N} u_{ik} x_i}{N_k}$$
$$\theta_k = \frac{\sum_{x_i \in C_k} x_i}{N_k}$$

where $N_k = \sum_{i=1}^{N} u_{ik}$ is the number of all data points assigned to cluster C_k, and $\sum_{x_i \in C_k} x_i = \sum_{i=1}^{N} u_{ik} x_i$ is the sum of data points that are assigned to cluster C_k.

- So the new cluster centroid is nothing but the **average** of all data points assigned to that cluster.

Observations

- Since K-Means uses distance-based measurements to determine the *similarity* between data points, it's recommended to **scale the data** since almost always the features in any data set would have different units of measurements (e.g. age vs income).

- Given K-Means Alternating Optimization approach, different initializations may lead to different clusters, as K-Means algorithm may be stuck in a *local optima* and not converge to the *global optima*. Therefore, it's recommended to run the algorithm using different initializations of centroids and pick the results of the run that yielded the lower sum of squared distance.

- One of the convergence criteria is to check whether the assignment of data points has not changed from one iteration to the next. This criteria of "assignment of points not changing" is the same as observing no change in the *within-cluster variation*:

$$\frac{1}{N_k} \sum_{i \in c_k} \|x_i - \theta_k\|_2^2$$

K-Means Applications

K-Means is a very popular algorithm and is commonly used in a variety of applications, such as: market segmentation, document clustering, image segmentation, image compression, etc.

The *goal* usually when we undergo a cluster analysis is either:

1. Get a meaningful intuition of the structure of the data we are dealing with.
2. Cluster-then-predict where different models will be built for different subgroups. An example of that is clustering patients into different subgroups (based on some feature map) and build a model for each subgroup to predict the probability of the risk of having a heart attack.

Let's take a look at two case applications for K-Means: Image Compression and Data Segmentation.