CSCC11 Week 10 Notes

Clustering:

- Clustering is an unsupervised learning problem in which the goal is to discover clusters in the data. A cluster is a collection of data points that are similar in some way.

K-Means:

- A simple method for clustering.
- Given N input data vectors ($\{y_i\}_{i=1}^N$) we wish to label each vector as belonging to 1 of k-clusters. This labelling will be specified with a binary matrix L, the elements are given by

$$L_{ij} = \begin{cases} 1, & \text{if data point } i \text{ belongs to cluster } j \\ 0, & \text{otherwise} \end{cases}$$

- We also assume that each data point is assigned to only 1 cluster.
  I.e. $\forall$ points i, $\sum_j L_{ij} = 1$

- We also want to find a representative for each cluster, C_j. For example, it could be the mean of the points assigned to that cluster.

  The obj-fun for k-Means Clustering is:
  $E(\{C_j\}_{j=1,\dots,k}, L) = \sum_{i,j} L_{ij} \|y_i - c_j\|^2$

This obj func penalizes the squared Euclidean dist btwn each data point and the center of the cluster to which it is assigned. To minimize this error, we want to bring cluster centers close to the points within their clusters and we want to assign each data point to the nearest cluster.

This optimization problem is NP-hard and can't be solved in close form. Furthermore, because it includes discrete variables (the labels L), we can't use gradient-based methods. Instead, we'll use block coordinate descent.

General Algo:

1. Initalize c_j's
2. Assign each point y_i to the closest c_j. Closest is computed using Euclidean dist
3. Update c_j to be the mean of the data points assigned to cluster j.
4. Repeat (2) and (3) until assignments are unchanged.

Note: There's a chance that you can get trapped in a local minima with this algo.

- In step 1 of the algo on the previous page we initalized C_j's. Poor initalization can lead to poor results. Here are a few strategies that can be used for initalization:

1. Random Labelling: Initalize the labelling L randomly and then run step (3) of the gen algo to determine the initial centers. This approach isn't recommended bc the initial centers will likely end up just being very close to the mean of the entire dataset.

2. Random Initial Centers: Choose the inital center values randomly. But it's very likely that some of the centers will fall into empty regions of the feature space and will be assigned no data. Getting a good initalization is difficult using this method.

3. Random data points as centers: Choose a random subset of the data points as the initial centers. Works somewhat better.

4. Multiple restarts: Run k-Means multiple times, each time with a different random initalization and keep the soln that gives the lowest value of the obj func.

5. K-Means++: The goal is to choose the initial centers to be relatively far from each other.

One of the simplest and best ways for initalization $\rightarrow$

    1. Choose 1 data point at random to be the first center.
    2. Compute the dist btwn each point and its closest center denoted $D(y_i)$ for the ith data point.
    3. Choose the next center from the remaining data points with prob proportion to $D(y_i)^2$ for the ith data point. 4. Repeat 2 and 3