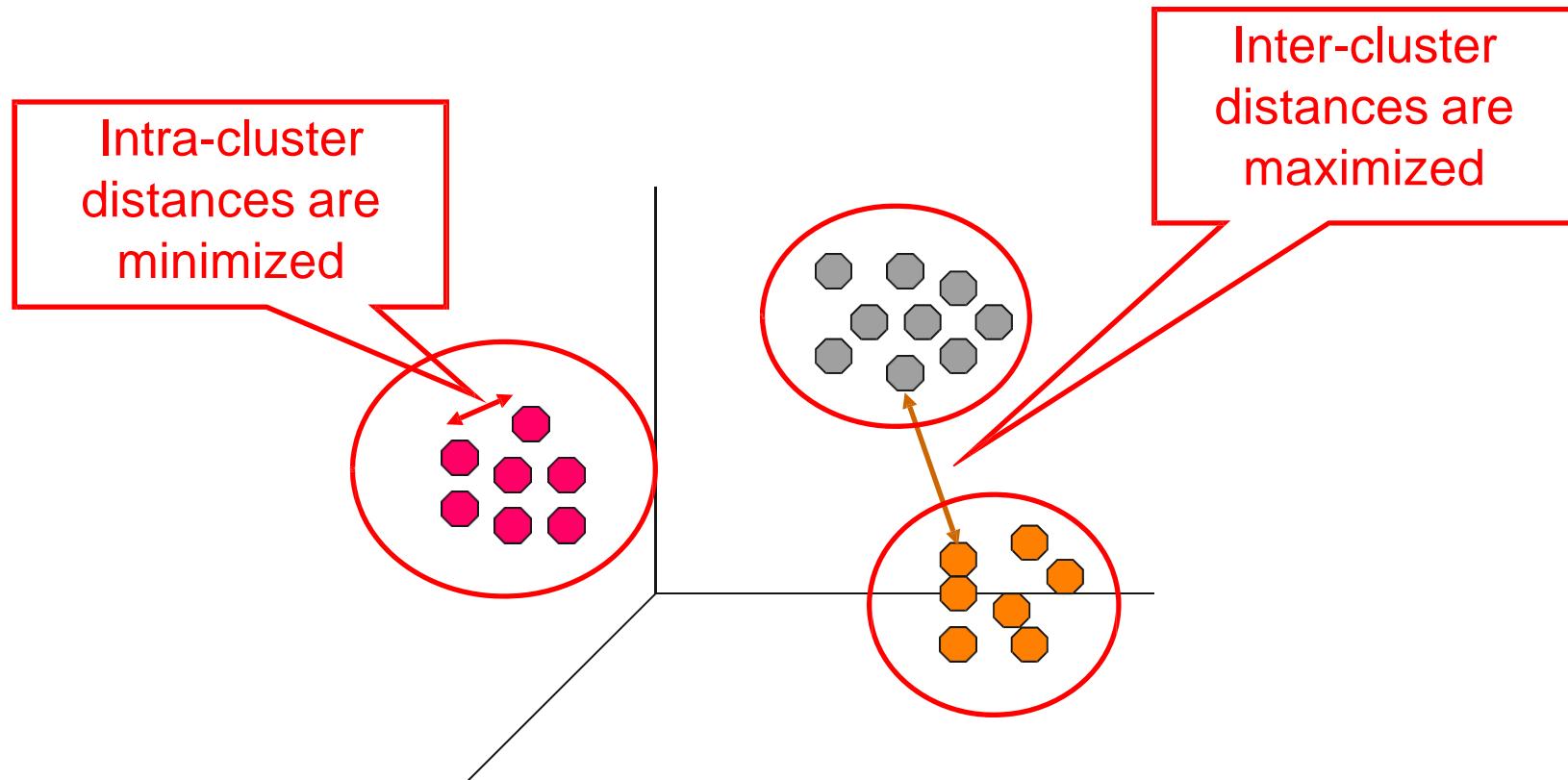


# Cluster Analysis

1. What is Cluster Analysis?
2. K-Means Clustering
3. Density-based Clustering
4. Hierarchical Clustering

# 1. What is Cluster Analysis?

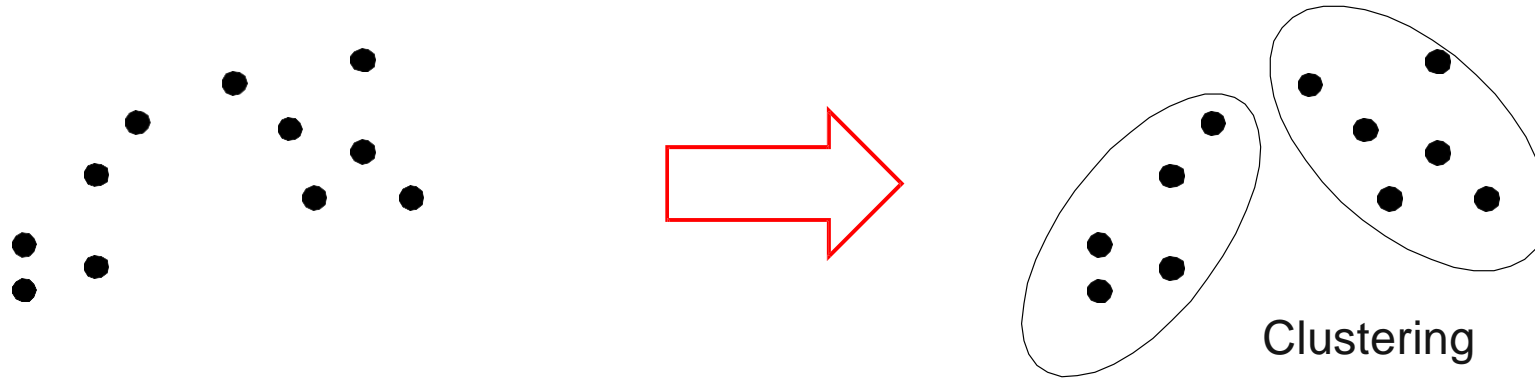
- Finding groups of objects such that
  - the objects in a group will be similar to one another
  - and different from the objects in other groups.
- Goal: Get a better understanding of the data



# Types of Clusterings

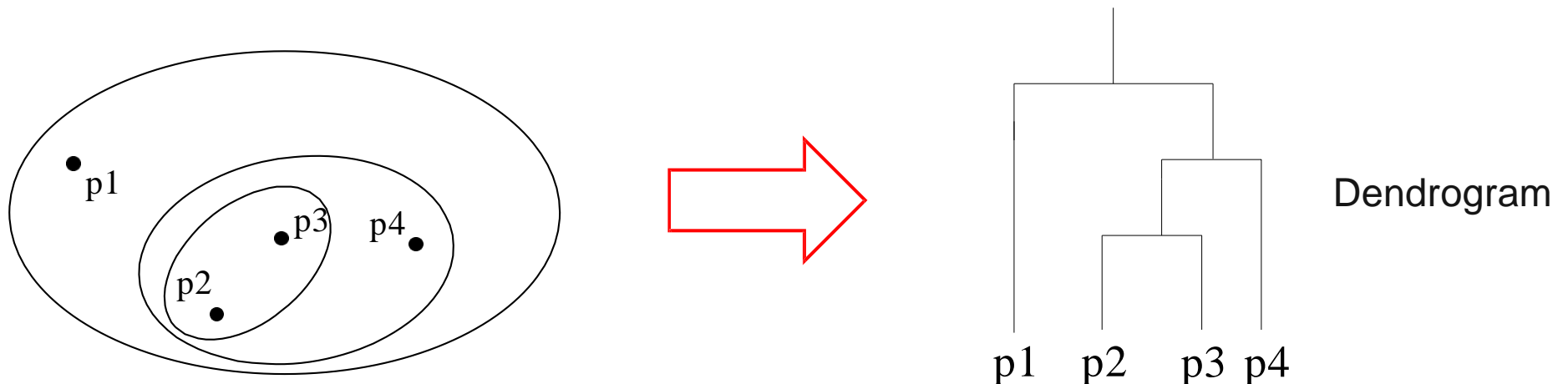
## – Partitional Clustering

- A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset



## – Hierarchical Clustering

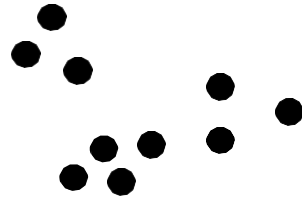
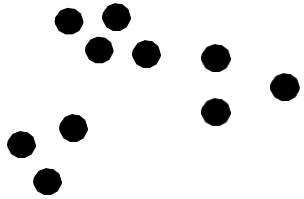
- A set of nested clusters organized as a hierarchical tree



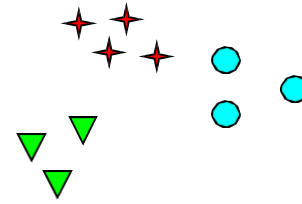
# Aspects of Cluster Analysis

- A clustering algorithm
  - Partitional algorithms
  - Density-based algorithms
  - Hierarchical algorithms
  - ...
- A proximity (similarity, or dissimilarity) measure
  - Euclidean distance
  - Cosine similarity
  - Data type-specific similarity measures
  - Domain-specific similarity measures
- Clustering quality
  - Intra-clusters distance  $\Rightarrow$  minimized
  - Inter-clusters distance  $\Rightarrow$  maximized
  - The clustering should be useful with regard to the goal of the analysis

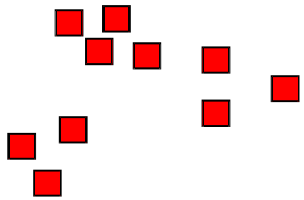
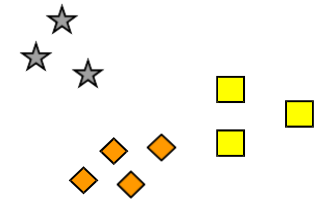
# The Notion of a Cluster is Ambiguous



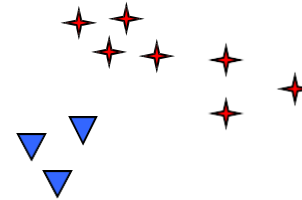
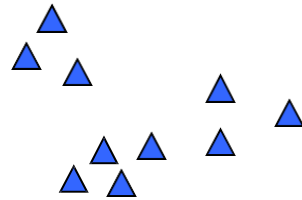
How many clusters do you see?



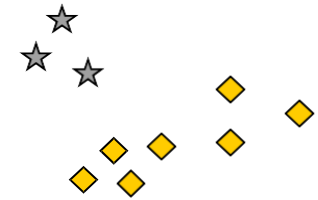
Six Clusters



Two Clusters



Four Clusters



The usefulness of a clustering depends  
on the **goal of the analysis**

# Cluster Analysis as Unsupervised Learning

- **Supervised learning:** Discover patterns in the data that relate data attributes with a target (class) attribute
  - these patterns are then utilized to predict the values of the target attribute in unseen data instances
  - the set of classes is known before
  - training data is often provided by human annotators
- **Unsupervised learning:** The data has no target attribute
  - we want to explore the data to find some intrinsic patterns in it
  - the set of classes/clusters is not known before
  - no training data is used
- Cluster Analysis is an unsupervised learning task

# Data Types and Representations

## Discrete vs. Continuous

### Discrete Feature

Has only a finite set of values e.g., zip codes, rank, or the set of words in a collection of documents

Sometimes, represented as integer variable

### Continuous Feature

Has real numbers as feature values e.g, temperature, height, or weight

Practically, real values can only be measured and represented using a finite number of digits

Continuous features are typically represented as floating-point variables



# Data Types and Representations

## Data representations

Data matrix (object-by-feature structure)

$$\begin{bmatrix} x_{11} & \dots & x_{1f} & \dots & x_{1p} \\ \dots & \dots & \dots & \dots & \dots \\ x_{i1} & \dots & x_{if} & \dots & x_{ip} \\ \dots & \dots & \dots & \dots & \dots \\ x_{n1} & \dots & x_{nf} & \dots & x_{np} \end{bmatrix}$$

- $n$  data points (objects) with  $p$  dimensions (features)
- **Two modes:** row and column represent different entities

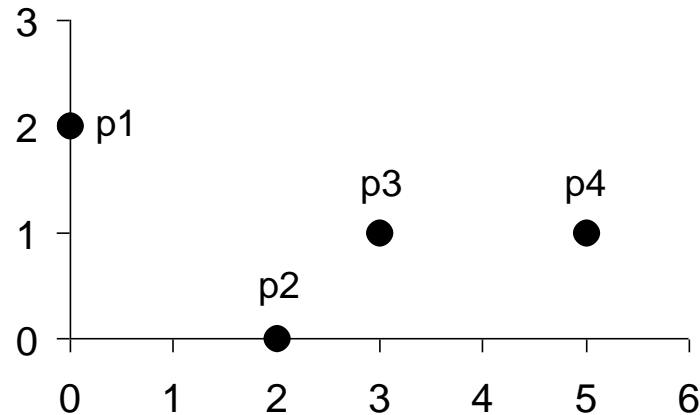
Distance/dissimilarity matrix (object-by-object structure)

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

- $n$  data points, but registers only the distance
- A symmetric/triangular matrix
- **Single mode:** row and column for the same entity (distance)

# Data Types and Representations

## Examples



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Data Matrix

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix (i.e., Dissimilarity Matrix) for Euclidean Distance

# Distance Measures

Minkowski Distance (Hermann Minkowski: [http://en.wikipedia.org/wiki/Minkowski\\_distance](http://en.wikipedia.org/wiki/Minkowski_distance))

For

$$\mathbf{x} = (x_1 \ x_2 \ \cdots \ x_n) \text{ and } \mathbf{y} = (y_1 \ y_2 \ \cdots \ y_n)$$

$$d(\mathbf{x}, \mathbf{y}) = \left( |x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p \right)^{\frac{1}{p}}, \quad p > 0$$

$p = 1$ : Manhattan (city block) distance

$$d(\mathbf{x}, \mathbf{y}) = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_n - y_n|$$

$p = 2$ : Euclidean distance

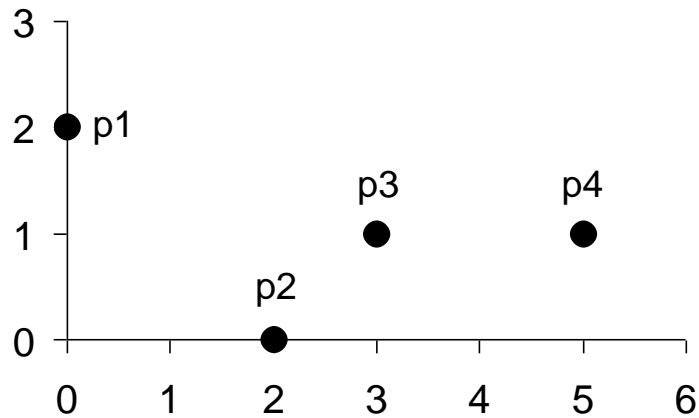
$$d(\mathbf{x}, \mathbf{y}) = \sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 + \cdots + |x_n - y_n|^2}$$

Do not confuse  $p$  with  $n$ , i.e., all these distances are defined based on all numbers of features (dimensions).

A generic measure: use appropriate  $p$  in different applications

# Distance Measures

Example: Manhattan and Euclidean distances



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

Data Matrix

L1	p1	p2	p3	p4
p1	0	4	4	6
p2	4	0	2	4
p3	4	2	0	2
p4	6	4	2	0

Distance Matrix for Manhattan Distance

L2	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix for Euclidean Distance

# Distance Measures

## Cosine Measure (Similarity vs. Distance)

For

$$\mathbf{x} = (x_1 \ x_2 \ \cdots \ x_n) \text{ and } \mathbf{y} = (y_1 \ y_2 \ \cdots \ y_n)$$

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{x_1 y_1 + \cdots + x_n y_n}{\sqrt{x_1^2 + \cdots + x_n^2} \sqrt{y_1^2 + \cdots + y_n^2}}$$

$$d(\mathbf{x}, \mathbf{y}) = 1 - \cos(\mathbf{x}, \mathbf{y})$$

- Property:  $0 \leq d(\mathbf{x}, \mathbf{y}) \leq 2$
- Nonmetric vector objects: keywords in documents, gene features in micro-arrays, ...
- Applications: information retrieval, biologic taxonomy, ...

# Distance Measures

Example: Cosine measure

$$\mathbf{x}_1 = (3, 2, 0, 5, 2, 0, 0), \mathbf{x}_2 = (1, 0, 0, 0, 1, 0, 2)$$

$$3 \times 1 + 2 \times 0 + 0 \times 0 + 5 \times 0 + 2 \times 1 + 0 \times 0 + 0 \times 2 = 5$$

$$\sqrt{3^2 + 2^2 + 0^2 + 5^2 + 2^2 + 0^2 + 0^2} = \sqrt{42} \approx 6.48$$

$$\sqrt{1^2 + 0^2 + 0^2 + 0^2 + 1^2 + 0^2 + 2^2} = \sqrt{6} \approx 2.45$$

$$\cos(\mathbf{x}_1, \mathbf{x}_2) = \frac{5}{6.48 \times 2.45} \approx 0.32$$

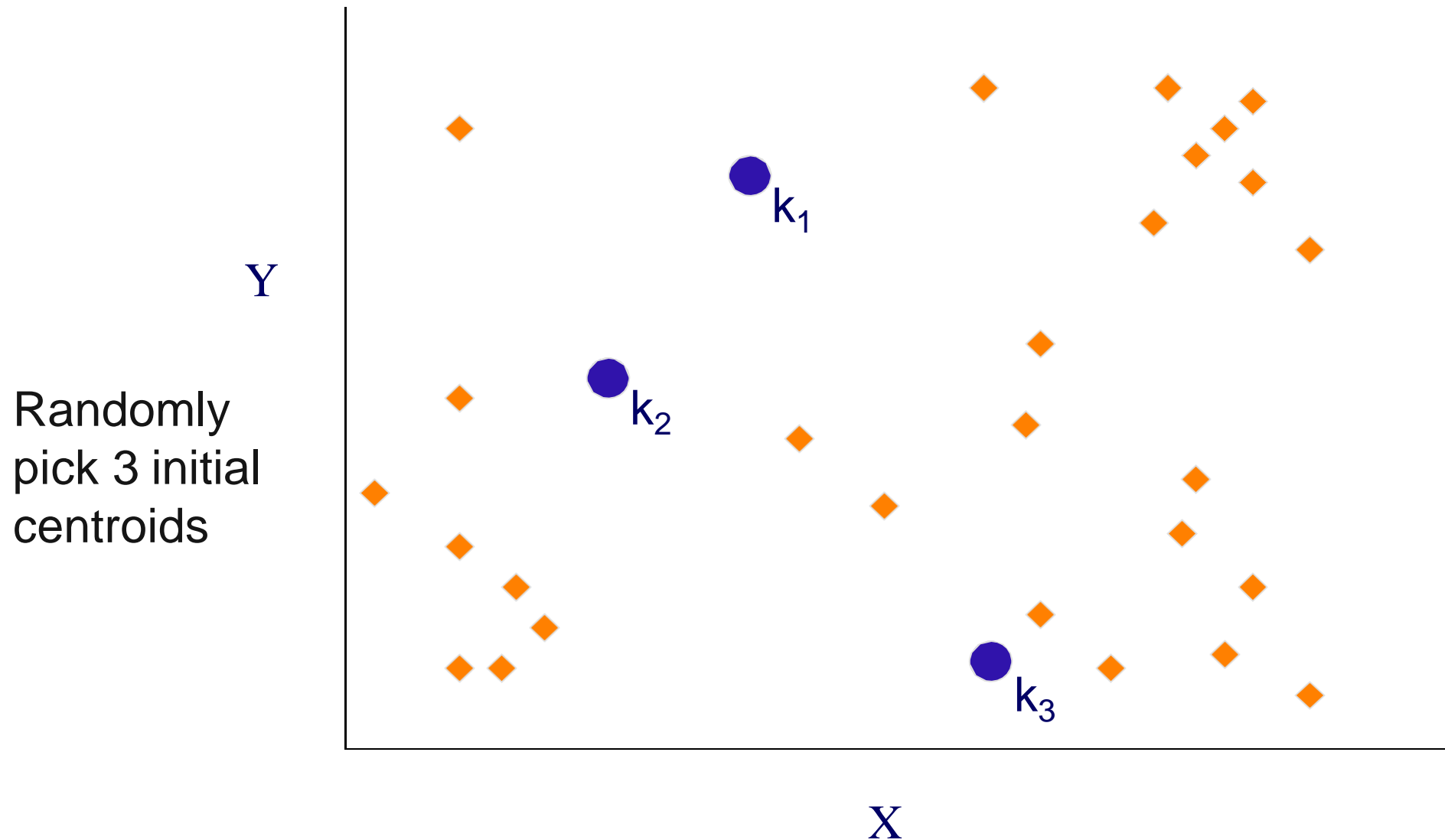
$$d(\mathbf{x}_1, \mathbf{x}_2) = 1 - \cos(\mathbf{x}_1, \mathbf{x}_2) = 1 - 0.32 = 0.68$$

## 2. K-Means Clustering

- Partitional clustering algorithm
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- **Number of clusters  $K$**  must be specified manually
- The K-Means algorithm is very simple:

- 
- 1: Select  $K$  points as the initial centroids.
  - 2: **repeat**
  - 3:   Form  $K$  clusters by assigning all points to the closest centroid.
  - 4:   Recompute the centroid of each cluster.
  - 5: **until** The centroids don't change
-

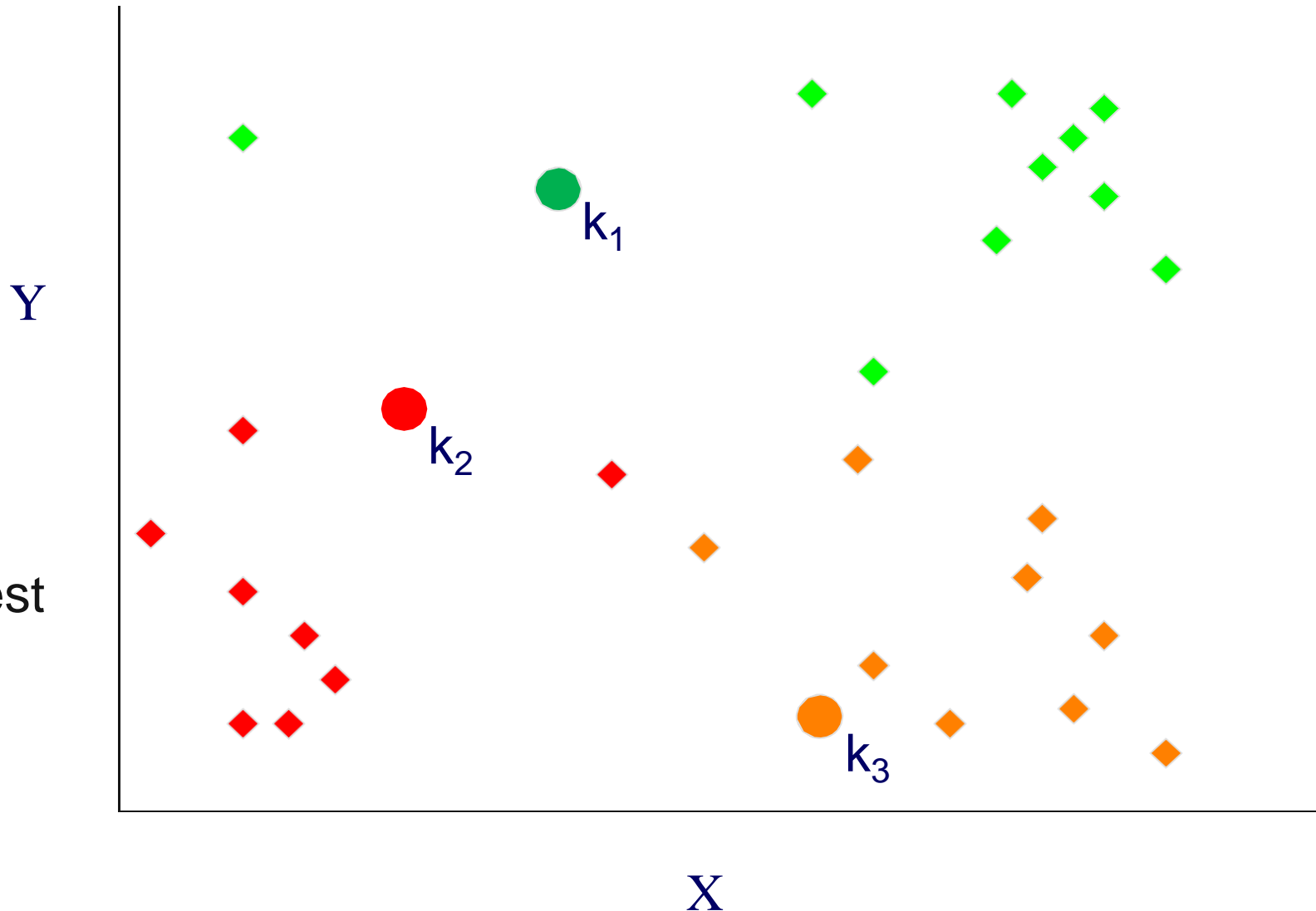
# K-Means Example, Step 1





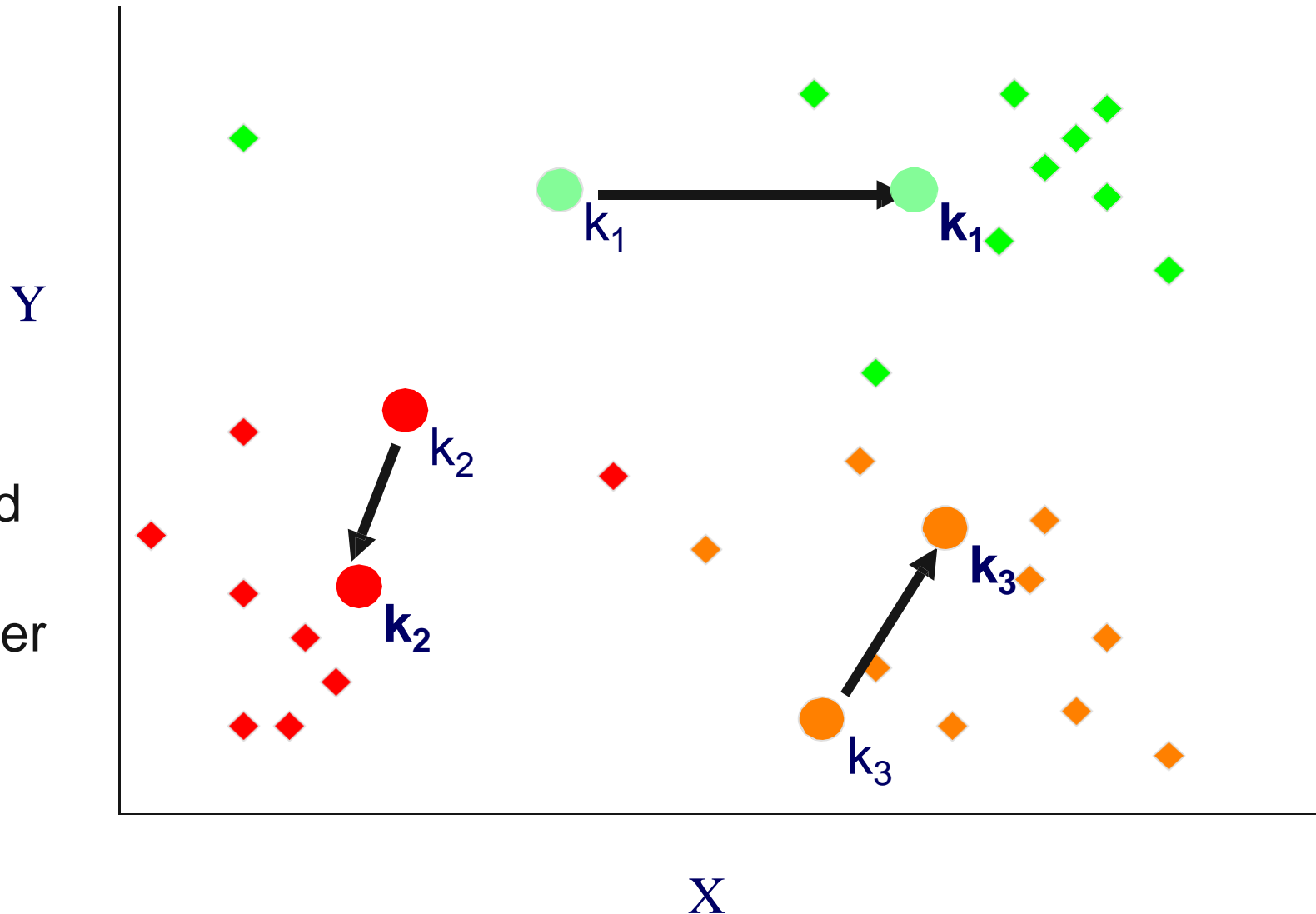
# K-Means Example, Step 2

Assign  
each point  
to the closest  
centroid



# K-Means Example, Step 3

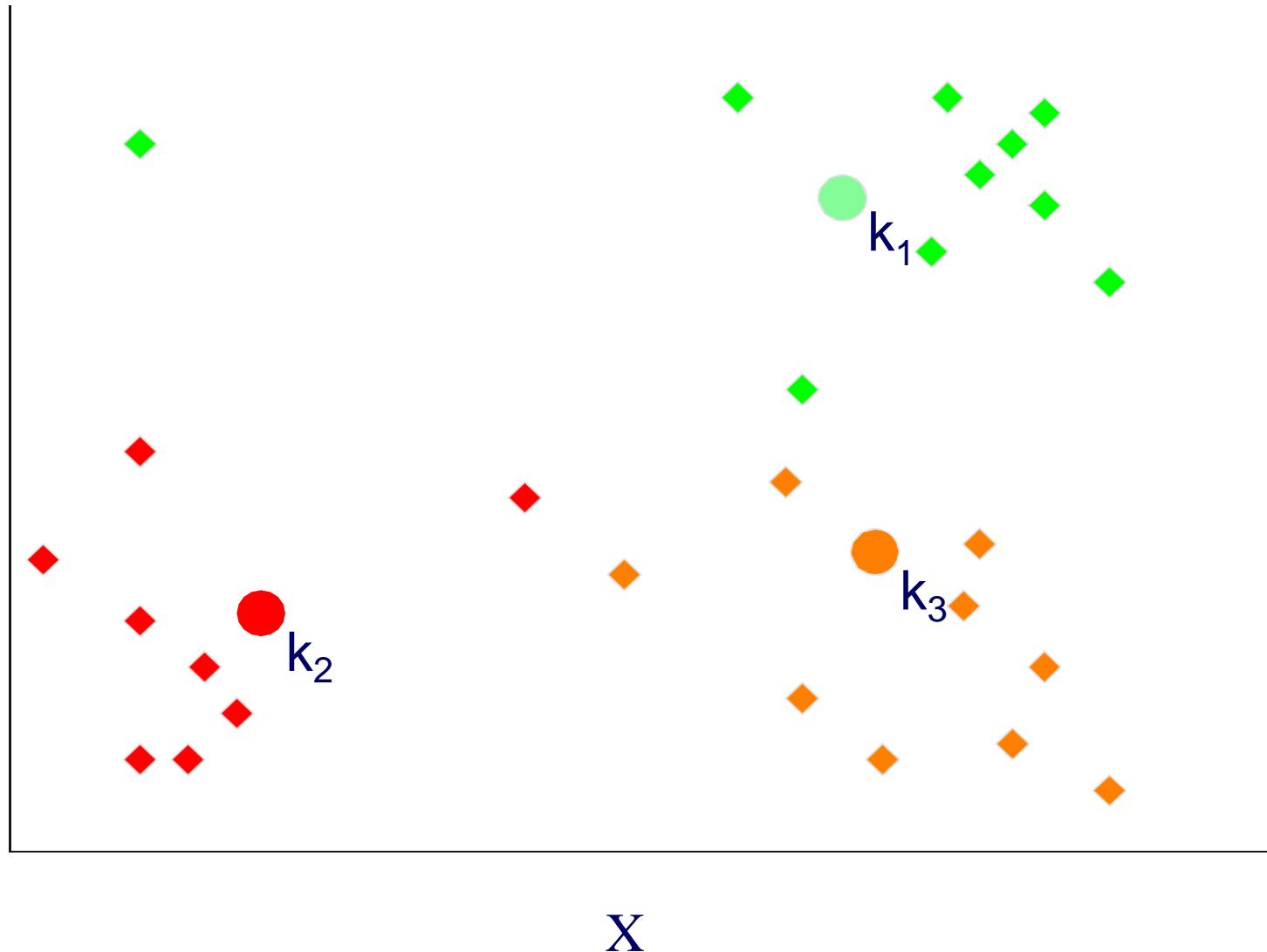
Move  
each centroid  
to **the mean**  
of each cluster



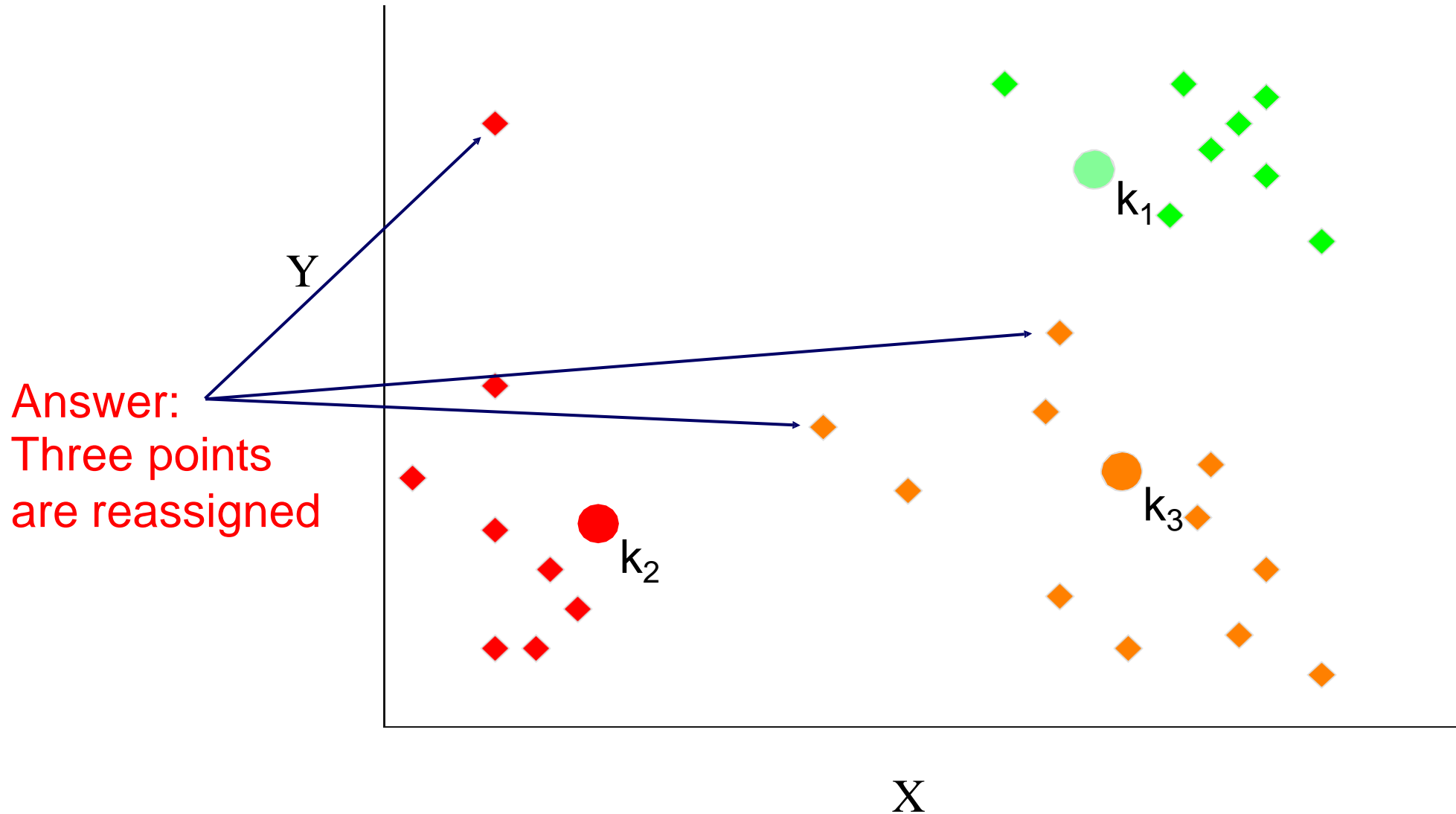
# K-Means Example, Step 4

Reassign  
points if they  
are now  
closer to a  
different  
centroid

Question:  
Which points  
are reassigned?

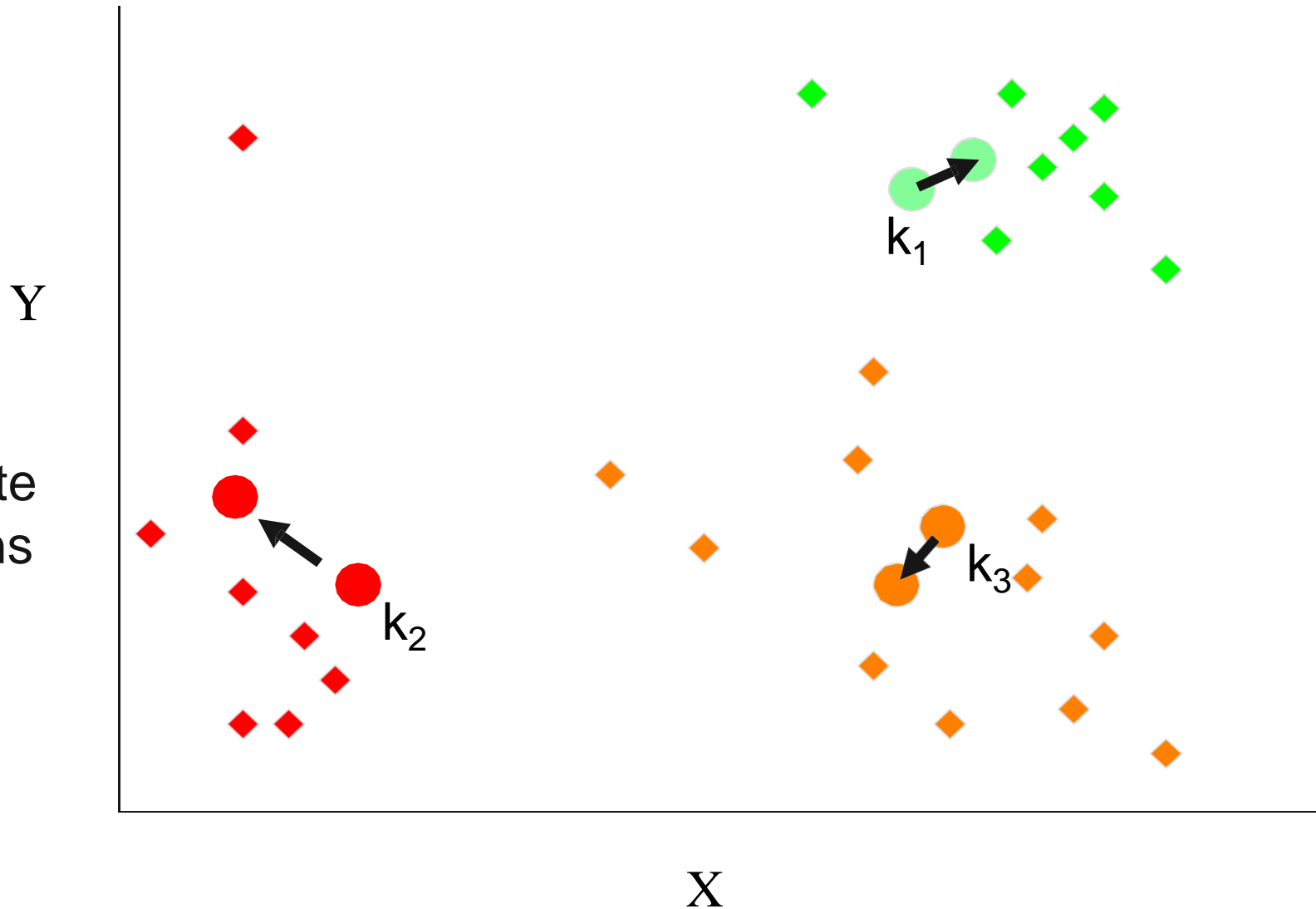


# K-Means Example, Step 4

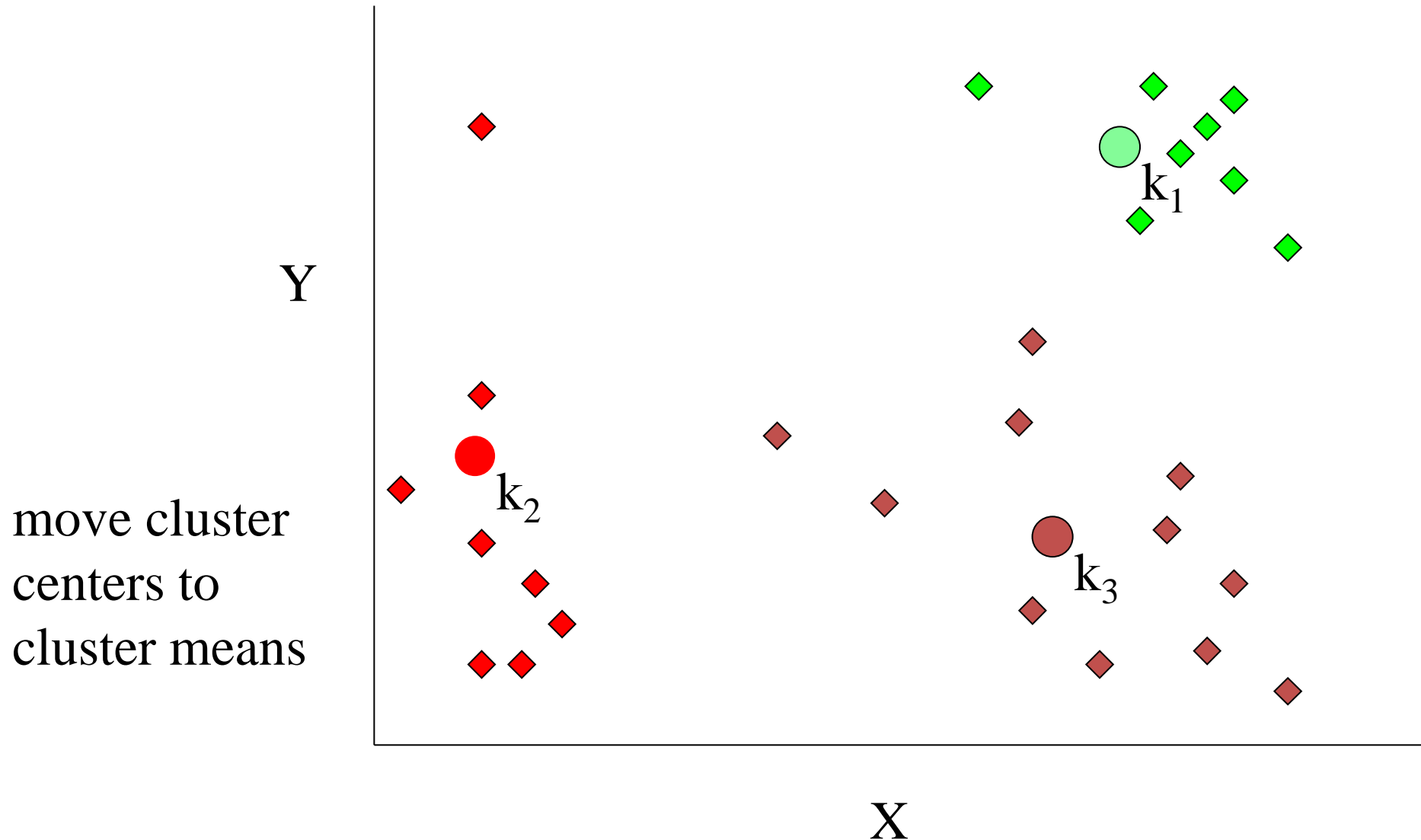


# K-Means Example, Step 5

1. Re-compute cluster means
2. Move centroids to new cluster means



## K-means example, step 5



# Convergence Criteria

Default convergence criterion

- no (or minimum) change of centroids

Alternative convergence criteria

1. no (or minimum) re-assignments of data points to different clusters
2. stop after x iterations
3. minimum decrease in the sum of squared error (SSE)
  - see next slide

# Evaluating K-Means Clusterings

- Widely used cohesion measure: **Sum of Squared Error (SSE)**
  - For each point, the error is the distance to the nearest centroid
  - To get SSE, we square these errors and sum them

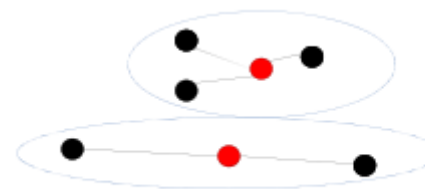
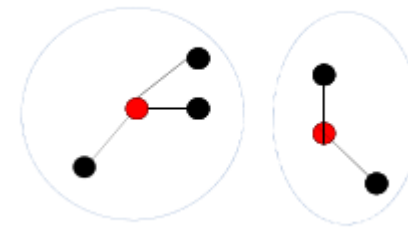
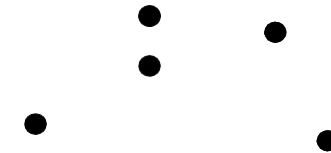
$$SSE = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} \text{dist}(\mathbf{x}, \mathbf{m}_j)^2$$

- $C_j$  is the  $j$ -th cluster
  - $\mathbf{m}_j$  is the centroid of cluster  $C_j$  (the mean vector of all the data points in  $C_j$ )
  - $\text{dist}(\mathbf{x}, \mathbf{m}_j)$  is the distance between data point  $\mathbf{x}$  and centroid  $\mathbf{m}_j$
- Given several clusterings (= groupings), we should prefer the one with the smallest SSE

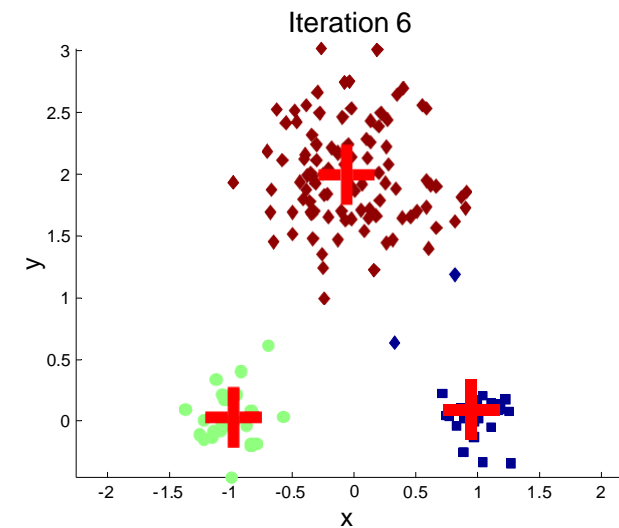
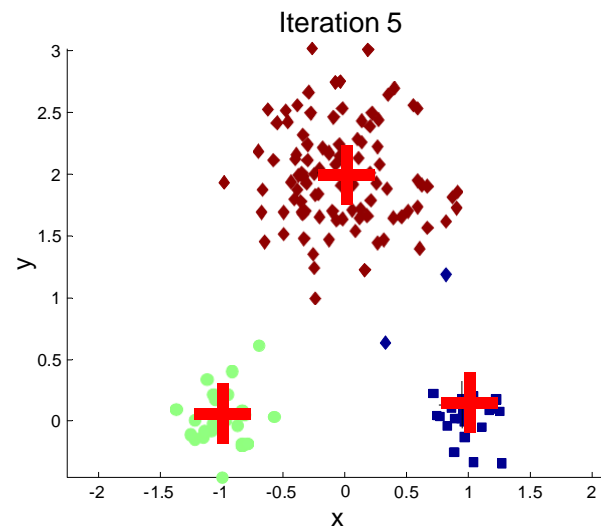
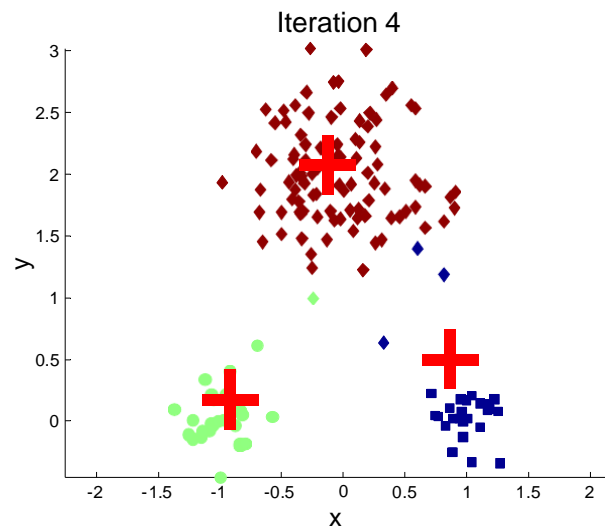
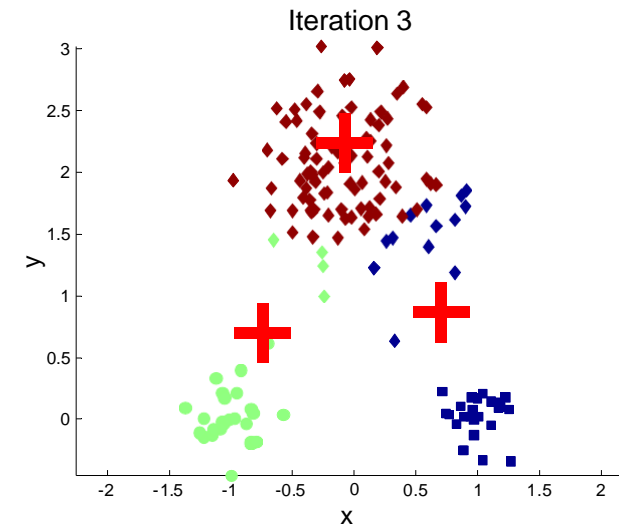
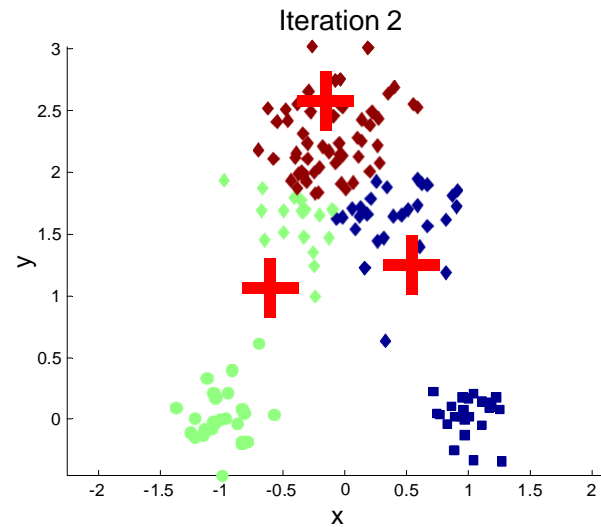
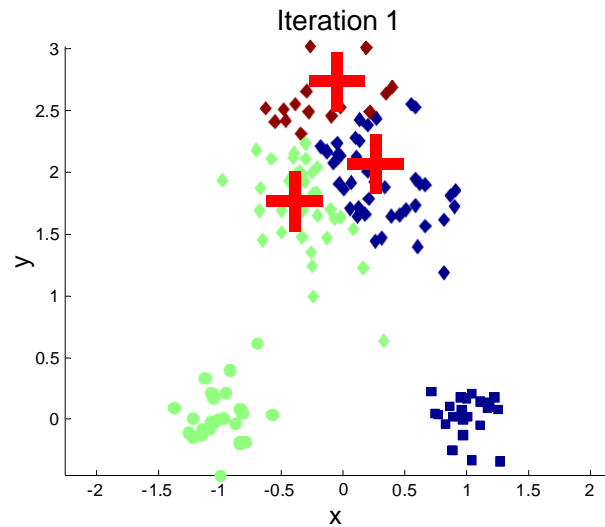


# Illustration: Sum of Squared Error

- Cluster analysis problem
- Good clustering
  - small distances to centroids
- Not so good clustering
  - larger distances to centroids

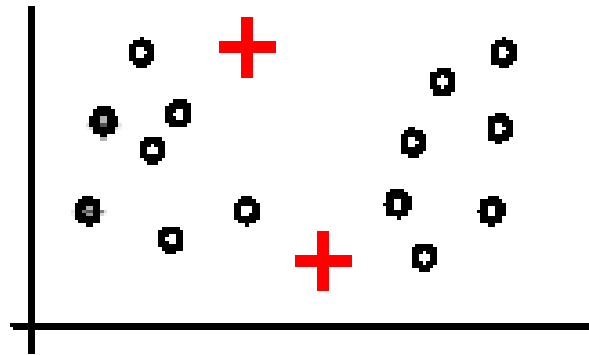


# K-Means Clustering – Second Example

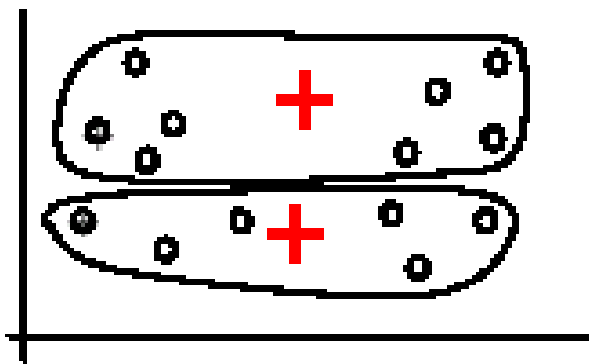


# Weaknesses of K-Means: Initial Seeds

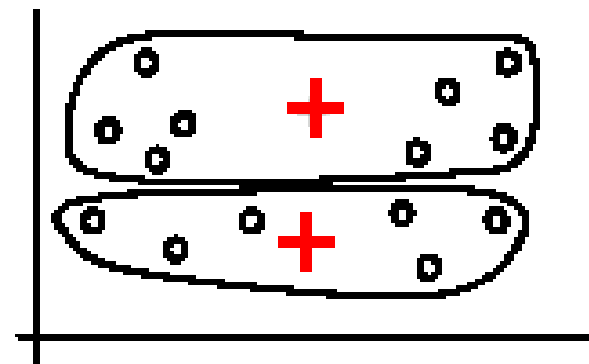
Clustering results may vary significantly depending on initial choice of seeds (**number** and **position** of seeds)



(A). Random selection of seeds (centroids)



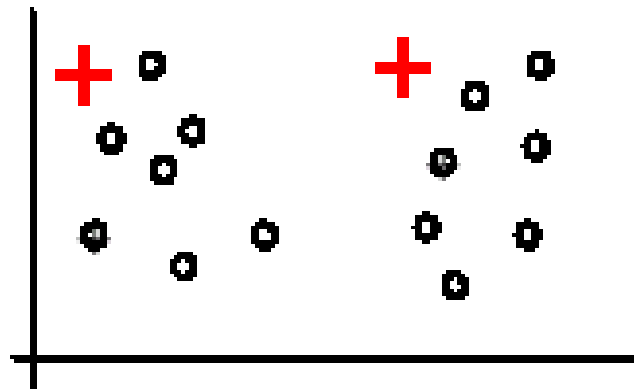
(B). Iteration 1



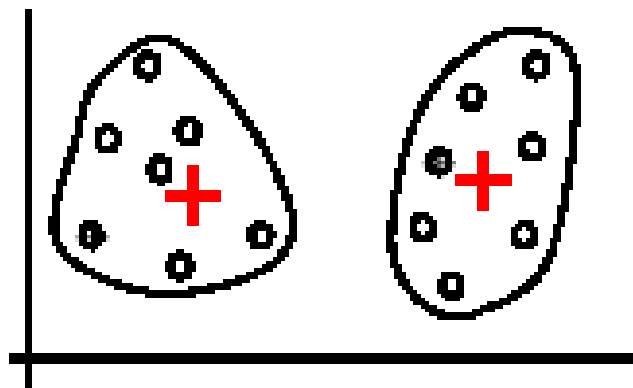
(C). Iteration 2

# Weaknesses of K-Means: Initial Seeds

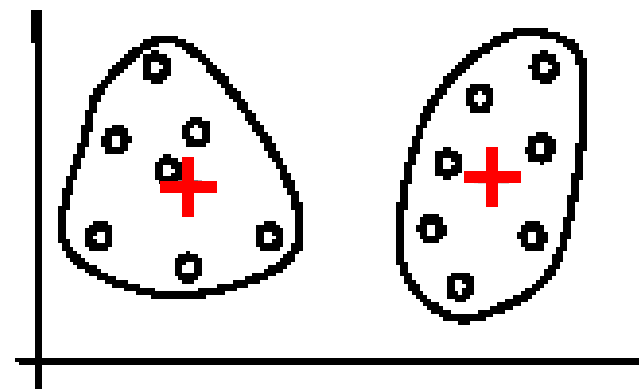
If we use **different seeds**, we get good results



(A). Random selection of  $k$  seeds (centroids)

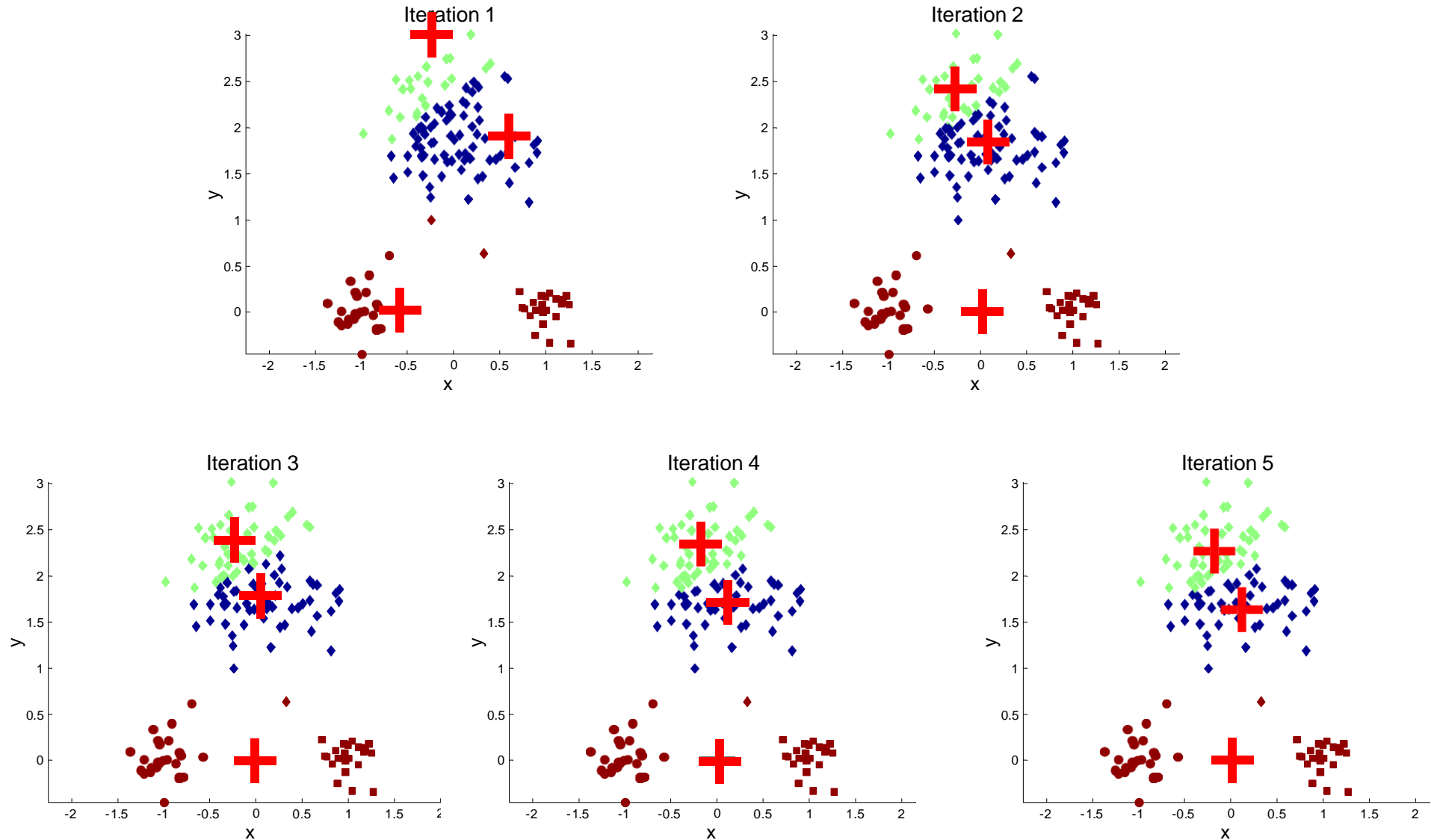


(B). Iteration 1



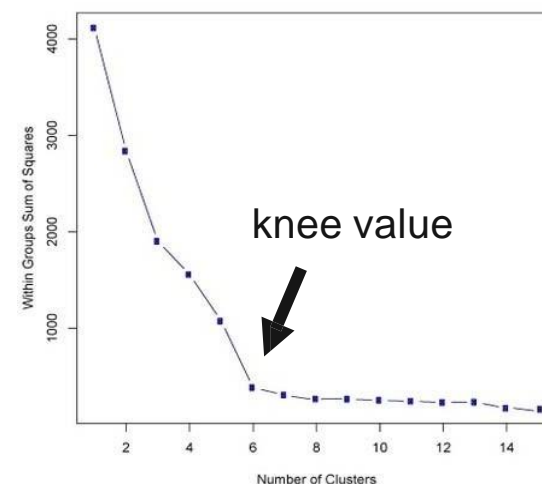
(C). Iteration 2

# Bad Initial Seeds – Second Example

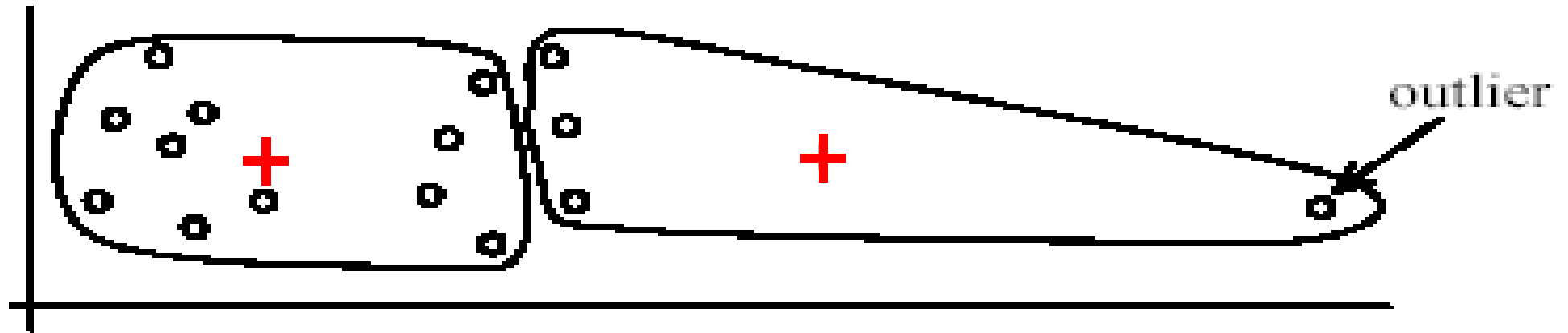


# Increasing the Chance of Finding Good Clusters

1. Restart a number of times with different random seeds
  - chose the resulting clustering with the smallest sum of squared error (SSE)
2. Run k-means with different values of k
  - The SSE for different values of k cannot directly be compared
  - think: what happens for  $k \rightarrow$  number of examples?
  - Workarounds
3. Choose k where SSE improvement decreases (knee value of k)
4. Employ X-Means
  - variation of K-Means algorithm that automatically determines k
  - starts with small k, then splits large clusters until improvement decreases



# Weaknesses of K-Means: Problems with Outliers



(A): Undesirable clusters



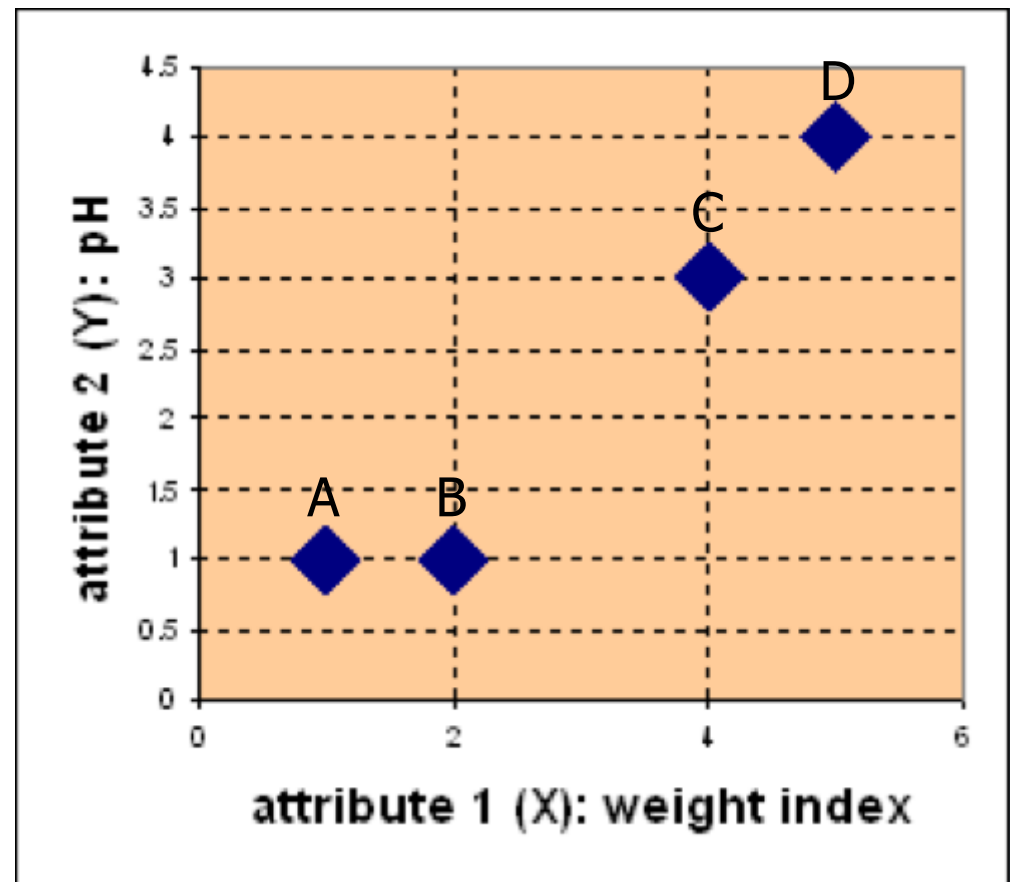
(B): Better clusters

# Example

## Problem

Suppose we have 4 types of medicines and each has two attributes (pH and weight index). Our goal is to group these objects into  $K=2$  groups of medicine.

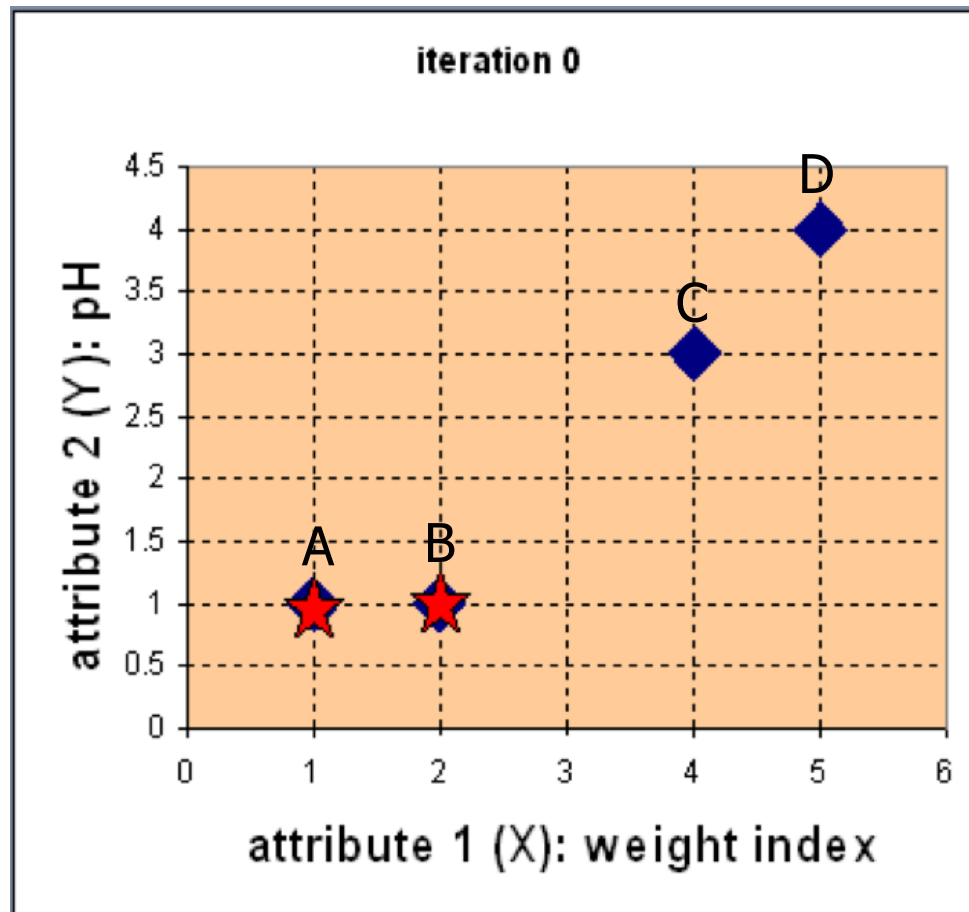
Medicine	Weight	pH-Index
A	1	1
B	2	1
C	4	3
D	5	4





# Example

Step 1: Use initial seed points for partitioning



$$c_1 = A, c_2 = B$$

$\mathbf{D}^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix}$	$\mathbf{c}_1 = (1,1)$	group - 1
	$\mathbf{c}_2 = (2,1)$	group - 2
$A \quad B \quad C \quad D$	Euclidean distance	
$\begin{bmatrix} 1 & 2 & 4 & 5 \end{bmatrix}$	$X$	
$\begin{bmatrix} 1 & 1 & 3 & 4 \end{bmatrix}$	$Y$	

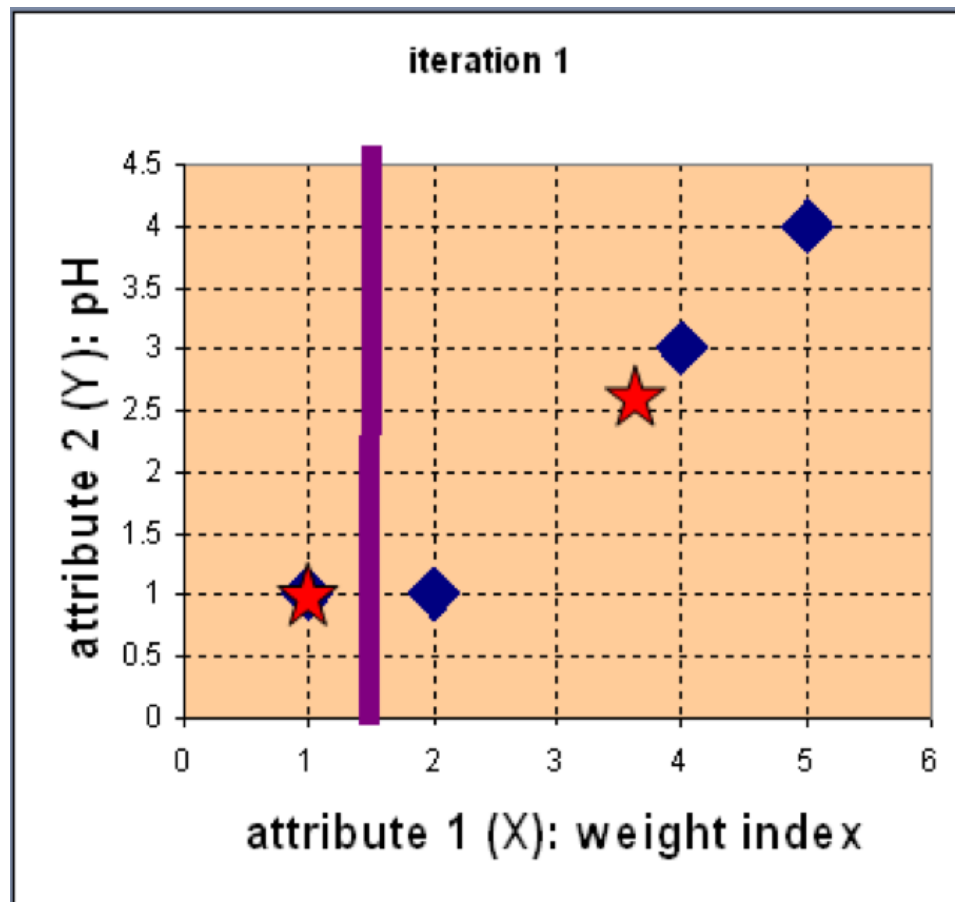
$$d(D, c_1) = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

$$d(D, c_2) = \sqrt{(5-2)^2 + (4-1)^2} = 4.24$$

Assign each object to the cluster with the nearest seed point

# Example

Step 2: Compute new centroids of the current partition



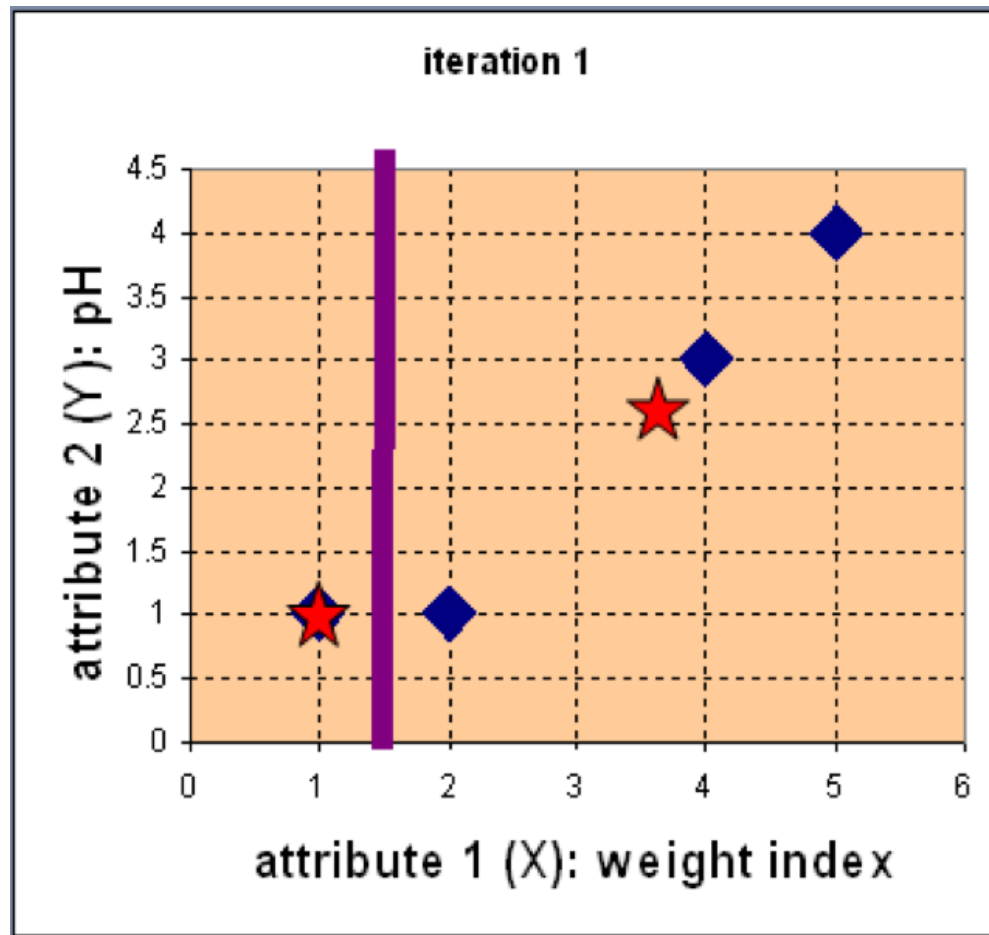
Knowing the members of each cluster, now we compute the new centroid of each group based on these new memberships.

$$c_1 = (1, 1)$$

$$\begin{aligned} c_2 &= \left( \frac{2 + 4 + 5}{3}, \frac{1 + 3 + 4}{3} \right) \\ &= \left( \frac{11}{3}, \frac{8}{3} \right) \end{aligned}$$

# Example

Step 2: Renew membership based on new centroids



Compute the distance of all objects to the new centroids

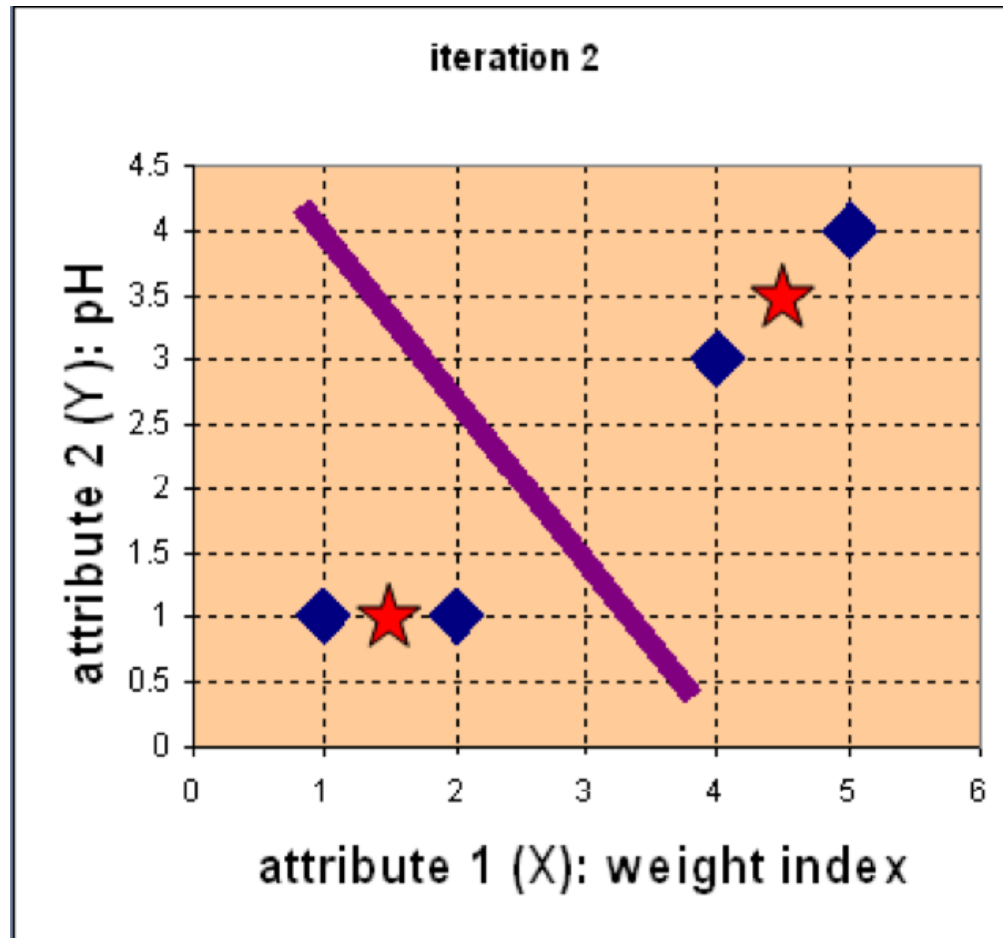
$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{array}{l} \mathbf{c}_1 = (1, 1) \text{ group-1} \\ \mathbf{c}_2 = (\frac{11}{3}, \frac{8}{3}) \text{ group-2} \end{array}$$

	A	B	C	D	
	1	2	4	5	X
	1	1	3	4	Y

Assign the membership to objects

# Example

Step 3: Repeat the first two steps until its convergence



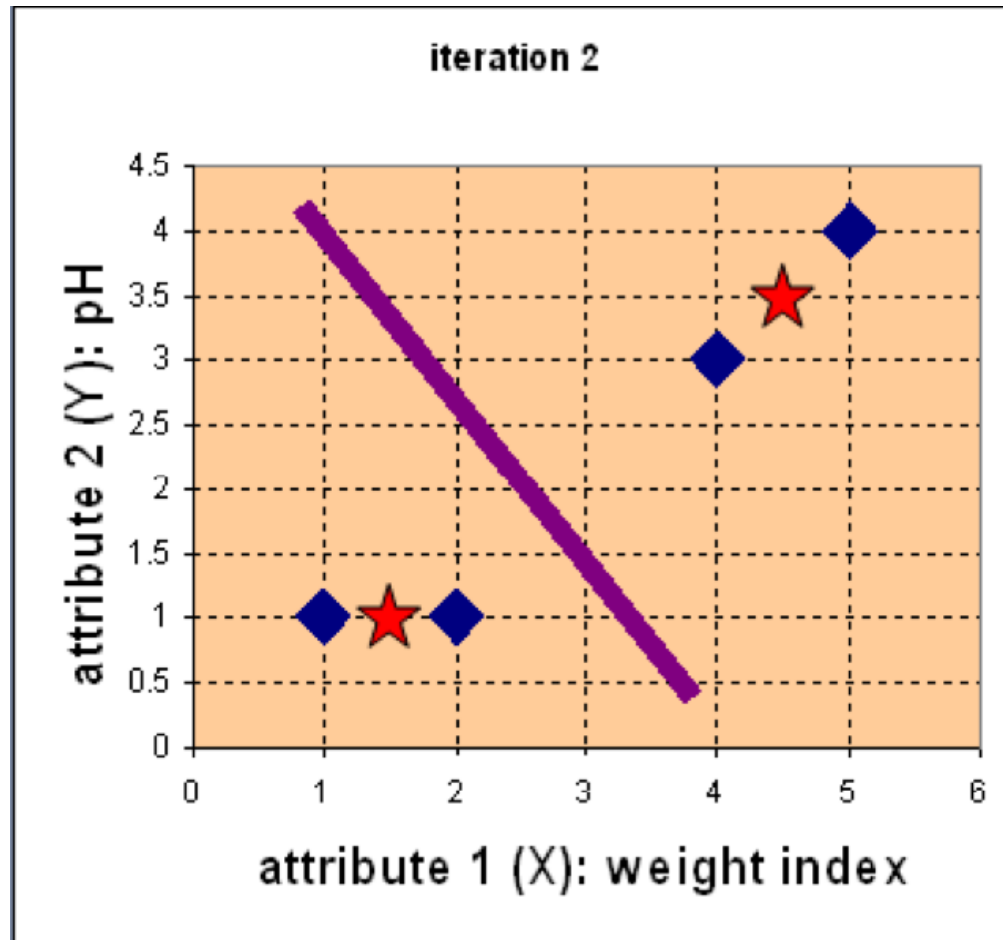
Knowing the members of each cluster, now we compute the new centroid of each group based on these new memberships.

$$c_1 = \left( \frac{1+2}{2}, \frac{1+1}{2} \right) = \left( 1\frac{1}{2}, 1 \right)$$

$$c_2 = \left( \frac{4+5}{2}, \frac{3+4}{2} \right) = \left( 4\frac{1}{2}, 3\frac{1}{2} \right)$$

# Example

Step 3: Repeat the first two steps until its convergence



Compute the distance of all objects to the new centroids

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{array}{l} \mathbf{c}_1 = (1\frac{1}{2}, 1) \text{ group-1} \\ \mathbf{c}_2 = (4\frac{1}{2}, 3\frac{1}{2}) \text{ group-2} \end{array}$$

	<i>A</i>	<i>B</i>	<i>C</i>	<i>D</i>	
	1	2	4	5	<i>X</i>
	1	1	3	4	<i>Y</i>

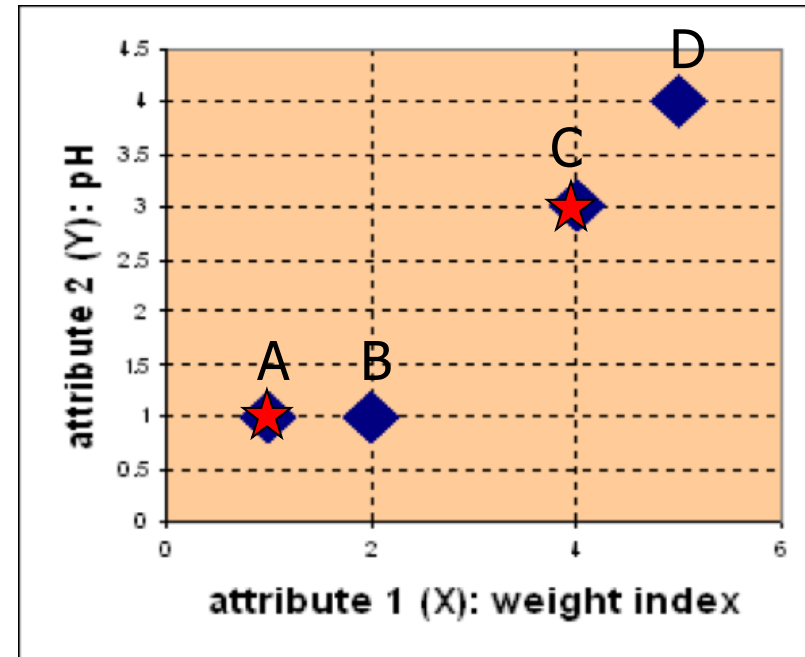
Stop due to no new assignment  
Membership in each cluster no longer change

# Exercise

For the medicine data set, use K-means with the **Manhattan** distance metric for clustering analysis by setting  **$K=2$**  and initialising seeds as  **$C_1 = A$  and  $C_2 = C$** . Answer three questions as follows:

1. How many steps are required for convergence?
2. What are memberships of two clusters after convergence?
3. What are centroids of two clusters after convergence?

Medicine	Weight	pH-Index
A	1	1
B	2	1
C	4	3
D	5	4



# Weaknesses of K-Means: Problems with Outliers

Approaches to deal with outliers:

## 1. K-Medoids

- K-Medoids is a K-Means variation that uses the **median** of each cluster instead of the mean
- Medoids are the most central **existing data points** in each cluster
- K-Medoids is more robust against outliers as the median is less affected by extreme values:
  - Mean and Median of 1, 3, 5, 7, 9 is **5**
  - Mean of 1, 3, 5, 7, 1009 is **205**
  - Median of 1, 3, 5, 7, 1009 is **5**

## 2. DBSCAN

- Density-based clustering method that **removes outliers**
  - see next section

# K-Means Clustering Summary

## Advantages

- Simple, understandable
- Efficient time complexity:  
 $O(n * K * I * d)$   
where
  - $n$  = number of points
  - $K$  = number of clusters
  - $I$  = number of iterations
  - $d$  = number of attributes

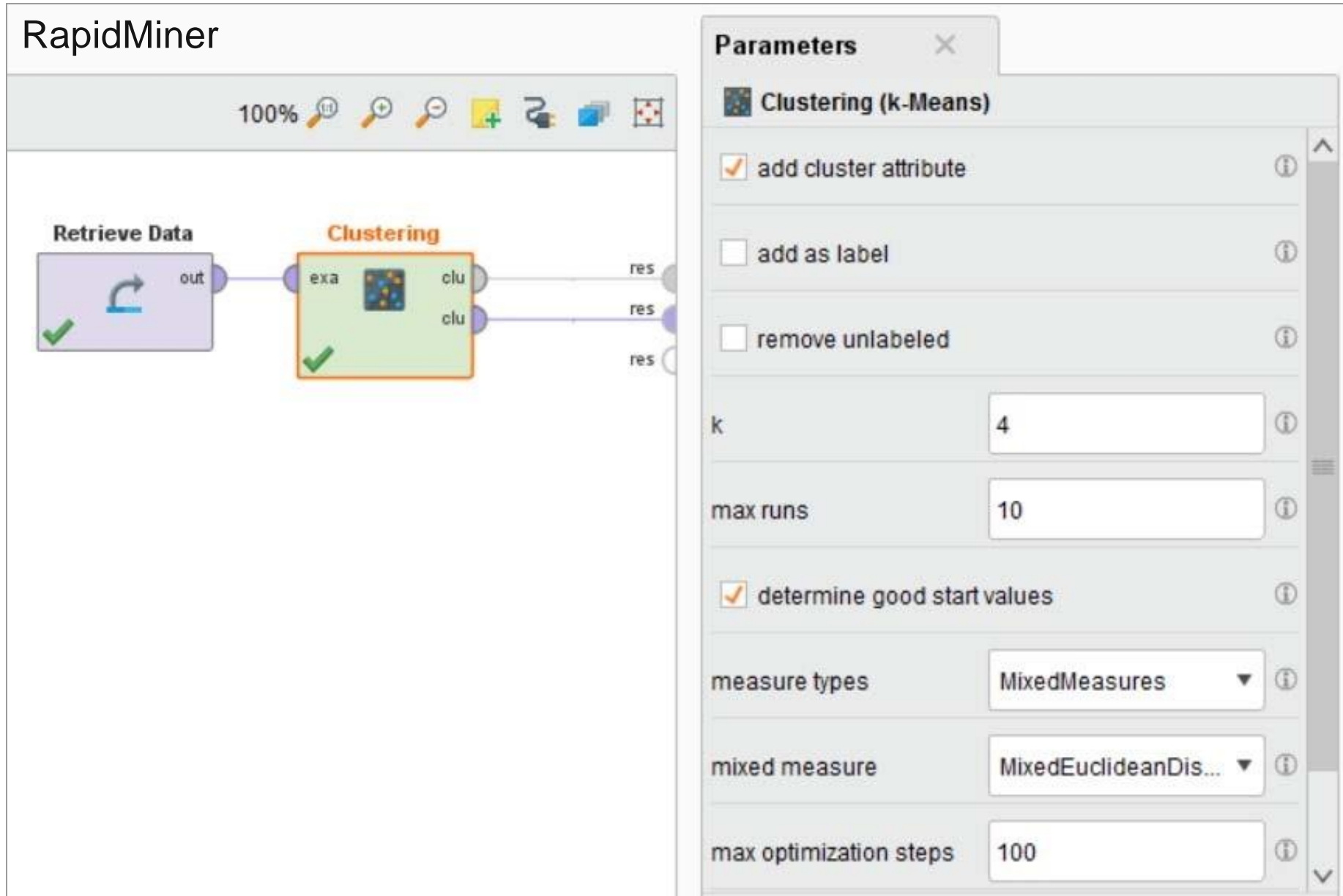
## Disadvantages

- Need to determine number of clusters
- All items are forced into a cluster
- Sensitive to outliers



# K-Means Clustering in RapidMiner and Python

RapidMiner



The image displays the RapidMiner software interface. On the left, a workflow is shown with two main nodes: 'Retrieve Data' (purple) and 'Clustering' (green). The 'Retrieve Data' node has an 'out' port connected to the 'Clustering' node's 'exa' port. The 'Clustering' node has two 'clu' ports, each connected to a 'res' port. A green checkmark is visible on the 'Clustering' node. The 'Parameters' panel on the right is titled 'Clustering (k-Means)' and contains the following settings:

- ☒ add cluster attribute
- ☐ add as label
- ☐ remove unlabeled
- k: 4
- max runs: 10
- ☒ determine good start values
- measure types: MixedMeasures
- mixed measure: MixedEuclideanDis...
- max optimization steps: 100

# K-Means Clustering Results

Result History

ExampleSet (Clustering) × Cluster Model (Clustering) ×

Open in Turbo Prep Auto Model

Filter (150 / 150 examples):

Row No.	id	label	cluster	a1	a2	a3	
1	id_1	Iris-setosa	cluster_1	5.100	5.500	1.400	0.200
2	id_2	Iris-setosa	cluster_1	4.900	3	1.400	0.200
3	id_3	Iris-se					
4	id_4	Iris-se					
5	id_5	Iris-se					
6	id_6	Iris-se					
7	id_7	Iris-se					
8	id_8	Iris-se					
9	id_9	Iris-se					
10	id_10	Iris-se					
11	id_11	Iris-se					
12	id_12	Iris-se					
13	id_13	Iris-se					
14	id_14	Iris-se					

Data

Statistics

Visualizations

Annotations

New cluster attribute

Result History

ExampleSet (Clustering) × Cluster Model (Clustering) ×

Description

Attribute	cluster_0	cluster_1	cluster_2
a1	6.913	5.006	6.252
a2	3.100	3.418	2.855
a3			
a4			

Folder View

Graph

Centroid Table

Result History

ExampleSet (Clustering) × Cluster Model (Clustering) ×

Description

### Cluster Model

Cluster 0: 32 items  
Cluster 1: 50 items  
Cluster 2: 40 items  
Cluster 3: 28 items  
Total number of items: 150

Folder View

Graph

Centroid Table

### 3. Density-based Clustering



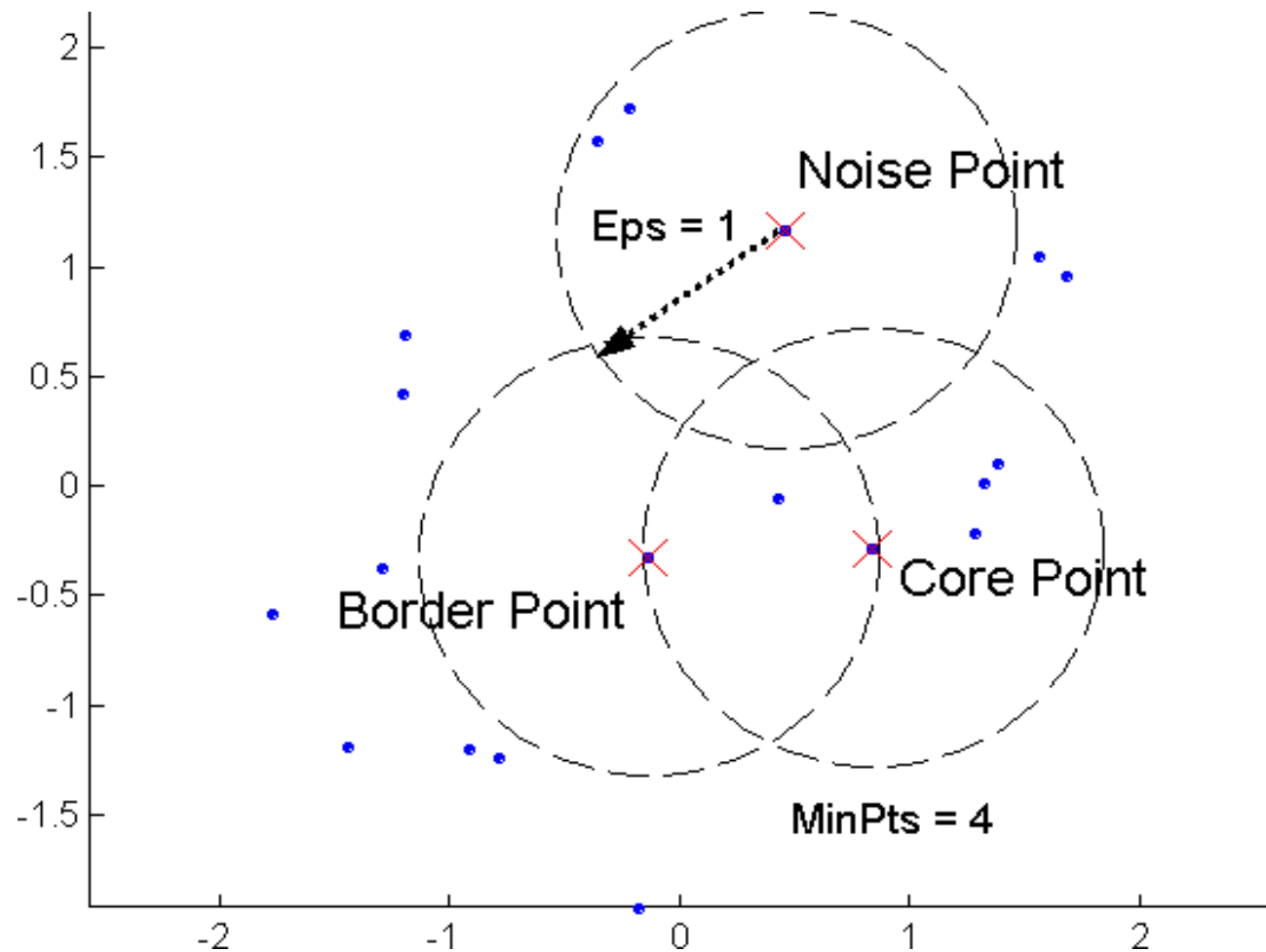
Challenging use case for K-Means because

- Problem 1: Non-globular shapes
- Problem 2: Outliers / noise points

# DBSCAN

- DBSCAN is a density-based algorithm
  - **Density** = number of points within a specified radius Epsilon (Eps)
- Divides data points into three classes:
  1. A point is a **core point** if it has at least a specified number of neighboring points (MinPts) within the specified radius Eps
    - the point itself is counted as well
    - these points form the interior of a dense region (cluster)
  2. A **border point** has fewer points than MinPts within Eps, but is in the neighborhood of a core point
  3. A **noise point** is any point that is not a core point or a border point

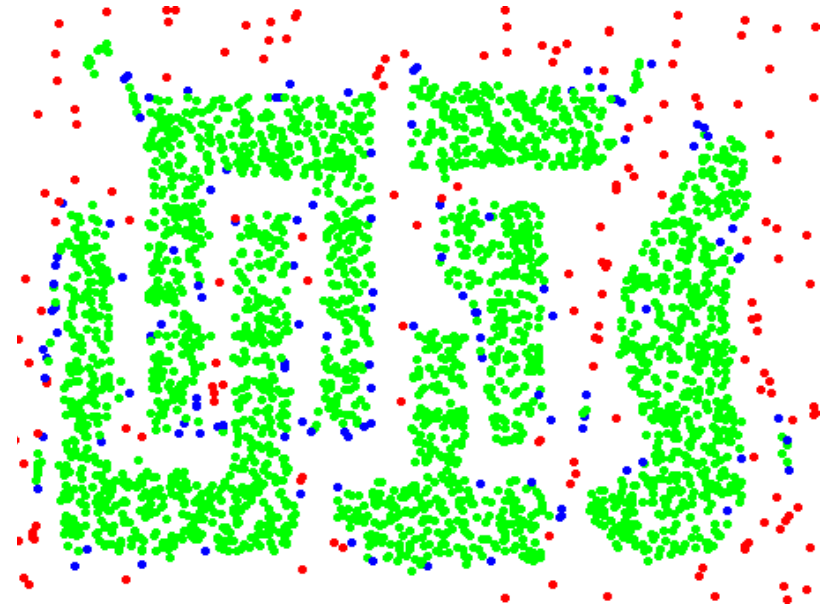
# Examples of Core, Border, and Noise Points 1



# Examples of Core, Border, and Noise Points 2



Original Points



Point types: **core**,  
**border** and **noise**

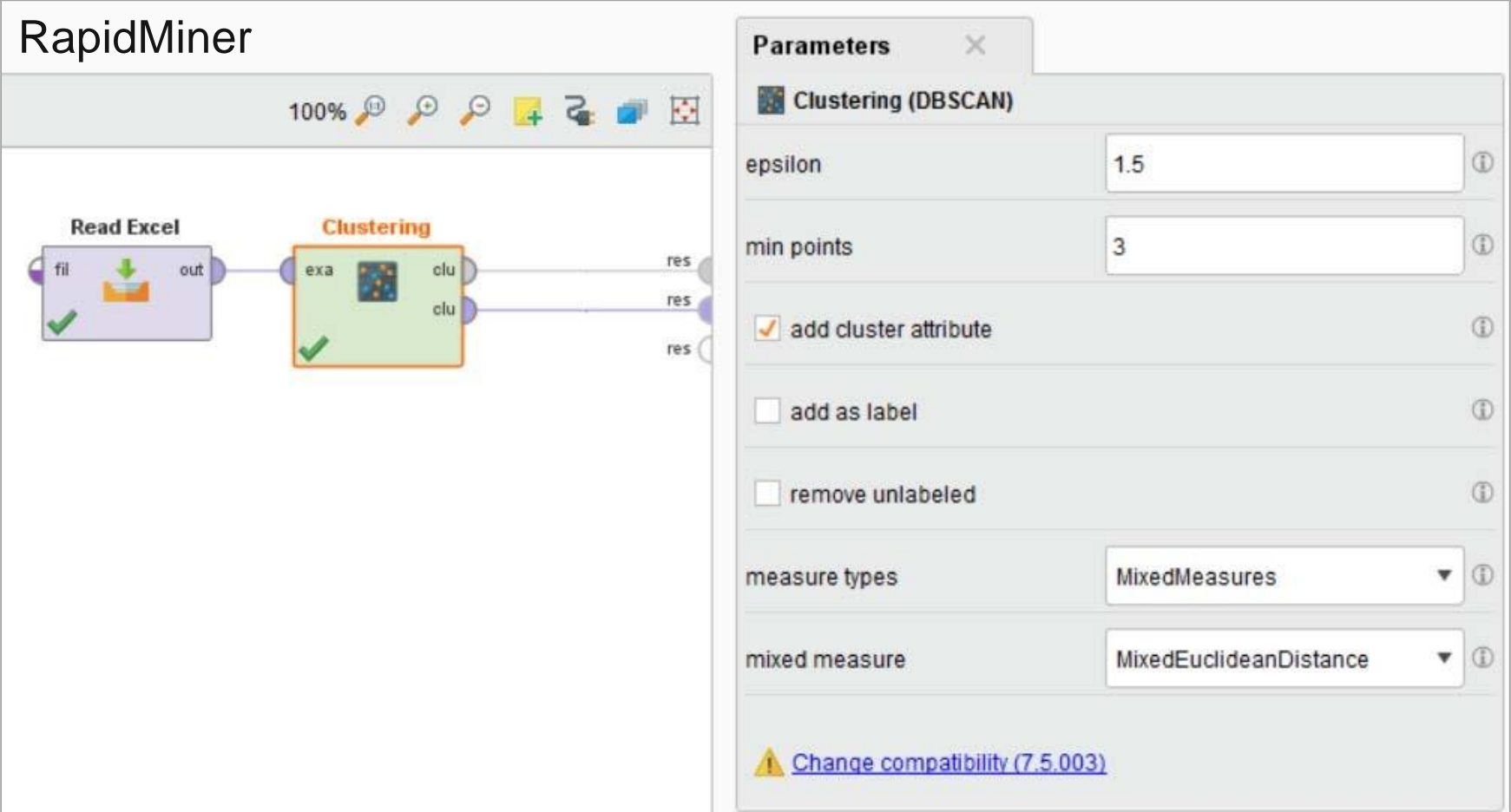
# The DBSCAN Algorithm

Eliminates noise points and returns clustering of the remaining points:

1. Label all points as core, border, or noise points
2. Eliminate all noise points
3. Put an edge between all core points that are within Eps of each other
4. Make each group of connected core points into a separate cluster
5. Assign each border point to one of the clusters of its associated core points
  - as a border point can be at the border of multiple clusters
  - use voting if core points belong to different clusters
  - if equal vote, than assign border point randomly

# DBSCAN in RapidMiner and Python

RapidMiner



The image displays the RapidMiner software interface. On the left, a workflow is visible with two main nodes: 'Read Excel' and 'Clustering'. The 'Read Excel' node has inputs 'fil' and 'out'. The 'Clustering' node has inputs 'exa' and 'clu', and outputs 'clu' and 'clu'. The 'Clustering' node is highlighted with an orange border. On the right, the 'Parameters' panel for 'Clustering (DBSCAN)' is open. It shows the following settings:

- epsilon: 1.5
- min points: 3
- ☒ add cluster attribute
- ☐ add as label
- ☐ remove unlabeled
- measure types: MixedMeasures
- mixed measure: MixedEuclideanDistance

At the bottom of the parameters panel, there is a warning icon and a link: [Change compatibility \(7.5.003\)](#).



# Hierarchical Clustering

# Outline

---

Introduction

Cluster Distance Measures

Agglomerative Algorithm

Example and Demo

---

# Hierarchical clustering

Given the input set  $S$ , the goal is to produce a hierarchy (dendrogram) in which nodes represent subsets of  $S$ .

Features of the tree obtained:

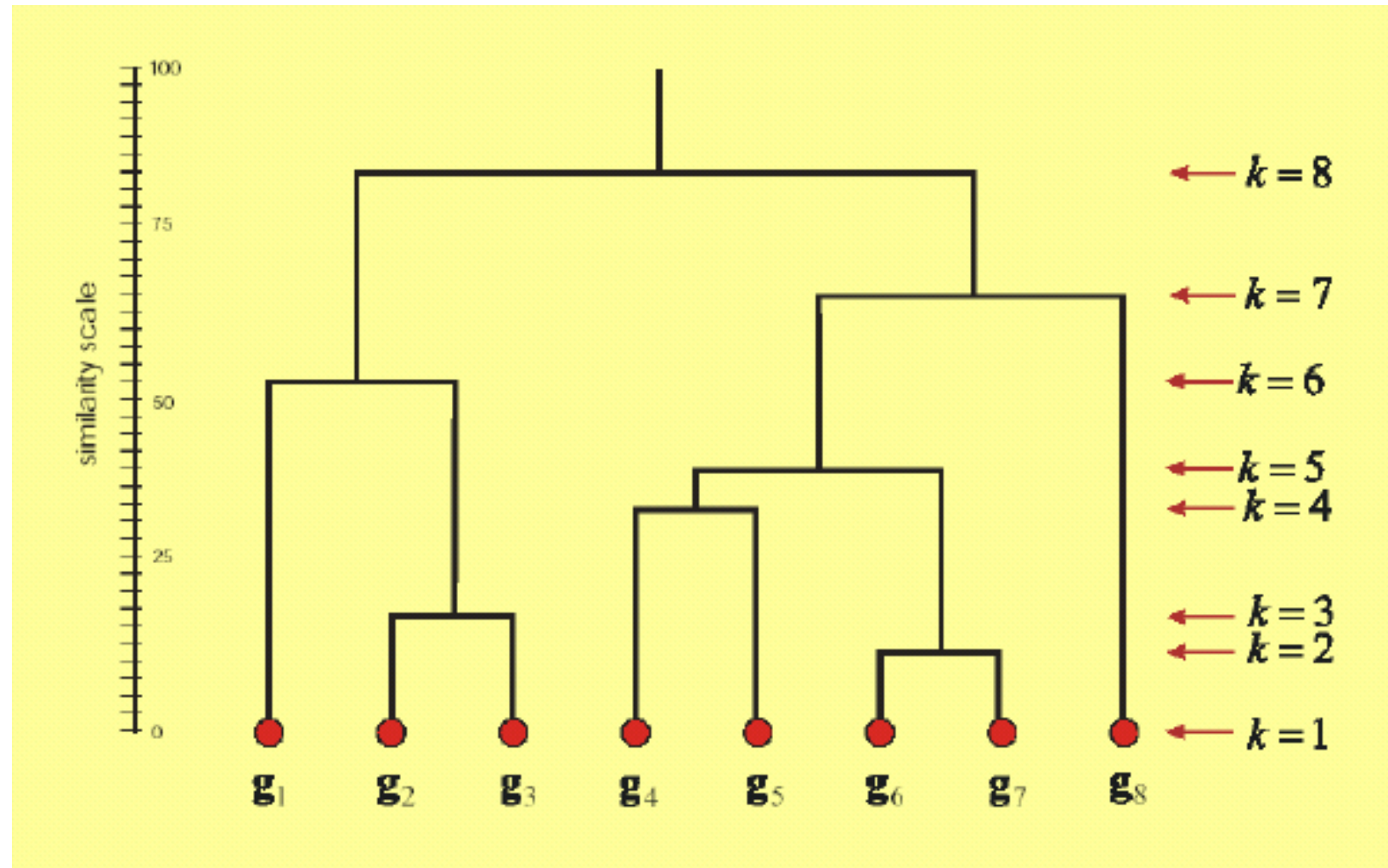
- The root is the whole input set  $S$ .

- The leaves are the individual elements of  $S$ .

- The internal nodes are defined as the union of their children.

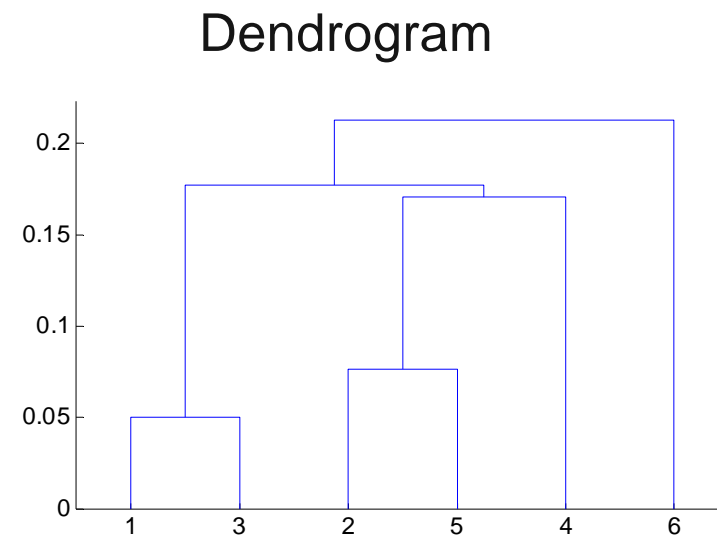
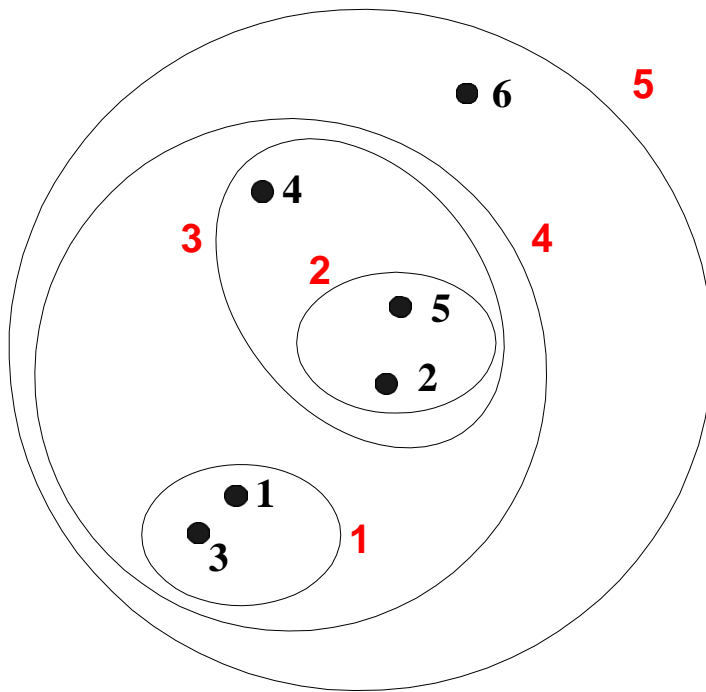
Each level of the tree represents a partition of the input data into several (nested) clusters or groups.

# Hierarchical clustering



## 4. Hierarchical Clustering

- Produces a set of **nested clusters** organized as a hierarchical tree
- Can be visualized as a **dendrogram**
  - A tree like diagram that records the sequences of merges or splits
  - The y-axis displays the former distance between merged clusters



# Two Main Types of Hierarchical Clustering

## – Agglomerative

- start with the points as individual clusters
- at each step, merge the closest pair of clusters until only one cluster (or  $k$  clusters) is left

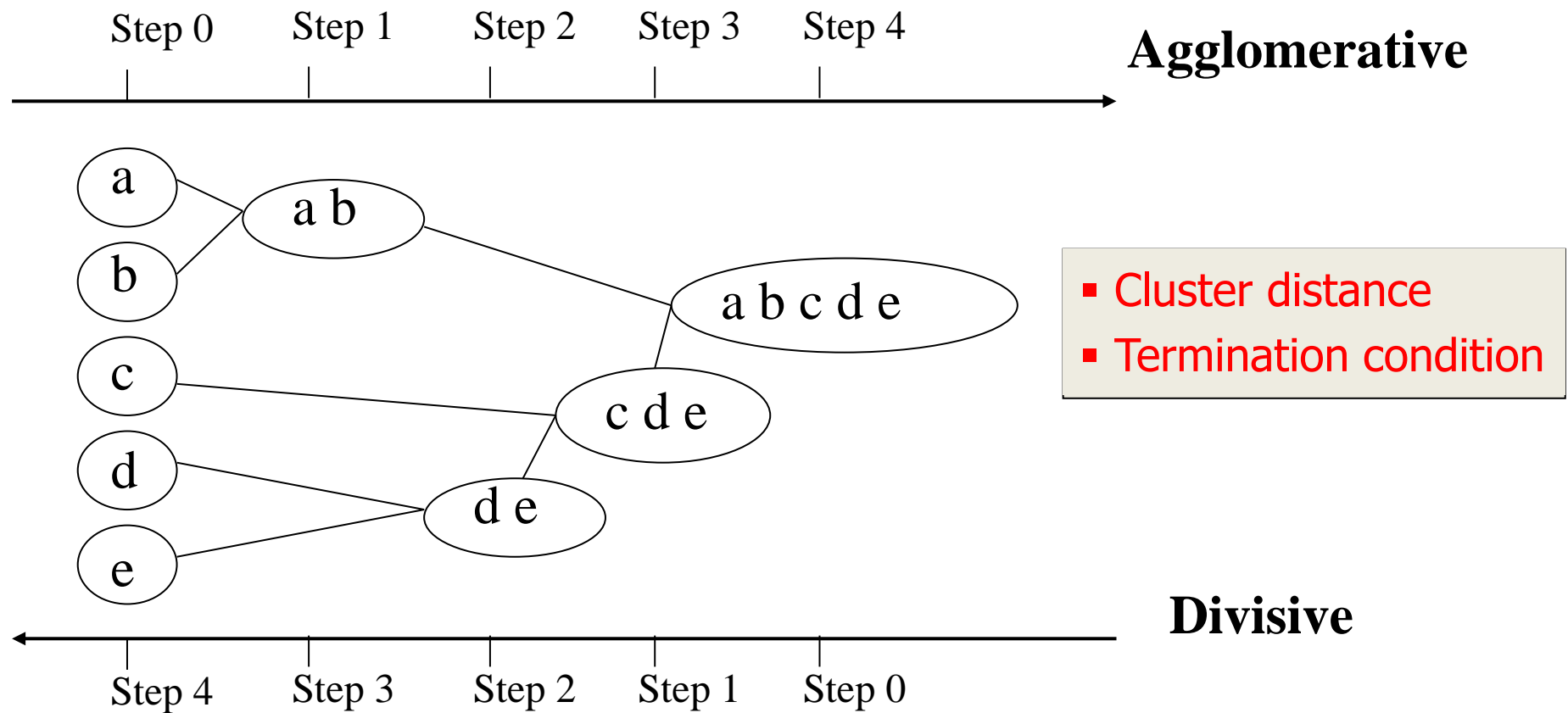
## – Divisive

- start with one, all-inclusive cluster
- at each step, split a cluster until each cluster contains a single point (or there are  $k$  clusters)

## – Agglomerative Clustering is more widely used

## Illustrative Example

Agglomerative and divisive clustering on the data set  $\{a, b, c, d, e\}$



# Agglomerative Clustering Algorithm

The basic algorithm is straightforward:

1. Compute the proximity matrix
2. Let each data point be a cluster
3. **Repeat**
  1. Merge the two closest clusters
  2. Update the proximity matrix

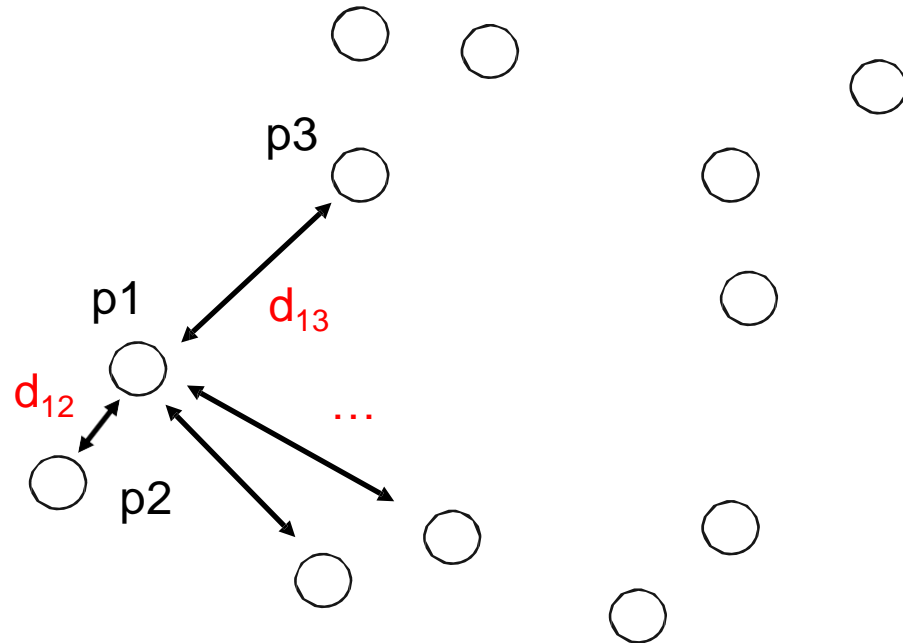
**Until** only a single cluster remains

- The key operation is the computation of the proximity of two clusters
- The different approaches to defining the distance between clusters distinguish the different algorithms



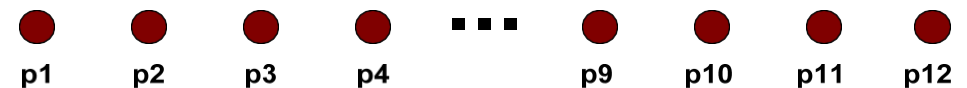
# Starting Situation

Start with clusters of individual points and a proximity matrix



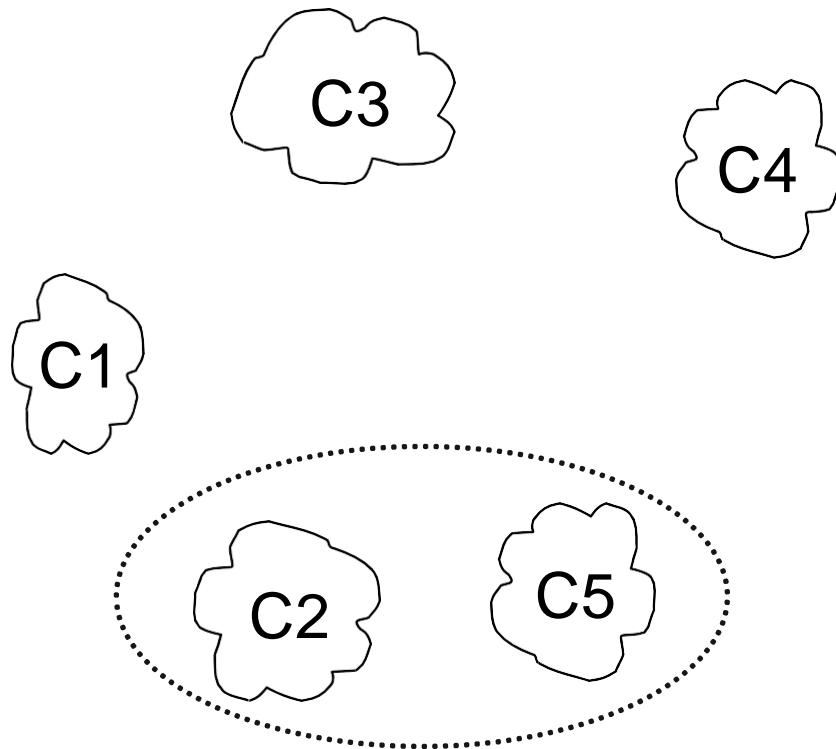
	p1	p2	p3	p4	p5	...
p1		$d_{12}$	$d_{13}$	...		
p2			...			
p3						
p4						
p5						
...						
...						

Proximity Matrix



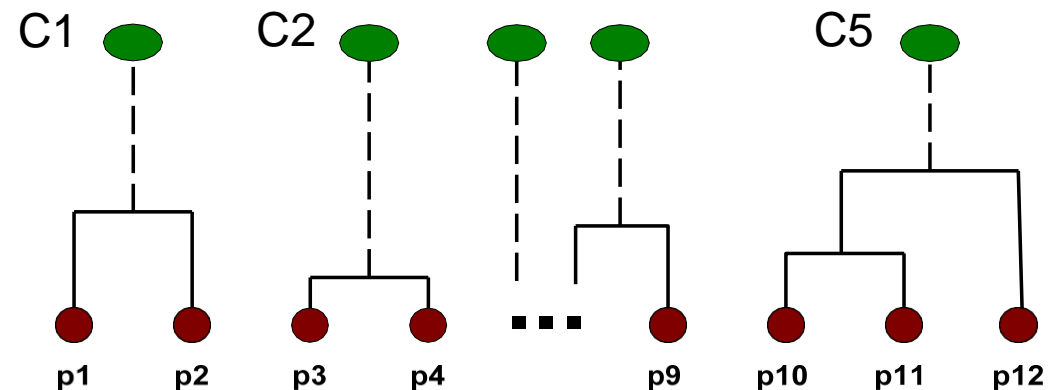
# Intermediate Situation

- After some merging steps, we have larger clusters.
- We want to keep on merging the two closest clusters (C2 and C5?)

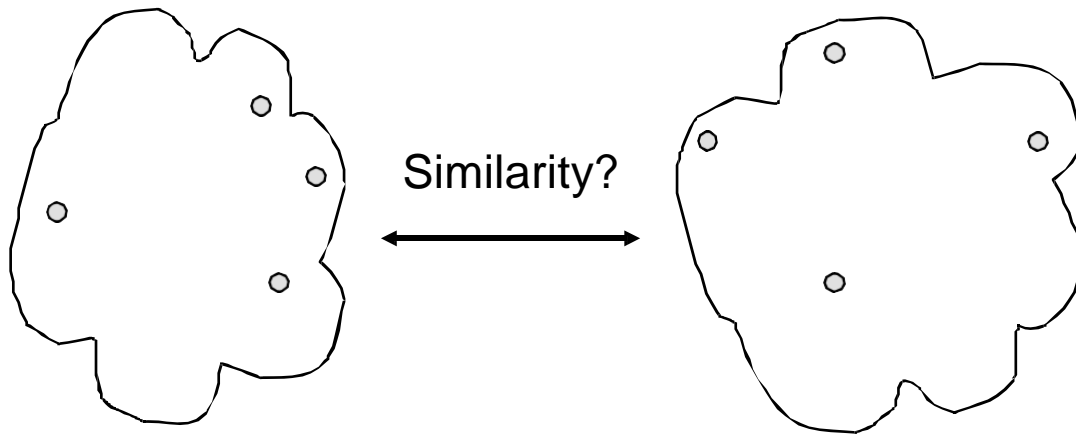


		C1	$\begin{matrix} C2 \\ \cup \\ C5 \end{matrix}$	C3	C4
C1			?		
$C2 \cup C5$		?	?	?	?
C3			?		
C4			?		

Proximity Matrix



# How to Define Inter-Cluster Similarity?



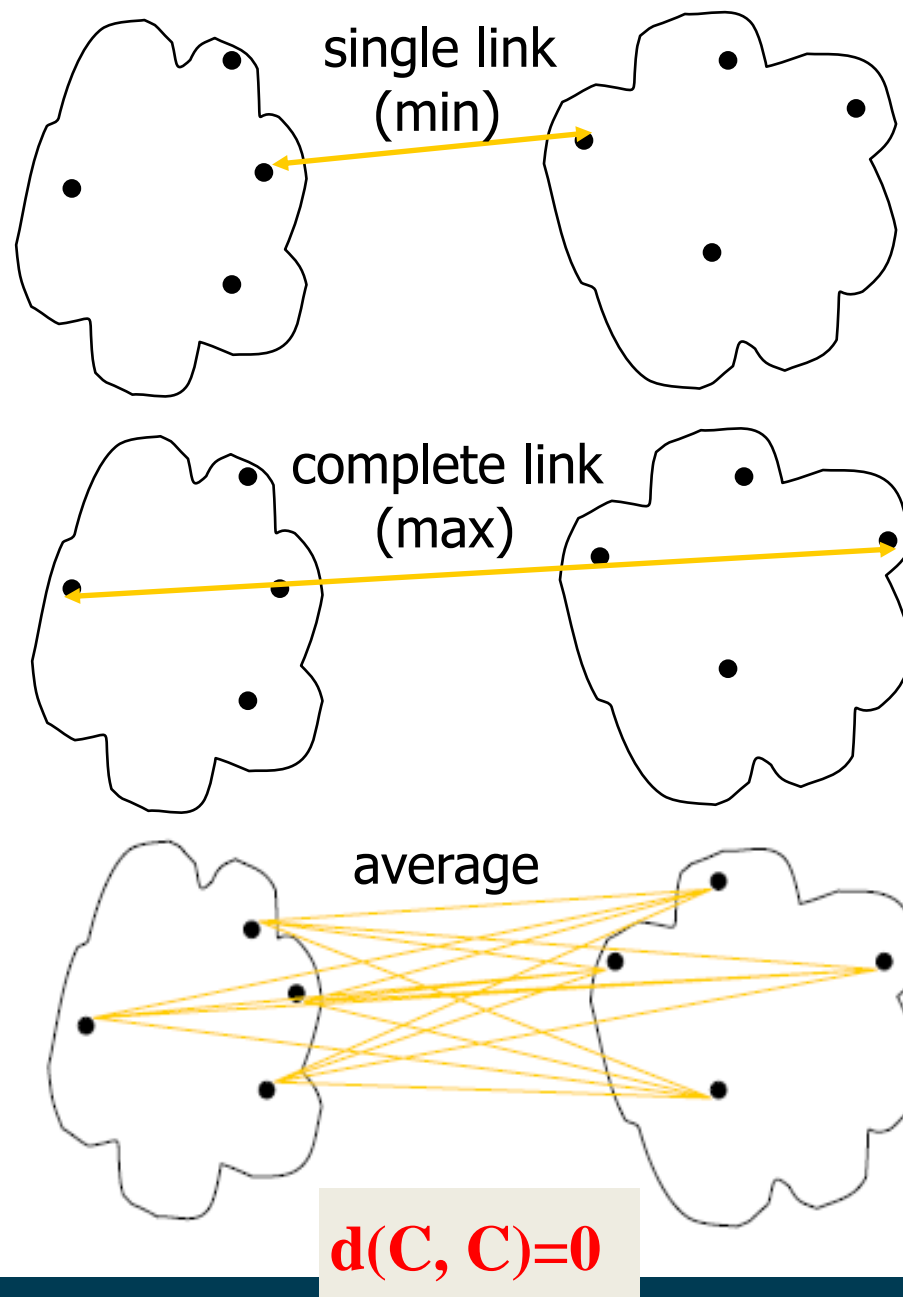
Different approaches are used:

1. Single Link
2. Complete Link
3. Group Average

**Single link:** smallest distance between an element in one cluster and an element in the other, i.e.,  $d(C_i, C_j) = \min\{d(x_{ip}, x_{jq})\}$

**Complete link:** largest distance between an element in one cluster and an element in the other, i.e.,  $d(C_i, C_j) = \max\{d(x_{ip}, x_{jq})\}$

**Average:** avg distance between elements in one cluster and elements in the other, i.e.,  $d(C_i, C_j) = \text{avg}\{d(x_{ip}, x_{jq})\}$



**Example:** Given a data set of five objects characterized by a single continuous feature, assume that there are two clusters:  $C_1: \{a, b\}$  and  $C_2: \{c, d, e\}$ .

	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
Feature	1	2	4	5	6

1. Calculate the distance matrix.
2. Calculate three cluster distances between  $C_1$  and  $C_2$ .

	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
<b>a</b>	0	1	3	4	5
<b>b</b>	1	0	2	3	4
<b>c</b>	3	2	0	1	2
<b>d</b>	4	3	1	0	1
<b>e</b>	5	4	2	1	0

**Single link**

$$\begin{aligned} \text{dist}(C_1, C_2) &= \min\{d(a, c), d(a, d), d(a, e), d(b, c), d(b, d), d(b, e)\} \\ &= \min\{3, 4, 5, 2, 3, 4\} = 2 \end{aligned}$$

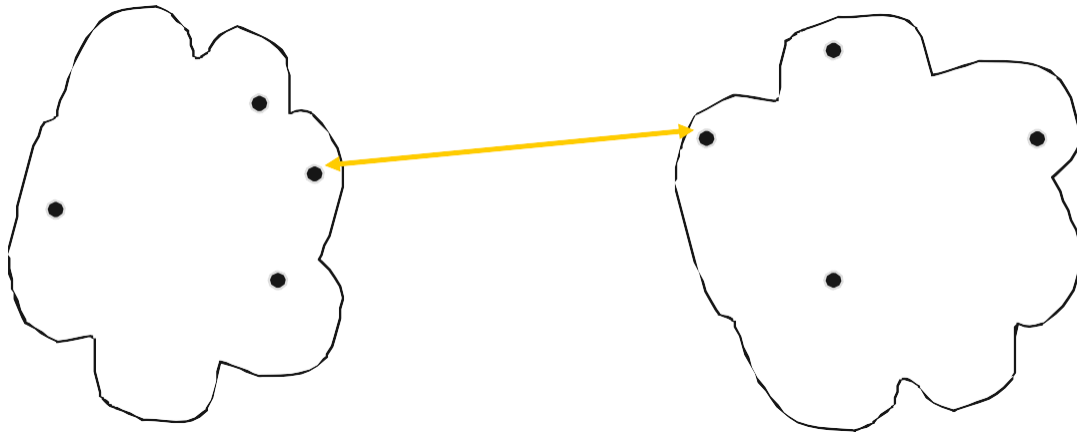
**Complete link**

$$\begin{aligned} \text{dist}(C_1, C_2) &= \max\{d(a, c), d(a, d), d(a, e), d(b, c), d(b, d), d(b, e)\} \\ &= \max\{3, 4, 5, 2, 3, 4\} = 5 \end{aligned}$$

**Average**

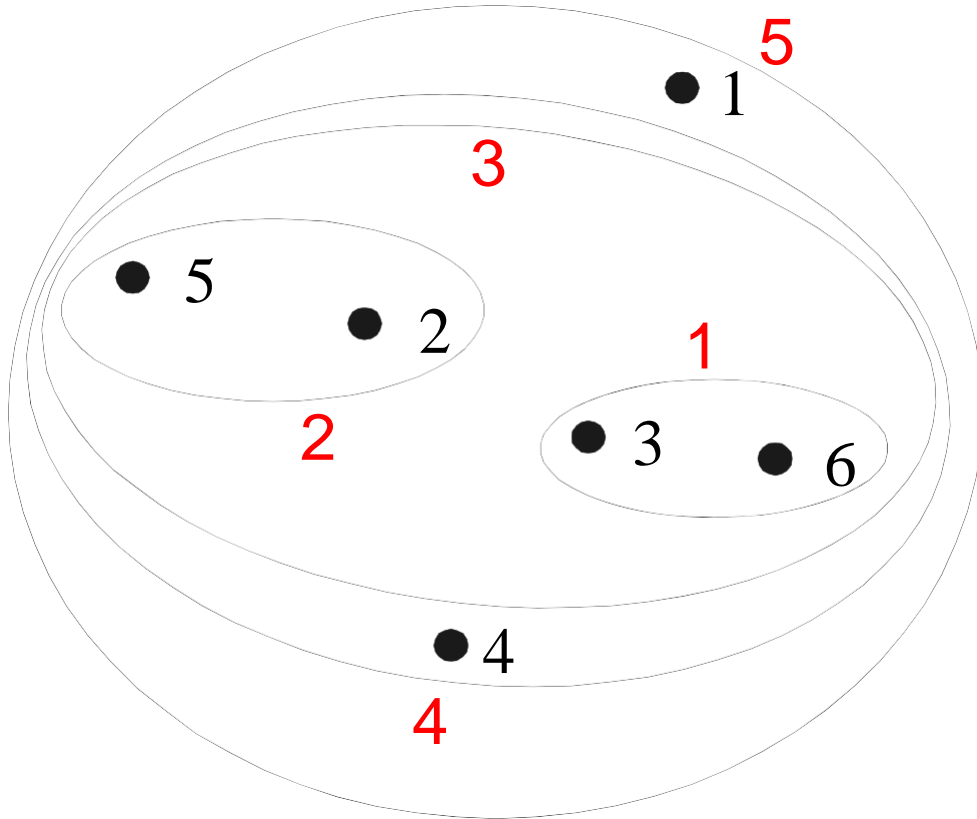
$$\begin{aligned} \text{dist}(C_1, C_2) &= \frac{d(a, c) + d(a, d) + d(a, e) + d(b, c) + d(b, d) + d(b, e)}{6} \\ &= \frac{3 + 4 + 5 + 2 + 3 + 4}{6} = \frac{21}{6} = 3.5 \end{aligned}$$

# Cluster Similarity: Single Link

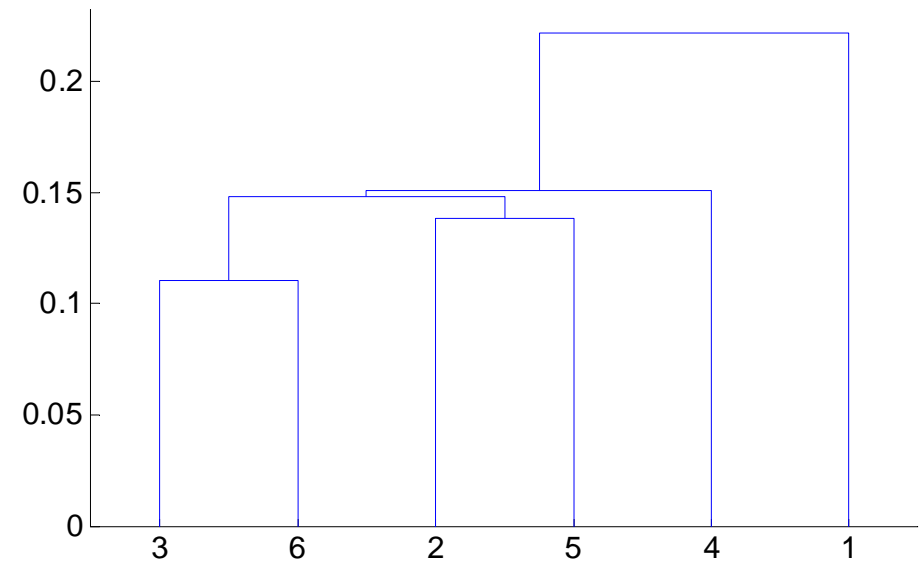


- Similarity of two clusters is based on the **two most similar (closest) points** in the different clusters
- Determined by one pair of points, i.e. by one link in the proximity graph

# Example: Single Link

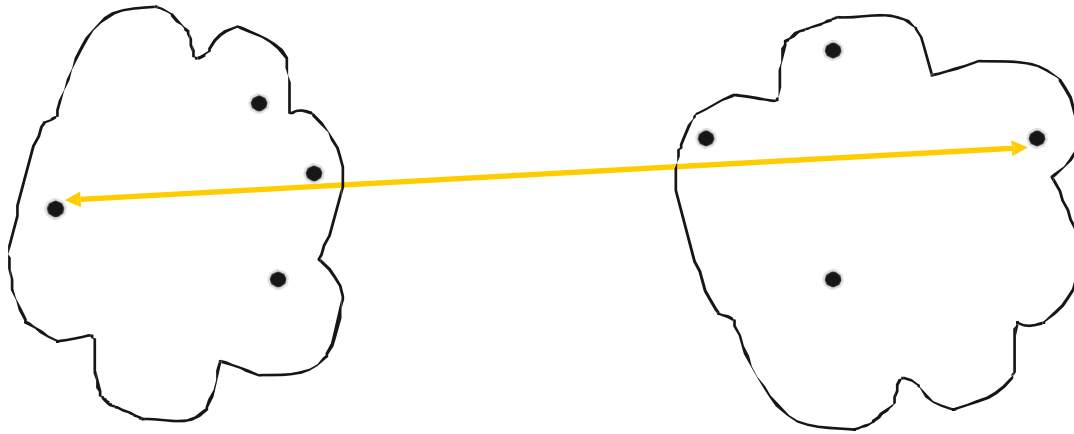


Nested Clusters



Dendrogram

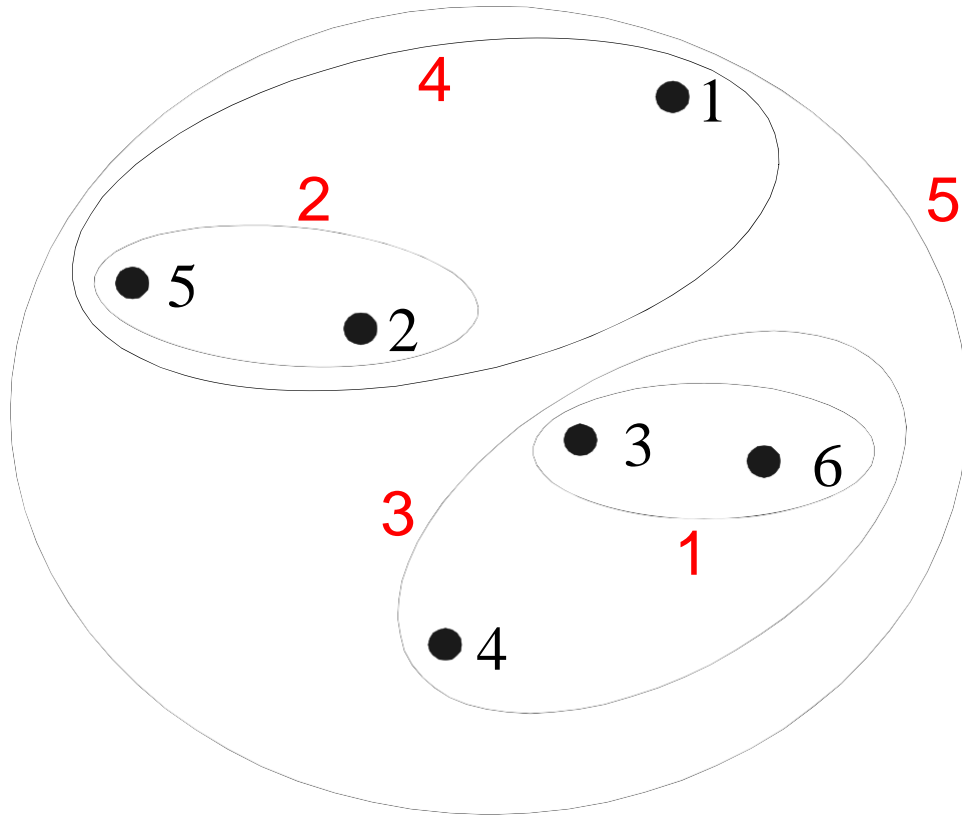
# Cluster Similarity: Complete Linkage



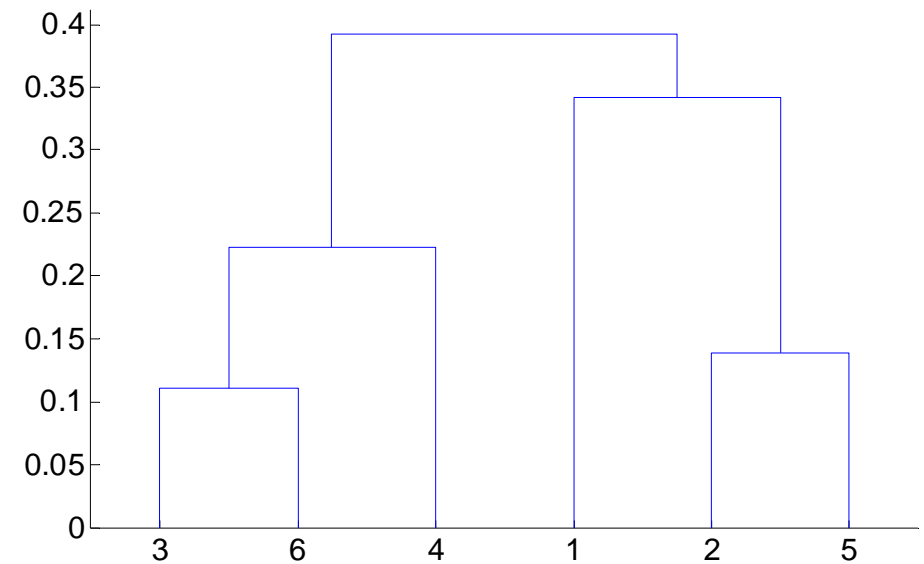
- Similarity of two clusters is based on the **two least similar (most distant) points** in the different clusters
- Determined by all pairs of points in the two clusters



# Example: Complete Linkage

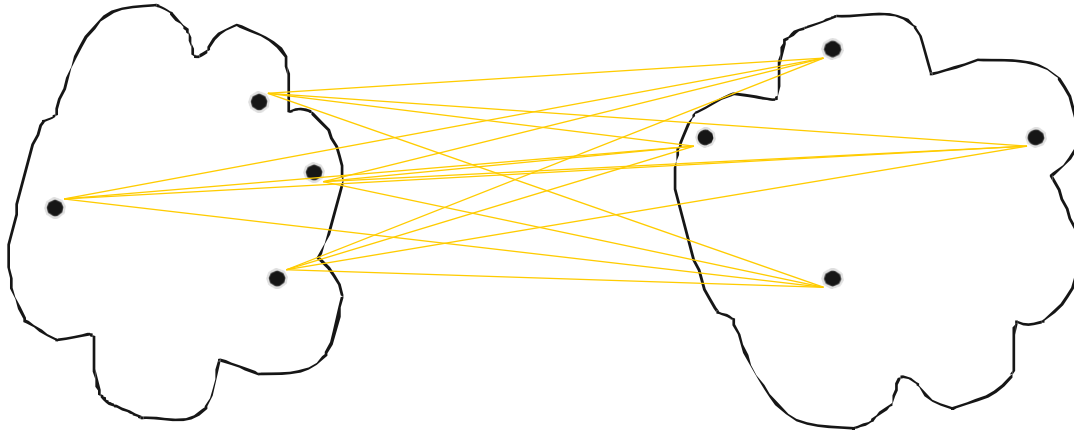


Nested Clusters



Dendrogram

# Cluster Similarity: Group Average

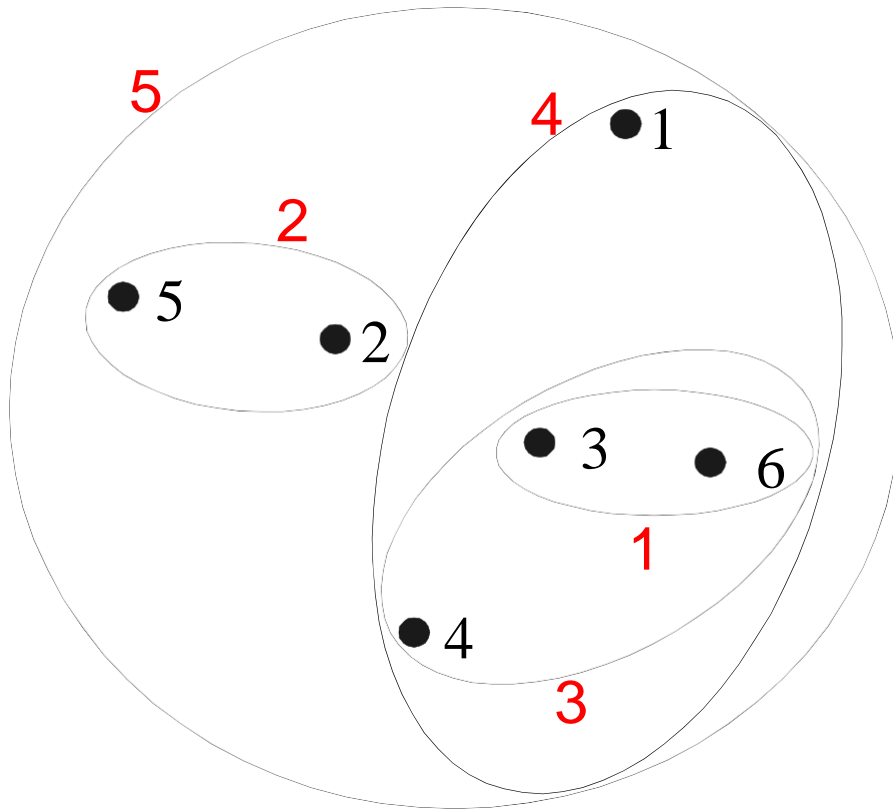


- Proximity of two clusters is the average of pair-wise proximity between all points in the two clusters.

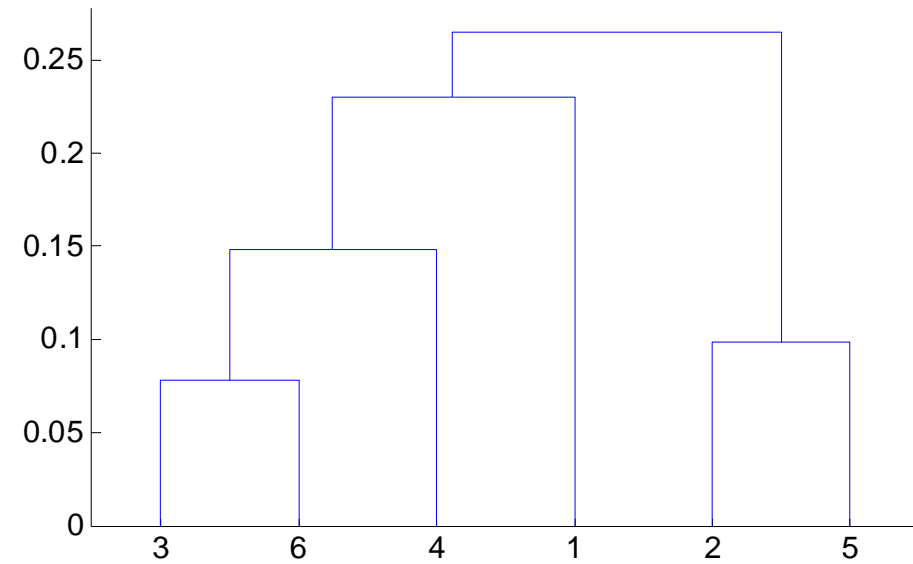
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

- Compromise between single and complete link  
Strength: Less sensitive to noise and outliers than single link

# Example: Group Average



Nested Clusters

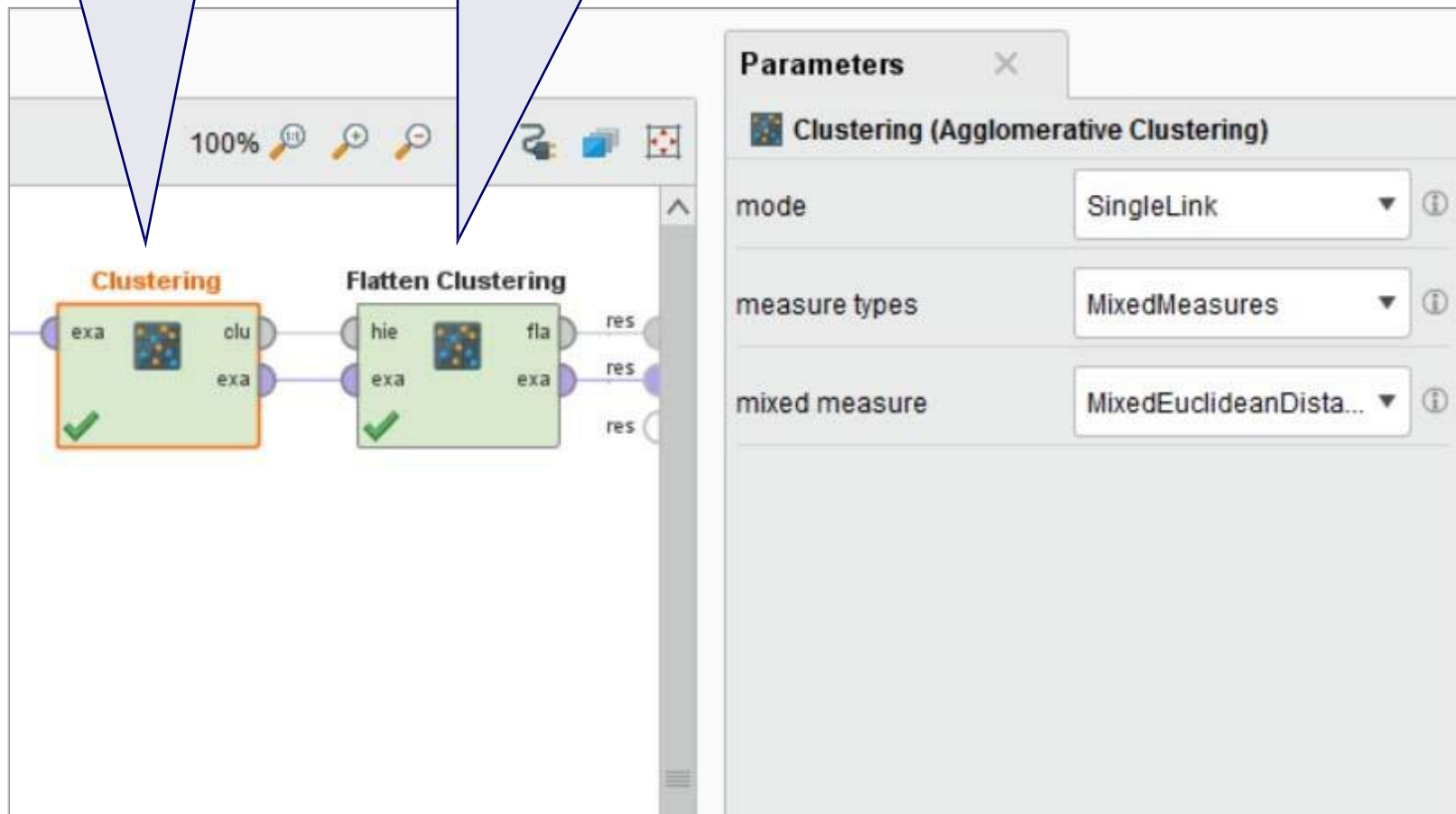


Dendrogram

# Agglomerative Hierarchical Clustering in RapidMiner

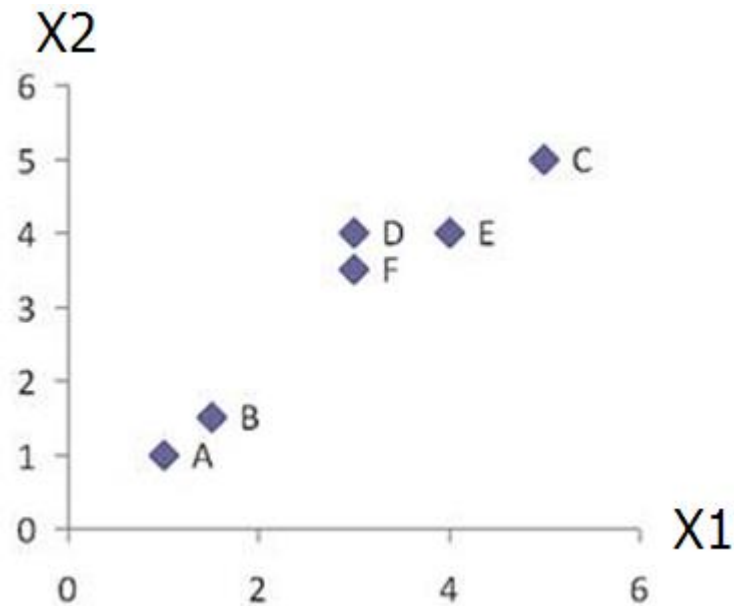
**Creates  
hierarchical clustering**

**Flattens clustering to a given  
number of clusters**



# Example

Problem: clustering analysis with agglomerative algorithm



$$d_{AB} = \left( (1-1.5)^2 + (1-1.5)^2 \right)^{\frac{1}{2}} = \sqrt{\frac{1}{2}} = 0.7071$$

$$d_{DF} = \left( (3-3)^2 + (4-3.5)^2 \right)^{\frac{1}{2}} = 0.5$$

Euclidean distance

	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

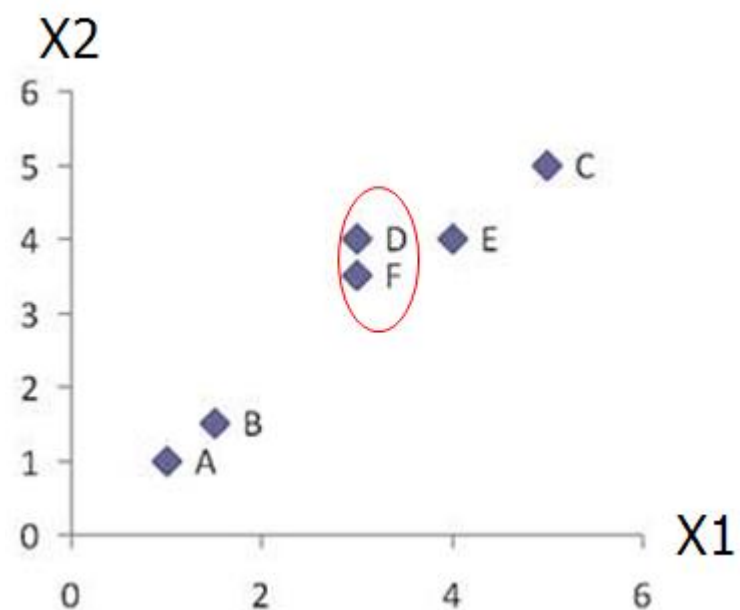
data matrix

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

distance matrix

# Example

Merge two closest clusters (iteration 1)



Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

# Example

Update distance matrix (iteration 1)

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

$$d_{(D,F) \rightarrow A} = \min(d_{DA}, d_{FA}) = \min(3.61, 3.20) = 3.20$$

$$d_{(D,F) \rightarrow B} = \min(d_{DB}, d_{FB}) = \min(2.92, 2.50) = 2.50$$

$$d_{(D,F) \rightarrow C} = \min(d_{DC}, d_{FC}) = \min(2.24, 2.50) = 2.24$$

$$d_{E \rightarrow (D,F)} = \min(d_{ED}, d_{EF}) = \min(1.00, 1.12) = 1.00$$

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

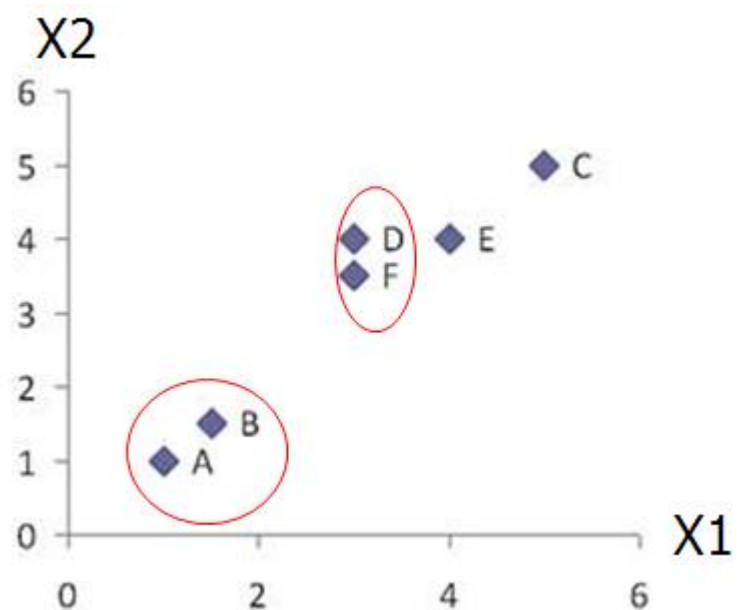
Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	3.20	2.50	2.24	0.00	1.00
E	4.24	3.54	1.41	1.00	0.00



# Example

Merge two closest clusters (iteration 2)



Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	3.20	2.50	2.24	0.00	1.00
E	4.24	3.54	1.41	1.00	0.00

Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0



# Example

Update distance matrix (iteration 2)

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	3.20	2.50	2.24	0.00	1.00
E	4.24	3.54	1.41	1.00	0.00

$$d_{C \rightarrow (A,B)} = \min(d_{CA}, d_{CB}) = \min(5.66, 4.95) = 4.95$$

$$d_{(D,F) \rightarrow (A,B)} = \min(d_{DA}, d_{DB}, d_{FA}, d_{FB}) = \min(3.61, 2.92, 3.20, 2.50) = 2.50$$

$$d_{E \rightarrow (A,B)} = \min(d_{EA}, d_{EB}) = \min(4.24, 3.54) = 3.54$$

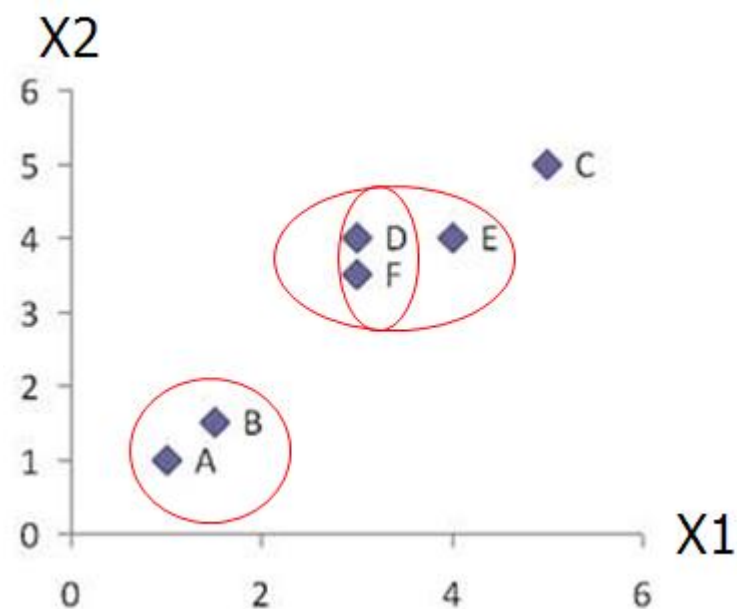
Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	4.95	0	2.24	1.41
(D, F)	2.50	2.24	0	1.00
E	3.54	1.41	1.00	0

# Example

Merge two closest clusters/update distance matrix (iteration 3)



Min Distance (Single Linkage)

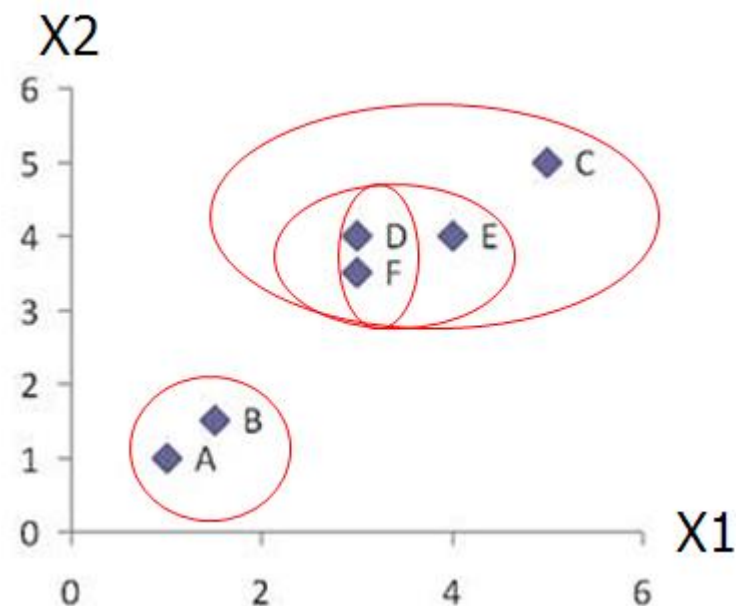
Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	4.95	0	2.24	1.41
(D, F)	2.50	2.24	0	1.00
E	3.54	1.41	1.00	0

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	2.50	1.41	0.00

# Example

Merge two closest clusters/update distance matrix (iteration 4)



## Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	2.50	1.41	0.00

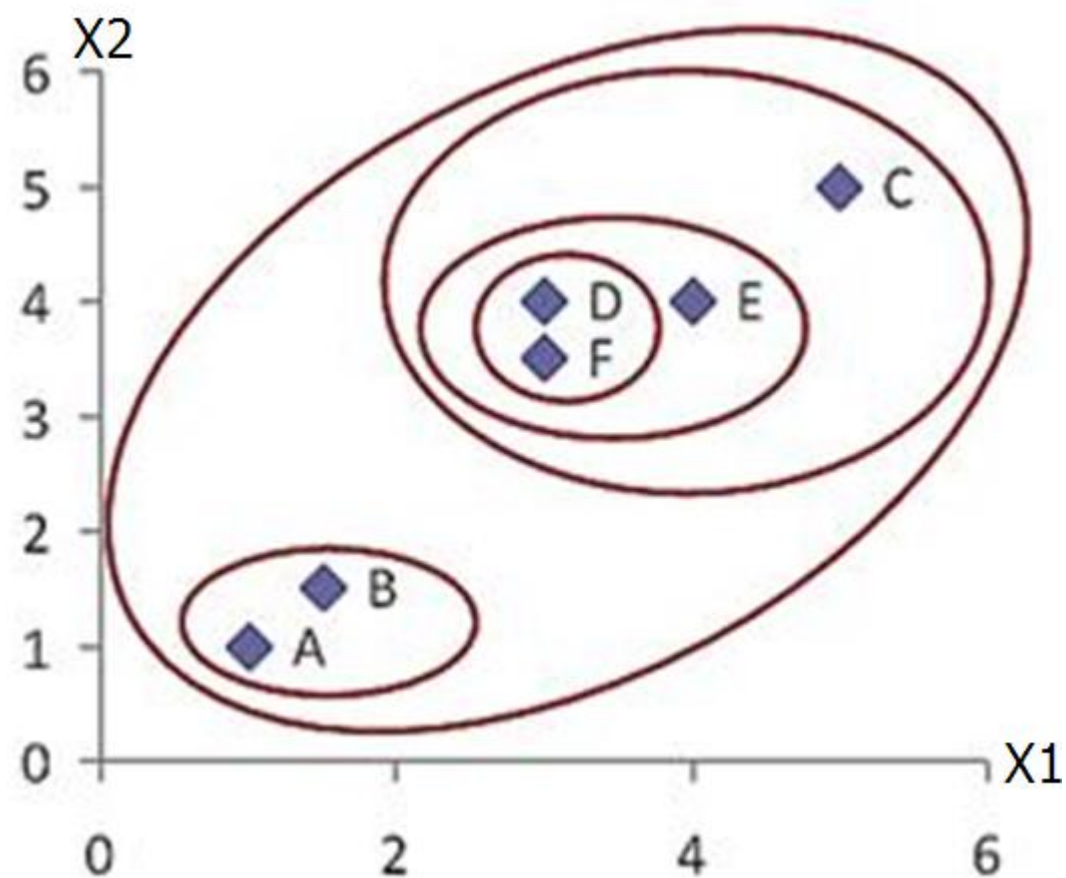
## Min Distance (Single Linkage)

Dist	(A,B)	((D, F), E), C
(A,B)	0.00	2.50
((D, F), E), C	2.50	0.00

# Example

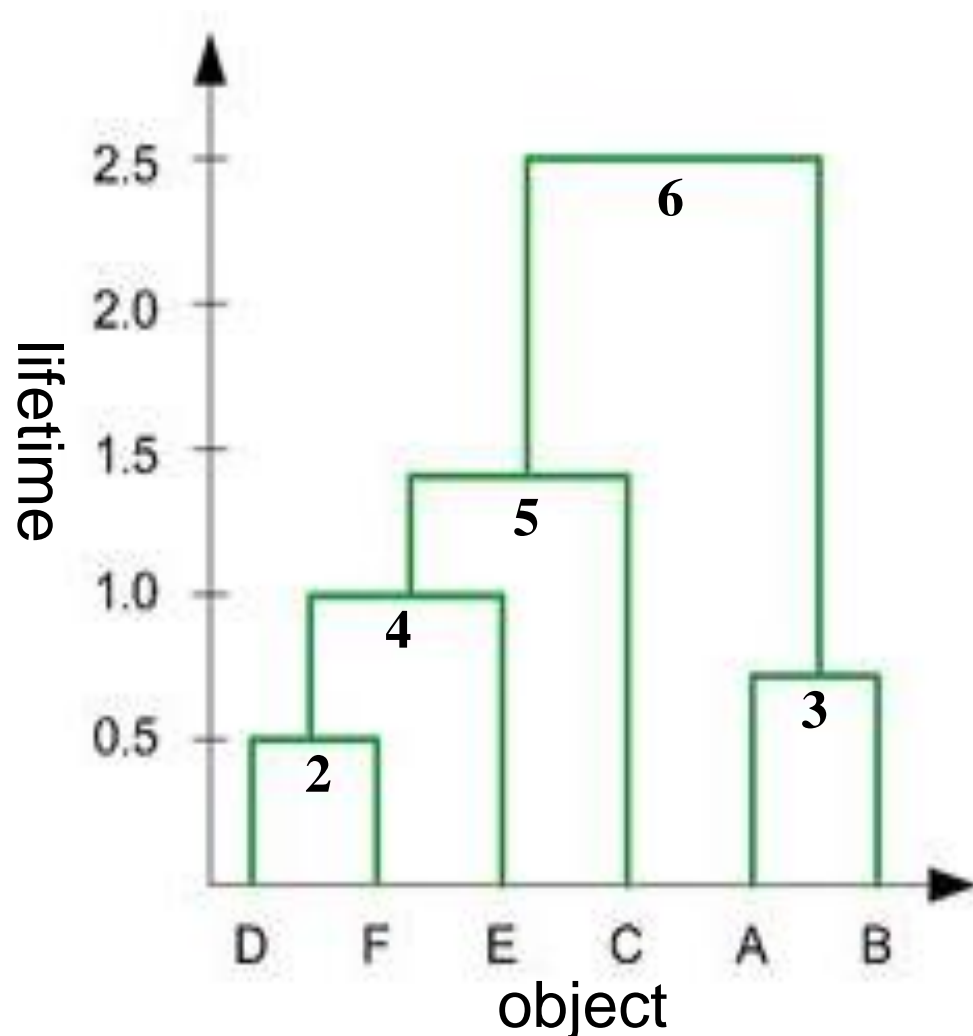
Final result (meeting termination condition)

	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5



# Example

## Dendrogram tree representation



1. In the beginning we have 6 clusters: A, B, C, D, E and F
2. We merge clusters D and F into cluster (D, F) at distance 0.50
3. We merge cluster A and cluster B into (A, B) at distance 0.71
4. We merge clusters E and (D, F) into ((D, F), E) at distance 1.00
5. We merge clusters ((D, F), E) and C into (((D, F), E), C) at distance 1.41
6. We merge clusters (((D, F), E), C) and (A, B) into ((((D, F), E), C), (A, B)) at distance 2.50
7. The last cluster contain all the objects, thus conclude the computation



Given a data set of five objects characterised by a single continuous feature:

	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
Feature	1	2	4	5	6

Apply the agglomerative algorithm with single-link, complete-link and averaging cluster distance measures to produce three dendrogram trees, respectively.

	<b>a</b>	<b>b</b>	<b>c</b>	<b>d</b>	<b>e</b>
<b>a</b>	0	1	3	4	5
<b>b</b>	1	0	2	3	4
<b>c</b>	3	2	0	1	2
<b>d</b>	4	3	1	0	1
<b>e</b>	5	4	2	1	0

# Hierarchical clustering: example

	BA	FI	MI	NA	RM	TO
BA	0	662	877	255	412	996
FI	662	0	295	468	268	400
MI	877	295	0	754	564	138
NA	255	468	754	0	219	869
RM	412	268	564	219	0	669
TO	996	400	138	869	669	0



# Hierarchical clustering: example using single linkage

