Clustering
-K-means
-Hierarchical clustering

Lin Guosheng
School of Computer Science and Engineering
Nanyang Technological University

Outline

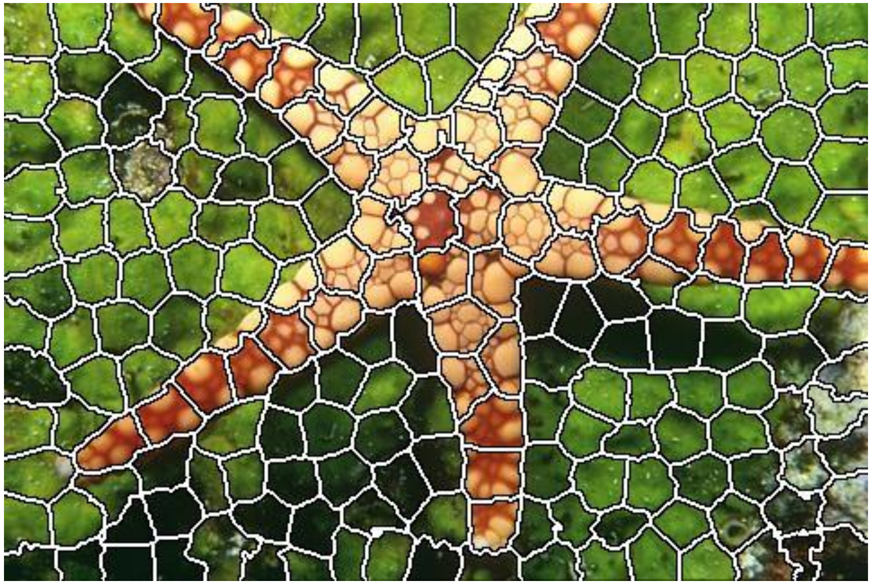
- **Basic methods for clustering:**
  - **K-means**
    - Algorithm
    - Extension: K-means++
    - K-means with clustroid **(non-examinable)**
    - Optimization problem **(non-examinable)**

    (non-examinable content may be useful for your coursework projects)
  - **Hierarchical Clustering**

Application

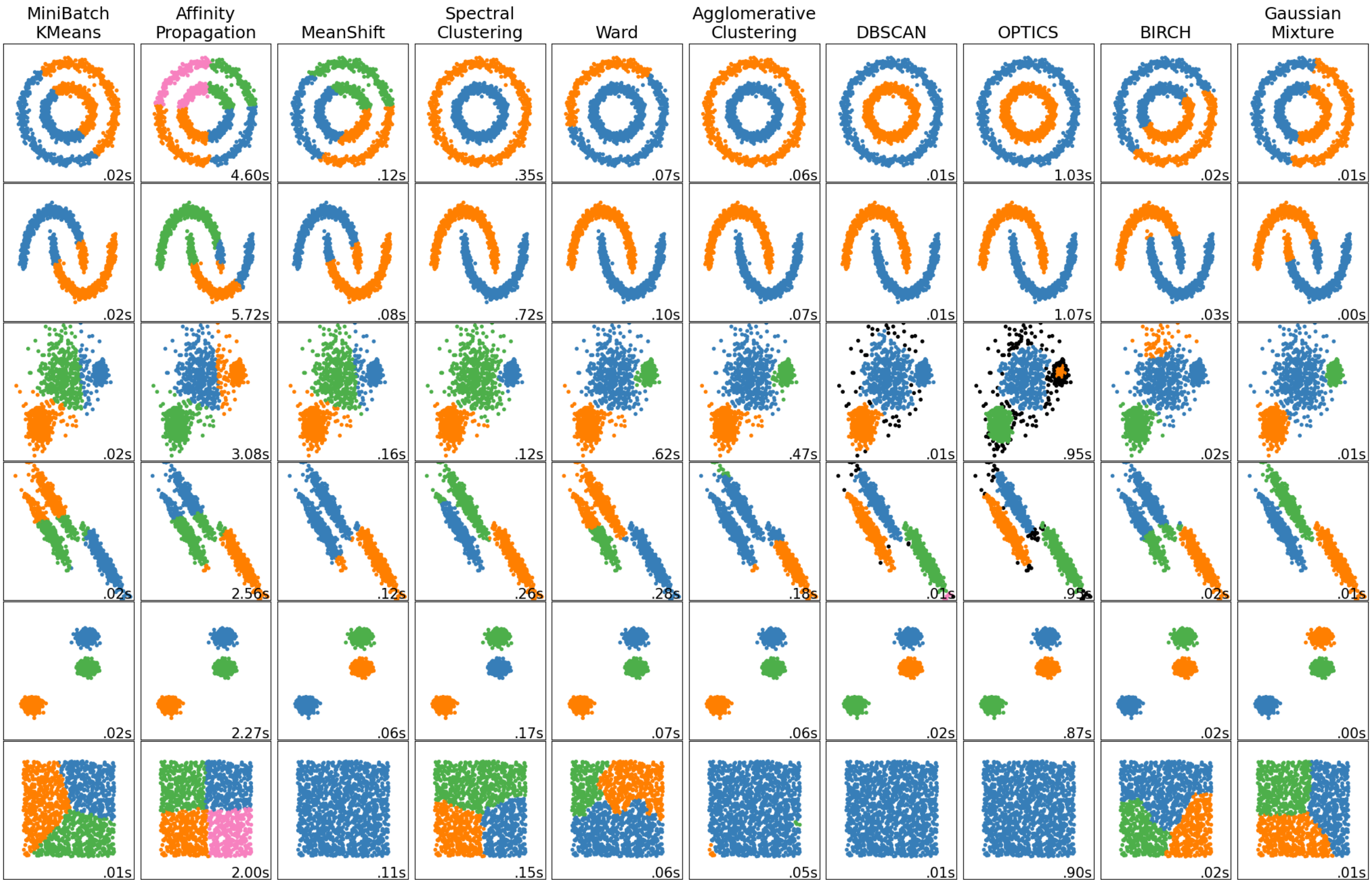
Image clustering

https://pythonawesome.com/image-classification-feature-visualization-and-transfer-learning-with-keras/



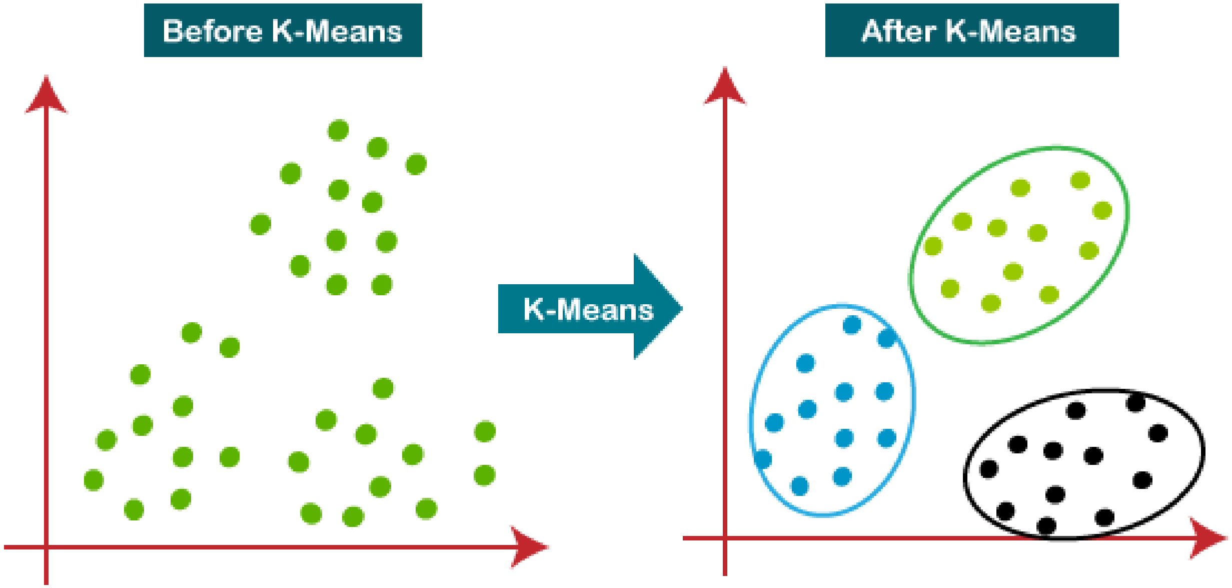
Cluster image pixels to generate supperpixels

https://www.epfl.ch/labs/ivrl/research/slic-superpixels/

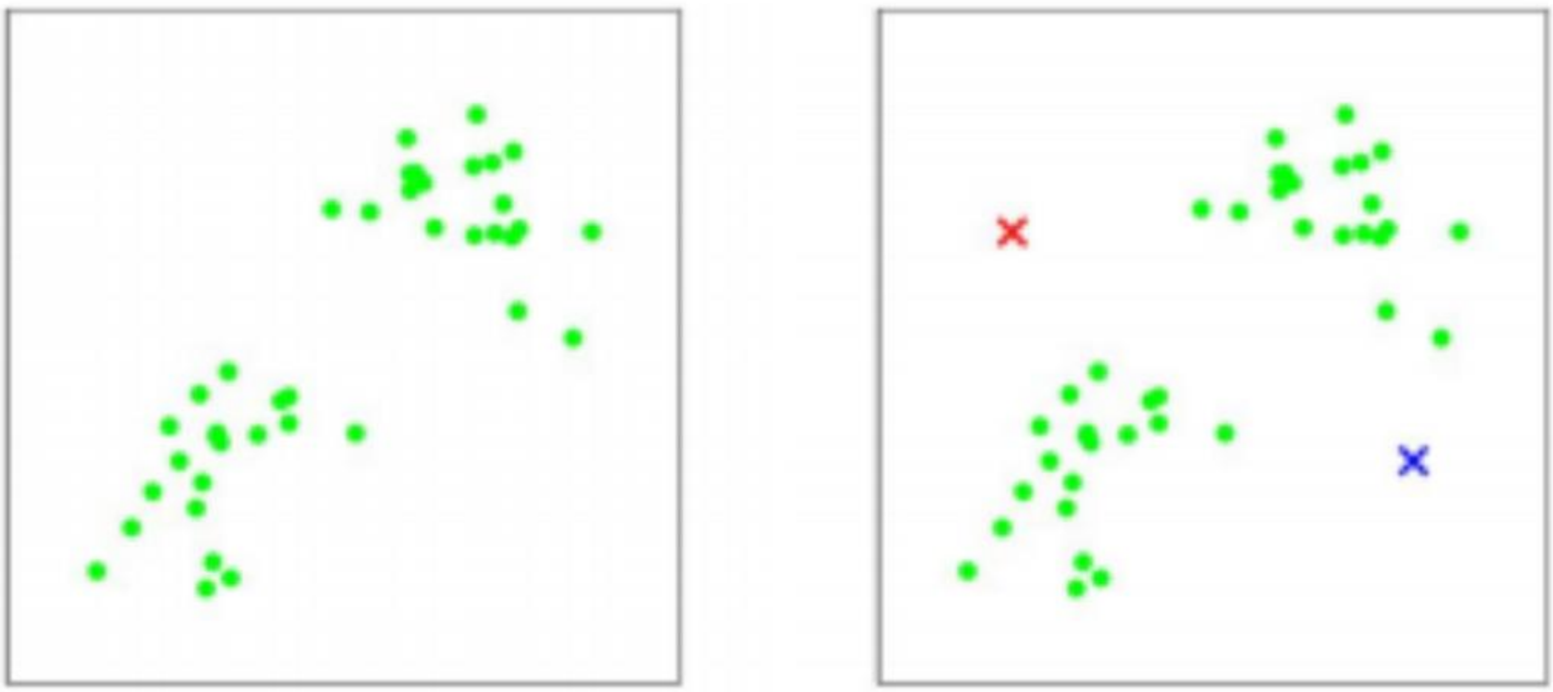


https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

K-means



A simple example

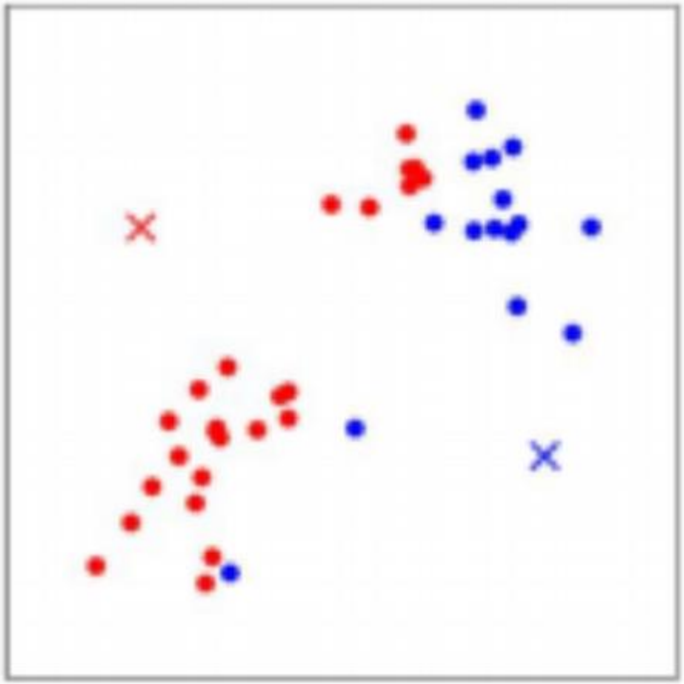


(a) (b)

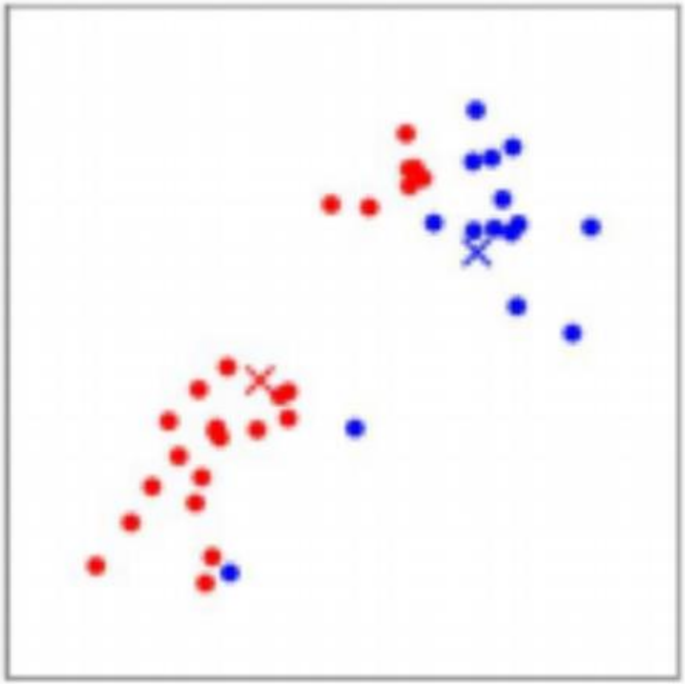
Input data points
(one point indicates one example,
feature dimension = 2)

Random initialized cluster centroids
(K=2)

Iteration 1:



(c)

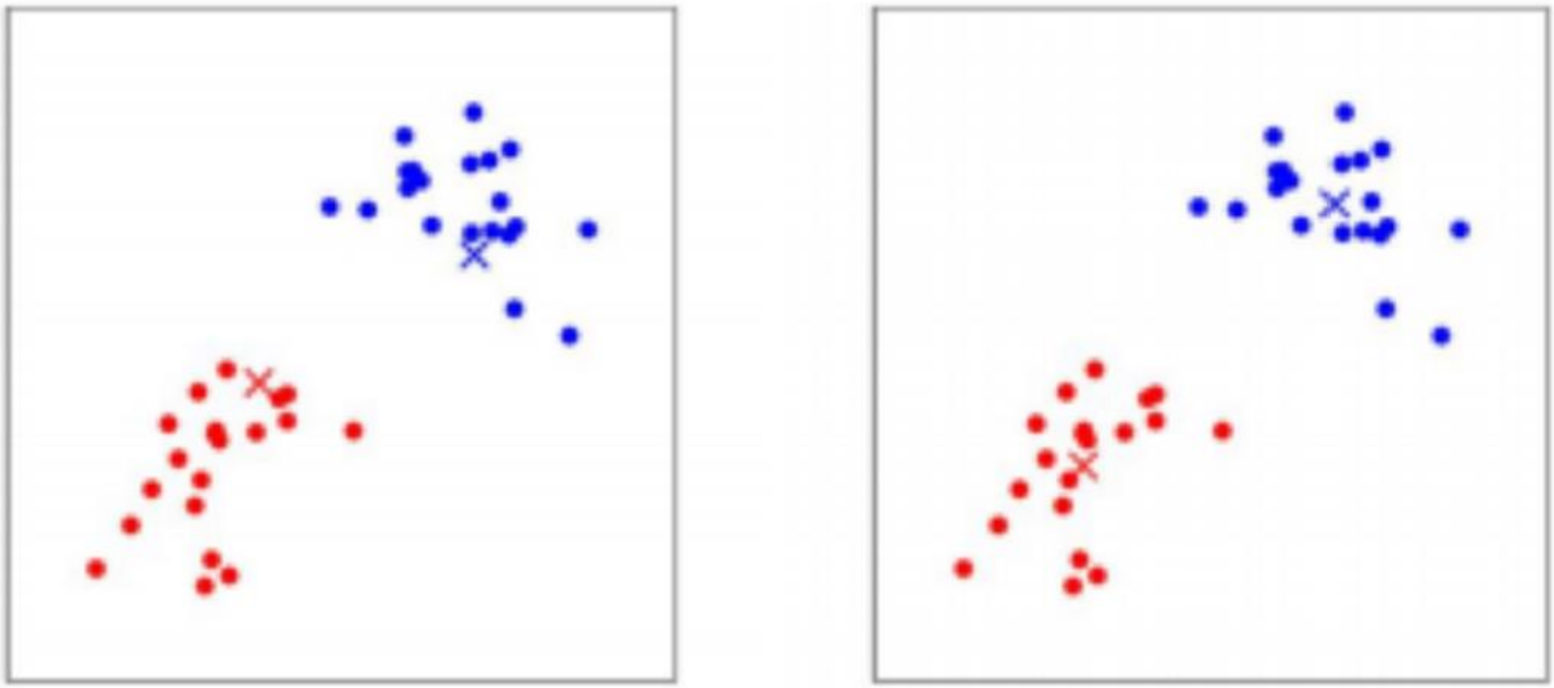


(d)

Assignment step
(assign each point to the nearest cluster)

Centroid update step
(compute the mean in each cluster)

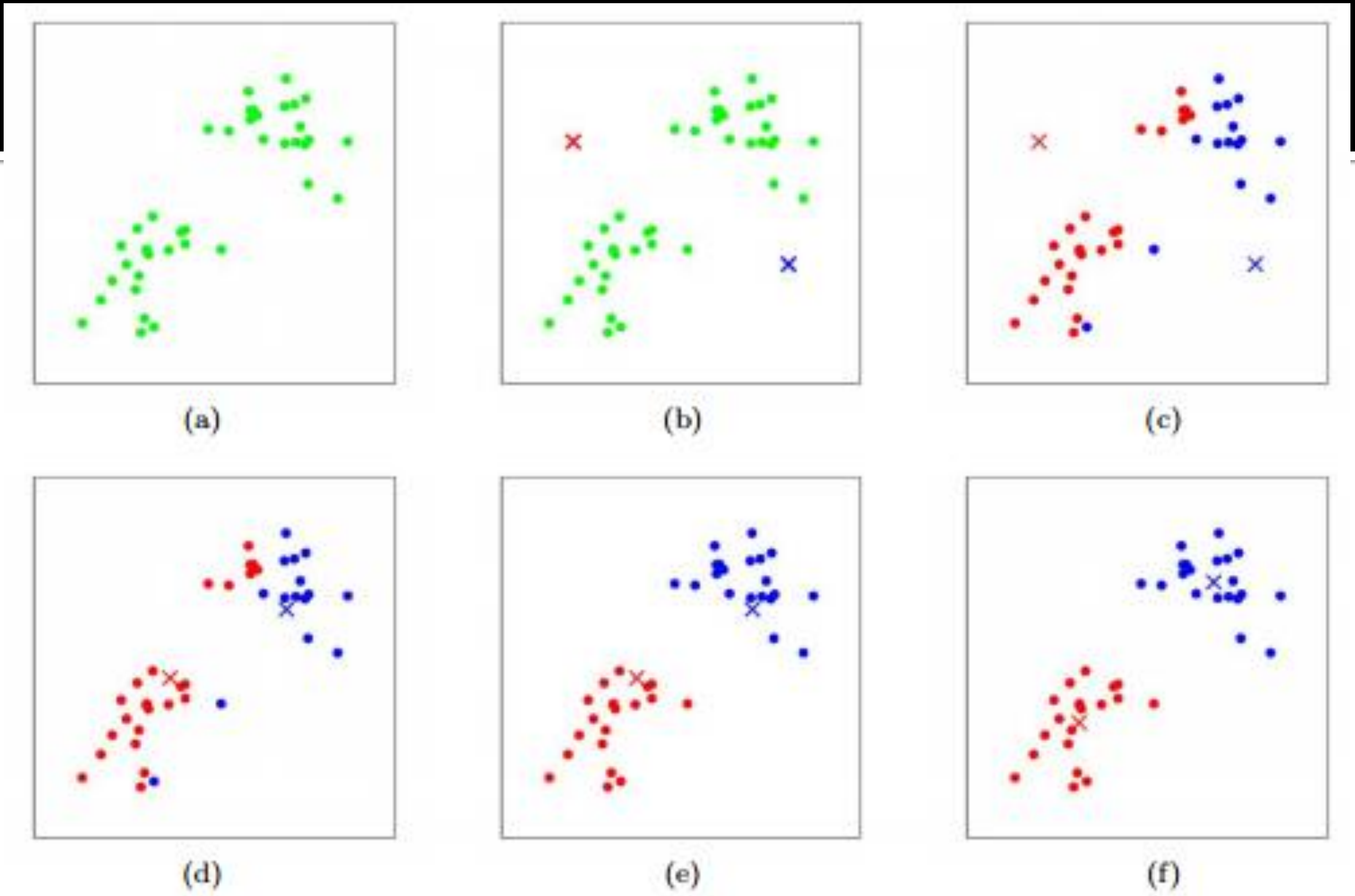
Iteration 2: (converged)



(e) (f)

Assignment step
(assign each point to the nearest cluster)

Centroid update step
(compute the mean in each cluster)



(a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means.

https://stanford.edu/~cpiech/cs221/handouts/kmeans.html

- K-means

Algorithm 1 k-means algorithm

Specify the number k of clusters to assign.
Randomly initialize k centroids.
repeat
    expectation: Assign each point to its closest centroid.
    maximization: Compute the new centroid (mean) of each cluster.
until The centroid positions do not change.

Assignment step

Centroid update step

https://realpython.com/k-means-clustering-python/

- Example
  - Given the following data vectors and initial centroids, perform K-means for 1 iteration.

Data points:

A	[-1, -1]
B	[-2, 0]
C	[1, 2]
D	[2, 1]

Initial centroids (2 clusters):

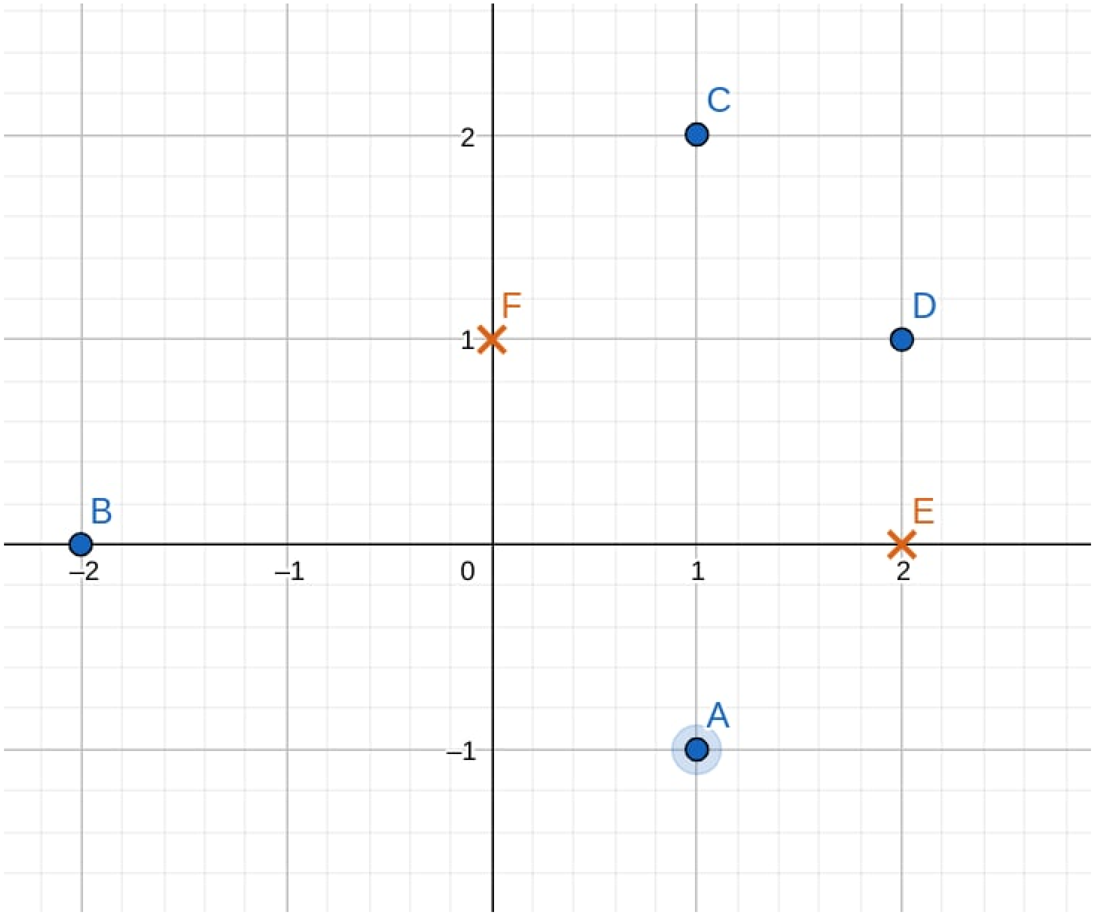
E (Cluster 1)	[2, 0]
F (Cluster 2)	[0, 1]

Data points:

A	[1, -1]
B	[-2, 0]
C	[1, 2]
D	[2, 1]

Initial centroids (2 clusters):

E (Cluster 1)	[2, 0]
F (Cluster 2)	[0, 1]



Iteration 1

Step 1: Cluster assignment

1) calculate squared L2 distance in Table 1

We can use squared L2 distance

as it will give the same comparison results as L2 distance.

E.g., SquaredL2(A, E) = $(1-2)^2 + (-1-0)^2 = 1 + 1 = 2$;

SquaredL2(A, F) = $(1-0)^2 + (-1-1)^2 = 1 + 4 = 5$;

Table 1 Squared L2 distance:

	E (2, 0) Cluster1	F (0, 1) Cluster2
A (1, -1)	2	5
B (-2, 0)	16	5
C (1, 2)	5	2
D (2, 1)	1	4

Squared L2 distance:

$$\|x - y\| = \sum_{i=1}^{d}(x_i - y_i)^2$$

L2 distance:

$$d_{L2}(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

2) Assign each point to its nearest centroid

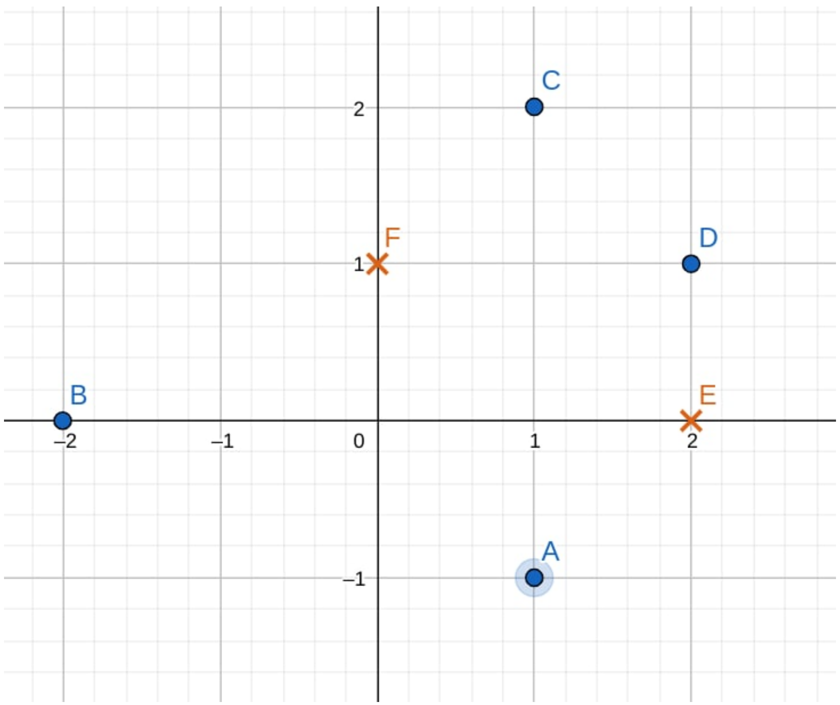
Cluster1: (A, D)   Cluster2 : (B, C)

Step 2: Centroid update by compute the mean of each cluster
Cluster1 E=[1.5, 0], F=[-0.5, 1]

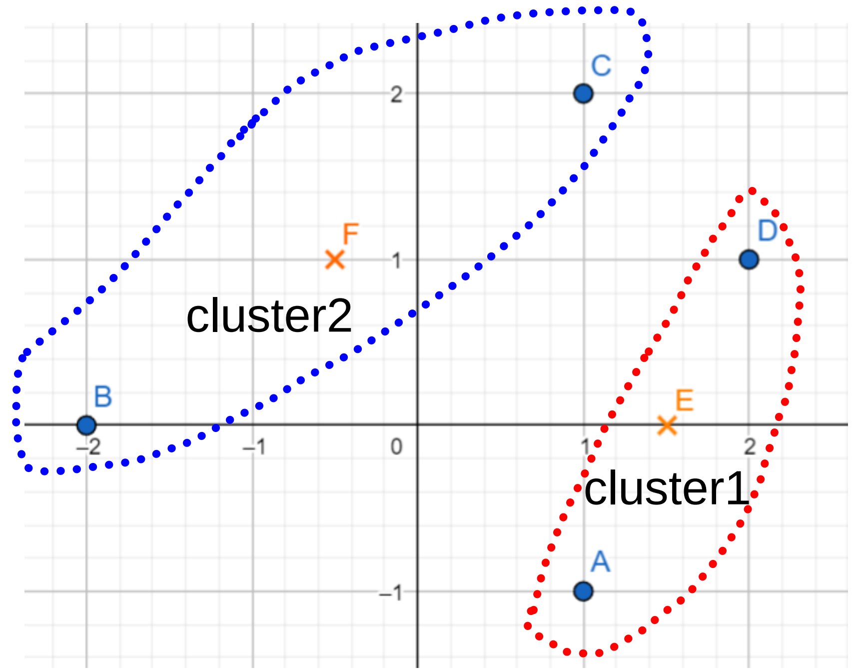
E_1= (1+2) /2 =1.5, E_2= (-1 + 1)/2=0, -> E=[1.5, 0]
F1= (-2 + 1)/2= -0.5, F2= (0 + 2)/2= 1, -> F=[-0.5, 1]

Table 1 Squared L2 distance:

	E (2, 0) Cluster1	F (0, 1) Cluster2
A (1, -1)	2	5
B (-2, 0)	16	5
C (1, 2)	5	2
D (2, 1)	1	4



Before iteration 1
(initial centroids)



After iteration 1

Iteration 2

Beyond the question, we perform the 2nd iteration:

Step 1: Cluster assignment

1) calculate squared L2 distance in Table 2

2) assign each point to its nearest centroid

Cluster C1: (A, D) Cluster C2: (B, C)

There is no changes of cluster assignments compared to the previous iteration, the algorithm is converged.

Step 2: Centroid update

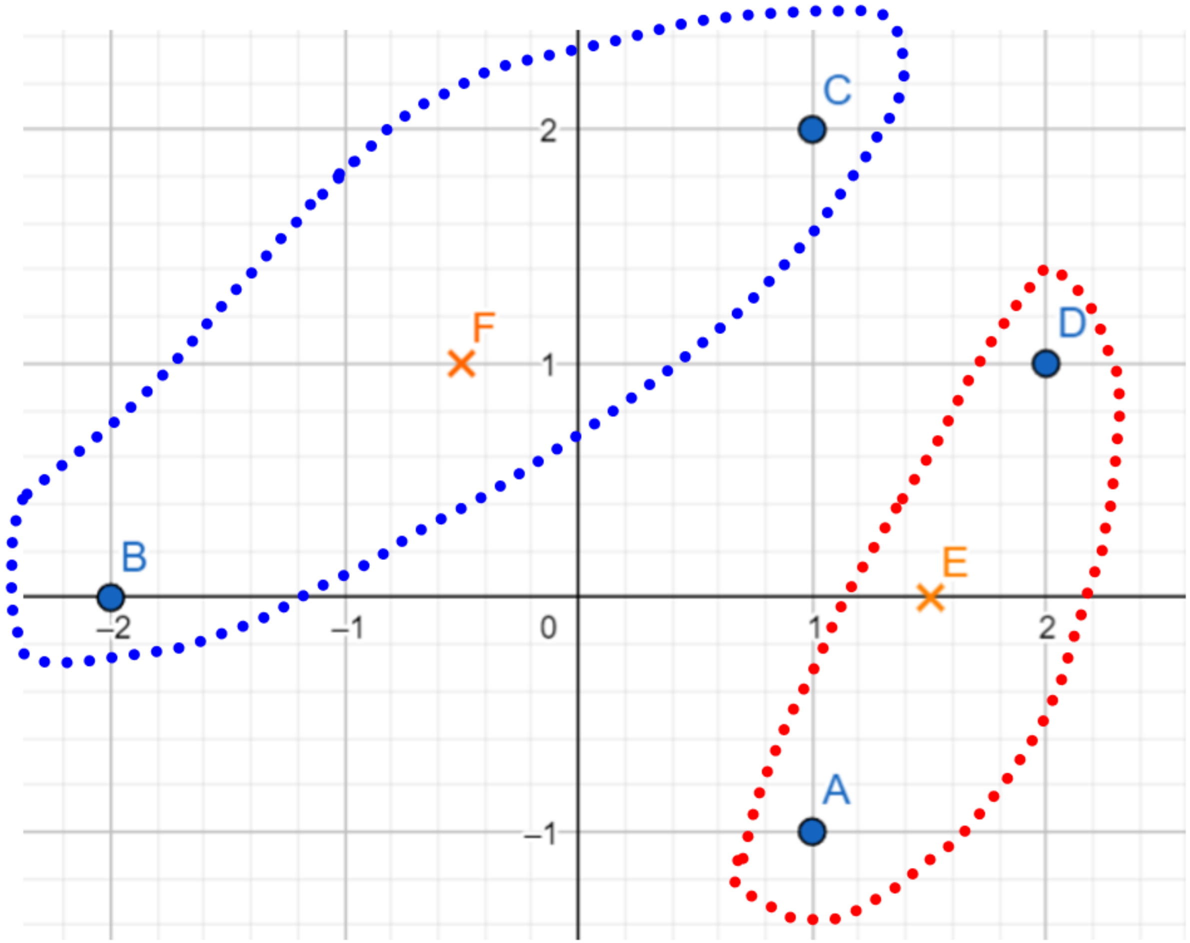
Cluster 1 (E): (1.5, 0)

Cluster 2 (F): (-0.5, 1)

Converged

Table: Squared L2 distance

	E(1.5, 0) C1	F (-0.5, 1) C2
A (1, -1)	**1.25**	6.25
B (-2, 0)	12.25	**3.25**
C (1, 2)	4.25	**3.25**
D (2, 1)	**1.25**	6.25



After iteration 2 (converged)

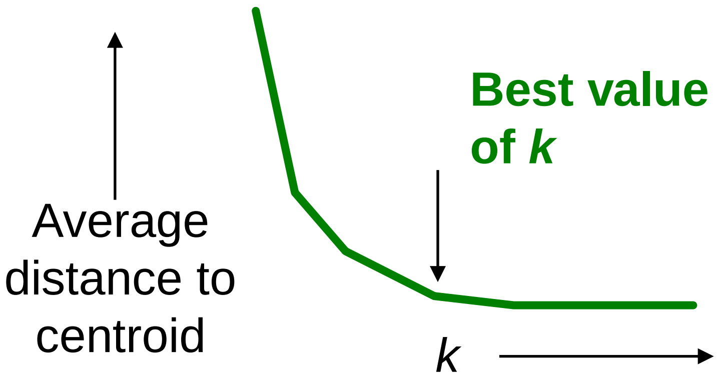
- How to determine the K parameter?
  - K is the number of clusters

https://en.wikipedia.org/wiki/K-means_clustering

How to select k?

- Try different k, looking at the change in the average distance to centroid as k increases
- Average distance to centroid falls rapidly at the beginning, then it changes slowly after a certain value of k

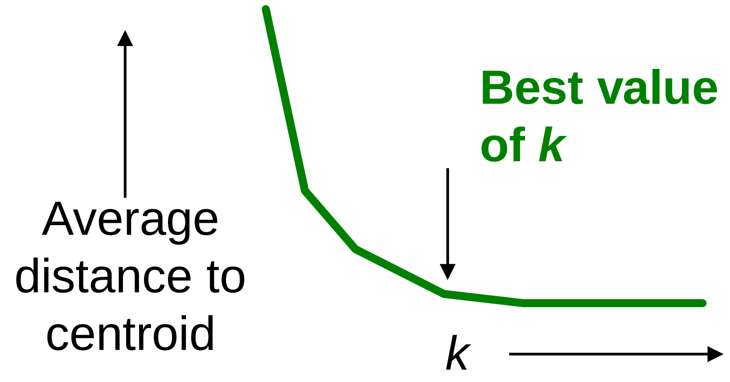
The elbow method



Select a small k that can produce small average distance to centroid

1. We prefer a low average distance to centroid
2. We prefer a small k

The elbow method



Average distance to centroid: E

$$E = \frac{1}{N}\left(\sum_{p_i \in C_1} \mathrm{d}(p_i, c_1) + \sum_{p_i \in C_2} \mathrm{d}(p_i, c_2) + \ldots + \sum_{p_i \in C_k} \mathrm{d}(p_i, c_k)\right)$$

d: distance, e.g., L2 distance or Squared L2 distance
p: a data point
c: a cluster centroid, C: a cluster

N: the total number of data points

Example

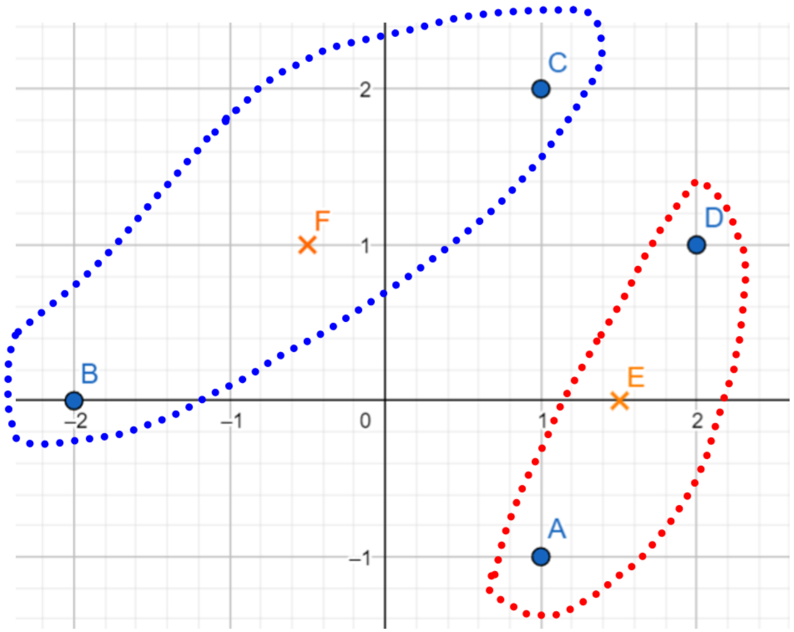


Table: Squared L2 distance

	E(1.5, 0) C1	F (-0.5, 1) C2
A (1, -1)	**1.25**	6.25
B (-2, 0)	12.25	**3.25**
C (1, 2)	4.25	**3.25**
D (2, 1)	**1.25**	6.25

k=2, use Squared L2 distance for d here:

y1 = d(A, C1) + d(D, C1) = 1.25 + 1.25 = 2.5

y2 = d(B, C2) + d(C, C2) = 3.25 + 3.25 = 6.5

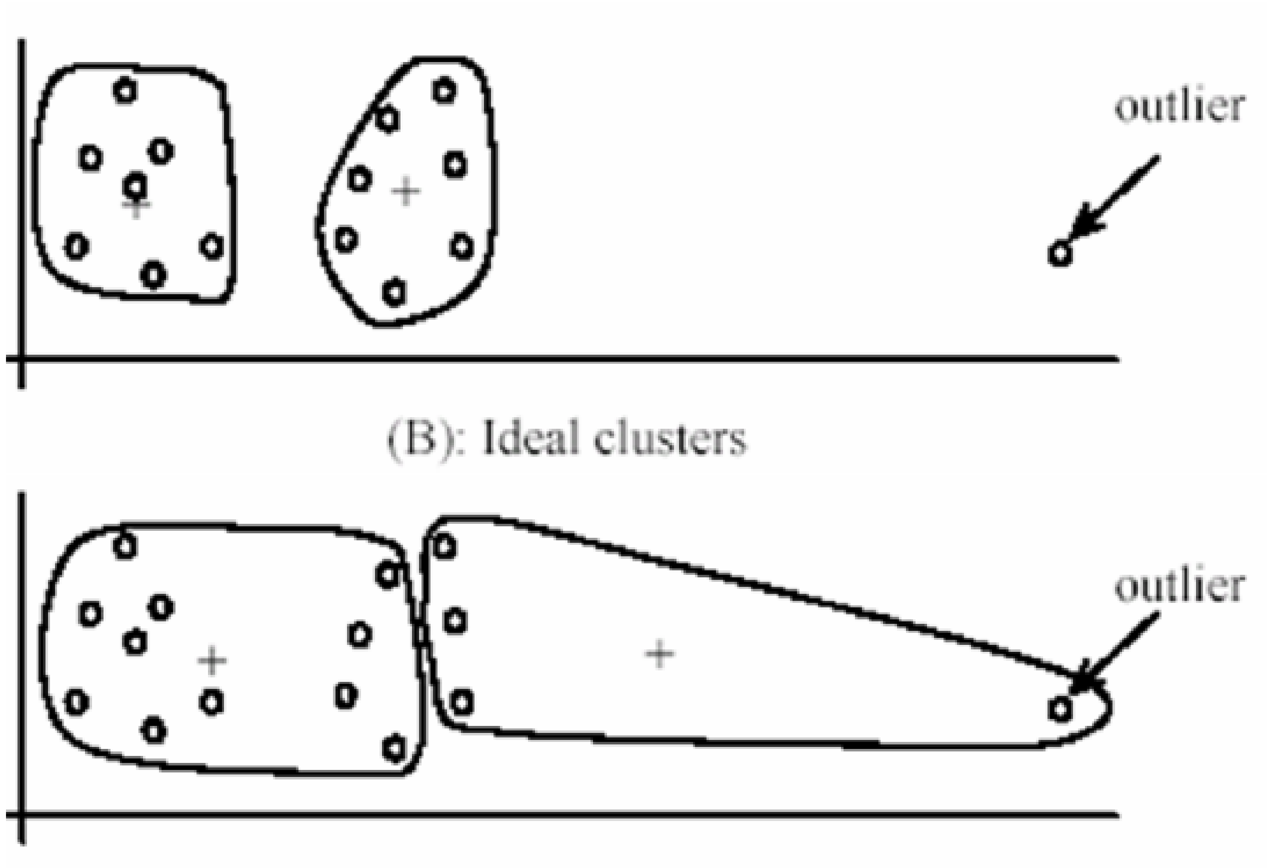
Average distance to centroid:

E = (y1+y2)/4 = 9/4 = 2.25

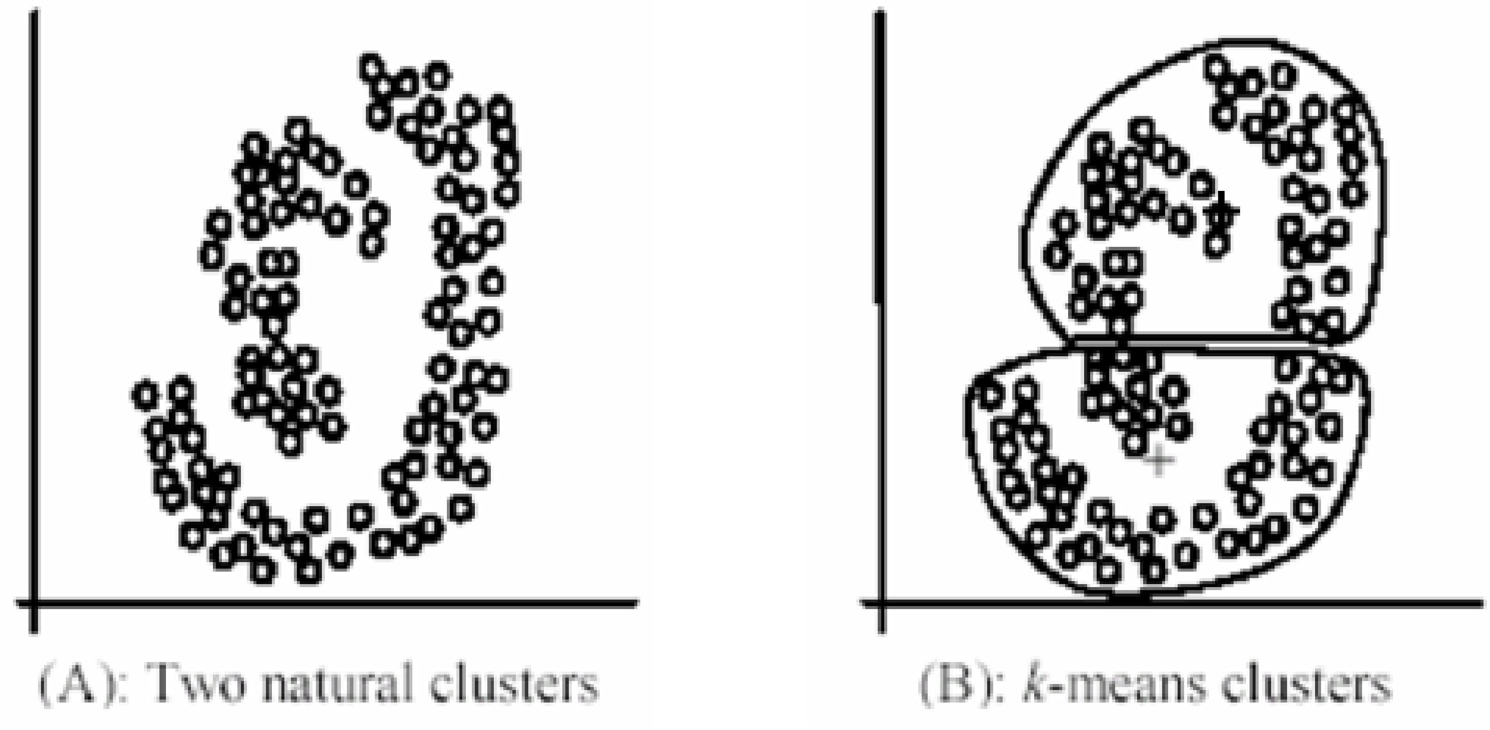
- K-means
  - distance functions
    - Euclidean distance (L2 distance)
      - most commonly used
    - L1 distance
    - Cosine distance
    - Other distance functions

K-means

- Pros
  - Simple and fast, Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Cannot work well on data with non-spherical shape
  - Sensitive to the initialization of the centroids

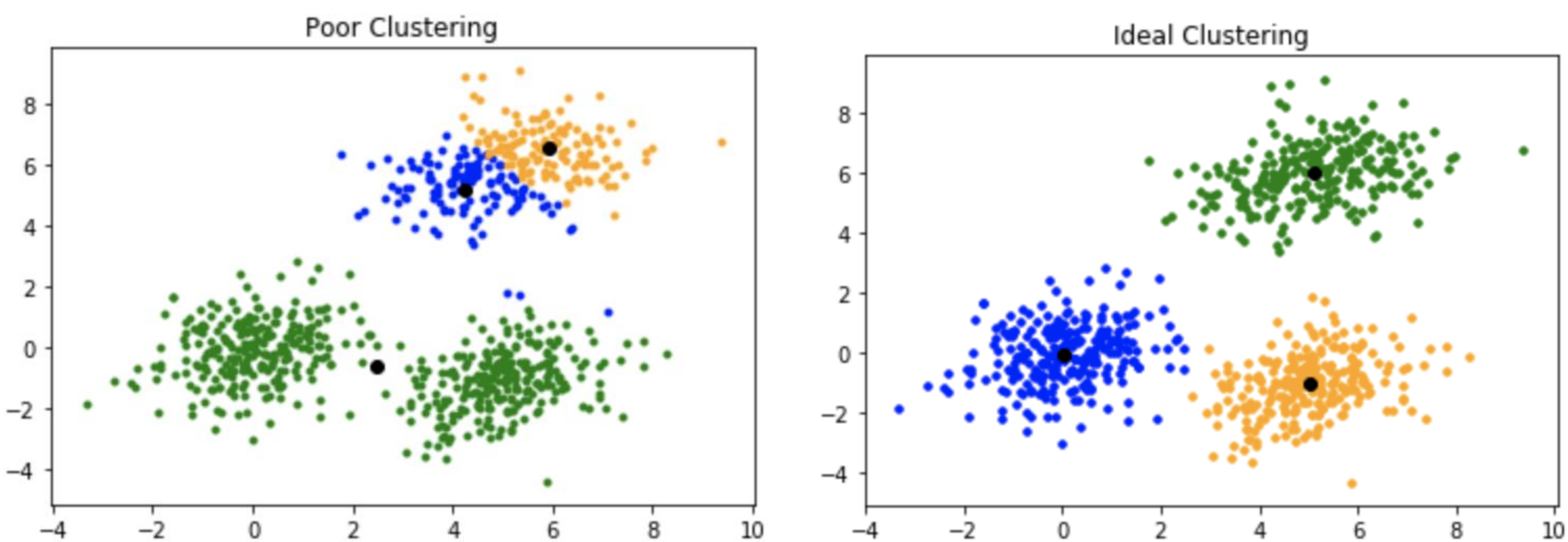


Limitation of K-means: sensitive to outliers



Limitation of K-means: cannot handle arbitrary shapes.
K-means is suitable for spherical or elliptical data.

- K-means algorithm is sensitive to the initialization of the centroids



https://www.geeksforgeeks.org/ml-k-means-algorithm/

- K-mean++:
  - Better initialization
  - K-means++ is the standard K-means algorithm coupled with a smarter initialization of the centroids.

K-mean++:

Idea: select initial centroids that are far away from each other.

1. Sequentially initialize the centroids.

2. When selecting one centroid, we aim to select a data point that is far away from existing centroids.

http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf

K-mean++:

let *D(x)* denote the shortest distance from a data point x to the existing centroids (c_1, c_2, c_3, …., c_n).

$$D(x) = \min_i \text{distance}(x, c_i)$$

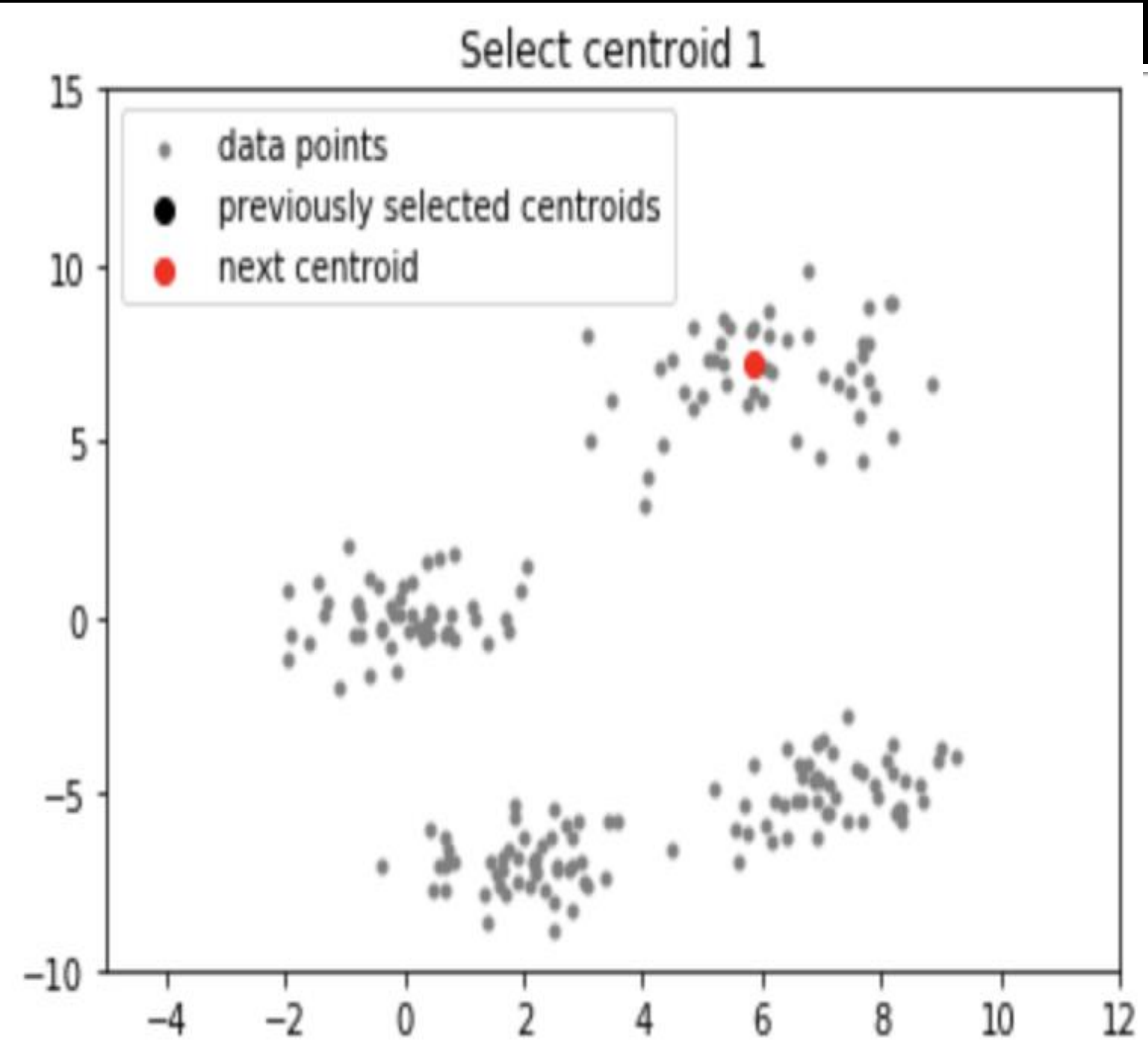
K-mean ++ (simplified version):

Step 1. choose the data point with the highest *D(x)* as the new centroid (C_{n+1}). The selected data point will be far away from all existing centroids.

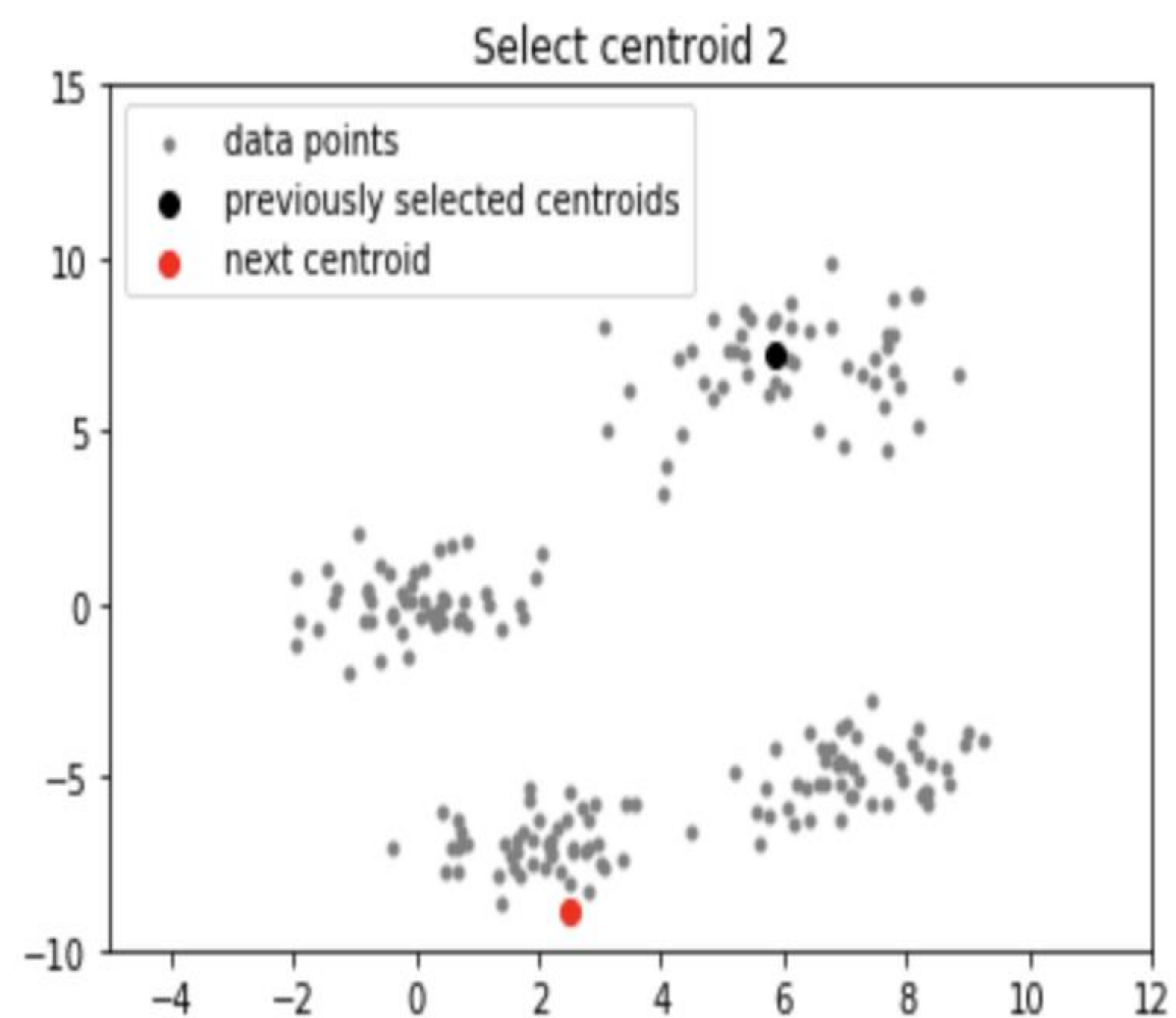
Step 2. Repeat step 1 until we have k centroids.

1. If the data point x is close to any existing centroids, D(x) will be small
2. If the data point x is far away from any existing centroids, D(x) will be large

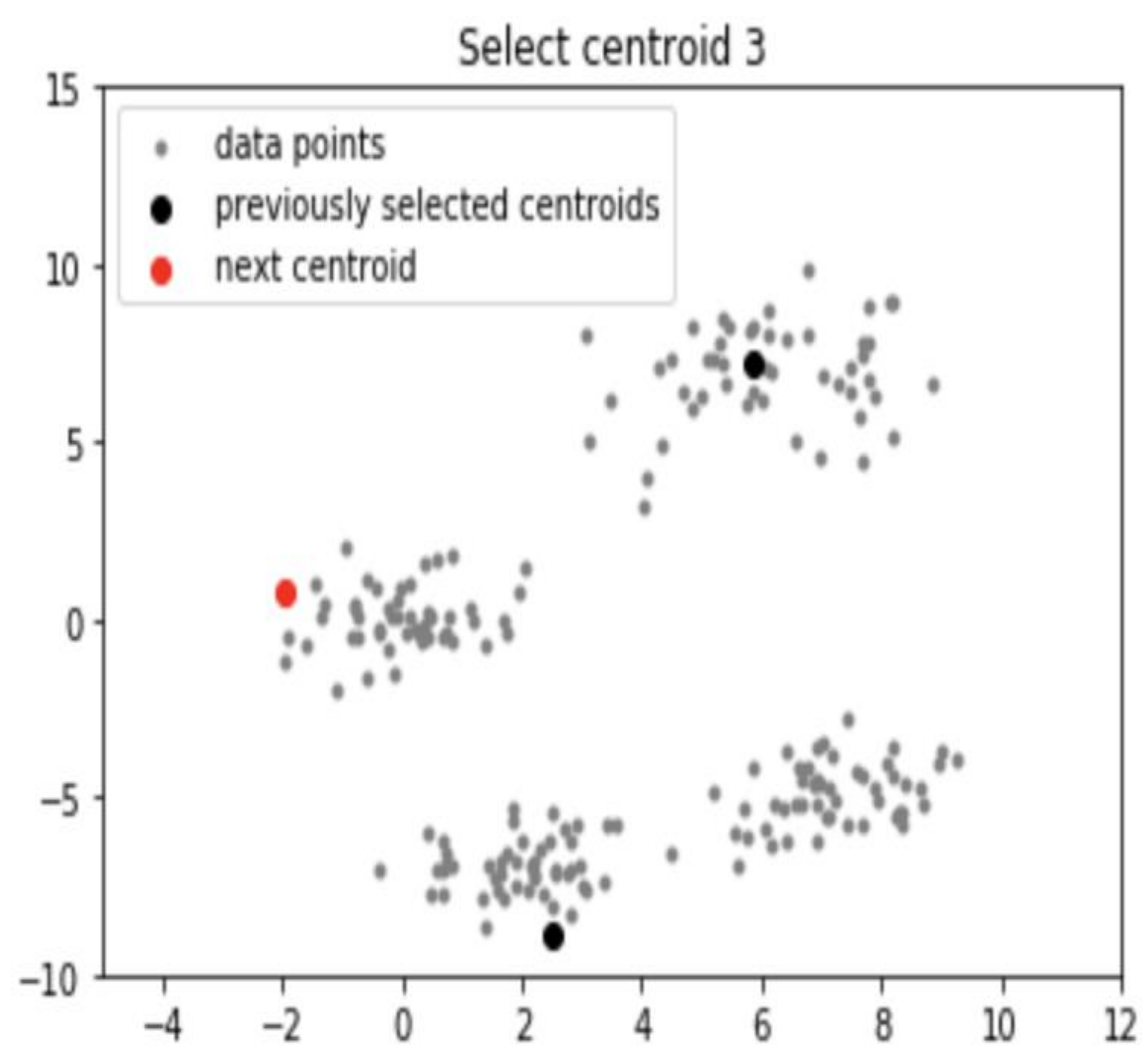
http://ilpubs.stanford.edu:8090/778/1/2006-13.pdf



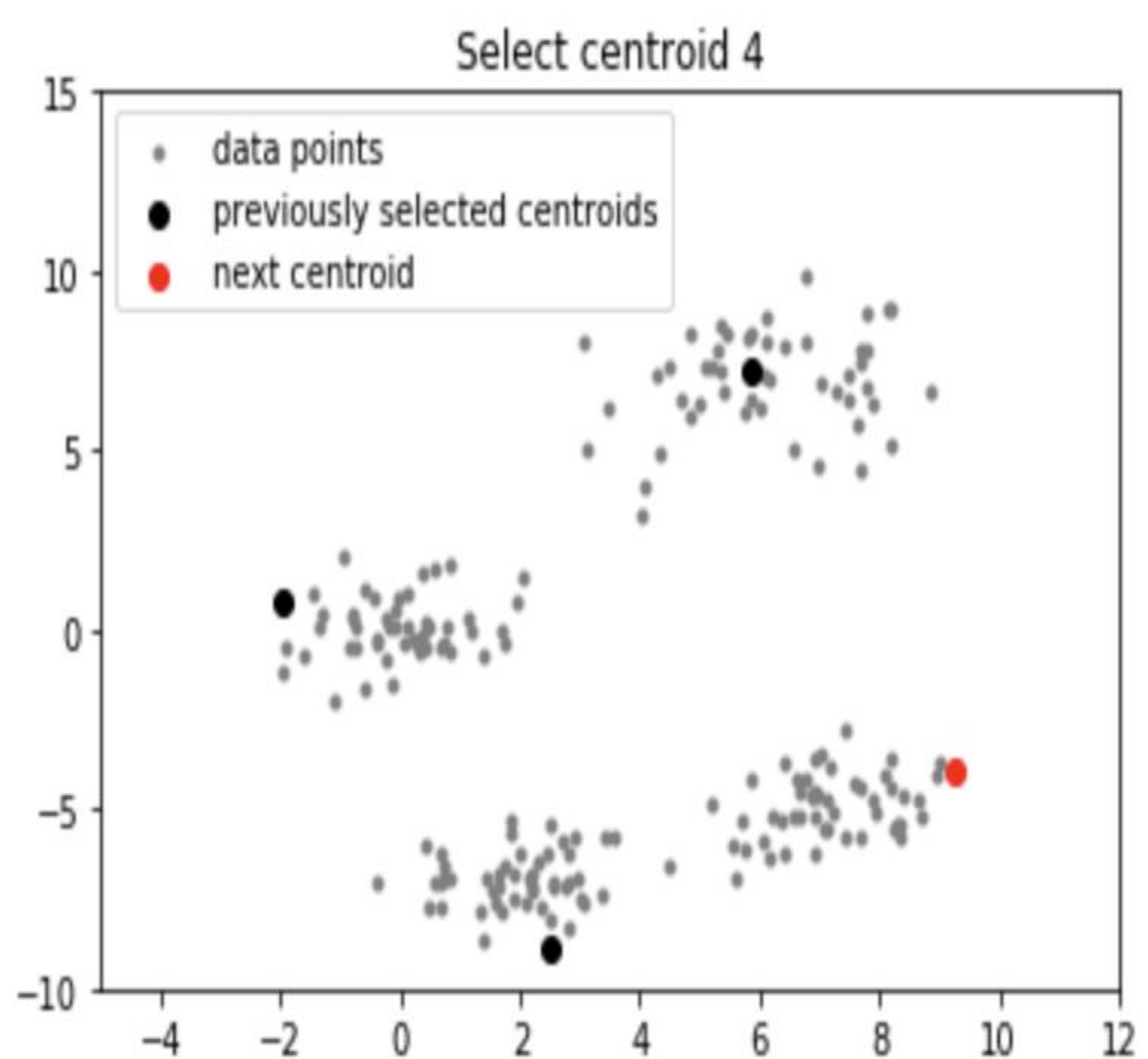
https://www.geeksforgeeks.org/ml-k-means-algorithm/



https://www.geeksforgeeks.org/ml-k-means-algorithm/



https://www.geeksforgeeks.org/ml-k-means-algorithm/

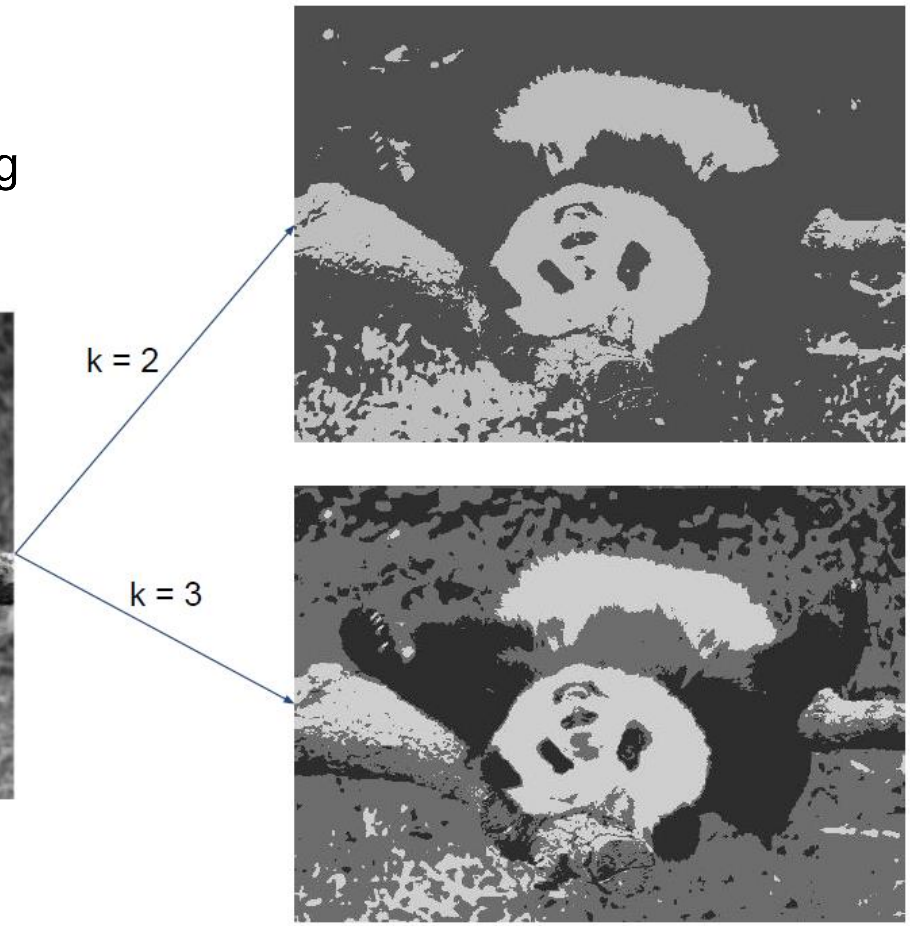


https://www.geeksforgeeks.org/ml-k-means-algorithm/

Segmentation as Clustering

- Let's just use the pixel intensities!

Segmentation: pixel grouping
Assign pixels to clusters

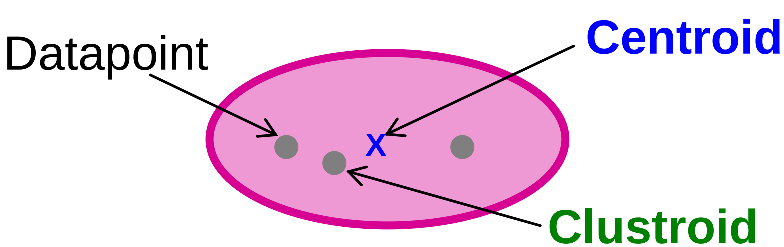


More examples:
https://github.com/topics/slic

More examples: SLIC for super-pixel segmentation
https://github.com/topics/slic

- K-means with clustroid
  - Use clustroid to replace centroid
    - Clustroid: is an existing (data) point that is "closest" to all other points in the cluster.

Non-examinable



A cluster with
3 data points

Centroid is the avg. of all (data) points in the cluster.

Centroid is not an existing point. It is an "artificial" point.

For a point *p*, calculate the average distance between *p* and all other members in the same cluster. Clustroid is the point that has the smallest average distance within the cluster.

Optimization problem

Non-examinable

- The optimization problem for K-means

The squared distance to centroid for one cluster:

$$S(C_i) = \sum_{x_j \in C_i} (c_i - x_j)^2$$

$$S(C_i) : \text{squared distance to centroid for cluster } C_i$$

$$x_j \in C_i : \text{a data point in cluster } C_i$$

$$c_i \in C_i : \text{the cluster centroid of } C_i$$

Slides from Stanford CS131

Optimization problem

Goal: minimize average squared distance to centroid (or minimize within-cluster variance)

Cluster center Data

Objective function:

$$c^*,\ \delta^* = \underset{c,\delta}{\arg\min} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} \left(c_i - x_j\right)^2$$

Asterisk * indicates the solutions to
1. cluster centroids
2. and assignments

Whether x_j is assigned to c_i

$$\text{if } x_j \text{ is a member of cluster } c_i, \quad \delta_{ij} = 1, ;$$
$$\text{otherwise,} \quad \delta_{ij} = 0.$$

Optimization problem

K-means clustering

1. Initialize ($t = 0$): cluster centers $c_1, \ldots, c_K$

2. Compute δ^t: assign each point to the closest center
   - δ^t denotes the set of assignment for each x_j to cluster c_i at iteration *t*

Assignment step

$$\delta^{\mathbf{t}} = \arg\min_{\delta} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij} (c_i^{t-1} - x_j)^2$$

Freeze c, solve for δ

3. Computer c^t: update cluster centers as the mean of the points

Centroid update step

$$\mathbf{c}^{\mathbf{t}} = \arg\min_{\mathbf{c}} \frac{1}{N} \sum_{j}^{N} \sum_{i}^{K} \delta_{ij}^{t} (c_i - x_j)^2$$

Freeze δ, solve for c

4. Update $t = t + 1$, Repeat Step 2-3 till stopped

Optimization problem

- Hard to solve the optimization problem
  - Hard to obtain global optimal solutions
- The iterative algorithm is also referred to as Lloyd's algorithm
  - Empirically works well, usually converges in a few iterations

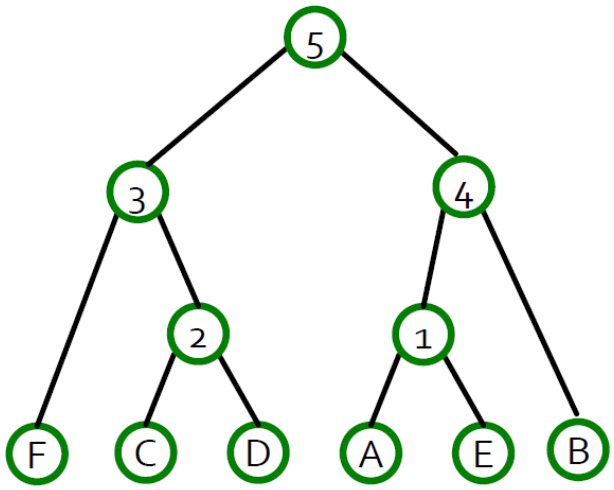
https://en.wikipedia.org/wiki/K-means_clustering

- Further reading:
  - Mini-batch K-means for large-scale data
    - Handle a small number of data points at a time, using stochastic gradient decent to gradually update the cluster centre .
    - Further reading:
      - Online tool: sklearn.cluster.MiniBatchKMeans

http://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html

Hierarchical clustering

- Hierarchical
  - Agglomerative (bottom up)
    - Initially, each point is a cluster
    - Repeatedly combine the two nearest clusters into one
  - Divisive (top down)
    - Start with one cluster and recursively split it



Hierachical clustering

Agglomerative clustering

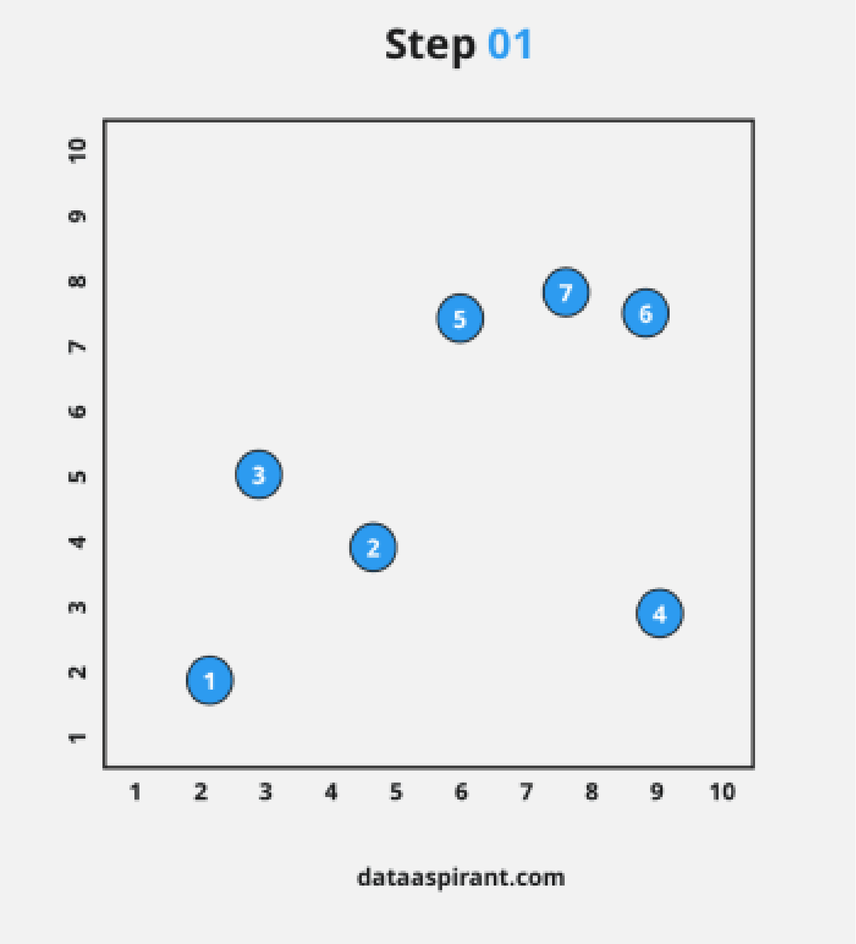
(or bottom-up hierarchical clustering)

Simple algorithm

- Initialization:
  - Every point is its own cluster
- Repeat:
  - Find “most similar” pair of clusters
  - Merge into a parent cluster
- Until:
  - The desired number of clusters has been reached
  - There is only one cluster

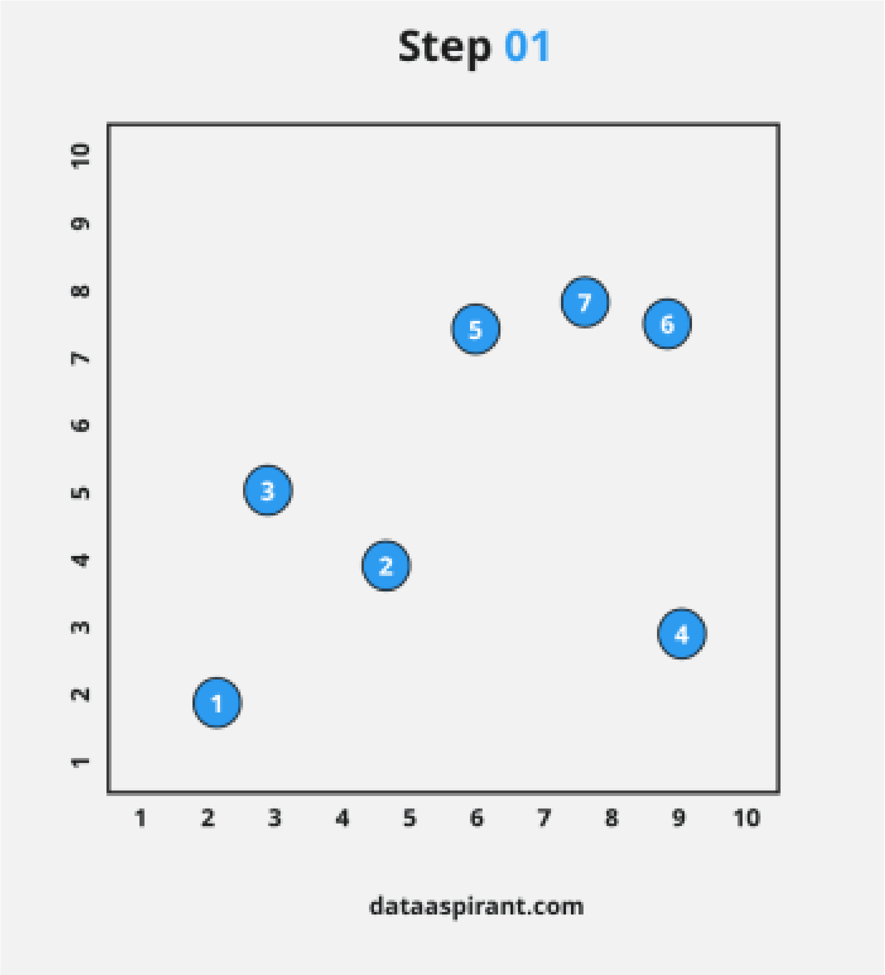
First, make each data point a cluster, which forms N clusters.

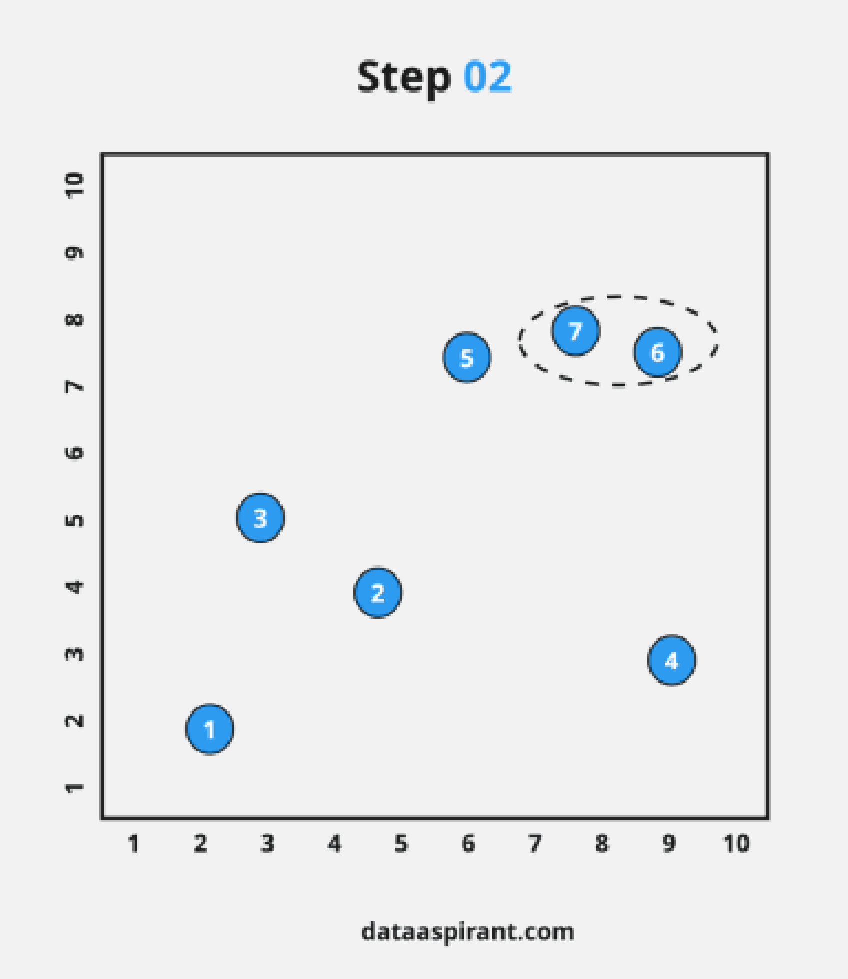
Example 1:



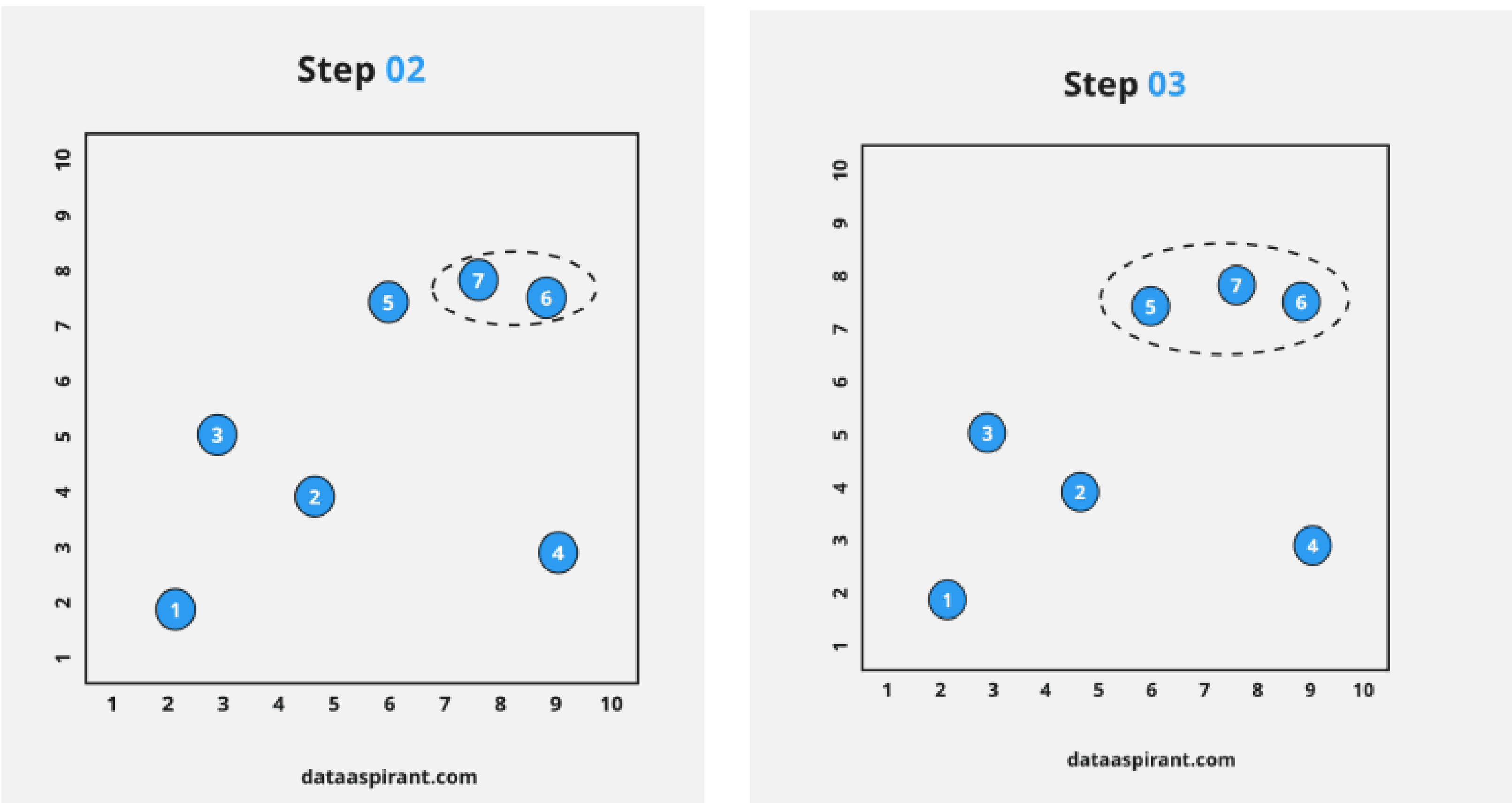
https://dataaspirant.com/hierarchical-clustering-algorithm/

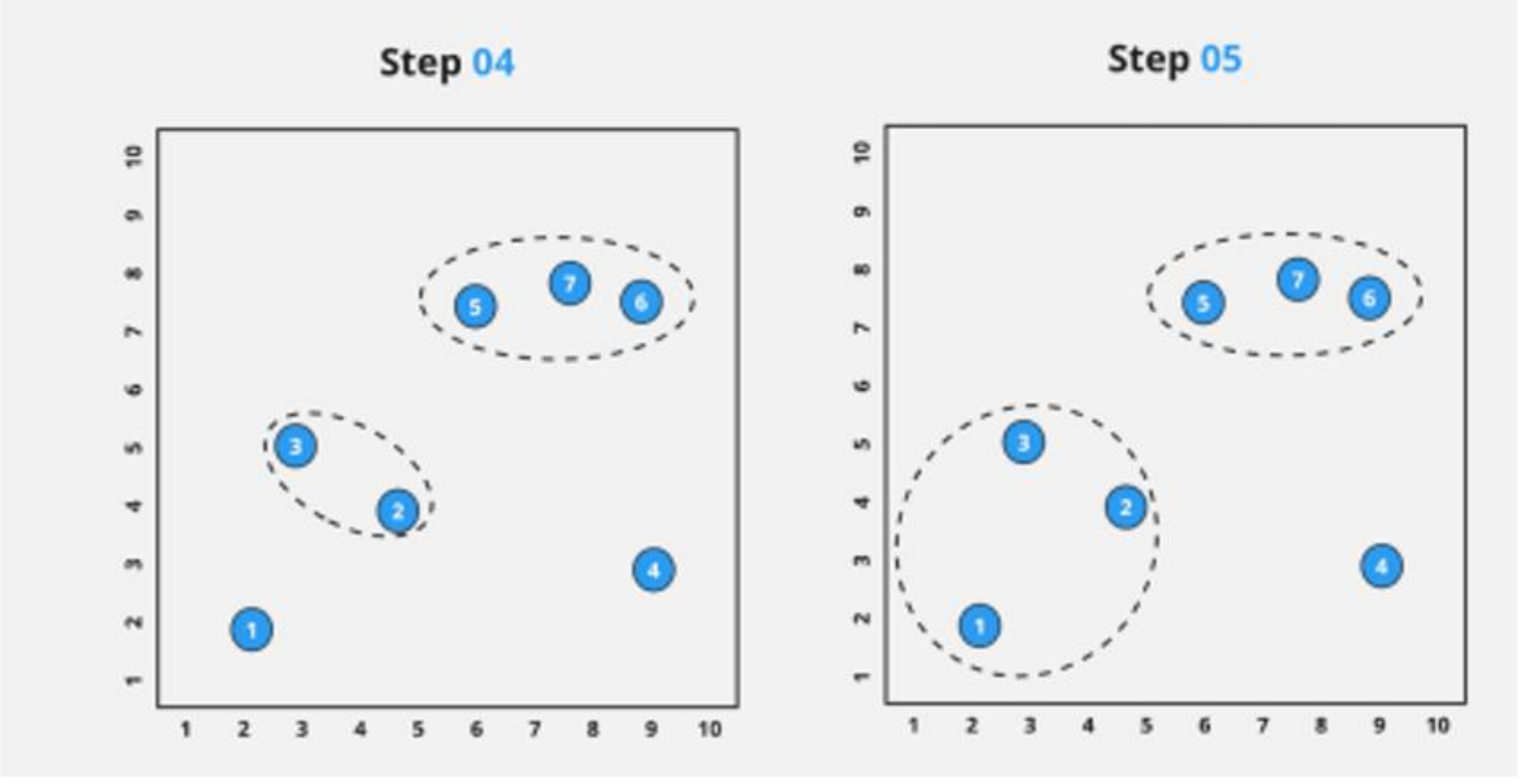
Take the next two nearest clusters and make them one cluster



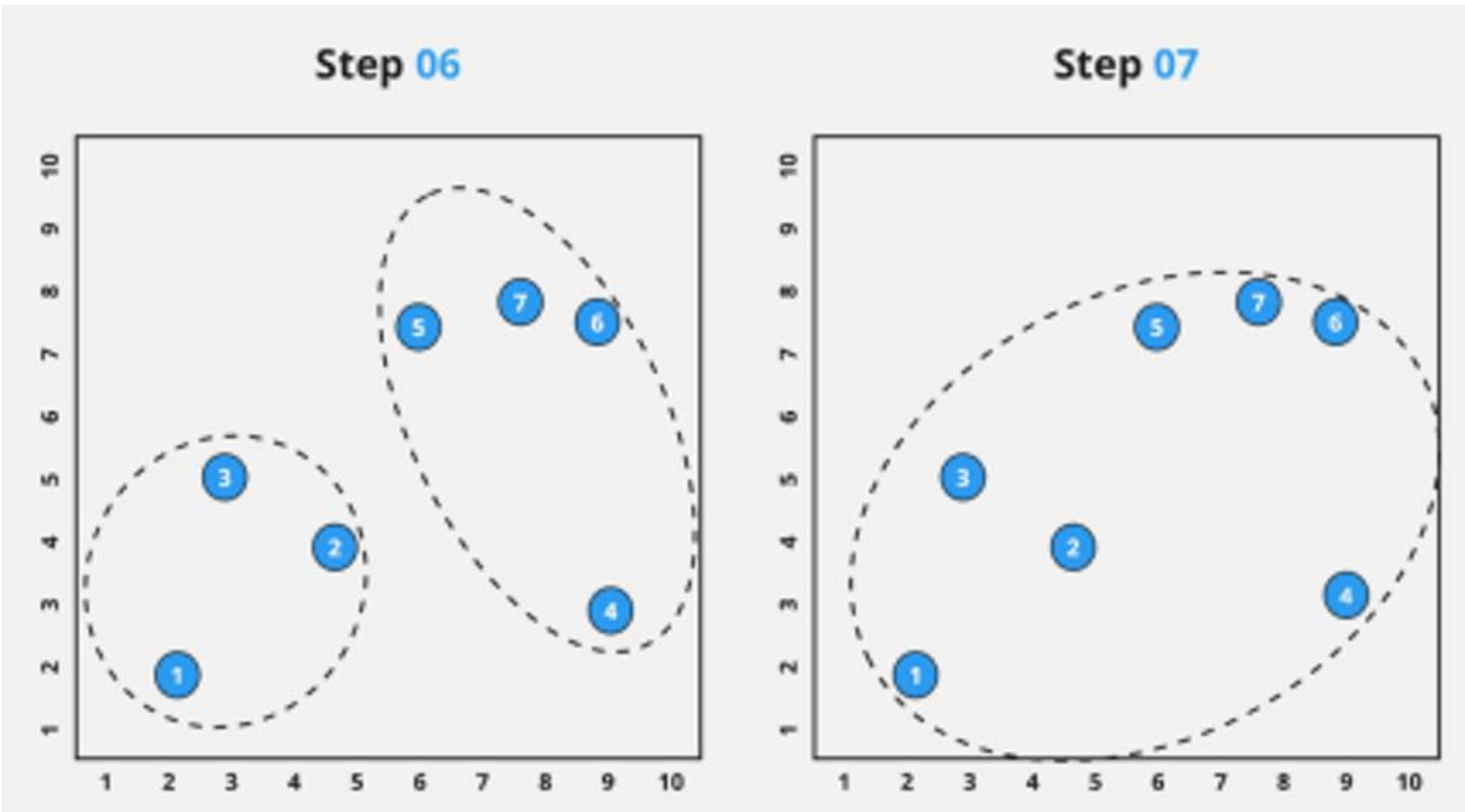


Again, take the two nearest clusters and make them one cluster

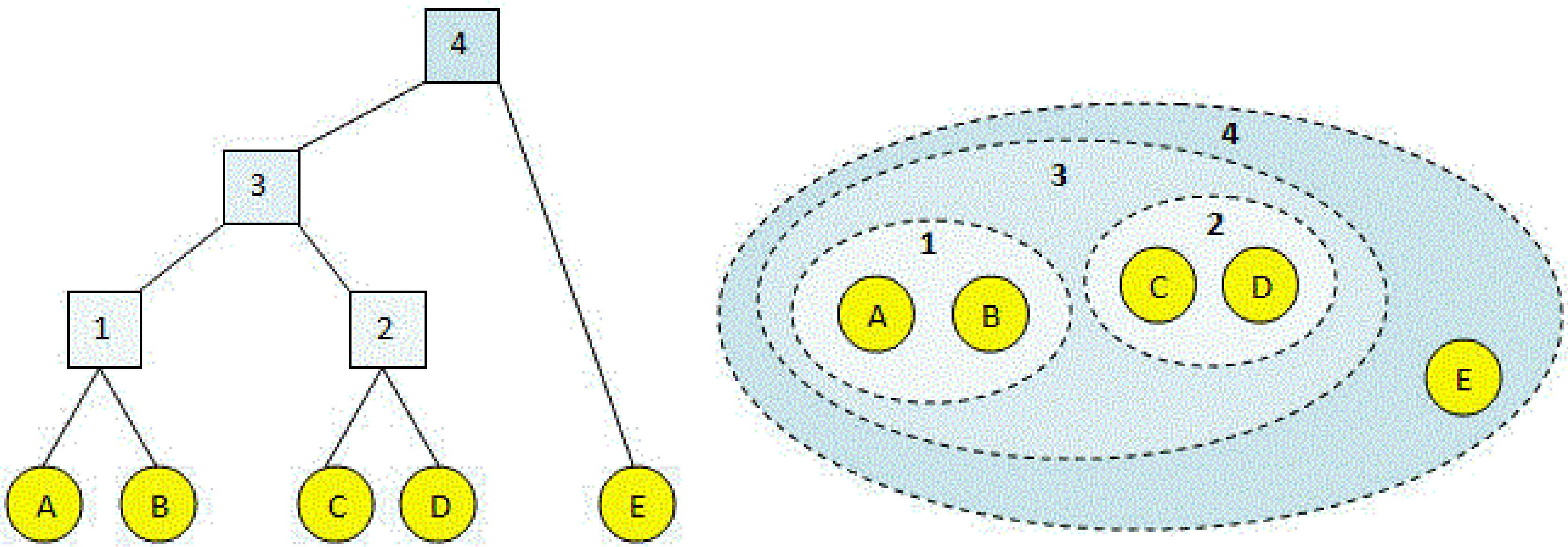




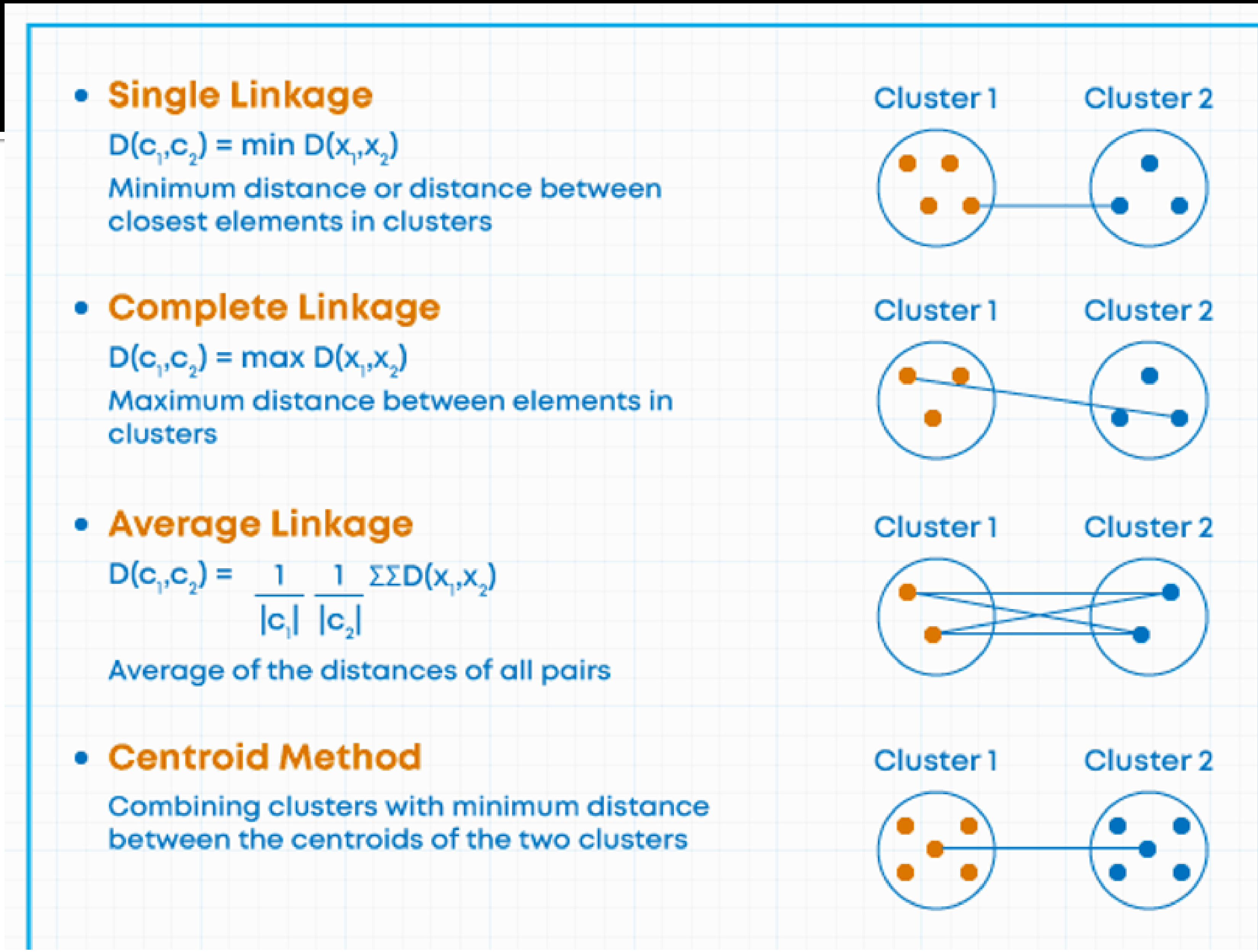
Step 04
Step 05


Step 06
Step 07
1
2
3
4
5
6
7
8
9
10

Example 2



- What is missing here?
  - How to define the similarity of two clusters?



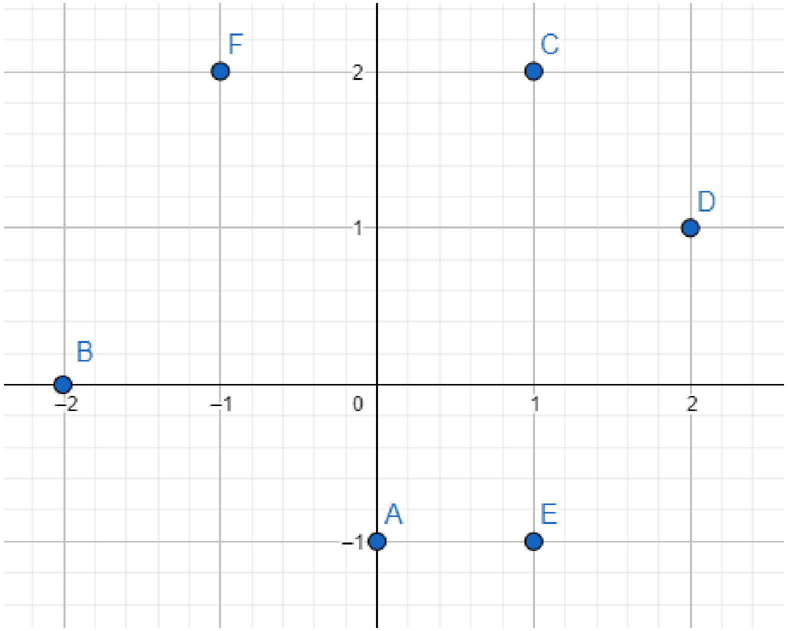
define cluster similarity

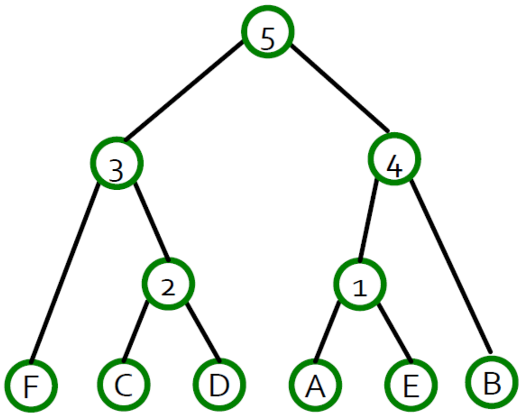
https://dataaspirant.com/hierarchical-clustering-algorithm/

- We will discuss the following calculation example in the tutorial class:

Data points

A	[0, -1]
B	[-2, 0]
C	[1, 2]
D	[2, 1]
E	[1, -1]
F	[-1, 2]





- When do we stop merging clusters?
  - When some number k of clusters are found
  - Keep merging until there is only 1 cluster

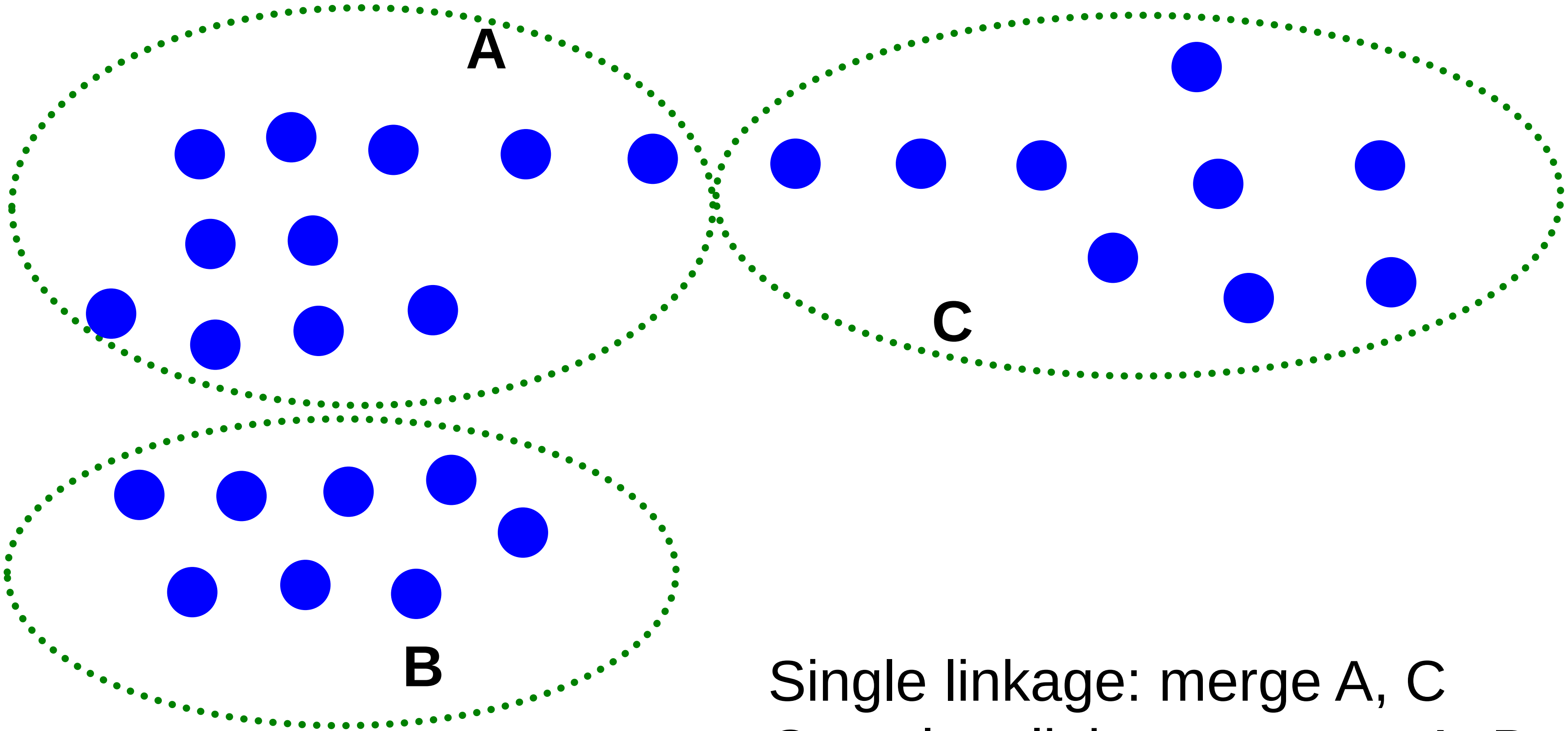
Hierarchical Clustering

- Pros
  - No need to set K: the number of clusters
  - No assumption on the shape of the cluster
  - Simple

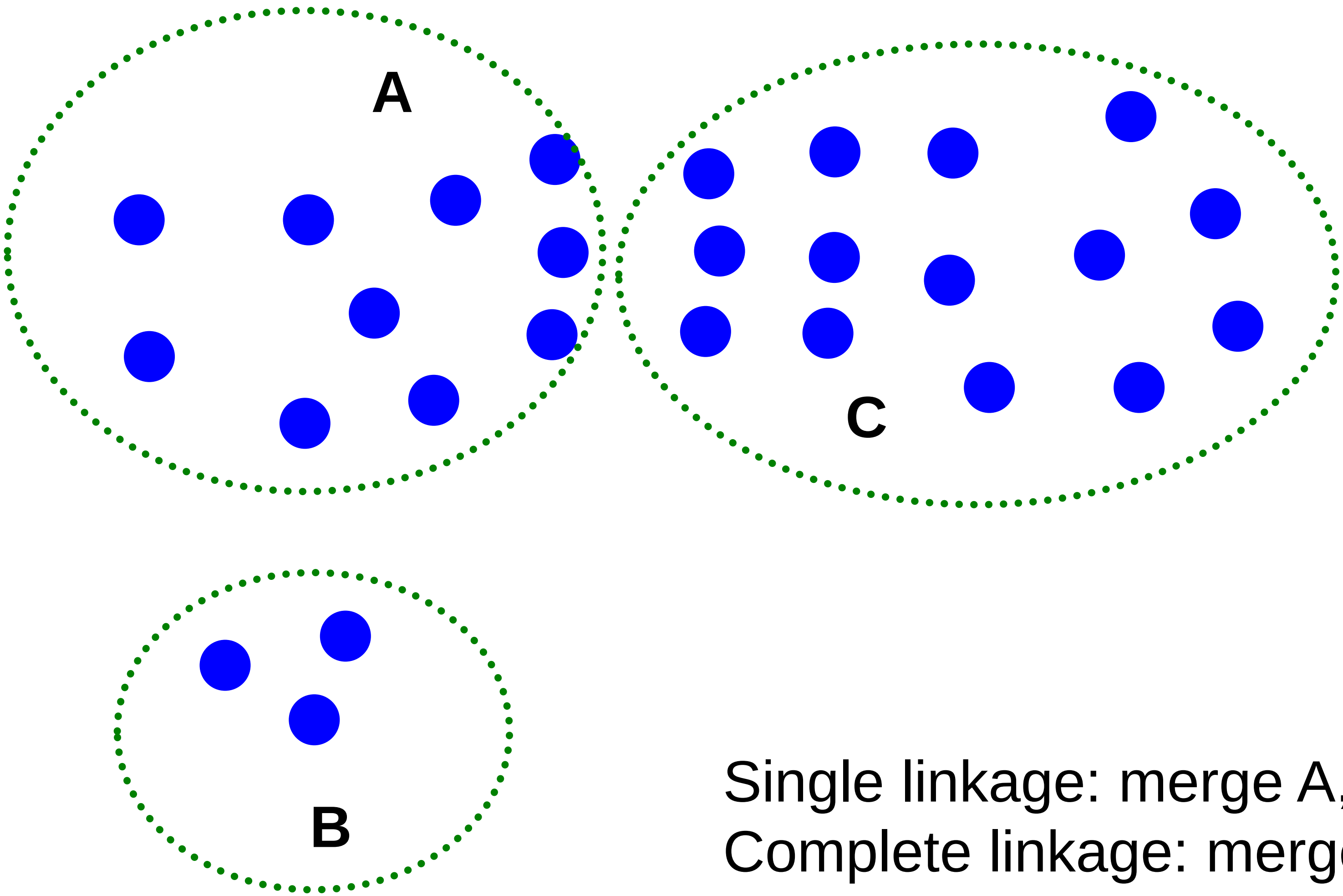
- Cons
  - The algorithm can never undo the grouping.
    The data points may can be incorrectly grouped at an earlier stage.

  - Different distance metrics may produce very different results.
    - Simple Linkage methods are sensitive to noise and outliers.
    - Complete linkage methods tend to break large clusters.

https://dataaspirant.com/hierarchical-clustering-algorithm/



Single linkage: merge A, C
Complete linkage: merge A, B



Single linkage: merge A, C
Complete linkage: merge A, B