PS5841

Data Science in Finance & Insurance

Regularization

Yubo Wang

Autumn 2022

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

K-Means Clustering

K-Means Clustering

- Pre-specify K clusters
- Each observation belongs to at least one of the K clusters
- No observation belongs to more than one cluster
- Clustering driven by minimizing within-cluster variations, e.g. squared Euclidean distance

K-Means Clustering objective

Global minimum

$$\min_{C_1,\dots C_K} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

$$= \min_{C_1,\dots C_K} \left\{ \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,j \in C_k} \sum_{l=1}^{p} \left(x_{il} - x_{jl}\right)^2 \right\}$$

$$= \min_{C_1,\dots C_K} \left\{ \sum_{k=1}^{K} 2 \sum_{i \in C_k} \sum_{l=1}^{p} (x_{il} - \bar{x}_{kl})^2 \right\}$$

Mean for feature l in cluster C_k: $\bar{x}_{kl} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{il}$

K-Means Clustering Algo

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

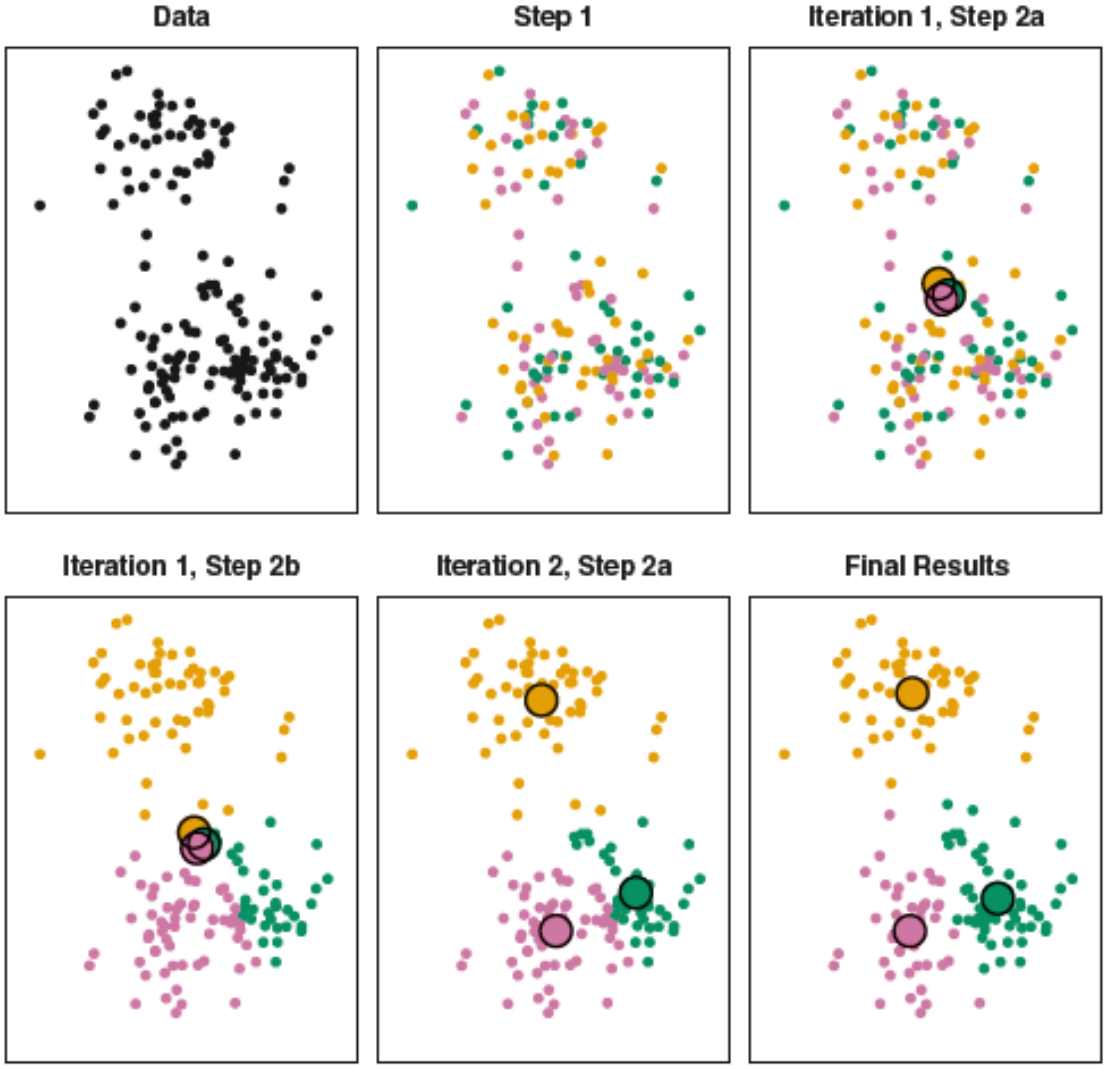
2. Iterate until the cluster assignments stop changing:

   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

K-means Clustering

random assignment calc cluster centroid



assign obs to the nearest centroid

within-cluster variation minimized

Coding

- R

```
base::kmeans()
```

- Python

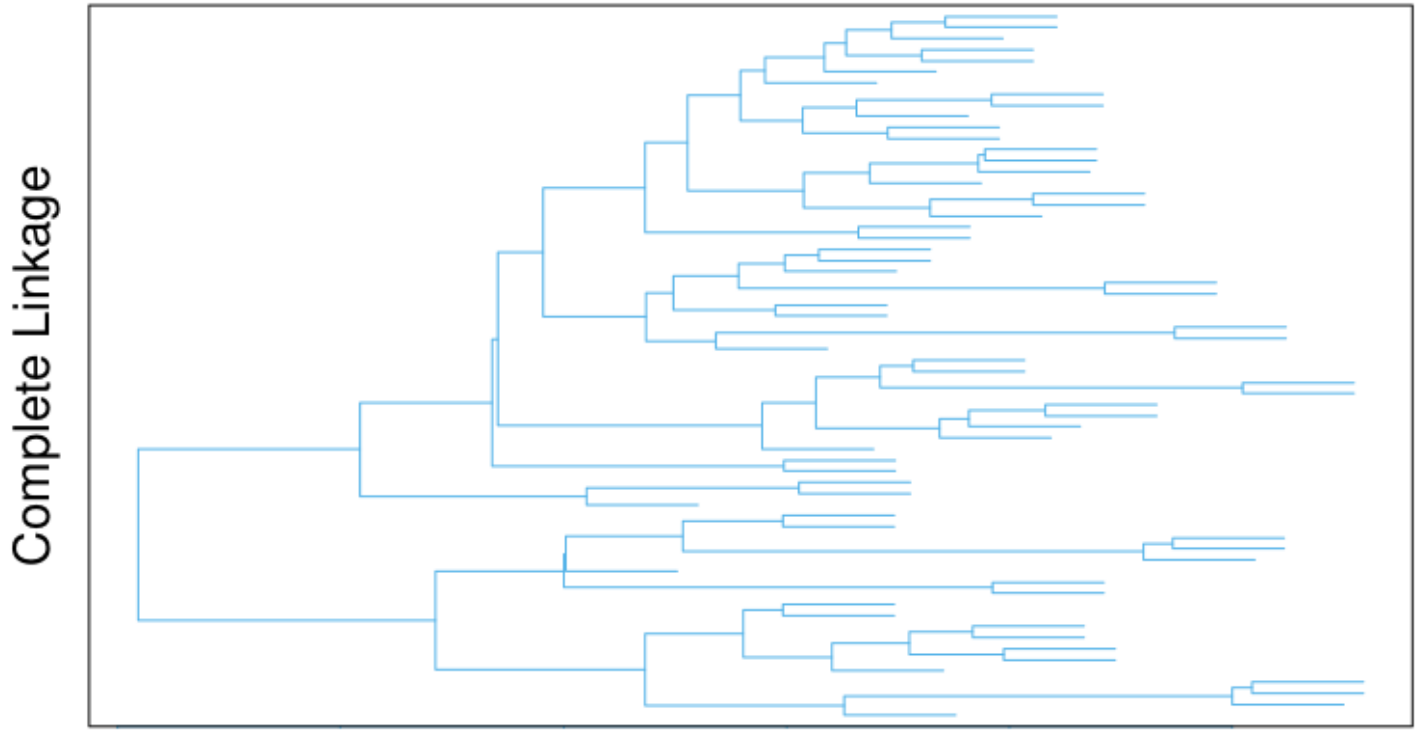
```
scipy.cluster.Kmeans()
```

Hierarchical Clustering

Hierarchical Clustering

- No need to prespecify the number of clusters
- Produces a dendrogram, a tree-based representation of the observations
- Bottom-up / agglomerative clustering
  - Start from the leaves and combine clusters up to the trunk

Dendrogram



HC Algo

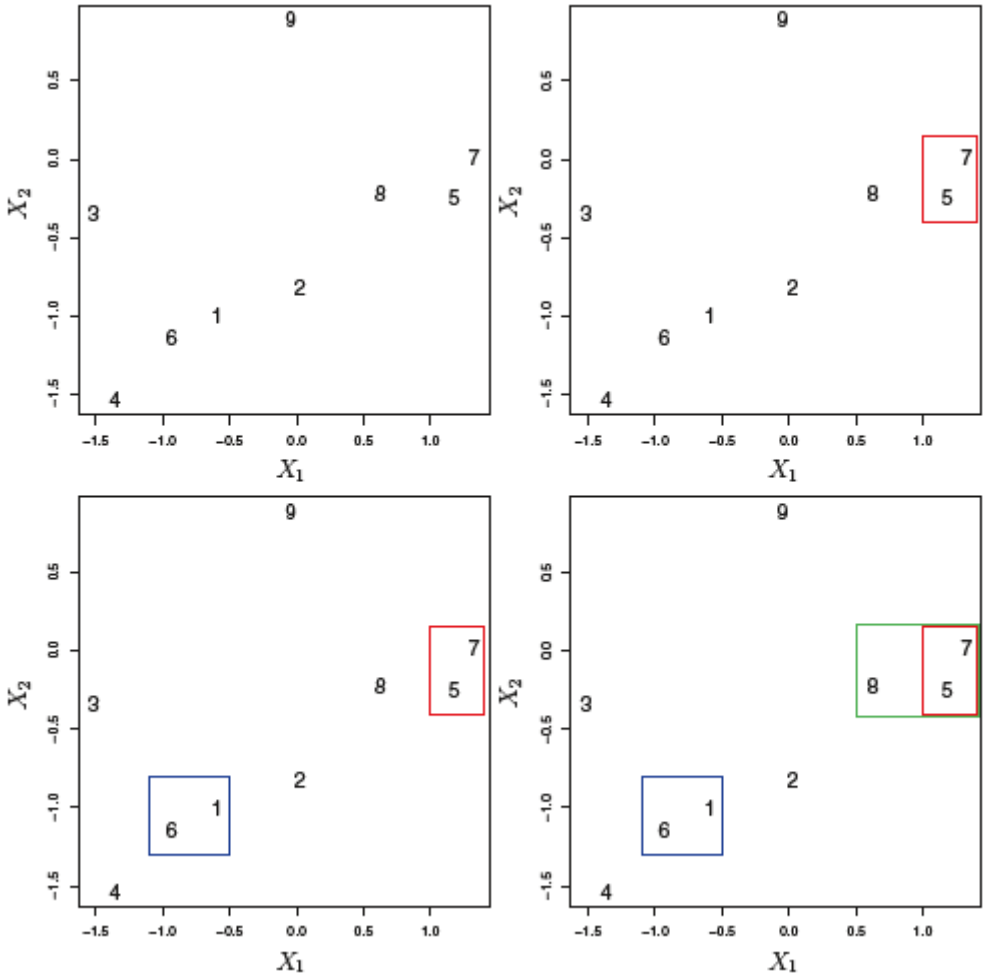
1. Begin with n observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

2. For $i = n, n-1, \ldots, 2$:

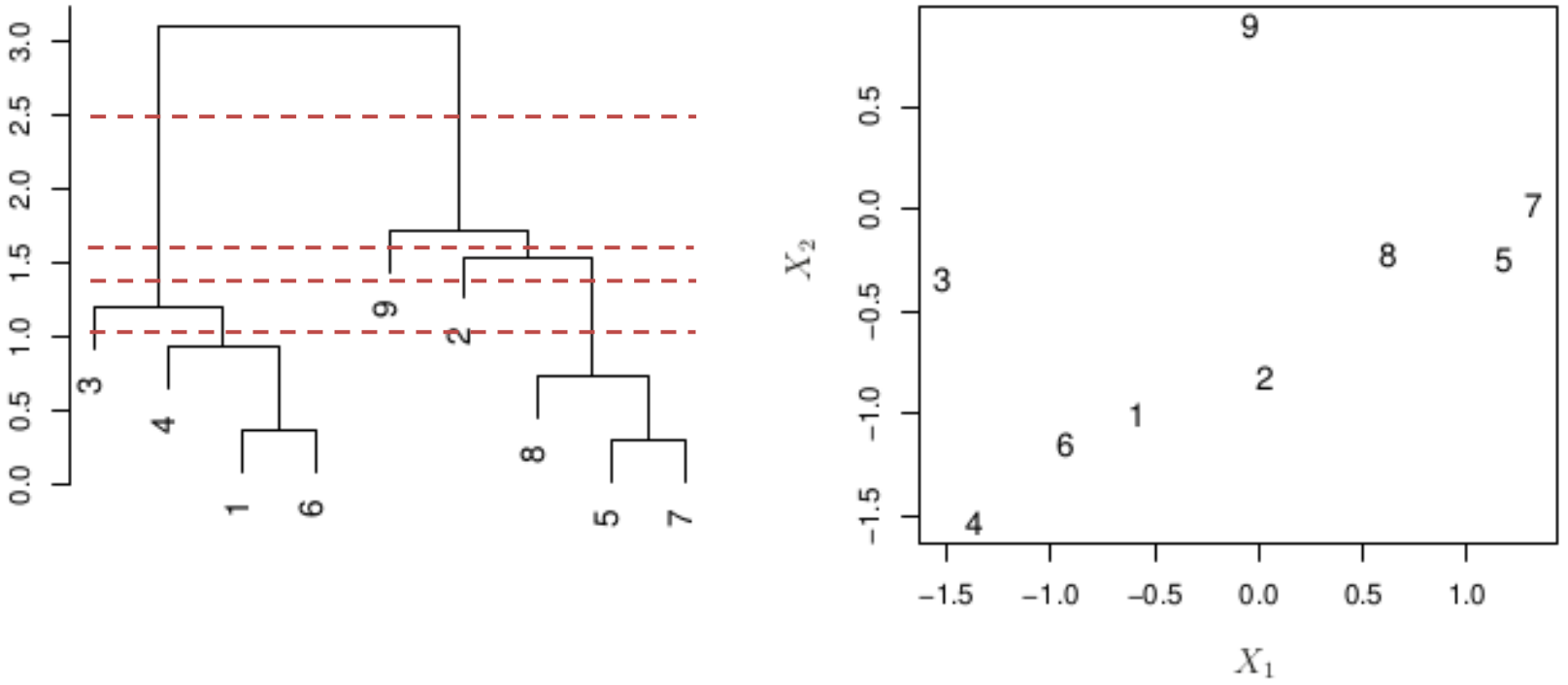
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

HC Algo



Interpreting a Dendrogram



[9] is no more similar to [2] than it is to [8],[5],[7]

Linkage (dissimilarity measure)

Linkage	*Description*
Complete	Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *largest* of these dissimilarities.
Single	Minimal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *smallest* of these dissimilarities. Single linkage can result in extended, trailing clusters in which single observations are fused one-at-a-time.
Average	Mean intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the *average* of these dissimilarities.
Centroid	Dissimilarity between the centroid for cluster A (a mean vector of length p) and the centroid for cluster B. Centroid linkage can result in undesirable *inversions*.

Hierarchical Clustering

- Applicable when clusters are nested, less competitive (than KMC) if not
  - Clusters obtained at a cut are nested within those from a higher cut
- Dendrogram
  - The height in the dendrogram (where a fusion is placed) is the dissimilarity between fused clusters

Coding

- R

```
base::hclust()
base::cutree()
```

- Python

```
scipy.cluster.hierarchy.linkage()
scipy.cluster.hierarchy.dendrogram()
scipy.cluster.hierarchy.cut_tree()
```

That was