Chapter 5 Unsupervised Learning: Clustering

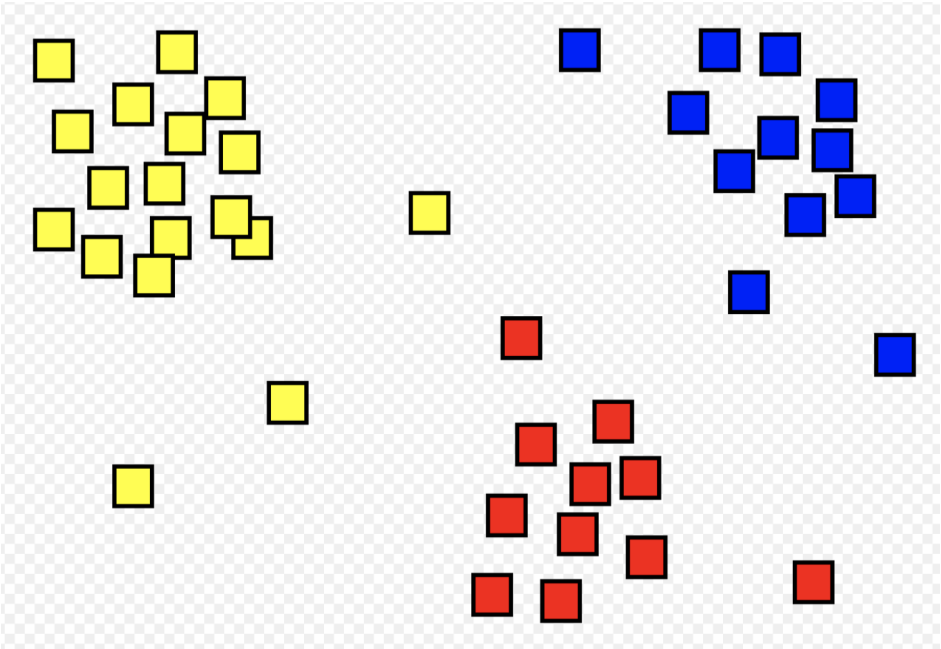
AMA4680
Fu Guanxing
guanxing.fu@polyu.edu.hk

Unsupervised learning:

- dimension reduction (PCA), and
- **Cluster Analysis**

Cluster Analysis

Cluster analysis, is to find groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups.



Cluster Analysis

- K-**means** (a case of prototype clustering)

K-means Clustering

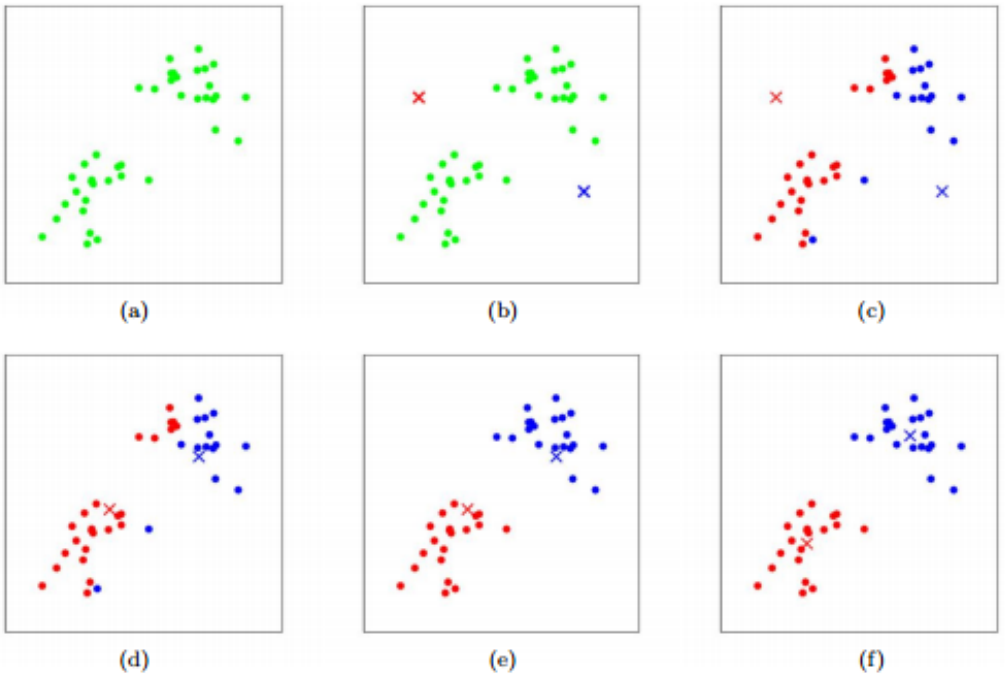
Given a dataset, divide it into K clusters, based on the distance between samples, such that the inter-cluster distance is as large as possible and the intra-cluster distance is as small as possible.

Step b Select K points as the initial centroids; $K = 2$ in the following example.

Step c Form K clusters by assigning all points to the closest centroid; ℓ_2 norm.

Step d Update the centroid of each cluster by calculating the mean of each cluster. Assign cluster of each sample based on the distance to the new centroids.

Step e-f Repeat c and d until the new centroids do not change any more.



K-means, *cont'd*

- Performance of K-means: **C**alinski-**H**arabasz index (Variance ratio criterion)[1].

$$CH = \frac{B(K)/(K-1)}{W(K)/(n-K)} \to \max,$$

where

- n is the sample size, K is the no. of clusters.
- $B(K) = \sum_{k=1}^{K} n_k \|c_k - c\|^2$ is the inter-cluster dispersion. Here n_k is the sample size of Cluster k, c_k is the centroid of Cluster k and c is the centroid of the dataset.
- $W(K) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} \|d_i - c_k\|^2$ is the intra-cluster dispersion. Here, d_i is sample i.

[1] Call `metrics.calinski_harabasz_score()` in python.

Example of K-means

Consider the dataset

Index	Feature 1	Feature 2	Index	Feature 1	Feature 2
1	0.697	0.460	16	0.593	0.042
2	0.774	0.376	17	0.719	0.103
3	0.634	0.264	18	0.359	0.188
4	0.608	0.318	19	0.339	0.241
5	0.556	0.215	20	0.282	0.257
6	0.403	0.237	21	0.748	0.232
7	0.418	0.149	22	0.714	0.346
8	0.437	0.211	23	0.483	0.312
9	0.666	0.091	24	0.478	0.437
10	0.243	0.267	25	0.525	0.369
11	0.245	0.057	26	0.751	0.489
12	0.343	0.099	27	0.532	0.472
13	0.639	0.161	28	0.473	0.376
14	0.657	0.198	29	0.725	0.445
15	0.360	0.370	30	0.446	0.459

Use K-means to divide into $K = 3$ clusters.

Example, *cont'd*

- Randomly choose d_6, d_{12} and d_{24} as initial centeroids:

$$\mu_1^{(0)} = (0.403, 0.237)', \quad \mu_2^{(0)} = (0.343, 0.099)', \quad \mu_3^{(0)} = (0.478, 0.437)'.$$

- Compute the distance of each sample to centroids. For instance, $d_1 = (0.697, 0.460)'$

$$\|d_1 - \mu_1^{(0)}\| = 0.369, \quad \|d_1 - \mu_2^{(0)}\| = 0.506, \quad \|d_1 - \mu_3^{(0)}\| = 0.22.$$

Thus, assign d_1 to Cluster 3...

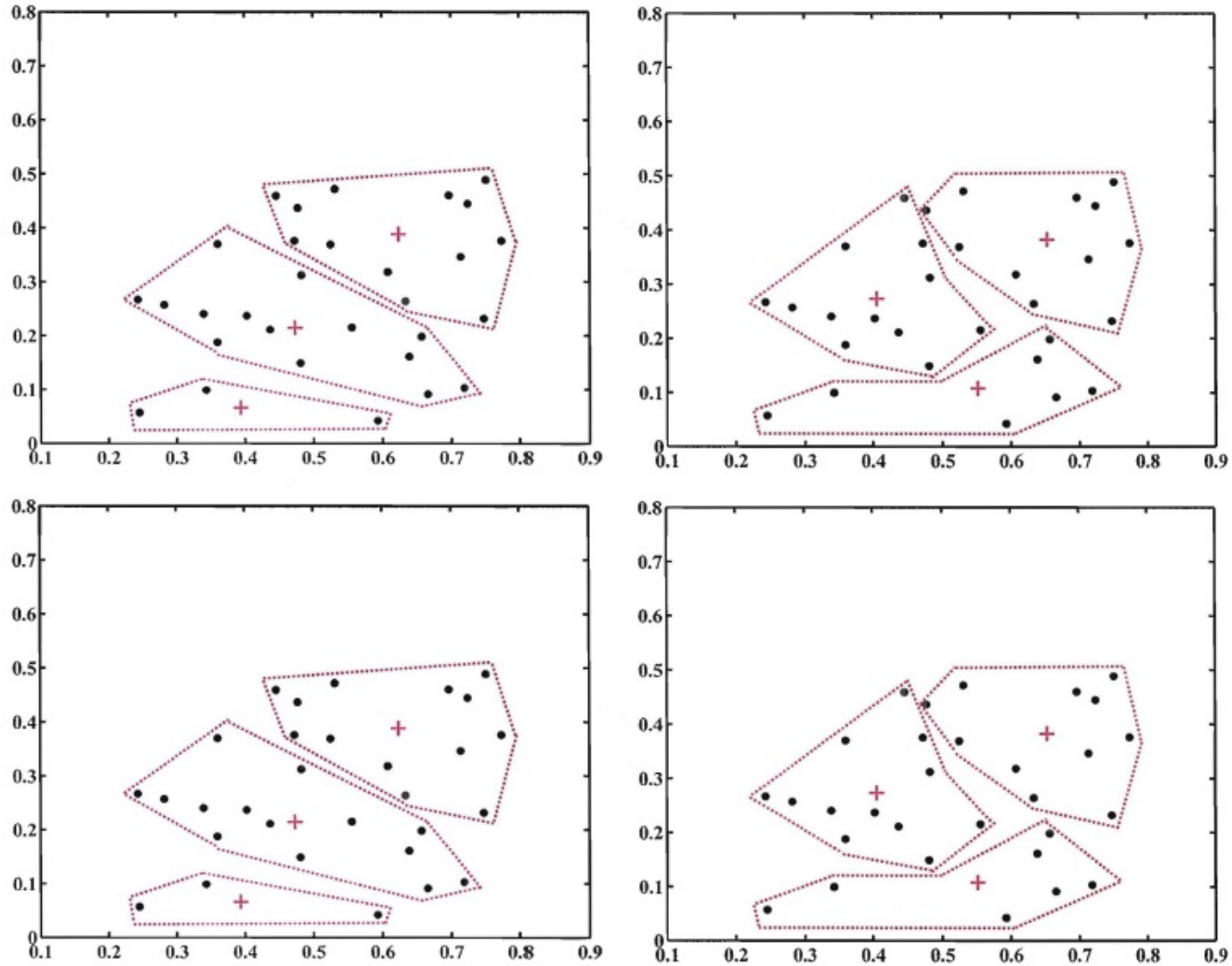
$$\text{Cluster}_1 = \{d_5, d_6, d_7, d_8, d_9, d_{10}, d_{13}, d_{14}, d_{15}, d_{17}, d_{18}, d_{19}, d_{20}, d_{23}\}$$
$$\text{Cluster}_2 = \{d_{11}, d_{12}, d_{16}\}$$
$$\text{Cluster}_3 = \{d_1, d_2, d_3, d_4, d_{21}, d_{22}, d_{24}, d_{25}, d_{26}, d_{27}, d_{28}, d_{29}, d_{30}\}.$$

- Update centroids:

$$\mu_1^{(1)} = (0.473, 0.214)', \quad \mu_2^{(1)} = (0.394, 0.066)', \quad \mu_3^{(1)} = (0.623, 0.388)'$$

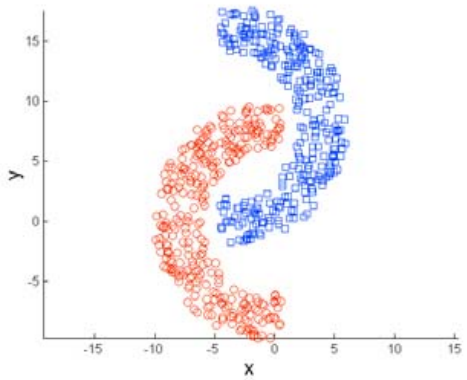
- Repeat.

Example, *cont'd*

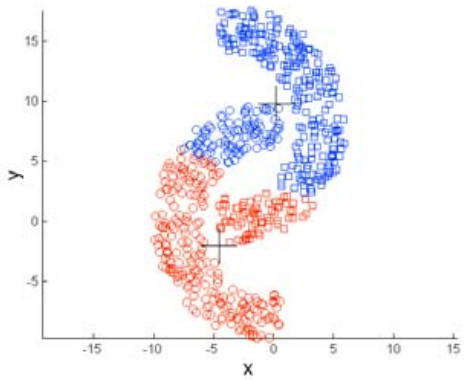


Shortcomings

- Multiple runs. Remedy: set the maximal iteration times `n_clusters`.[2]
- Significantly rely on the initial choice. Remedy: run several times with different initial centroids and choose the best one.[3]
- May get bad results when points are distributed on manifold yet ambient distance is used.



Original Points



K-means (2 Clusters)

[2] By default `n_clusters`=300 in sklearn.

[3] By default, `n_init`=10.

K-means from an optimization point of view

- Given the dataset $\{x_i \in \mathbb{R}^d\}_{i=1}^n$, K-means is equivalent to
  - Step 1. For each fixed $i \in \{1, \cdots, n\}$, calculate the distance between x_i and all centroids $\mu_1, \cdots, \mu_k$. Select the index with the smallest distance, i.e. $z_i = \arg\min_{j=1,\cdots,k} \|x_i - \mu_j\|^2$.
  - Step 2. Updating the centroid μ_j for cluster $j = 1, \cdots, k$ by minimizing the loss function

$$L_j(\mu_j) = \sum_{i \in \{1,\cdots,n:\ z_i = j\}} \|x_i - \mu_j\|^2.$$

- Equivalently, minimize the following loss function by choosing the centroids of k clusters $\mu = (\mu_1, \cdots, \mu_k)^\top$

$$\begin{aligned} L(\mu) &= \sum_{i=1}^n \min_{j \in \{1\cdots,k\}} \|x_i - \mu_j\|^2 \\ &= \sum_{\{i: x_i \in \mathsf{C}_1\}} \|x_i - \mu_1\|^2 + \cdots + \sum_{\{i: x_i \in \mathsf{C}_k\}} \|x_i - \mu_k\|^2 \end{aligned}$$

Shortcoming of K-means and remedy

- K-means is sensitive to outliers. Consider the dataset with two features

x_1	-1	-1	1	1	10
x_2	-1	1	-1	1	0

  Calculate the mean, and the mean by changing $(10, 0)^\top$ to $(100, 0)^\top$.

- Remedy: K-median clustering.
  - For each sample $i \in \{1, \cdots, n\}$, let

$$z_i = \arg \min_{j=1,2,\cdots,k} \|x_i - \mu_j\|_{\ell_1}$$

  - Update the centroid μ_j for cluster $j = 1, \cdots, k$ by minimizing the loss function

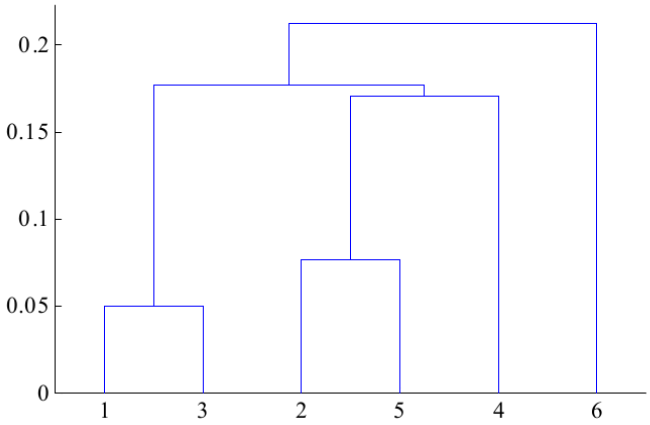
$$L_j(\mu_j) = \sum_{i \in \{1,\cdots,n:\ z_i = j\}} \|x_i - \mu_j\|_{\ell_1}.$$

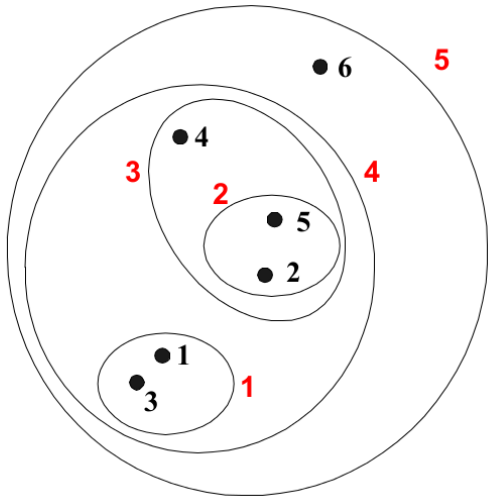
Cluster Analysis

- K-means
- **Agglomerative clustering** (a case of hierarchical clustering)

Hierarchical Clustering

- Produce hierarchical tree based on similarities between samples.
- Usually visualized as a dendrogram (cluster tree), which is a tree-like diagram that records the sequence of merges or splits.
- Advantage:
  - Do not have to assume any particular number of clusters.
  - The cluster tree corresponds to meaningful taxonomies.





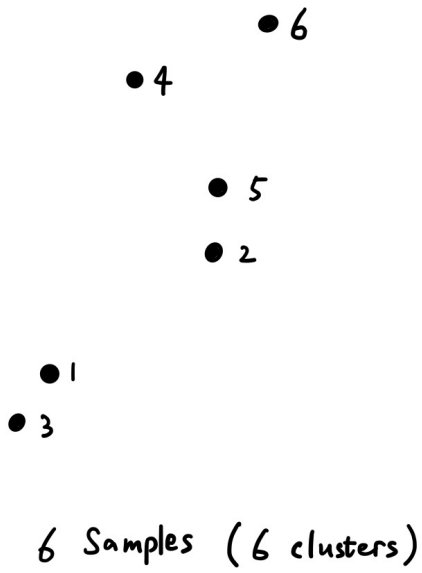
Agglomerative Clustering Algorithm (1/5)

- A popular hierarchical clustering algorithm
- Procedure of agglomerative clustering (AGNES)

Preparation: Compute the proximity matrix
Step 1: Let each data point be a cluster
Step 2: Merge the two closest clusters
Step 3: Update the proximity matrix
Step 4: Repeat Step 2 and Step 3 until only one single cluster remains.

Agglomerative clustering algorithm (2/5)

Start with clusters of individual points and a proximity matrix (which defines the distances between each point).

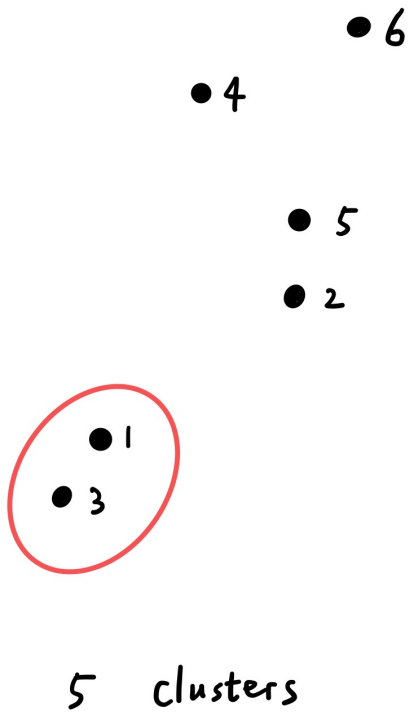


	p_1	p_2	p_3	p_4	p_5	p_6
p_1	0	0.175	0.05	*	*	*
p_2	0.175	0	*	*	0.75	*
p_3	0.05	*	0	*	*	*
p_4	*	*	*	0	0.17	0.21
p_5	*	0.75	*	0.17	0	*
p_6	*	*	*	0.21	*	0

proximity matrix

Agglomerative clustering algorithm (3/5)

- Merge 1 and 3 and get 5 clusters.



- How to compute distance (similarity, linkage) between clusters? ↷

Agglomerative clustering (4/5)

Given two clusters C_1 and C_2, inter-cluster similarity can be computed by

- Single linkage:

$$\mathrm{dist}(C_1, C_2) = \min_{x \in C_1, y \in C_2} \mathrm{dist}(x, y),$$

- Complete linkage:

$$\mathrm{dist}(C_1, C_2) = \max_{x \in C_1, y \in C_2} \mathrm{dist}(x, y),$$

- Group average linkage:

$$\mathrm{dist}(C_1, C_2) = \frac{1}{\#C_1 \times \#C_2} \sum_{x \in C_1, y \in C_2} \mathrm{dist}(x, y),$$

where $\mathrm{dist}(x, y)$ is the Euclidean distance (ℓ_2 distance) between sample x and sample y.

Agglomerative clustering algorithm (5/5)

With the following distance matrix, plot a cluster tree with single linkage and mark out the merging level.

	p1	p2	p3	p4
p1	0.00	0.82	0.10	0.35
p2	0.82	0.00	0.91	0.65
p3	0.10	0.91	0.00	0.44
p4	0.35	0.65	0.44	0.00