Clustering

Tran Giang Son, tran-giang.son@usth.edu.vn

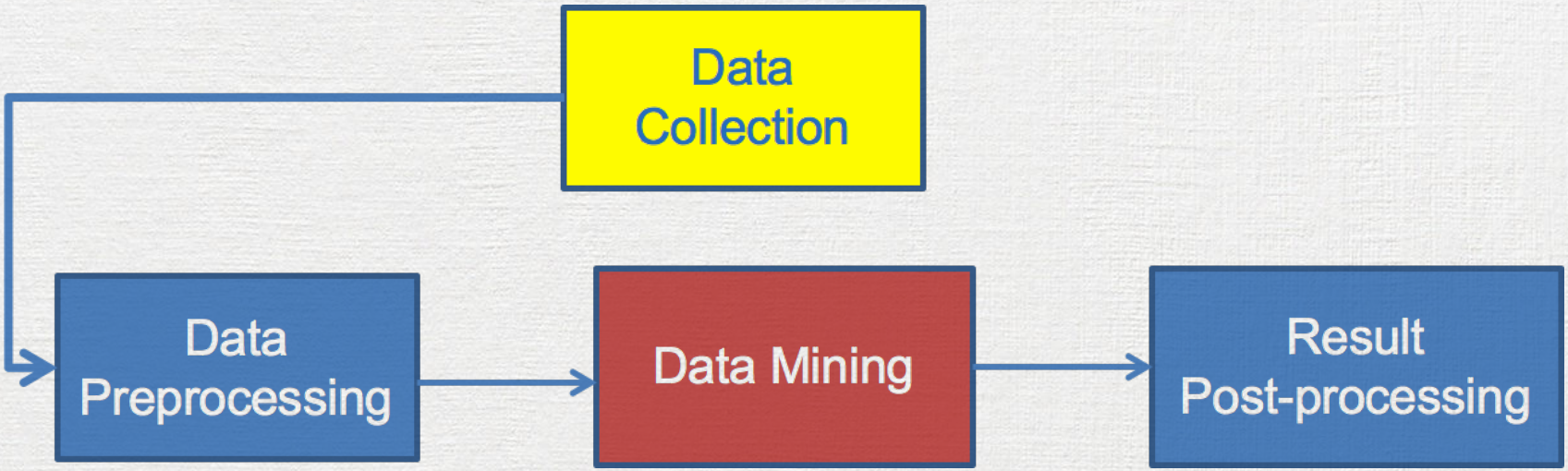
ICT Department, USTH

Clustering

Data Mining

- What: Extract information from data
- Why: A lot of data. Data is $$$

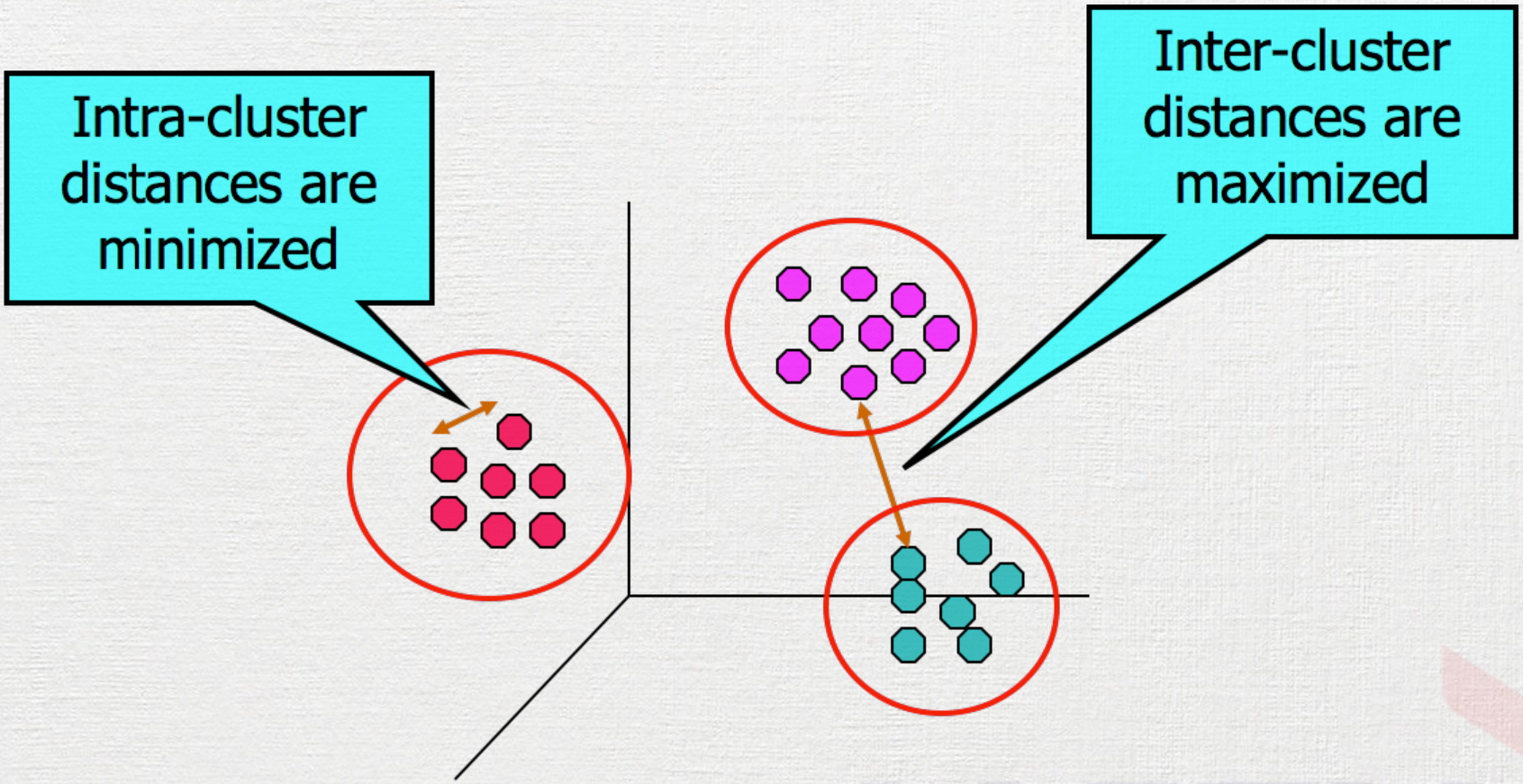
Data Mining


Data Collection
Data Preprocessing
Data Mining
Result Post-processing

Clustering

- Unsupervised learning
- Input: objects
- Output: groups of objects
  - Objects in the same group are similar
  - Objects in different group are unrelated
  - Each group is called a cluster

Clustering


Intra-cluster distances are minimized
Inter-cluster distances are maximized

Clustering

- Interpretability
- Attribute shape
- Different types of attributes
- Noisy data

Clustering

- Applications
  - Image processing: segmentation of objects
  - Biology, bioinformatics: grouping of species
  - Mobile communication: grouping of users
  - Medicine: medical imaging
  - Economics: market research, shopping items

Clustering

- Hierarchical clustering
- K-means and its variants
- DBSCAN

Hierarchical Clustering

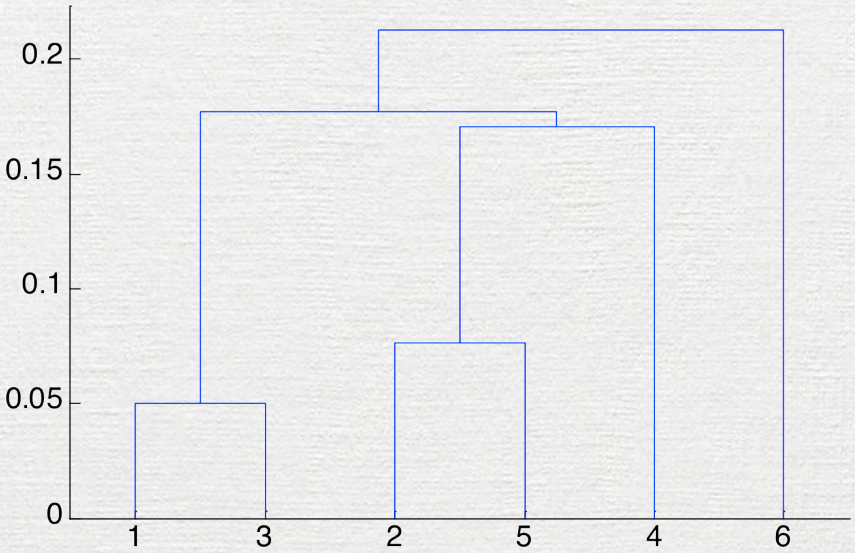
Hierarchical Clustering

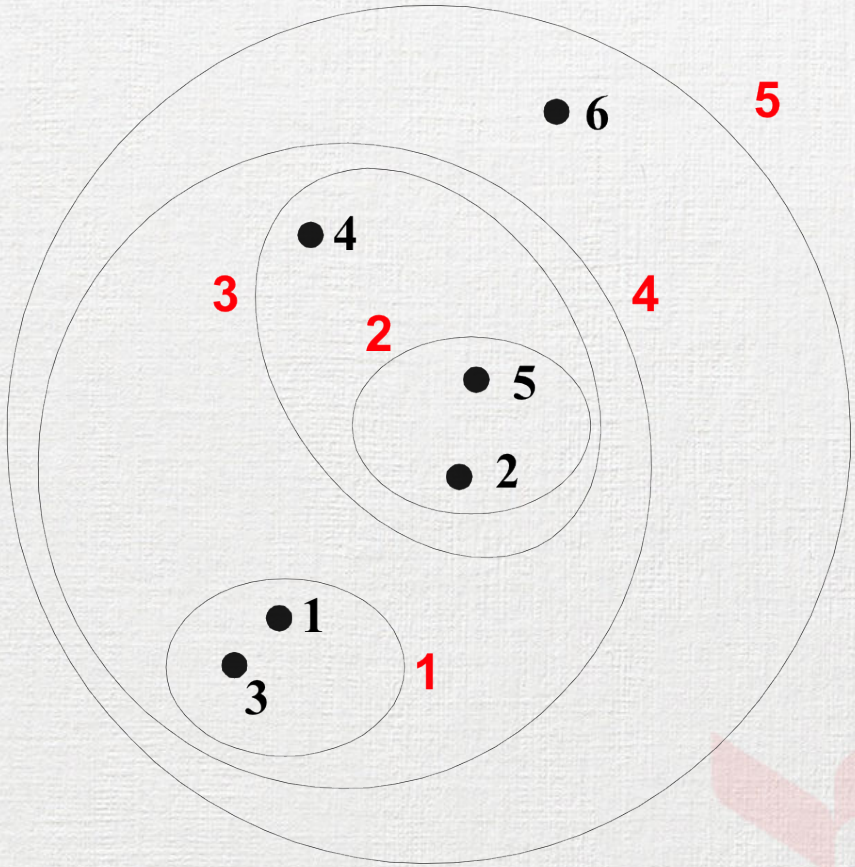
- Two main types of hierarchical clustering
  - Agglomerative: bottom up
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive: top down
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains a point (or there are k clusters)

Hierarchical Clustering

- Traditional hierarchical clustering
  - Similarity or distance matrix
  - Merge or split one cluster at a time
- Produces a set of nested clusters
  - A hierarchical tree
  - Visualized as a dendrogram
    - A tree like diagram that records the sequences of merges or splits

Hierarchical Clustering





Hierarchical Clustering

- No assumption of number of clusters
  - Simply cutting the dendogram at the proper level for a specific number of clusters
- Meaningful taxonomies
  - Biological sciences: animal kingdom
  - Image analysis: finger, palm, hand, limb, human

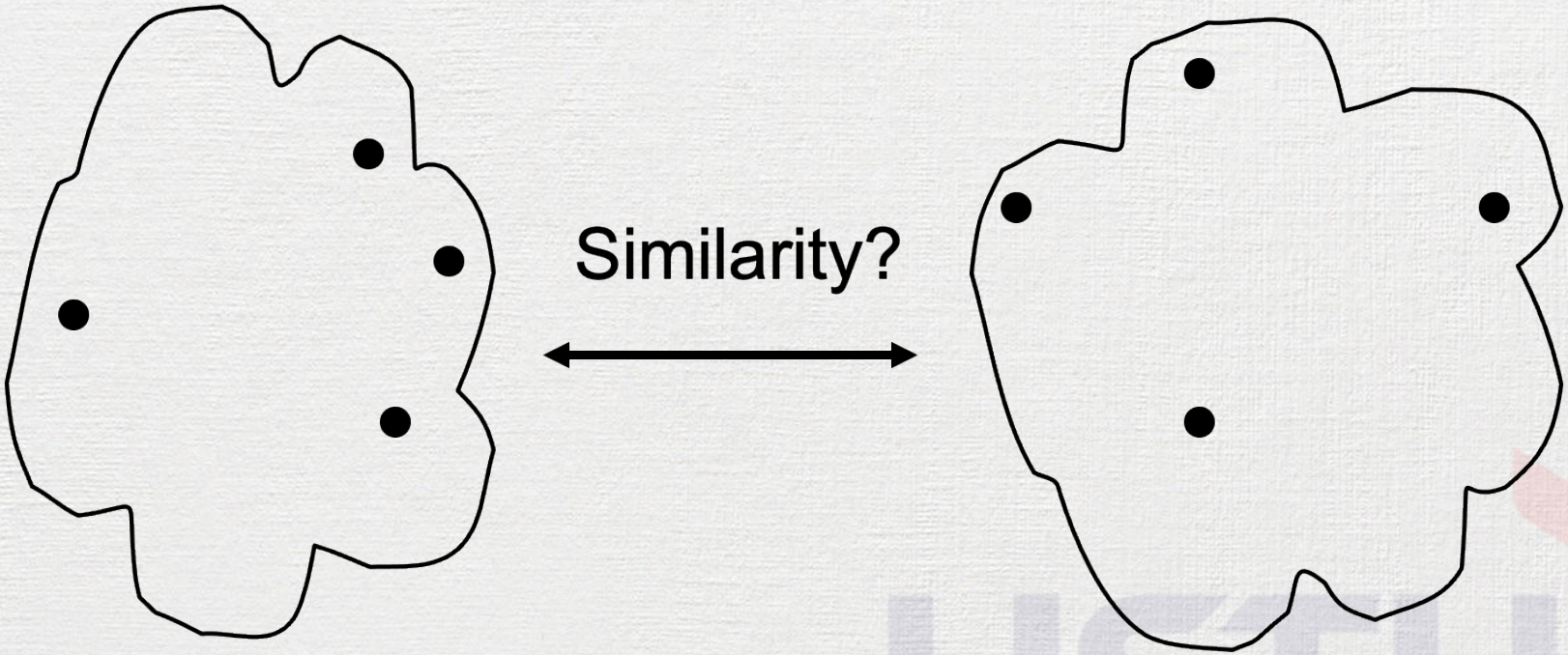
Agglomerative Hierarchical Clustering

- More popular than divisive methods
- Basic algorithm

```
Compute the proximity matrix
Let each data point be a cluster
Repeat
    Calculate distance between clusters
    Merge the two closest clusters
Until only a single cluster remains
```

Agglomerative Hierarchical Clustering: Proximity

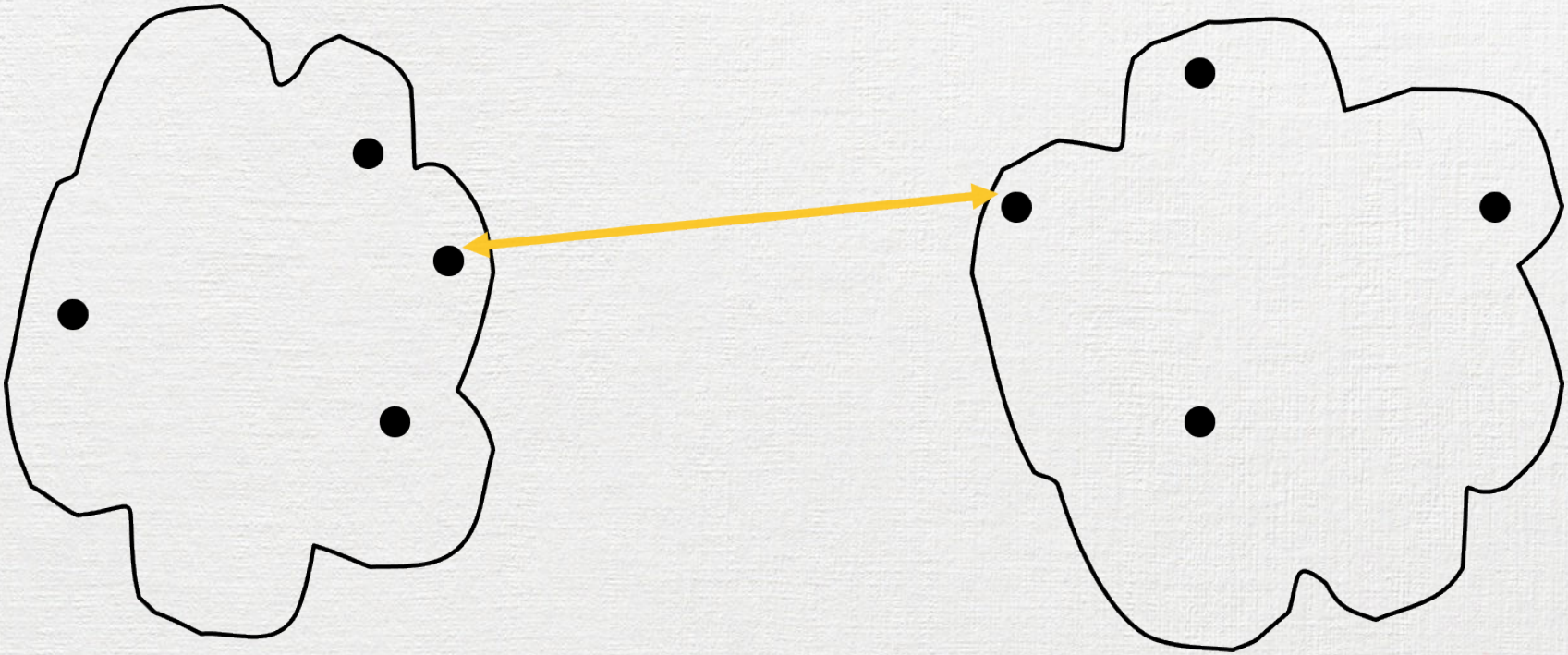
- Proximity matrix: distance between pairs of points.
- Proximity of two clusters?



Agglomerative Hierarchical Clustering: Proximity

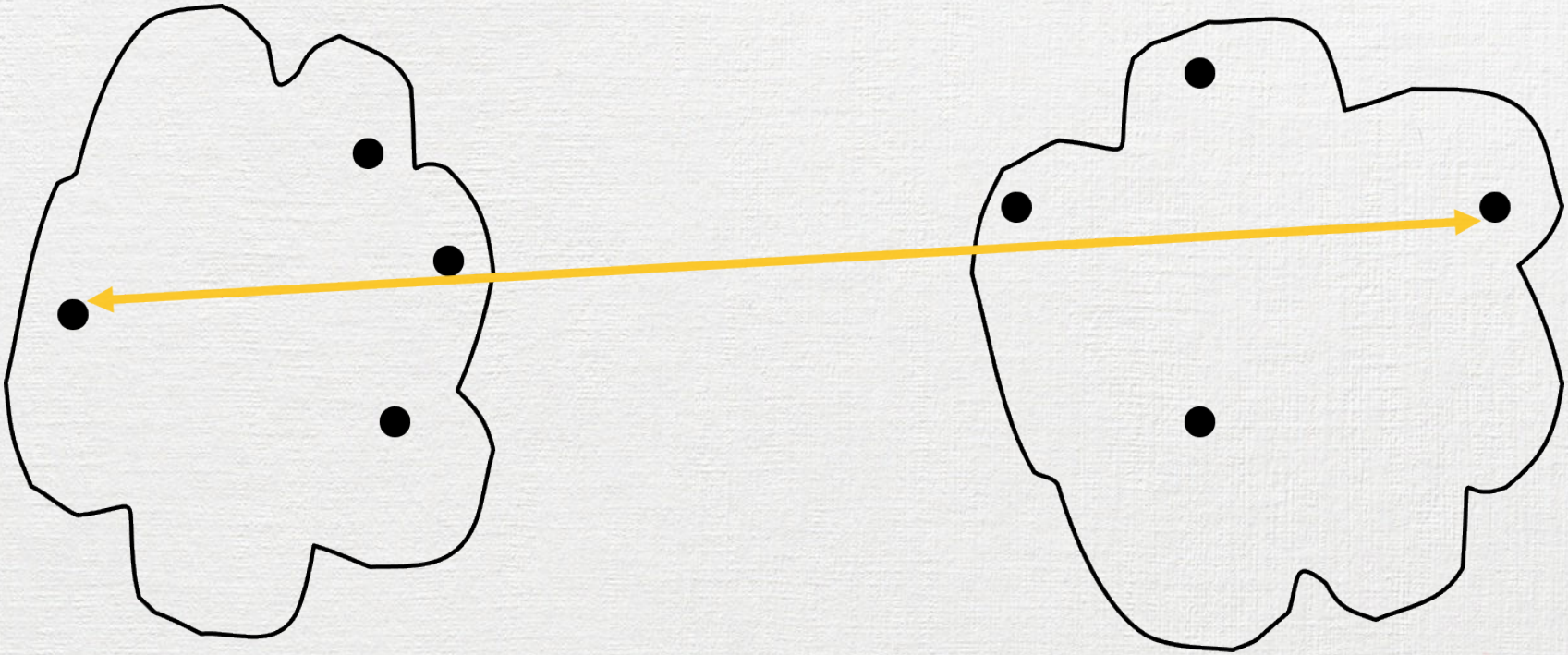
- Proximity of two clusters?
  - Min
  - Max
  - Average
  - Centroid

Agglomerative Hierarchical Clustering



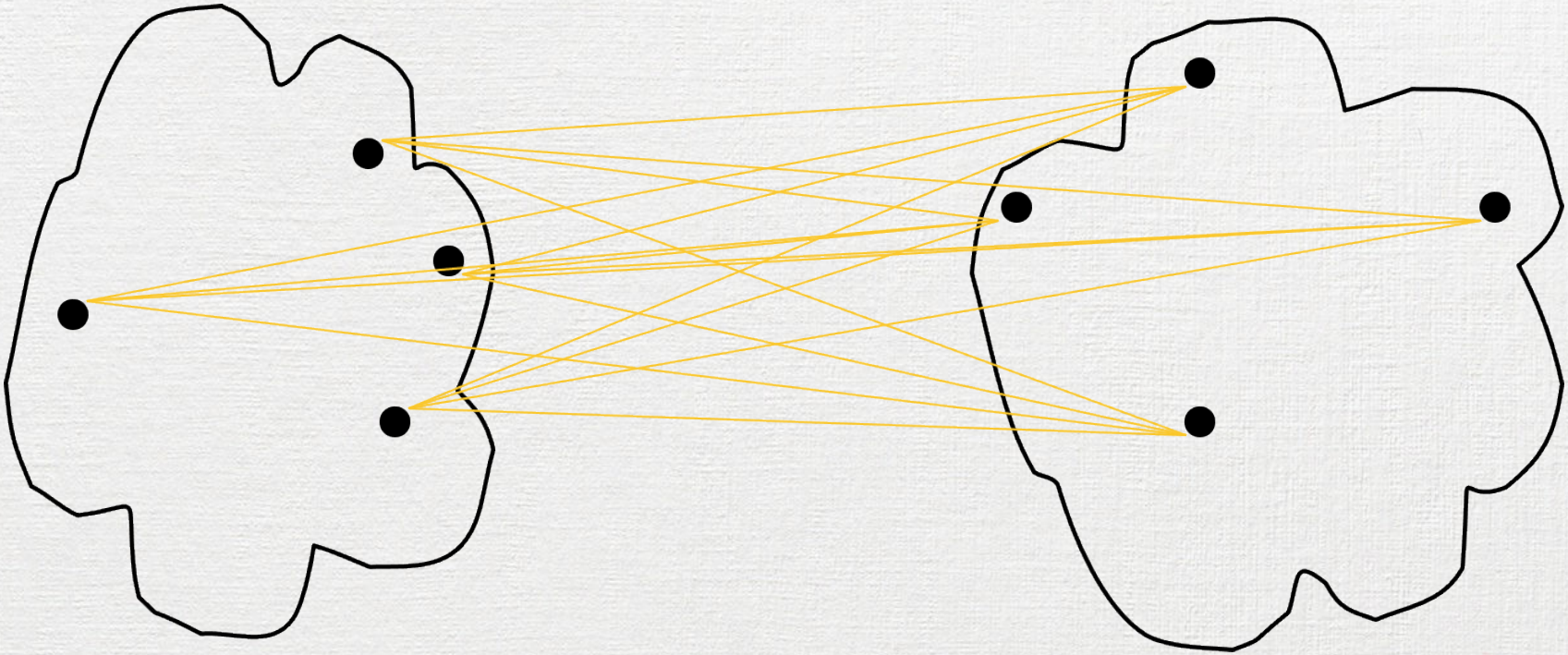
Min item distance

Agglomerative Hierarchical Clustering



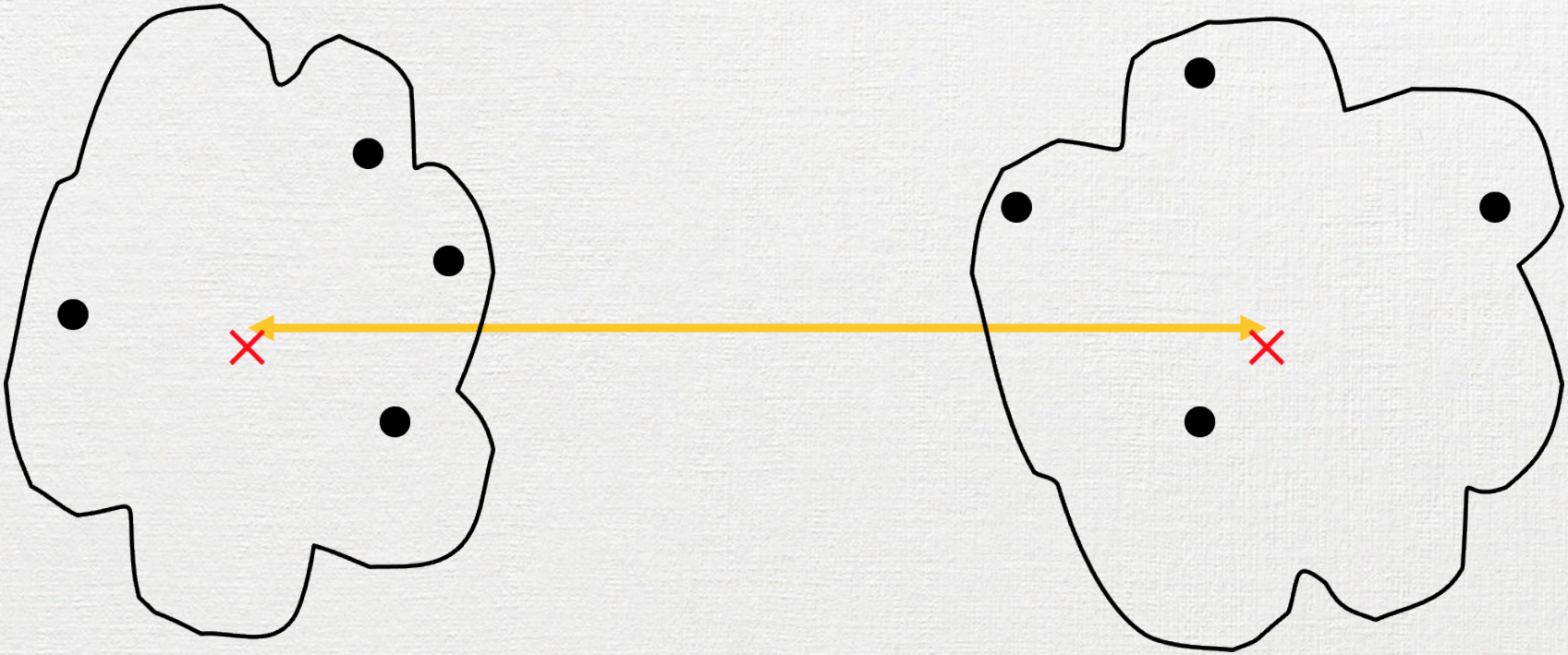
Max item distance

Agglomerative Hierarchical Clustering



Average item distance

Agglomerative Hierarchical Clustering

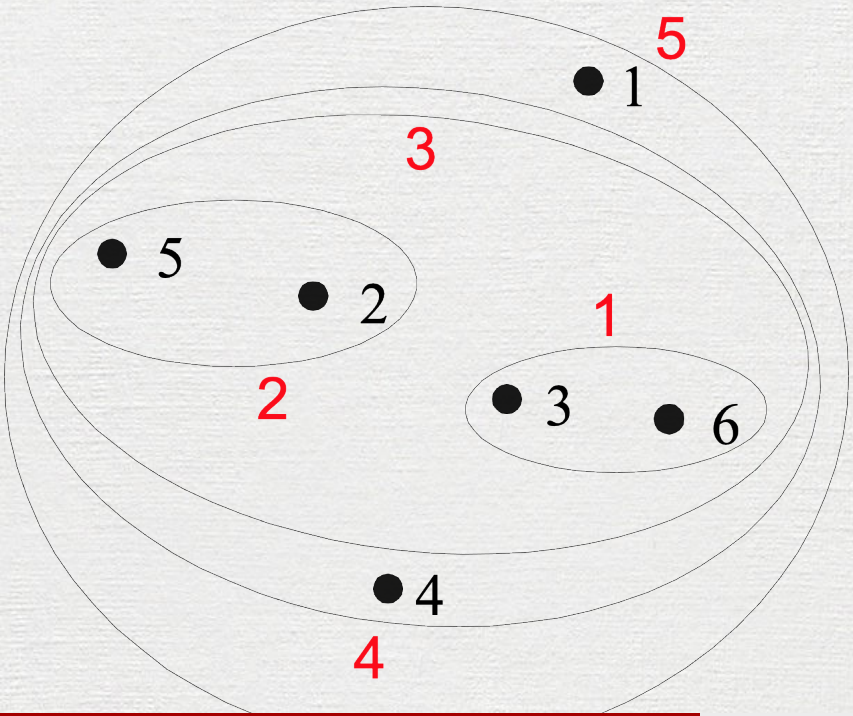


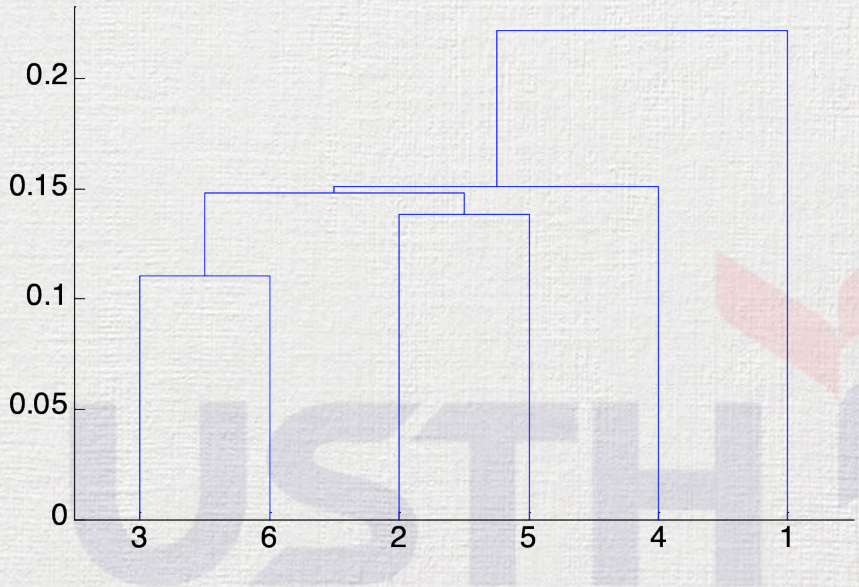
Centroid distance

Agglomerative Hierarchical Clustering

- Min

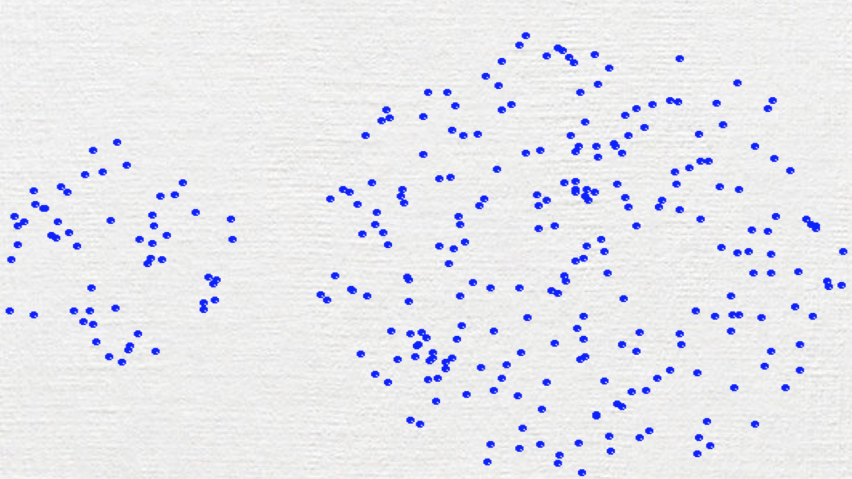
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



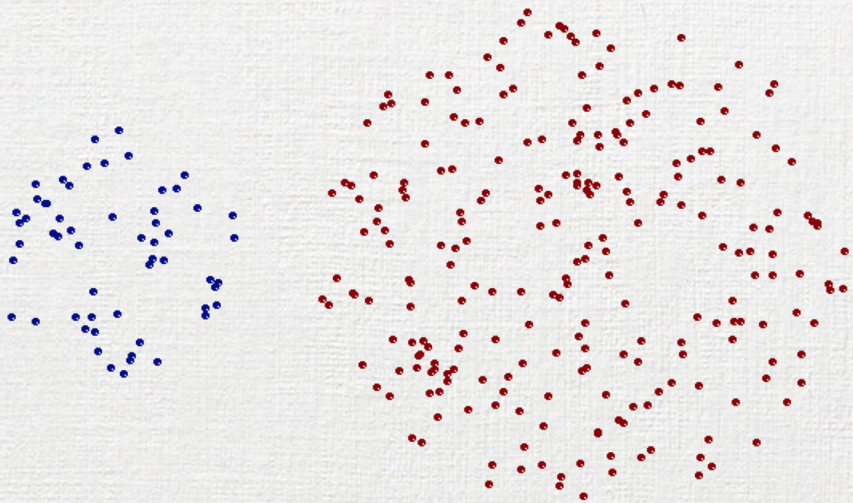


Agglomerative Hierarchical Clustering

- Min: can handle non-elliptical shapes...



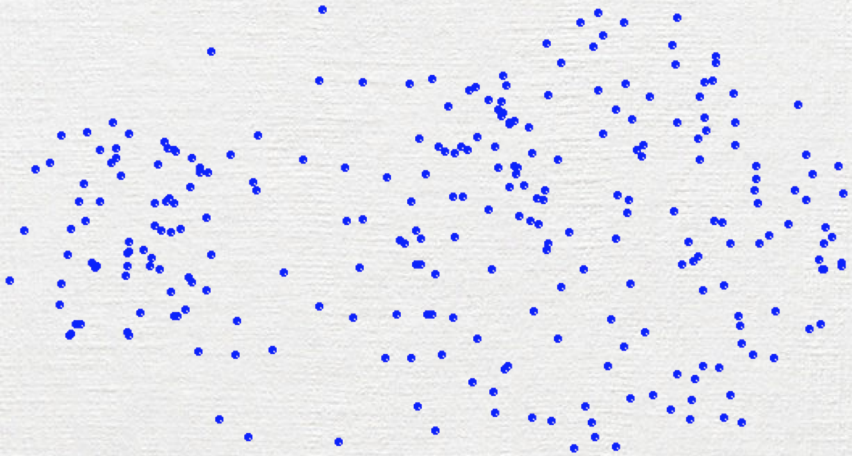
Input



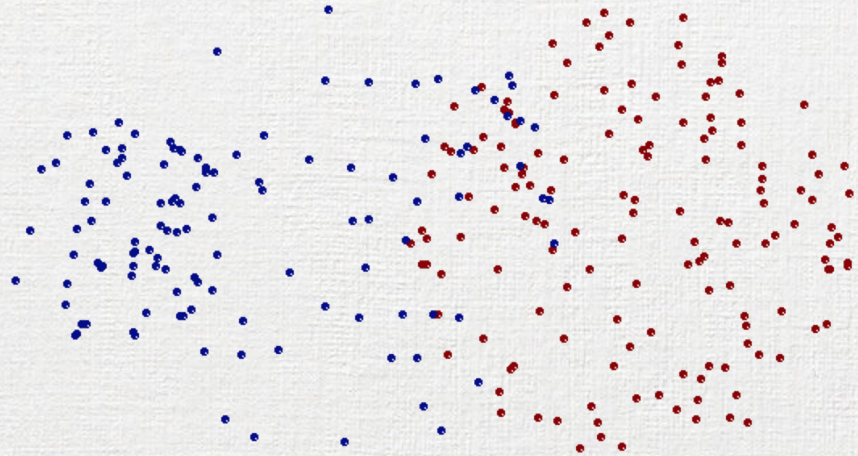
2 clusters

Agglomerative Hierarchical Clustering

- Min: . . . but prone to noise and outliers



Input

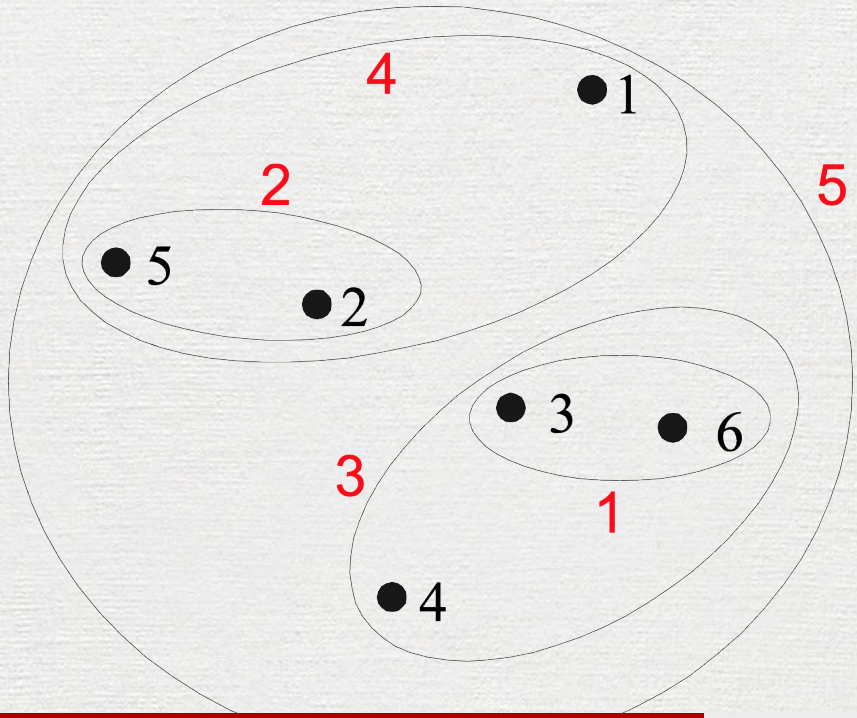


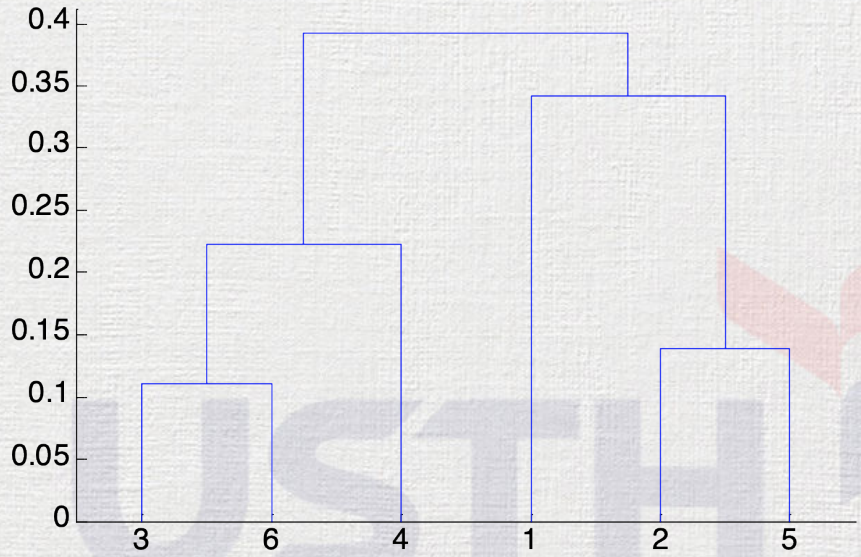
2 clusters

Agglomerative Hierarchical Clustering

- Max

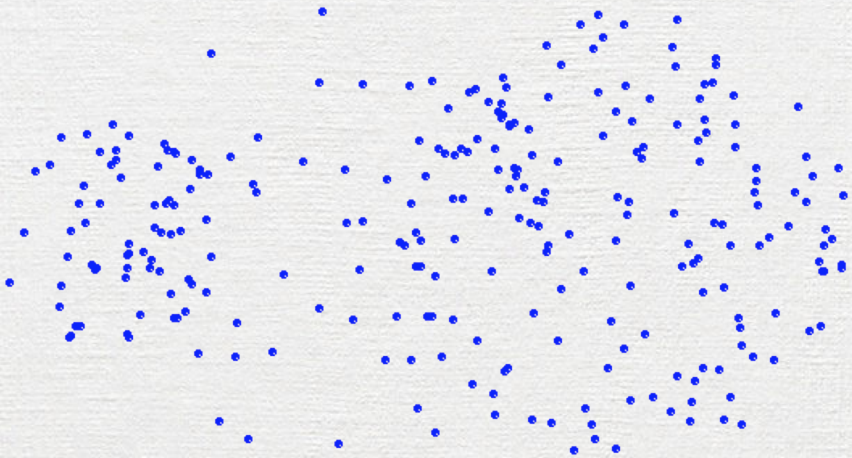
	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0



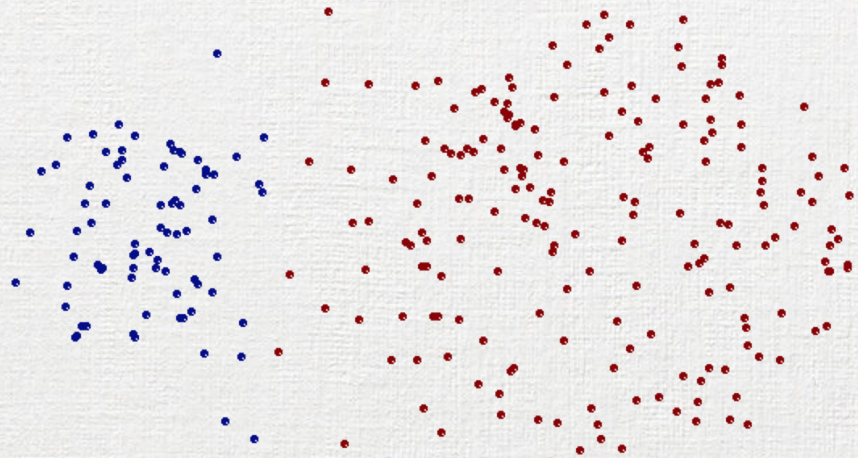


Agglomerative Hierarchical Clustering

- Max: less sensitive to noise...



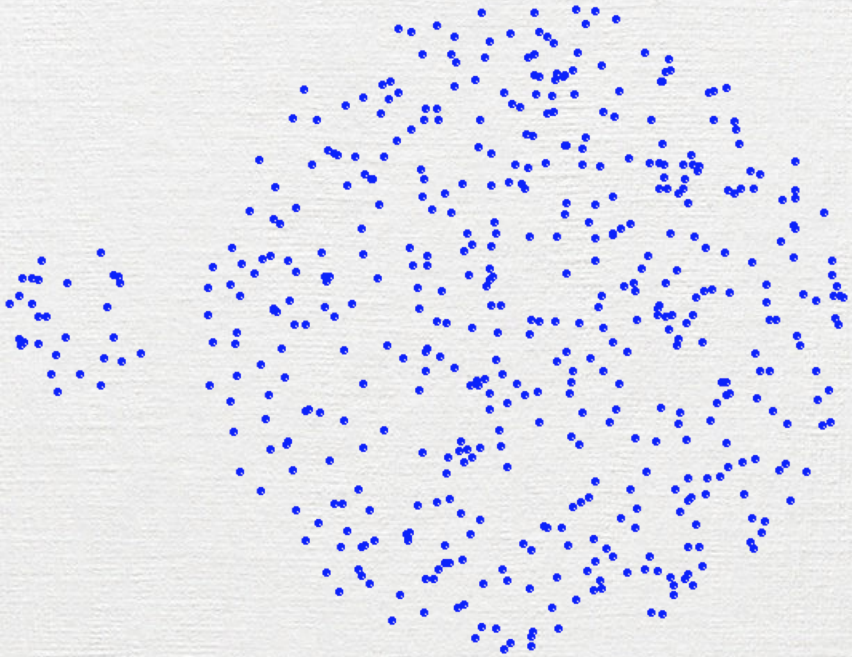
Input



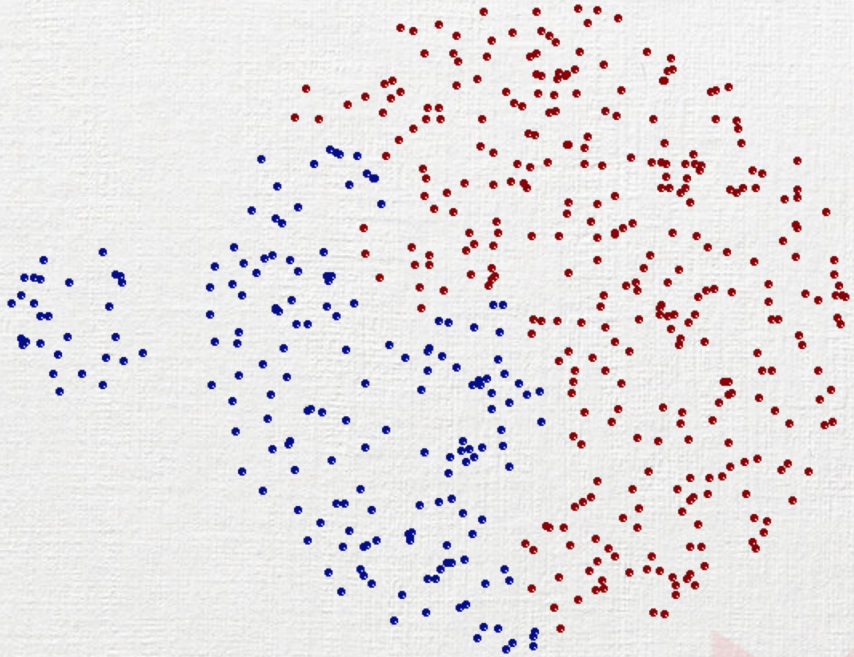
2 clusters

Agglomerative Hierarchical Clustering

- Min: . . . but can break large clusters



Input



2 clusters

Agglomerative Hierarchical Clustering

- Average: average of pairwise distance between points in two clusters

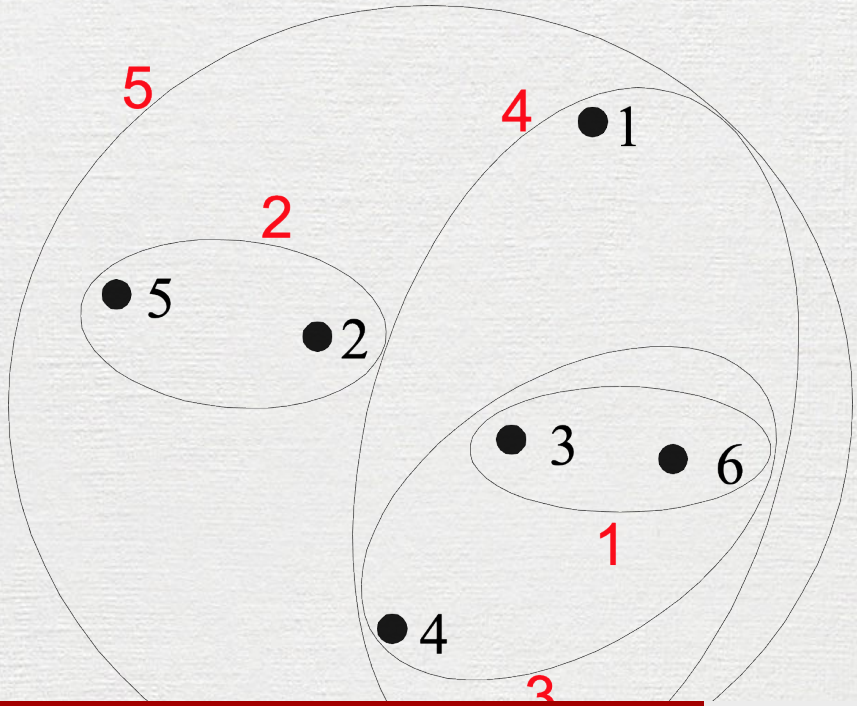
$$d(c_i, c_j) = \frac{\sum_{p_i \in c_i, p_j \in c_j} d(p_i, p_j)}{\mid c_i \mid * \mid c_j \mid} \quad (1)$$

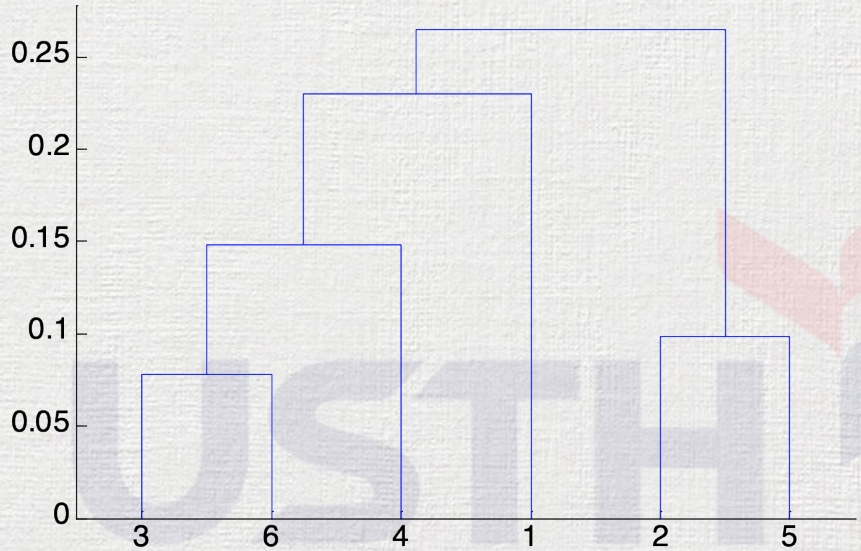
- More complex in time

Agglomerative Hierarchical Clustering

- Average

	1	2	3	4	5	6
1	0	.24	.22	.37	.34	.23
2	.24	0	.15	.20	.14	.25
3	.22	.15	0	.15	.28	.11
4	.37	.20	.15	0	.29	.22
5	.34	.14	.28	.29	0	.39
6	.23	.25	.11	.22	.39	0

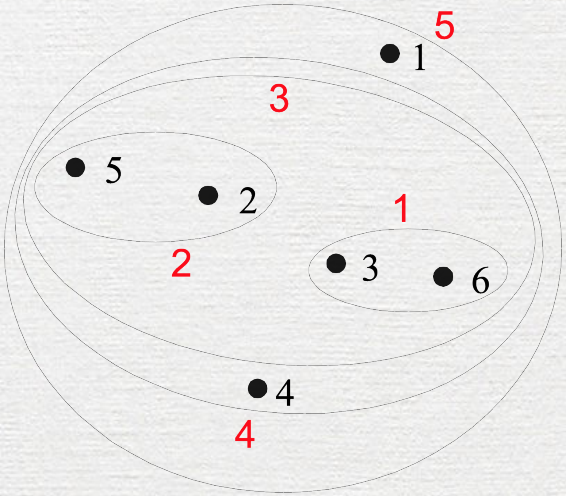




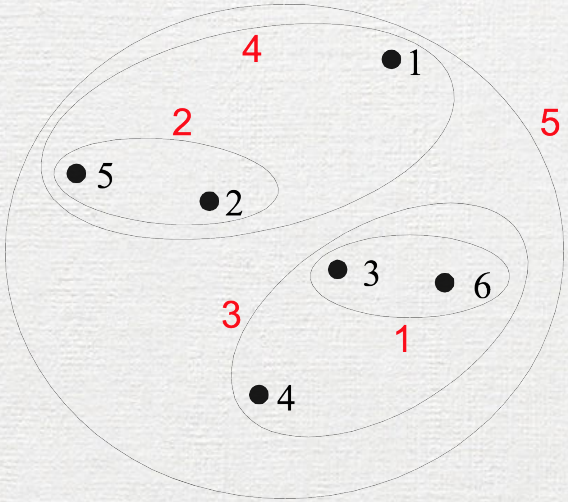
Agglomerative Hierarchical Clustering

- Average
  - Less sensitive to noise and outliers
  - Biased toward globular clusters

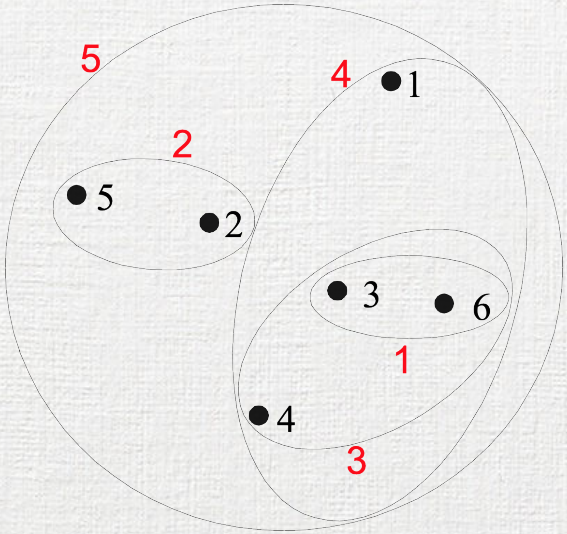
Agglomerative Hierarchical Clustering: Summary



Min



Max



Average

Agglomerative Hierarchical Clustering: Complexity

- Space: $O(N^2)$
  - N: number of points
- Time: $O(N^3)$
  - N steps
  - N^2 updating proximity matrix

Practical work 3: Hierarchical Clustering

- Implement hierarchical clustering
  - Min, max
  - Cluster the reviews into 3 clusters using its length
    - Expected: short, medium, long reviews
  - Name your source code «03.review.length.clustering.py»
- Push your code to corresponding forked Github repository

K-means

What

- Partitional clustering method
  - Assigns items to each cluster
  - Find mutually exclusive cluster of spherical shape based on distance
    - One data point belongs to only one cluster
- K: number of clusters, predefined parameter
- Centroid: center point of a cluster

Why

- Simple!
- Less computationally intensive
- Can be suitable for large dataset
- Guaranteed convergence, quite fast

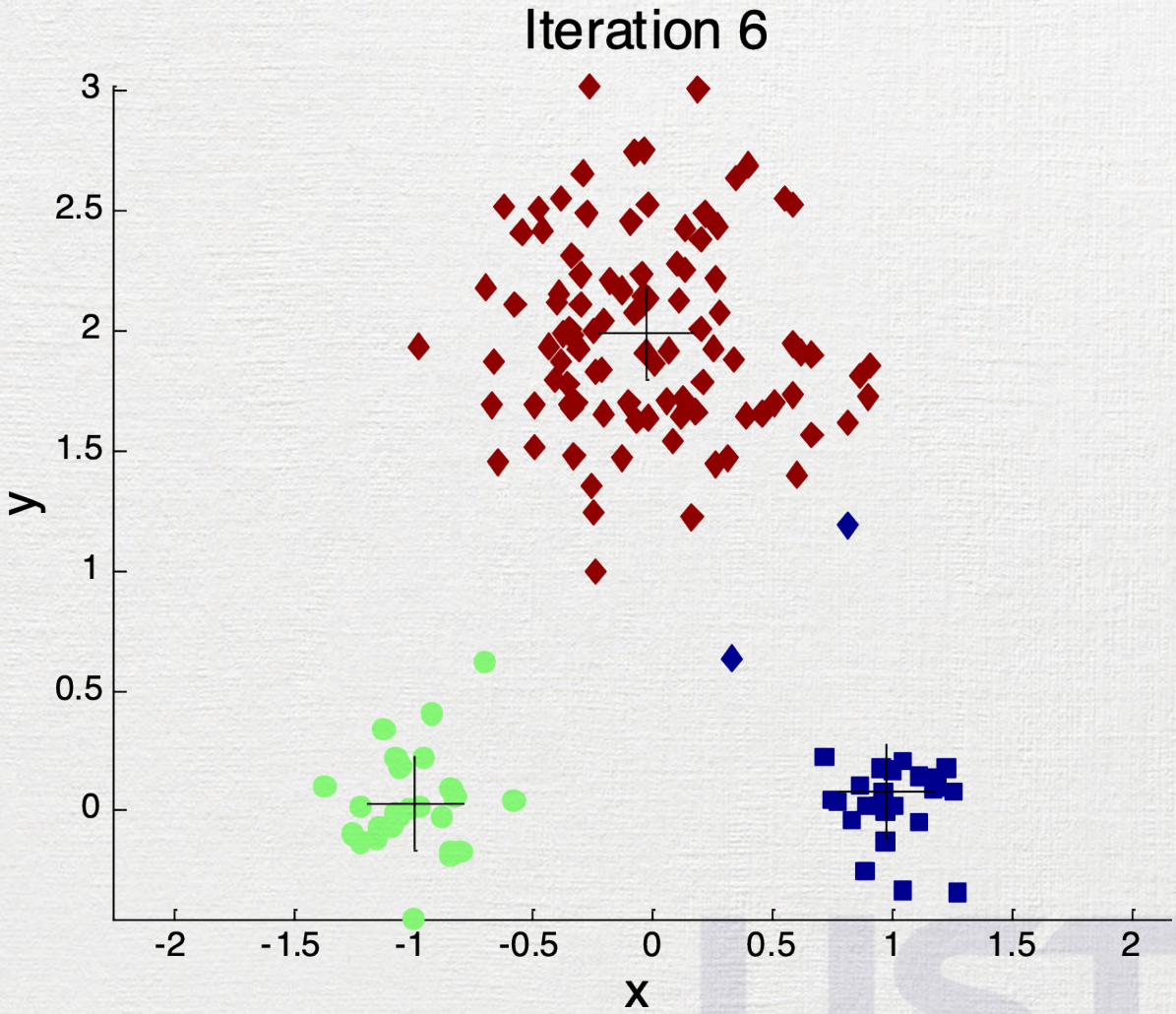
How

Algorithm 1: K-means clustering method

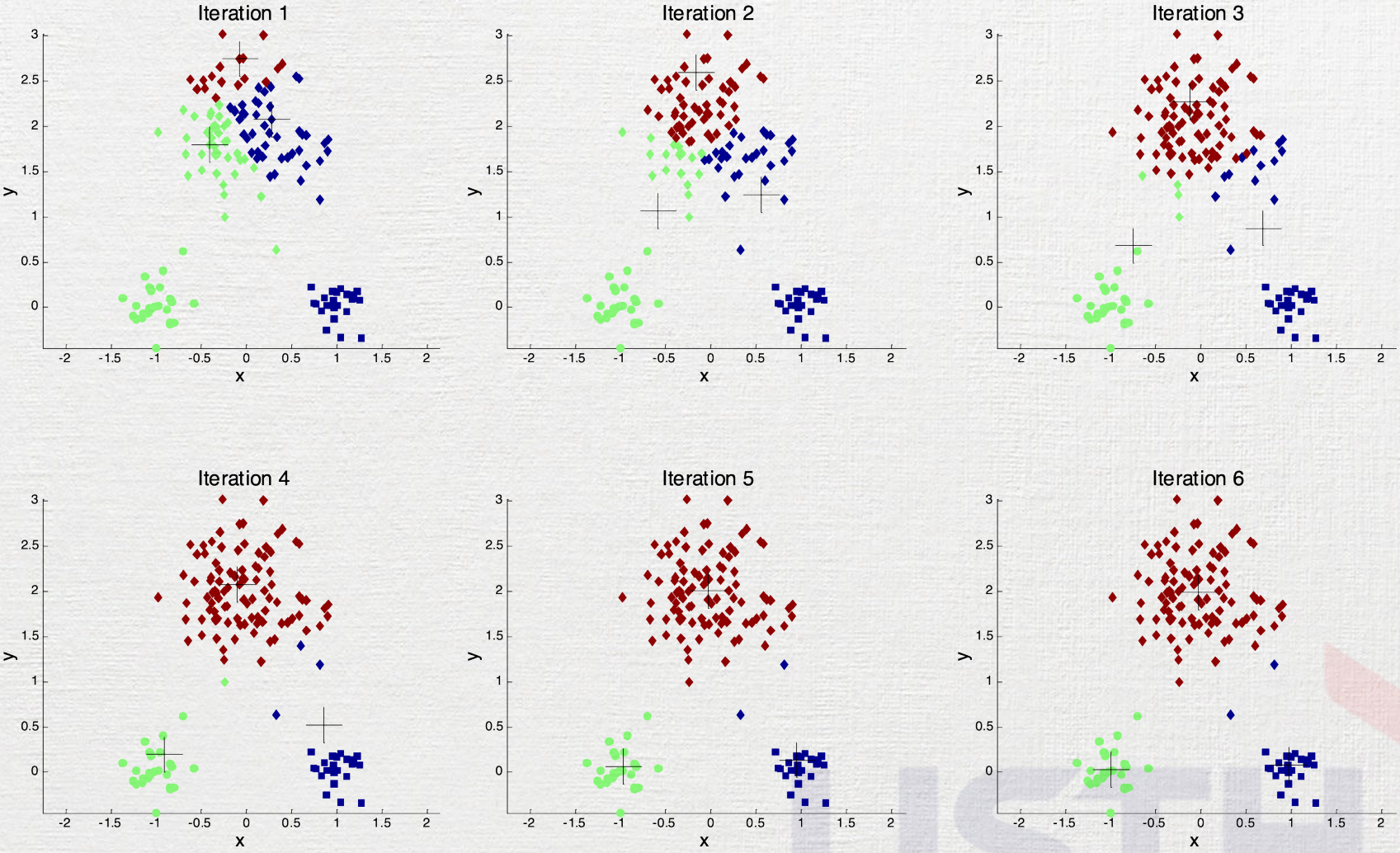
Input : Set of points P_i
Output: Set of clusters $\hat{C}$

```
1 Select K initial centroids;
2 repeat
3 |   Assign all points to the closest centroid;
4 |   Recompute centroids;
5 until Centroids don't change;
```

Example



Example


Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5
Iteration 6
x
y

How

- Time complexity: $O(n * K * I * d)$
  - n: number of points
  - K: number of clusters
  - I: number of iterations
  - d: number of attributes

Formulation

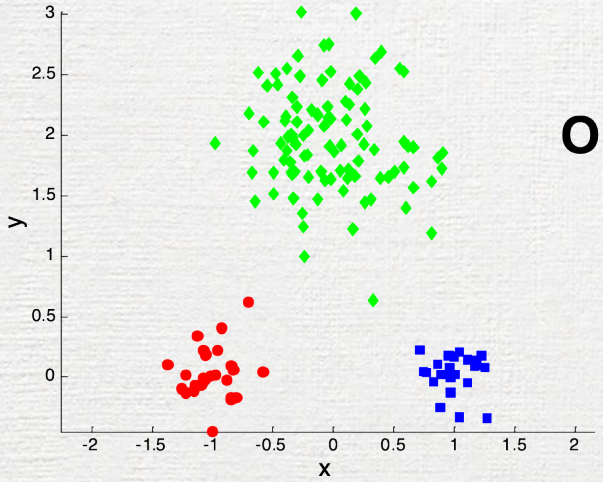
- K-means as an optimization problem
  - Find centroids m_i of cluster C_i
  - Subject to minimizing Sum of Squared Errors (SSE)

$$\text{Minimize } SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(x, m_i) \tag{2}$$

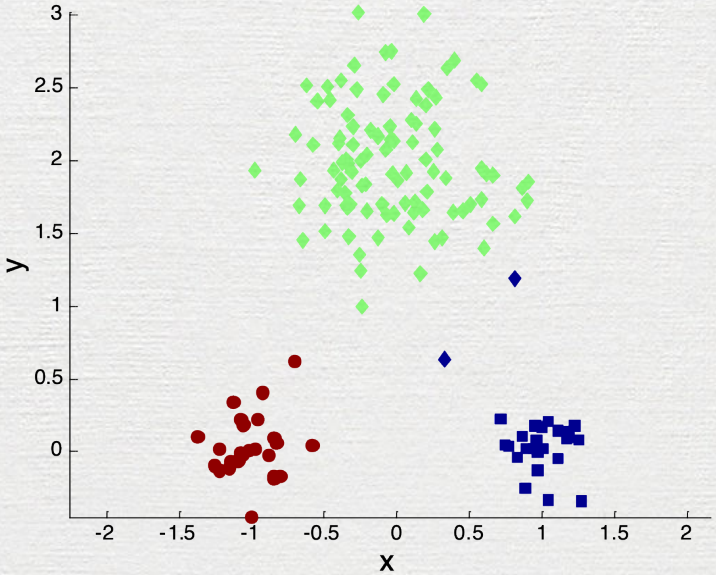
How

- Selecting initial centroids
  - Hardcoded (!)
  - Randomly
  - Farthest
- Centroid: center of a cluster
  - Mean
  - Median
- Distance between points
  - Manhattan (L1)
  - Euclidean (L2)

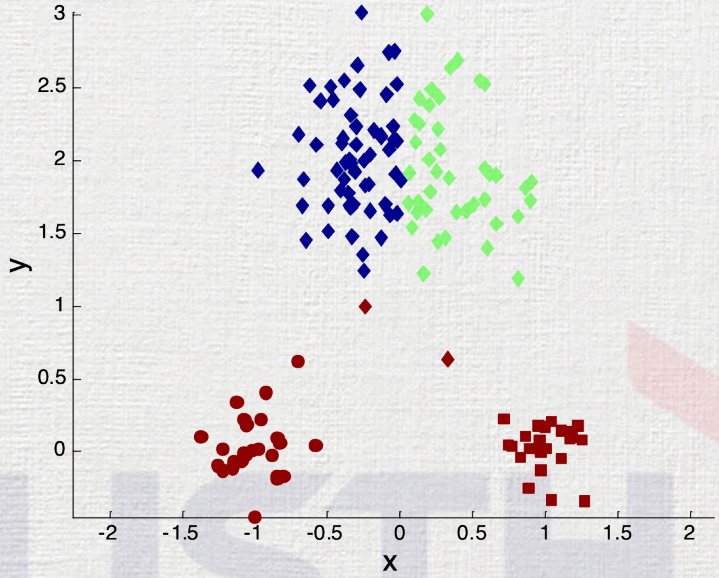
Importance of initial centroids



Original Points

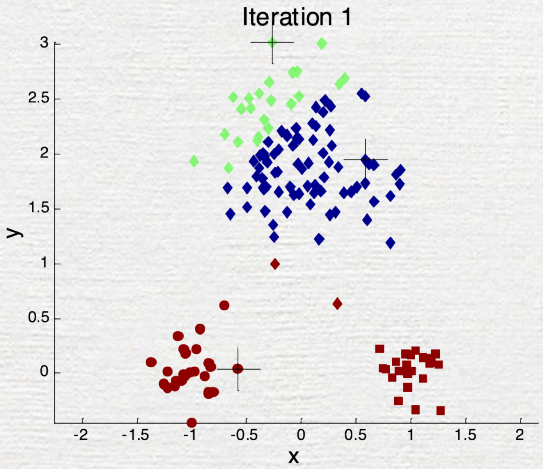


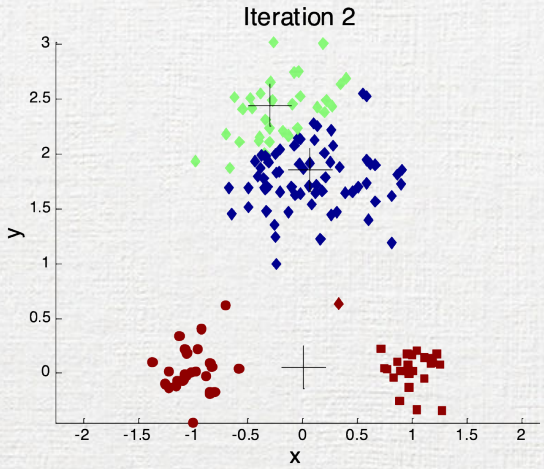
Optimal Clustering

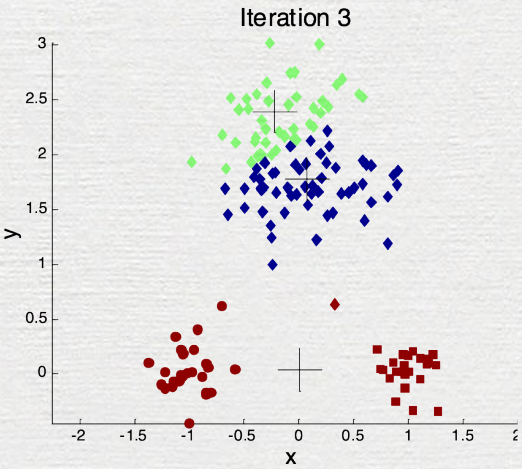


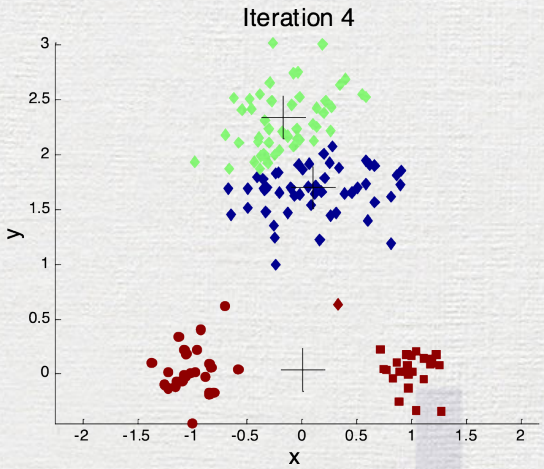
Sub-optimal Clustering

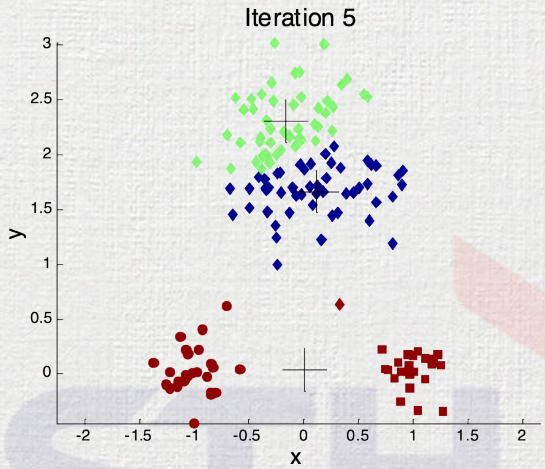
Importance of initial centroids











Practical work 4: K-means Clustering

- Implement K-means clustering
  - Random initial centroids
  - Mean centroids
  - Euclidean distance
  - Cluster the reviews into 3 clusters using its length
    - Expected: short, medium, long reviews
  - Compare the results with practical work 3
  - Name your source code «04.review.length.kmeans.py»
- Push your code to corresponding forked Github repository

Mean-shift

What

- An unsupervised, nonparametric clustering technique
- Centroid-based algorithm
- No prior knowledge of the number of clusters
- No constrain the shape of the clusters

Why

- No more K
- No more initial centroid problem
- Density based method
- Better cluster shape prediction
  - Not only globular
  - Nonconvex shapes
- No local minima problems like K-means
- Guaranteed convergence

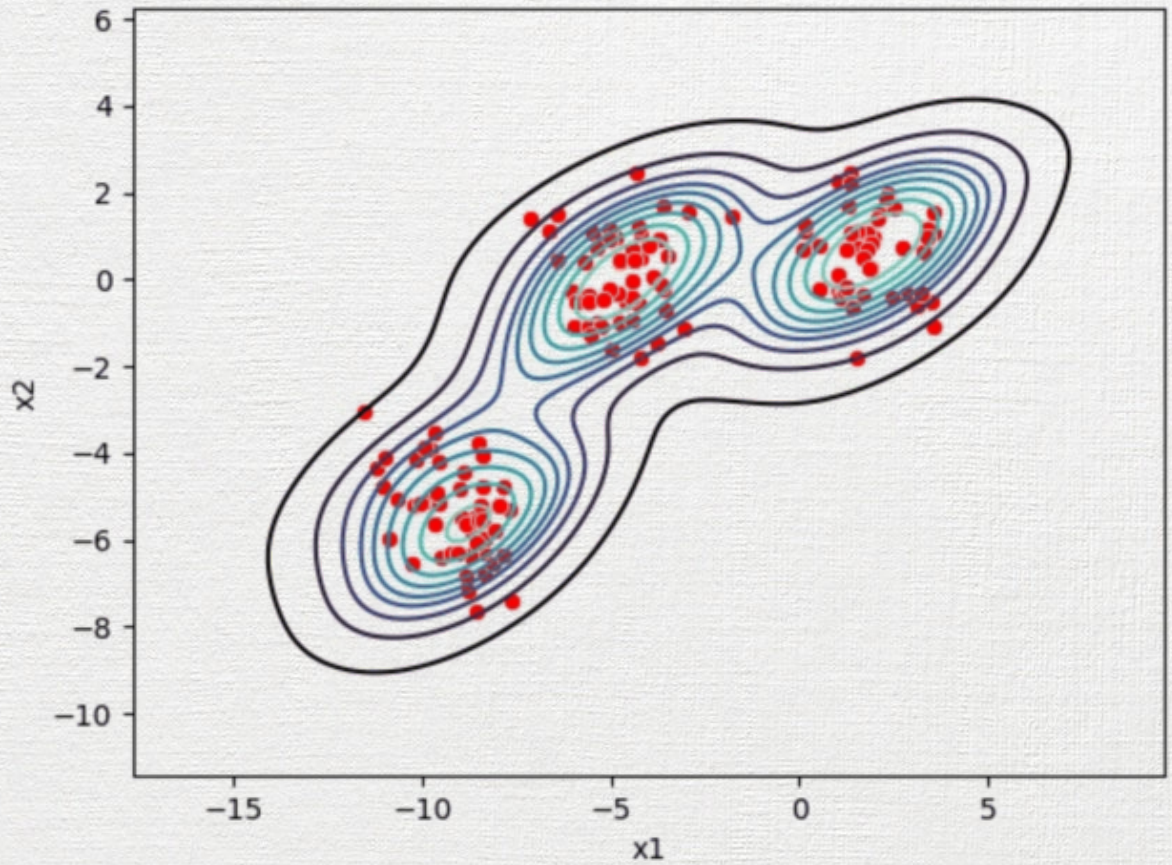
Formal definition

- Input: a set of n points $x_i, 1 \leq i \leq n$
- Model
  - Non parametric statistical model
  - A probability density function
- Output: for any given point:
  - Closest local mode of density function

How

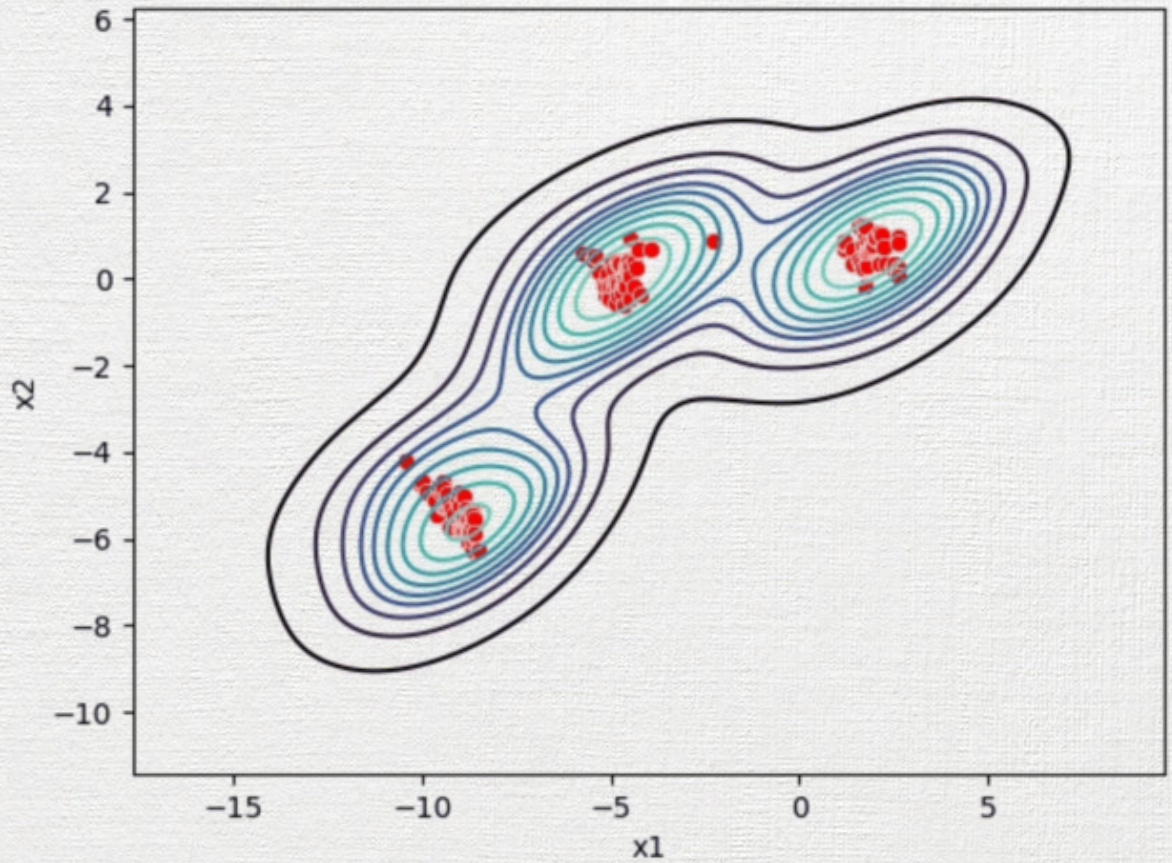
- Iterative method
- Mode: estimated center mass of the area around a point
- For each data point
  - Shift its mode to the highest density center
  - Repeat the previous shift until its mode converges
- Merge all data points having the same mode into a cluster

Example



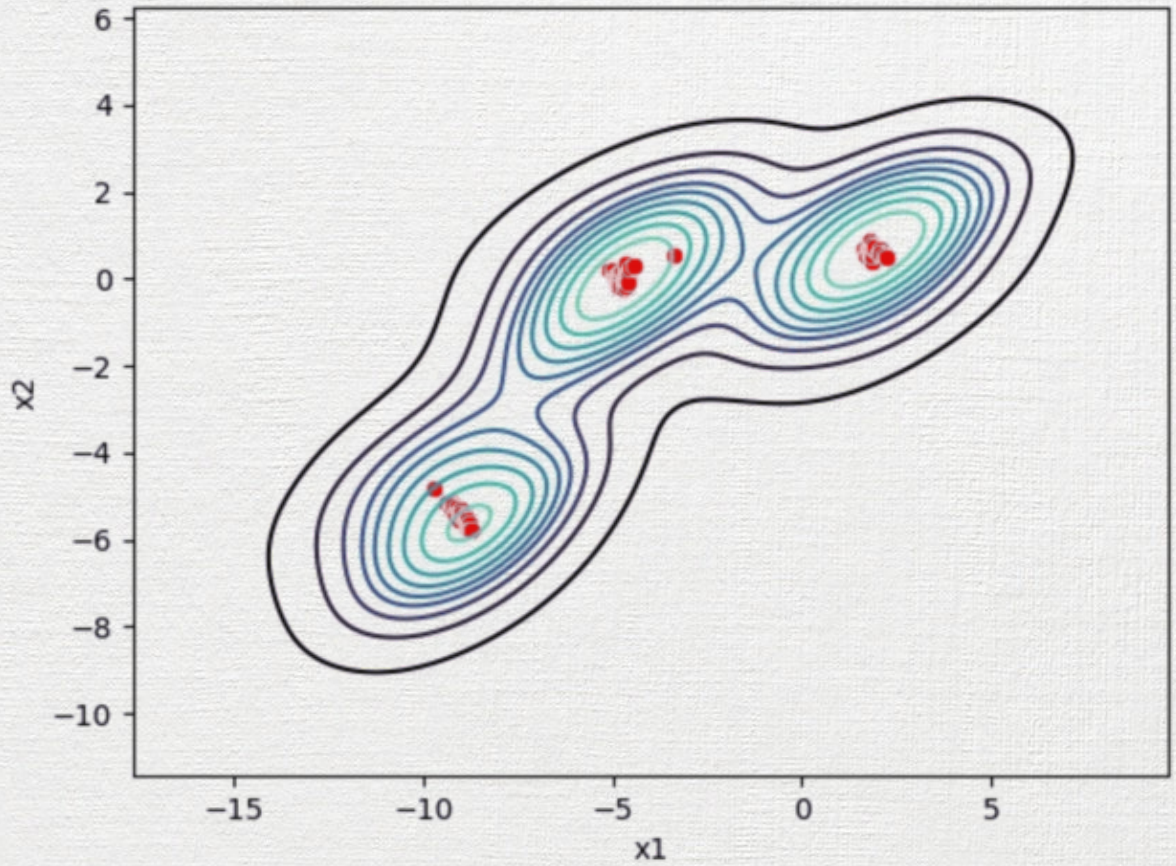
Input data

Example



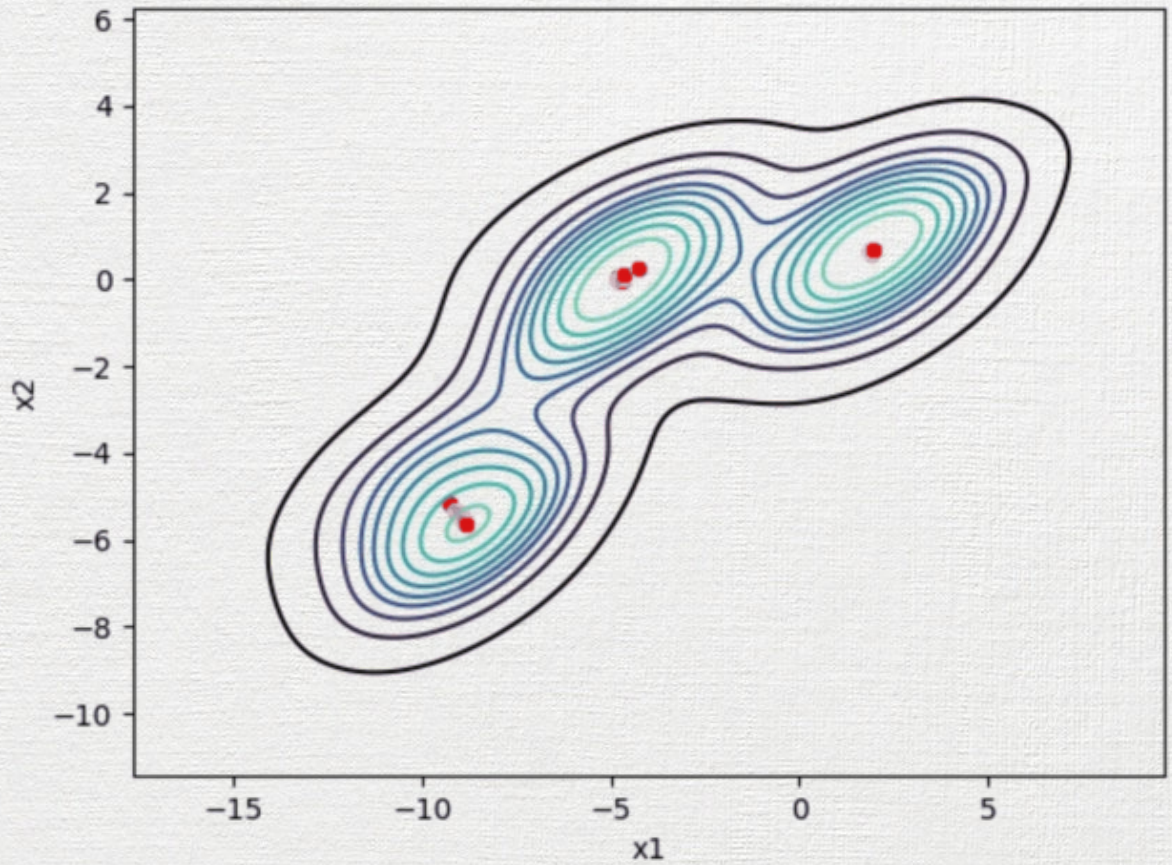
Iteration 1

Example



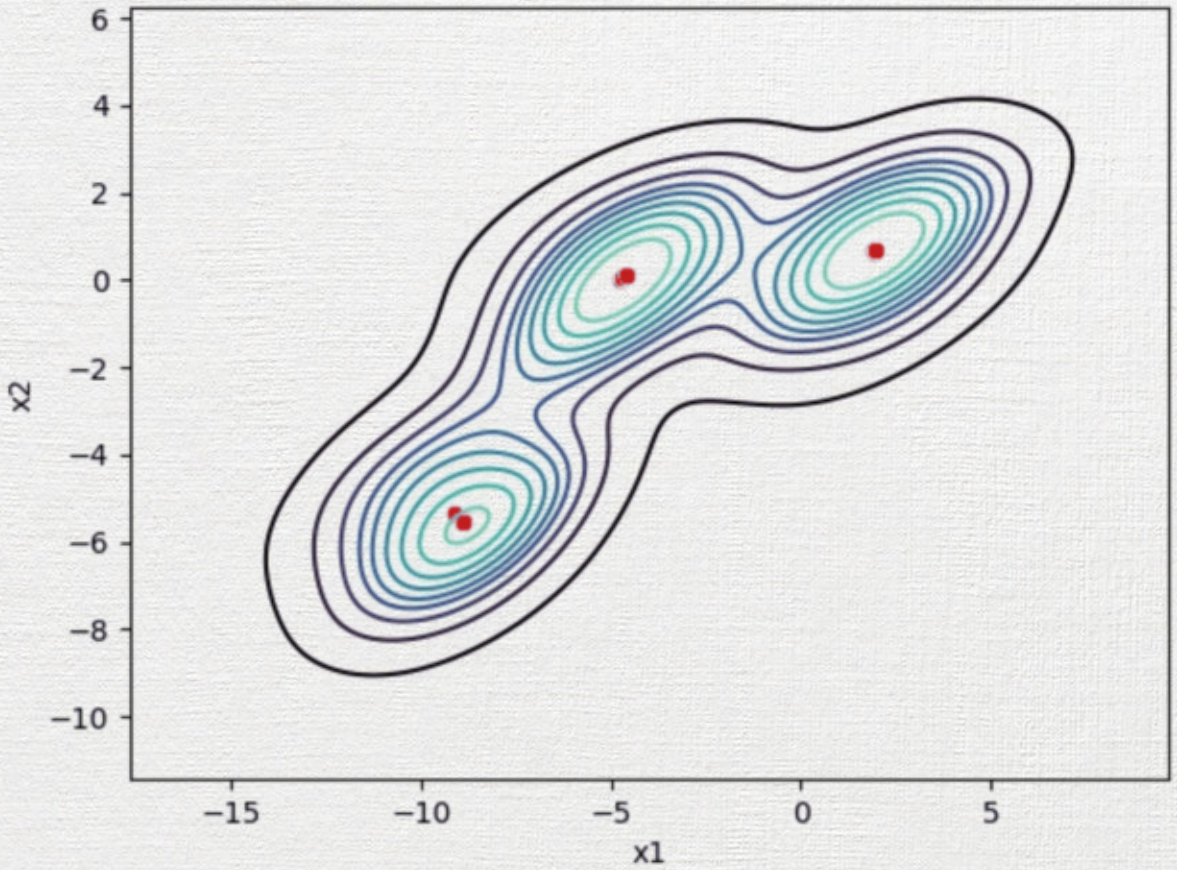
Iteration 2

Example



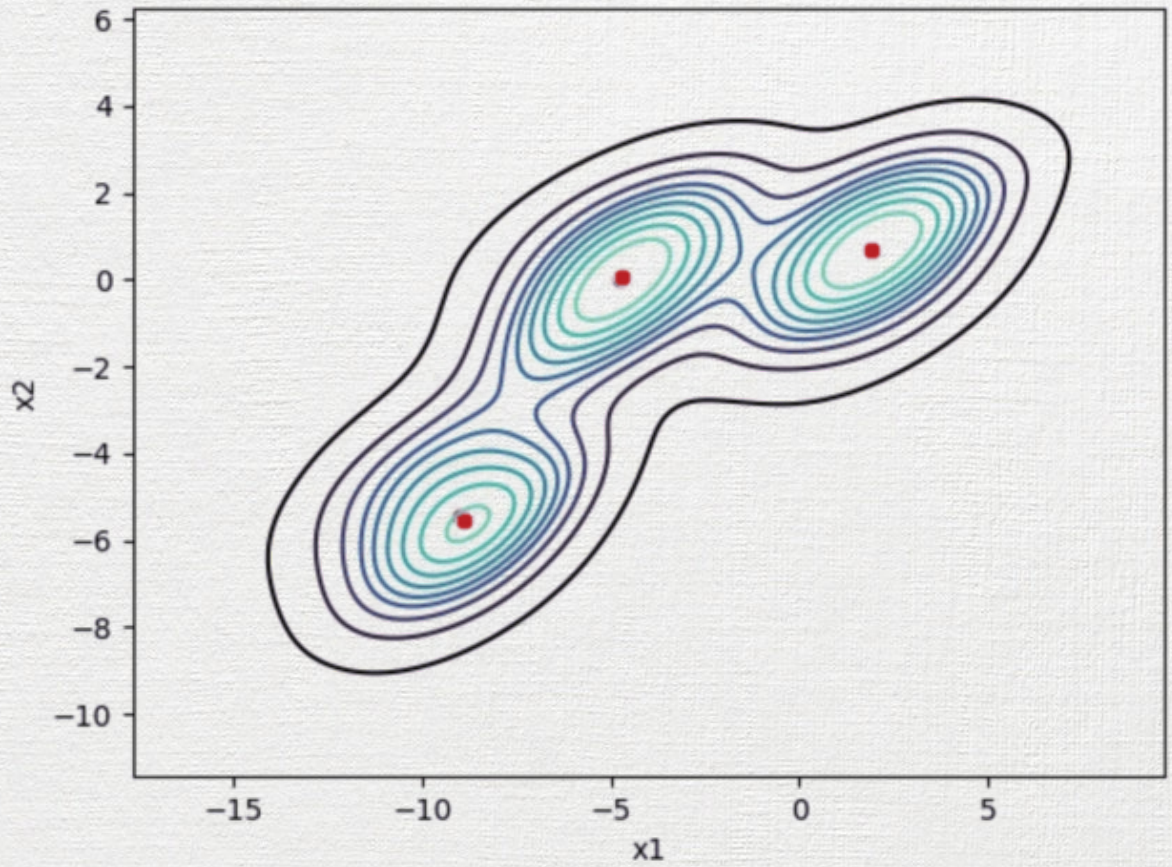
Iteration 3

Example



Iteration 4

Example



Iteration 5

How

Algorithm 2: Mean-shift clustering method

Input : Set of points x_i, bandwidth h, threshold τ
Output: Set of modes $\hat{C}$

```
// shift mode to highest density centers;
for i = 0 to n − 1 do
    m = 0;
    mode[i][m] = x_i;
    repeat
        mode[i][m + 1] = shiftMode(mode[i][m]);
        m = m + 1;
    until dist(mode[i][m], mode[i][m − 1]) < τ;
    mode[i][0] = mode[i][m];
end
Ĉ = [];
for i = 0 to n − 1 do
    if mode[i][0] not in Ĉ then
        Ĉ += [mode[i][0]];
    end
end
return Ĉ;
```

How

- How to estimate density of an area?
- How to calculate distance to shift in `shiftMode()`?

How: PDF estimation

- Kernel:
  - A non-negative real-valued integrable function $K(x)$
  - For estimating probability density function gradient
  - Bandwidth h: radius parameter to estimate density

How: PDF estimation

- Kernel:
  - A non-negative real-valued integrable function $K(x)$
  - For estimating probability density function gradient
  - Bandwidth h: radius parameter to estimate density
- In English: a function used to estimate the density of an area

How: PDF estimation

- Kernel requirements
  - Normalized

$$\int_{\mathbb{R}^d} K(x)\,dx = 1 \tag{3}$$

  - Symmetrics

$$\int_{\mathbb{R}^d} xK(x)\,dx = 0 \tag{4}$$

  - Exponential weight decay

$$\lim_{||x||\to\infty} ||\ x\ ||^d\ K(x) = 0 \tag{5}$$

How: PDF estimation

- Common kernel profiles for previous Kernels
  - Flat kernel

$$k_F(x) = \begin{cases} 1 & \text{if } x \leq h, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

  - Gaussian kernel

$$k_G(x) = e^{-\frac{1}{2}x^2} \tag{7}$$

How: Shifting mode

- Starting with a given mode $x_i^0 = x_i$, $x_i \in \mathbb{R}^d$
- Next mode position x_i^{m+1} can be calculated from current mode position x_i^m

$$x_i^{m+1} = \frac{\sum_{j=1}^{n} x_j k(x_j - x_i^m)}{\sum_{j=1}^{n} k(x_j - x_i^m)}. \tag{8}$$

Practical work 5: Mean-shift Clustering

- Implement Mean-shift clustering
  - Cluster the reviews using its length
    - Expected: short, medium, long reviews
    - Try with different bandwidth h values
  - Compare the results with practical work 3 and 4
  - Name your source code «05.review.length.meanshift.py»
- Push your code to corresponding forked Github repository