Lab 13: Clustering

Johnny Hong

Stat 154, Fall 2017

Introduction

In this lab, we will explore two methods of clustering: K-means clustering and hierarchical clustering.

K-means clustering

In K-means clustering, K refers to the desired number of distinct non-overlapping clusters. Do NOT confuse the K-means algorithm with the K-nearest neighbors method, which is used for classification. Given a dataset X and the desired number of clusters K, the K-means algorithm is as follows:

1. Randomly select K rows of the dataset and treat them as the initial cluster centroids $g_1, ..., g_K$.
2. For each observation x_i,
   a. compute $d(x_i, g_k) = ||x_i - g_k||_2^2$ for each k.
   b. Find $k^* = \arg\min_{k=1,\ldots,K} d(x_i, g_k)$.
   c. Assign x_i to cluster k^*.
3. For each cluster C_k, compute the cluster centroid $g_k = |C_k|^{-1} \sum_{i \in C_k} x_i$.
4. Repeat Step 2 and Step 3 until convergence (that is, the cluster assignment does not change).

Your turn

- Implement a function `my_kmeans()` that performs the K-means algorithm.
- `my_kmeans()` should take the following inputs:
  - `X`: a $n \times p$ matrix
  - `k`: the desired number of clusters
- `my_kmeans()` should return a list of the following:
  - `cluster_sizes`: a length-K vector of cluster sizes
  - `cluster_means`: a $K \times p$ matrix in which each row corresponds to a cluster centroid
  - `clustering_vector`: a vector of length n containing the cluster assignment for each observation
  - `wss_cluster`: a vector of length K containing the within-cluster sum-of-squares
  - `bss_over_tss`: a scalar, BSS/TSS, where BSS is the between sum-of-squares and TSS is the total sum-of-squares.

- Run `my_kmeans()` with `X` being the first four columns of `iris` and `K = 3`.
- Compare your answer with `kmeans()`.

Remark: Do not panic even if your output does not match with $kmeans()$. The local optimum achieved in K-means depends on the initialization of the centroids. Try running `my_kmeans()` and `kmeans()` a couple times and see how the cluster assignments change.

Hints: The within-sum-of-squares for cluster k is

$$WSS_k = \sum_{j=1}^{p} \sum_{i \in C_k} (x_{ij} - \bar{x}_{kj})^2, k = 1, ..., K,$$

where $\bar{x}_{kj}$ is the mean of cluster k, and the between-sum-of-squares is

$$BSS = TSS - \sum_{k=1}^{K} WSS_k.$$

Hierarchical clustering

One issue with the K-means algorithm is that the number of clusters K must be specified before running the algorithm. In hierarchical clustering, we do not have to commit to a particular K. The idea of the most common type of hierarchical clustering is to build the cluster assignment bottom-up: we first begin with each of the observation being a cluster (and hence there are n clusters initially, where n is the number of observations in the dataset) and gradually merge the clusters until there is only one cluster left.

To decide which clusters to merge at each step, we have to define a suitable "distance" between two clusters (that is, two groups of observations). Once such a distance is defined, we can simply compute the pairwise distances among all of the clusters and merge the two clusters with the shortest distance from each other. There are four common types of cluster dissimilarity measures, also known as *linkages*: complete linkage, average linkage, single linkage, and centroid linkage. See Table 10.2 in ISL for a detailed description of each linkage. In this lab, we will not explore the centroid linkage.

Your turn

This part follows closely ISL 10.5.

- Use `hclust()` to perform hierarchical clustering on the `iris` dataset. Do complete linkage, average linkage and single linkage. Save the resulting fits to `hc.complete`, `hc.average` and `hc.single` respectively.
- For each linkage,
  - plot the dendrogram using `plot()`.
  - use `cutree()` with `k=3`.
- Based on the results from `cutree()`, how does each linkage perform? Do they all separate (roughly) the observations into correct group?