



**POLITECNICO**  
MILANO 1863

*Credits for the material in these slides go to:*

*Prof. Pier Luca Lanzi lectures on  
DATA MINING*

*Prof. Salvatore Orlando lectures on  
DATA AND WEB MINING*

# Clustering Algorithms

Matteo Matteucci, PhD ([matteo.matteucci@polimi.it](mailto:matteo.matteucci@polimi.it))

*Artificial Intelligence and Robotics Laboratory  
Politecnico di Milano*

**AIRLAB**  
ARTIFICIAL INTELLIGENCE AND ROBOTICS LAB

# Machine Learning Paradigms

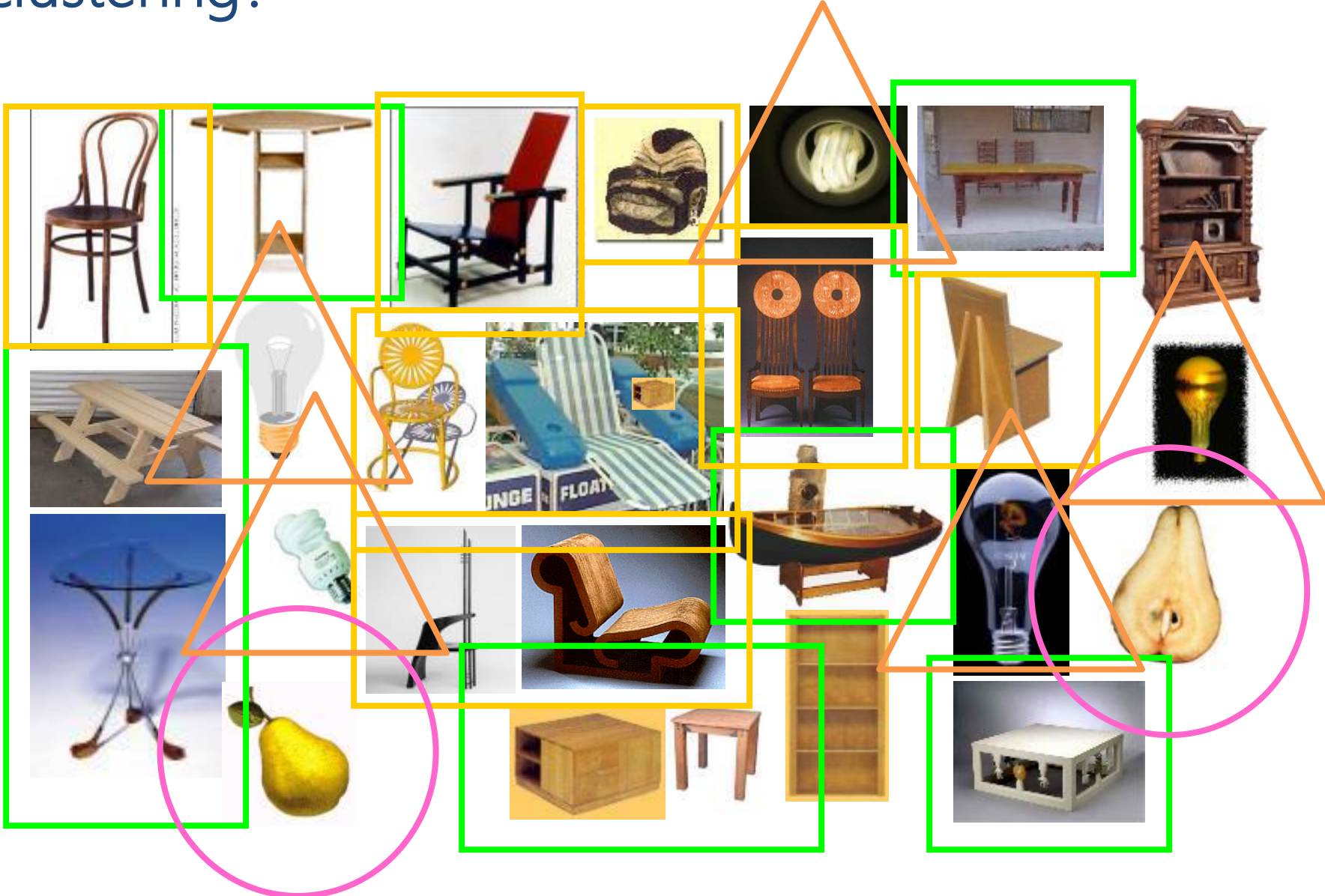
Imagine you have a certain experience  $E$ , i.e., data, and let's name it

$$D = x_1, x_2, x_3, \dots, x_N$$

- **Supervised learning**: given the desired outputs  $t_1, t_2, t_3, \dots, t_N$  learn to produce the correct output given a new set of input
- **Unsupervised learning**: exploit regularities in  $D$  to build a representation to be used for reasoning or prediction
- **Reinforcement learning**: producing actions  $a_1, a_2, a_3, \dots, a_N$  which affect the environment, and receiving rewards  $r_1, r_2, r_3, \dots, r_N$  learn to act in order to maximize rewards in the long term

*We will focus on  
Clustering ...*

# What is clustering?



*"The process of organizing objects into groups whose members are similar in some way"*

J.A. Hartigan, 1975

*"An algorithm by which objects are grouped in classes, so that intra-class similarity is maximized and inter-class similarity is minimized"*

J. Han and M. Kamber, 2000

*"... grouping or segmenting a collection of objects into subsets or clusters, such that those within each cluster are more closely related to one another than objects assigned to different clusters"*

T. Hastie, R. Tibshirani, J. Friedman, 2009







# Brief insights about Clustering

Clustering is an *unsupervised learning* algorithm

- “Exploit regularities in the inputs to build a representation that can be used for reasoning or prediction”

Among the possible unsupervised models it focuses on:

- Groups/Classes (vs outliers)
- Distance/Similarity



*This all is embedded in the distance or similarity used*

What clustering is useful for:

- Data reduction (representatives for homogeneous groups)
- Natural data types (unknown properties of “natural” clusters)
- Useful data classes (useful and suitable groupings)
- Outlier detection (unusual data objects)



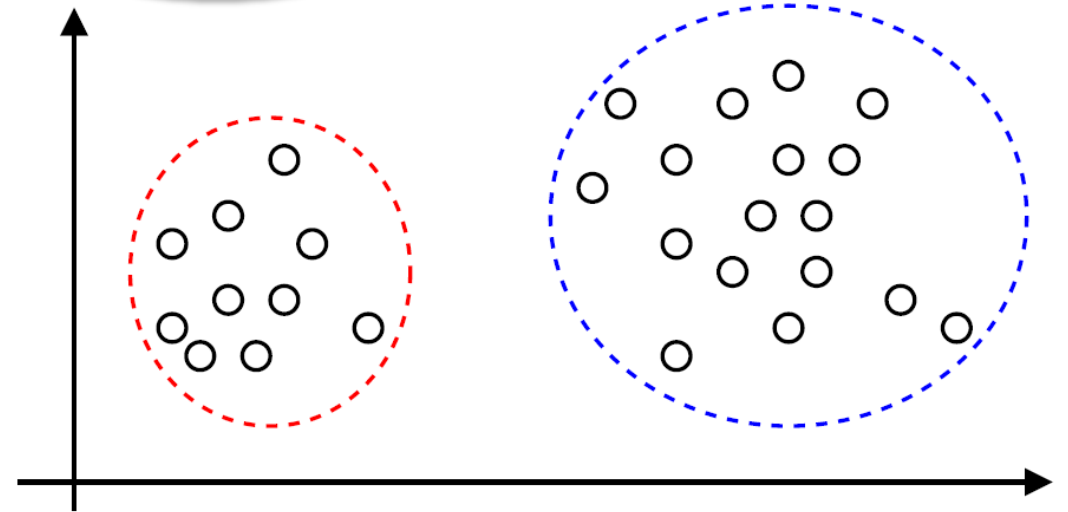
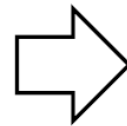
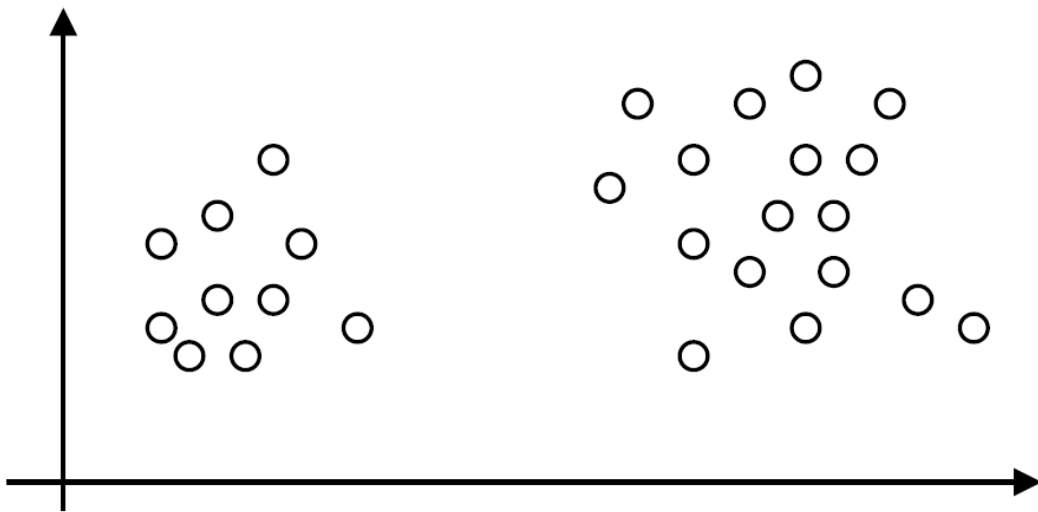
# Clustering ... what are we looking for?

**Cluster:** a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters

A good clustering method will produce high **quality clusters** with

- High intra-class similarity
- Low inter-class similarity

*If (dis)similarity criterion is distance we have distance-based clustering.*



# Clustering ... what are we looking for?

**Cluster:** a collection of objects which are “similar” between them and are “dissimilar” to the objects belonging to other clusters

A good clustering method will produce high *quality clusters* with

- High *quality*  
Cosine similarity,  
Pearson correlation,  
Levenstain distance ...

*If (dis)similarity criterion is distance we have distance-based clustering.*

*Attributes might be weighted too ...*

The clustering result depends on a similarity measure.

- One numeric attribute A -> Distance(X,Y) =  $A(X) - A(Y)$
- Several numeric attributes -> Distance(X,Y) = Euclidean distance  $\|X - Y\|_2$
- Nominal attributes -> Hamming distance (X,Y) = if different 1, else 0
- ...

# What about Euclidean distance?

The goal of clustering is to determine the intrinsic grouping in a set of **unlabeled** data. But there is no absolute “best” criterion which would be independent of the final aim of the clustering. It is the user which must supply the criterion through a distance measure.

With numerical data, in higher dimensions, we could also use a distance measure called Minkowski Metric:

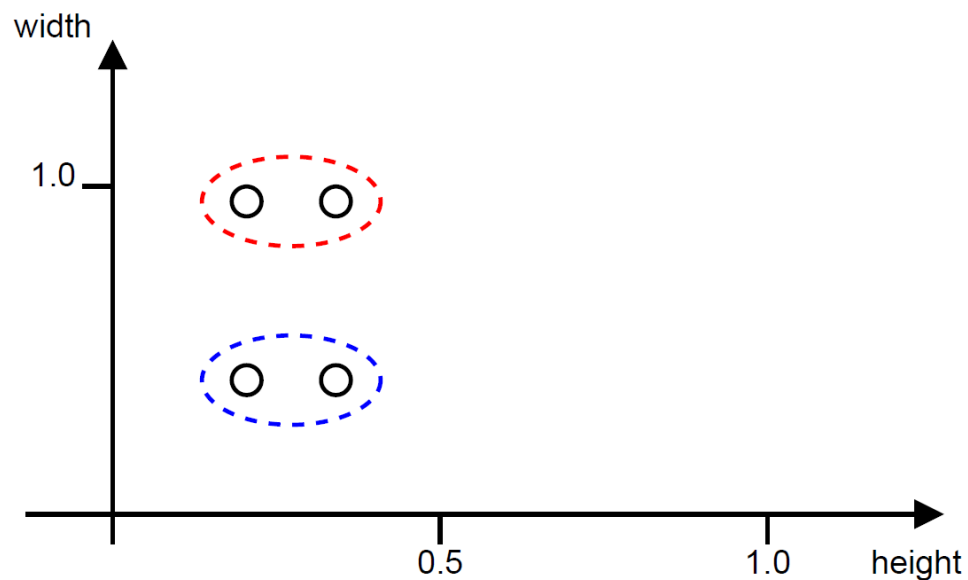
$$d_p(x_i, x_j) = \left( \sum_{l=1}^L |x_i^{(l)} - x_j^{(l)}|^p \right)^{1/p}$$

**Euclidean distance** is a special case where  $p = 2$ , the **Manhattan distance** has  $p = 1$ , and the **Max distance** has  $p = \infty$ .

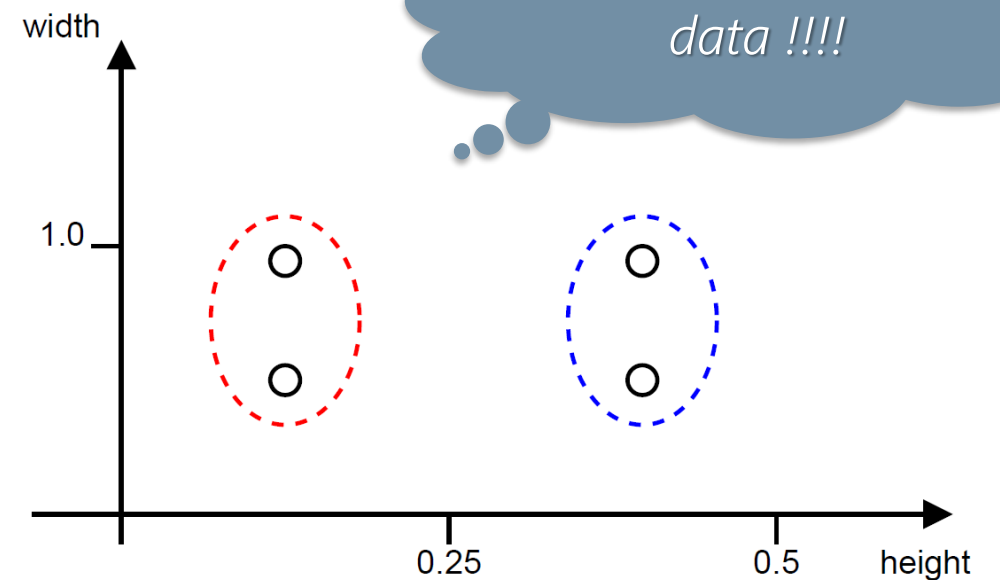
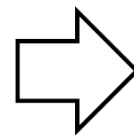
*Very sensitive to scaling ...*

# What about Euclidean distance?

The goal of clustering is to determine the intrinsic grouping in a set of **unlabeled** data. But there is no absolute "best" criterion which would be independent of the final aim of the clustering. It is the user which must supply the criterion through a distance measure.



Before Scaling



After Scaling

*These are the same data !!!!*

# Applications for Clustering

## Biology and natural sciences

- Grouping of plants and animals given their features

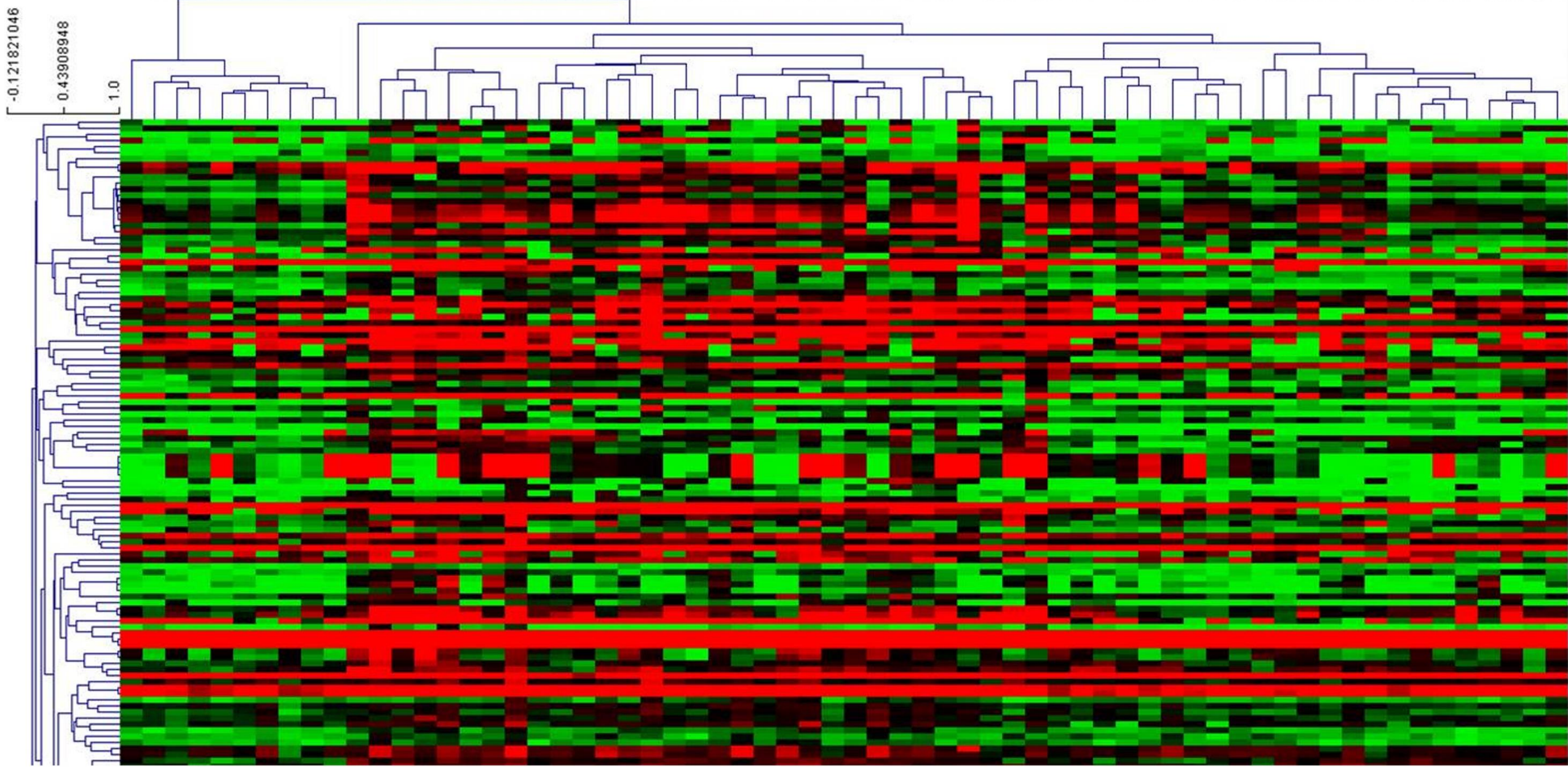
## Market research

- Groups of similar customers for targeted advertising
- Identify frauds by innsurance, telephone companies

## On the Web

- Document classification
- Discover groups of similar access patterns in logs
- Recommendation systems: "If you liked this, you might also like that" (<http://www.gnod.com/>)





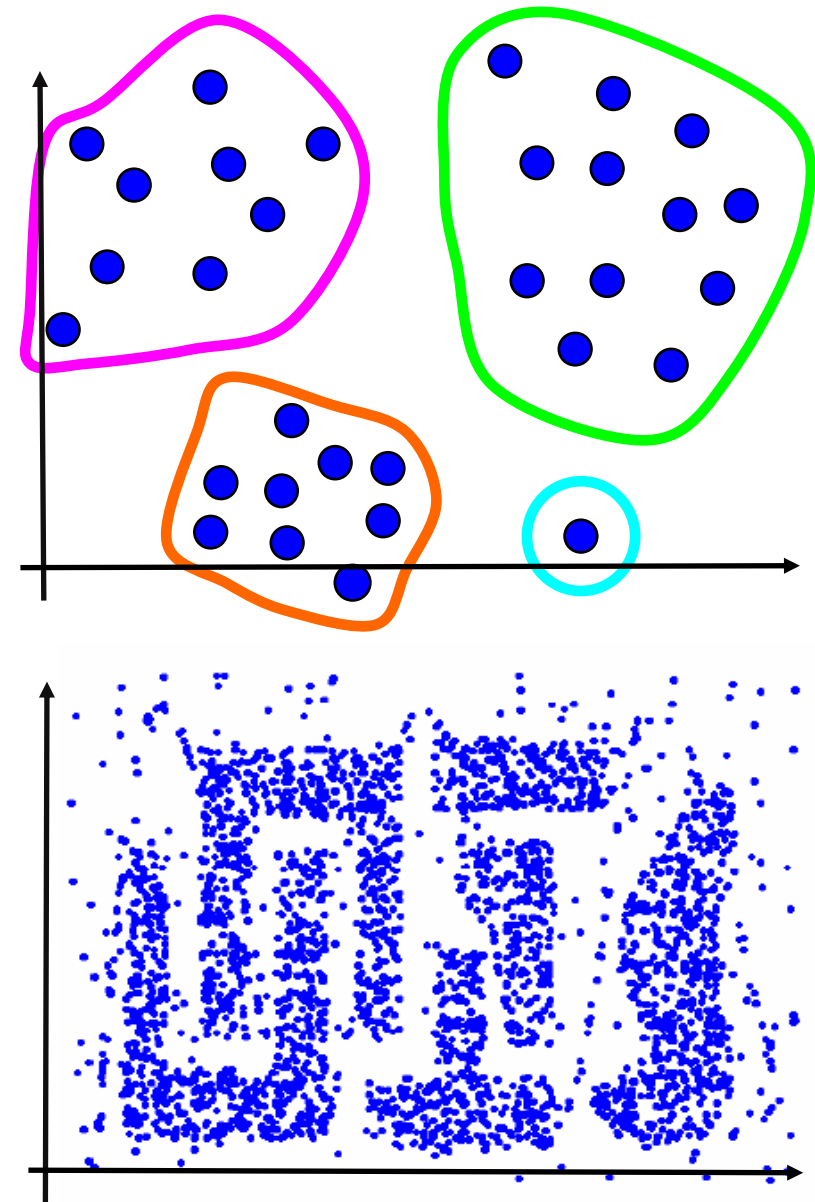
# Cluster/Clustering Characteristics

Different data types:

- Symbols
- Numbers
- ...

Different metrics:

- Euclidean
  - real-valued dimensions
  - "dense" points in some space
  - notion of average of two points
- Non-Euclidean
  - based on properties of points



# A Brief Clustering Taxonomy

Clustering methods can be divided in two main typologies

*Hierarchical Clustering*: division of samples in a hierarchy of clusters

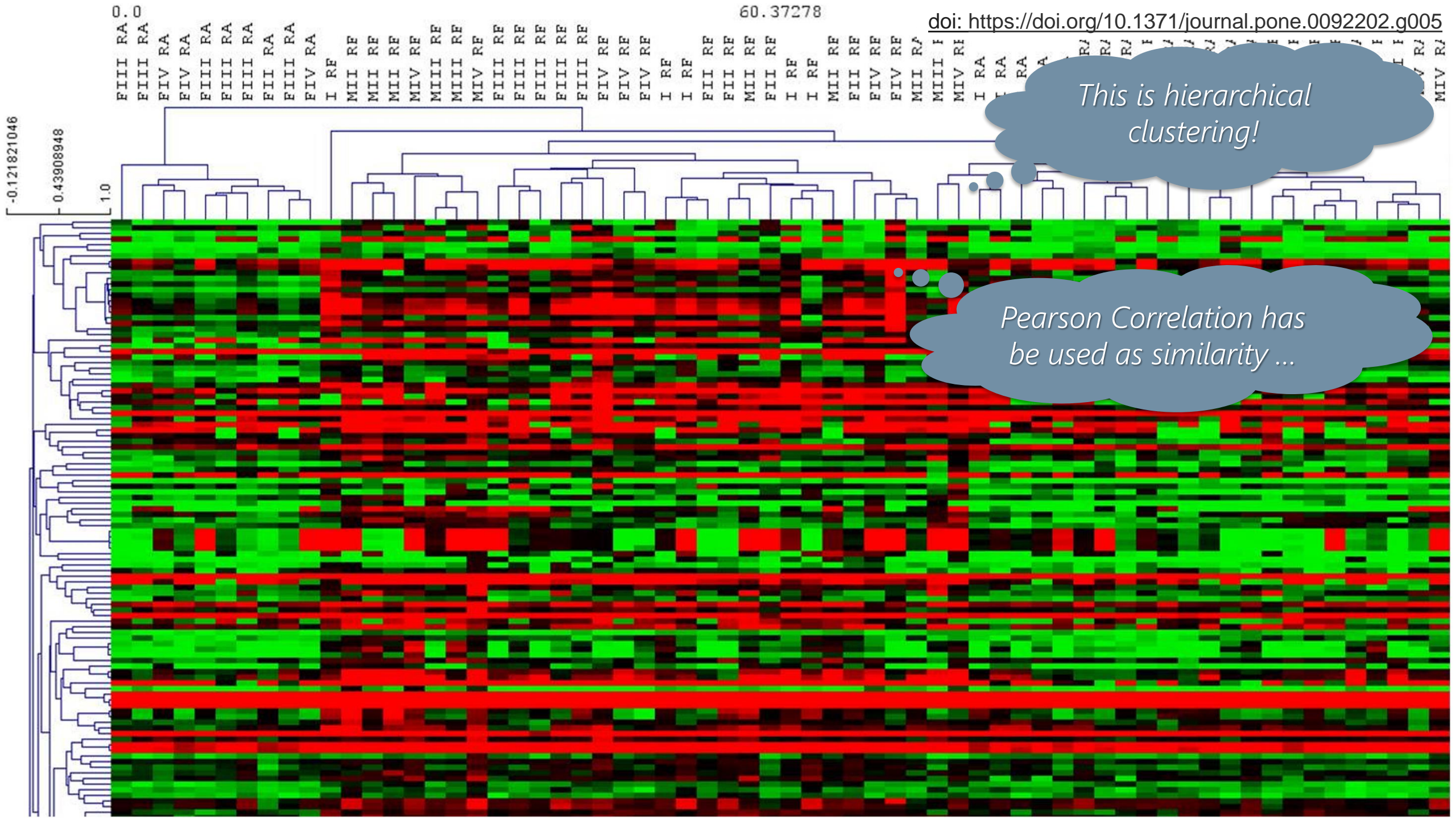
- Bottom-up or agglomerative
- Top-down

*Partitioning Clustering*: division of the data into  $k$  clusters

Resulting Clusters can be:

- *Exclusive Clusters*: examples belong to only one cluster
- *Overlapping Clusters*: examples may belong to more clusters
- *Probabilistic Clusters*: examples belong to one cluster with certain probability



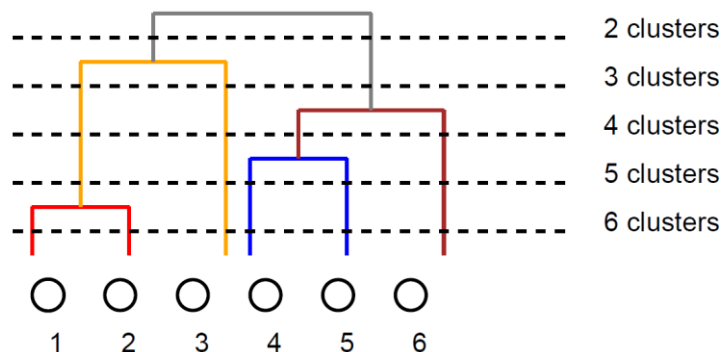
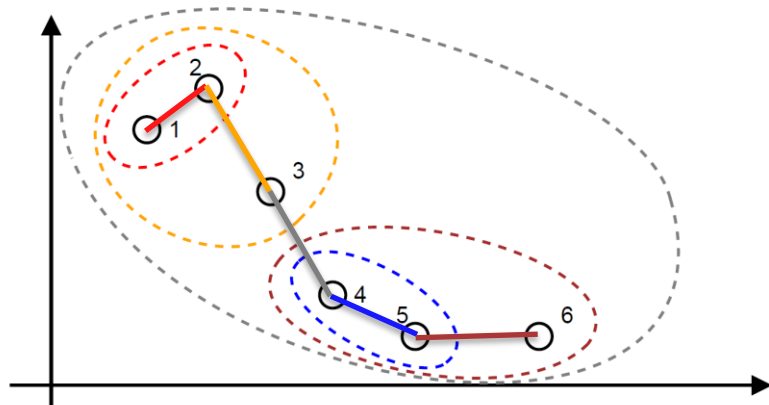


*This is hierarchical clustering!*

*Pearson Correlation has be used as similarity ...*

# Bottom up approach

Given  $N$  items to be clustered, and an  $N \times N$  distance (or similarity) matrix, follow the following algorithm (S.C. Johnson 1967):



---

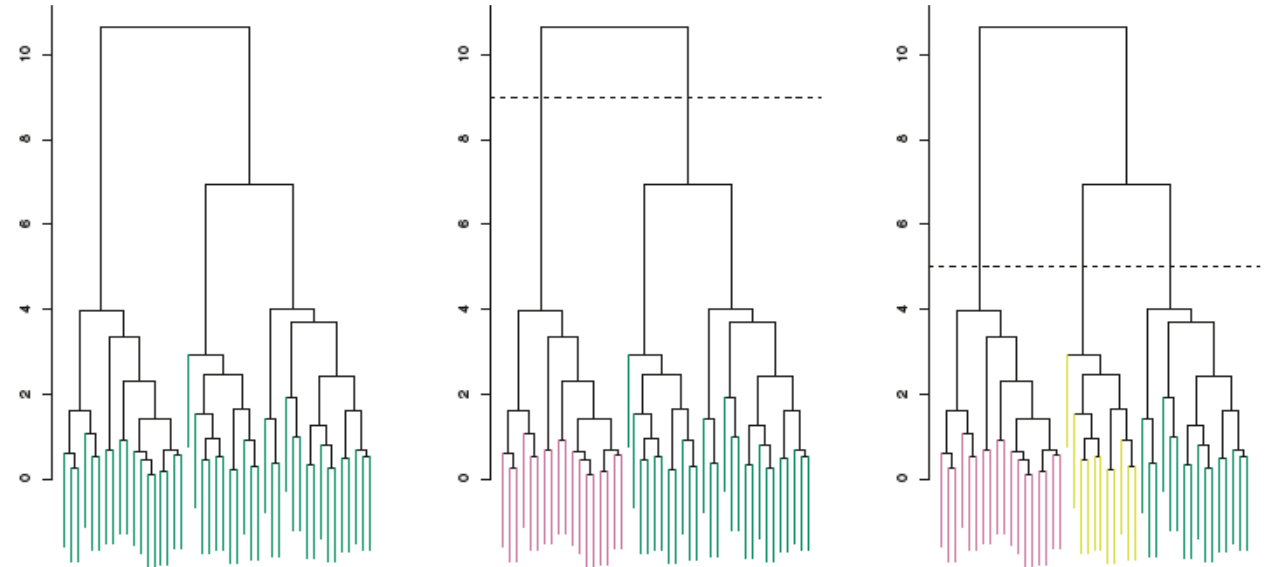
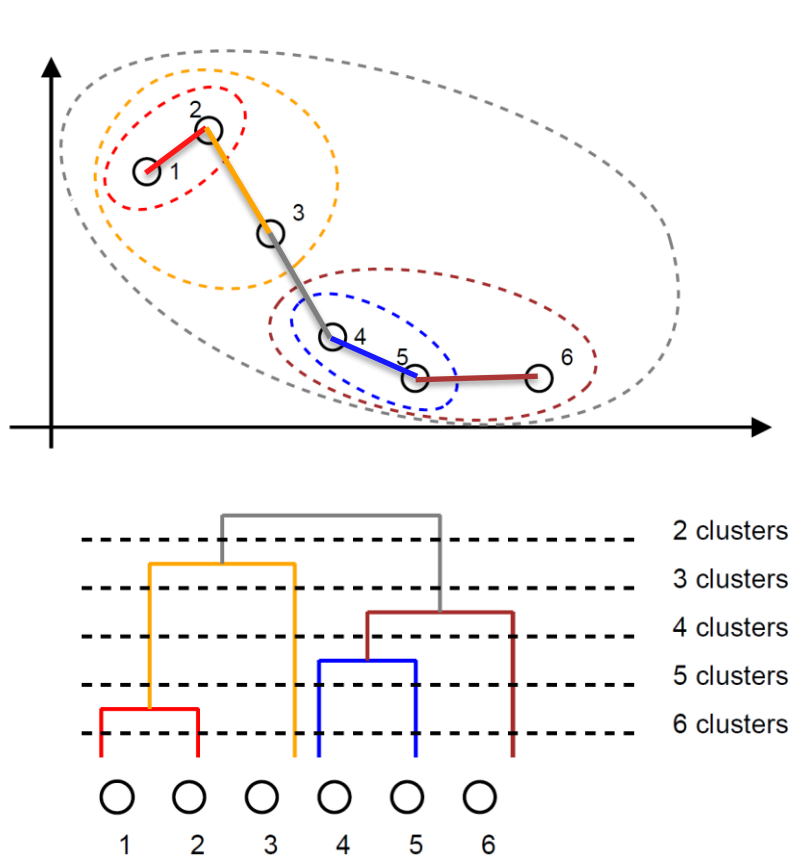
## Algorithm 10.2 Hierarchical Clustering

---

1. Begin with  $n$  observations and a measure (such as Euclidean distance) of all the  $\binom{n}{2} = n(n-1)/2$  pairwise dissimilarities. Treat each observation as its own cluster.
  2. For  $i = n, n-1, \dots, 2$ :
    - (a) Examine all pairwise inter-cluster dissimilarities among the  $i$  clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.
    - (b) Compute the new pairwise inter-cluster dissimilarities among the  $i-1$  remaining clusters.
-

# Bottom up approach

Given  $N$  items to be clustered, and an  $N \times N$  distance (or similarity) matrix, follow the following algorithm (S.C. Johnson 1967):



**FIGURE 10.9.** Left: dendrogram obtained from hierarchically clustering the data from Figure 10.8 with complete linkage and Euclidean distance. Center: the dendrogram from the left-hand panel, cut at a height of nine (indicated by the dashed line). This cut results in two distinct clusters, shown in different colors. Right: the dendrogram from the left-hand panel, now cut at a height of five. This cut results in three distinct clusters, shown in different colors. Note that the colors were not used in clustering, but are simply used for display purposes in this figure.

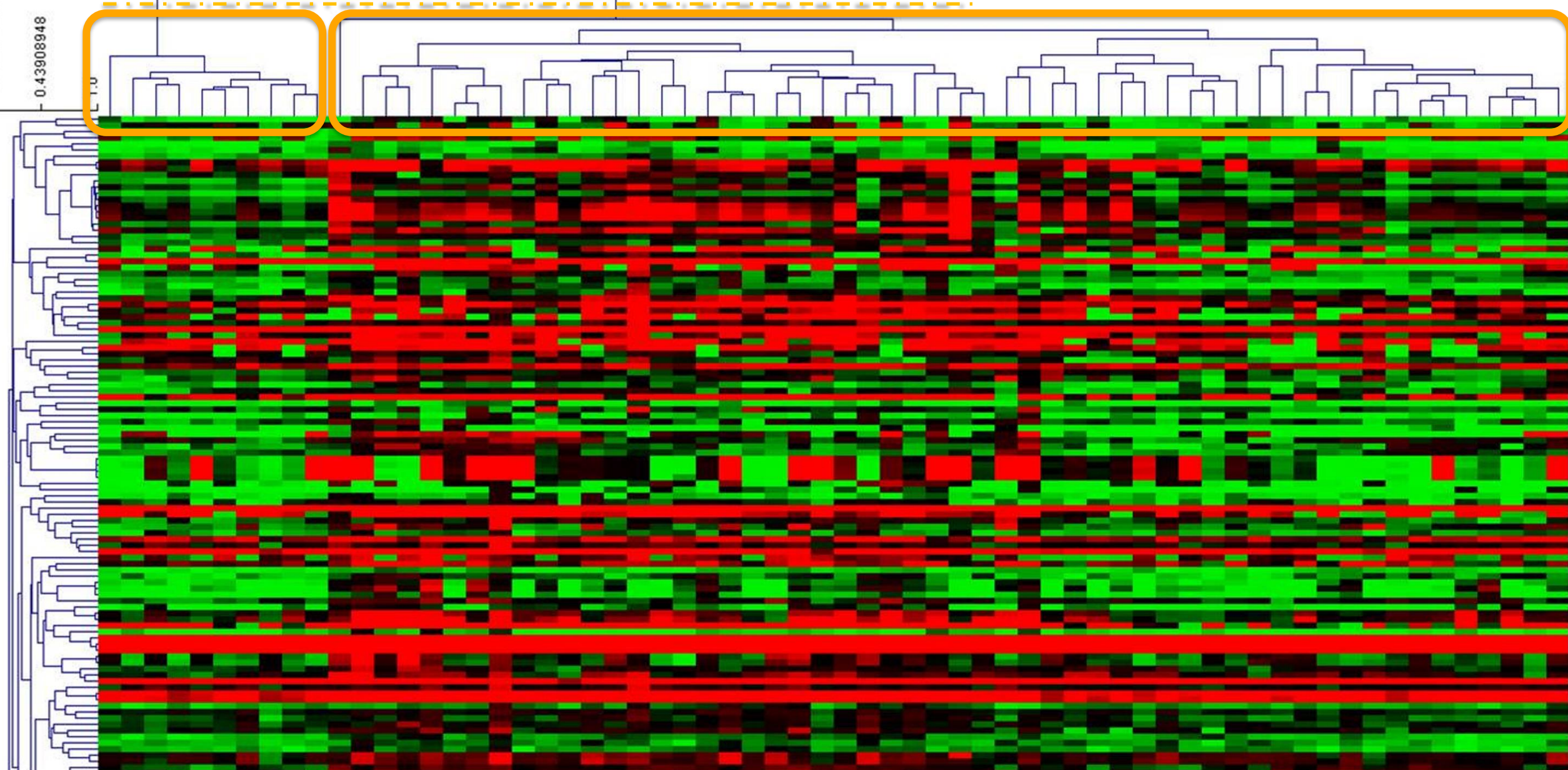
-0.121821046

-0.43908948

0.0  
FIII RA.0  
FIII RA  
FIV RA  
FIV RA  
FIII RA  
FIII RA  
FIII RA  
FII RA  
FIII RA  
FIV RA  
I RE  
MII RE  
MII RE  
MII RE  
MIV RE  
MIII RE  
MIII RE  
MIV RE  
FIII RE  
FIII RE  
FIII RE  
FIII RE  
FIII RE  
FIII RE  
FIV RE  
FIV RE  
FIV RE  
I RE  
I RE  
FII RE  
FII RE  
MII RE  
FII RE  
I RE  
I RE  
MII RE  
FII RE  
FIV RE  
FIV RE  
MII R/  
MIII F  
MIV R/  
I RA  
I RA  
I RA  
I RA  
I RA  
FII R/  
FII R/  
FII R/  
FIII F  
FII R/  
MII R/  
MII R/  
MII R/  
MII R/  
MII R/  
FII R/  
MIII F  
MIII F  
MIII F  
MIV R/  
MIII F  
MIII F  
MIV R/  
MIV R/  
MIV R/

60.37278

doi: <https://doi.org/10.1371/journal.pone.0092202.g005>



-0.121821046

-0.43908948

0.0

FIII RA.0  
FIII RA  
FIV RA  
FIV RA  
FIII RA  
FIII RA  
FIII RA  
FII RA  
FIII RA  
FIV RA

I RE  
MII RE  
MII RE  
MII RE  
MIV RE  
MIII RE  
MIII RE  
MIV RE  
FII RE  
FIII RE  
FIII RE  
FII RE  
FIII RE  
FIV RE  
FIV RE  
FIV RE

I RE  
I RE  
FII RE  
FII RE  
MII RE  
FII RE  
I RE  
I RE

MII RE  
FII RE  
FIV RE  
FIV RE  
MII R/  
MIII F  
MIV R/  
I RA  
I RA  
I RA  
I RA  
I RA

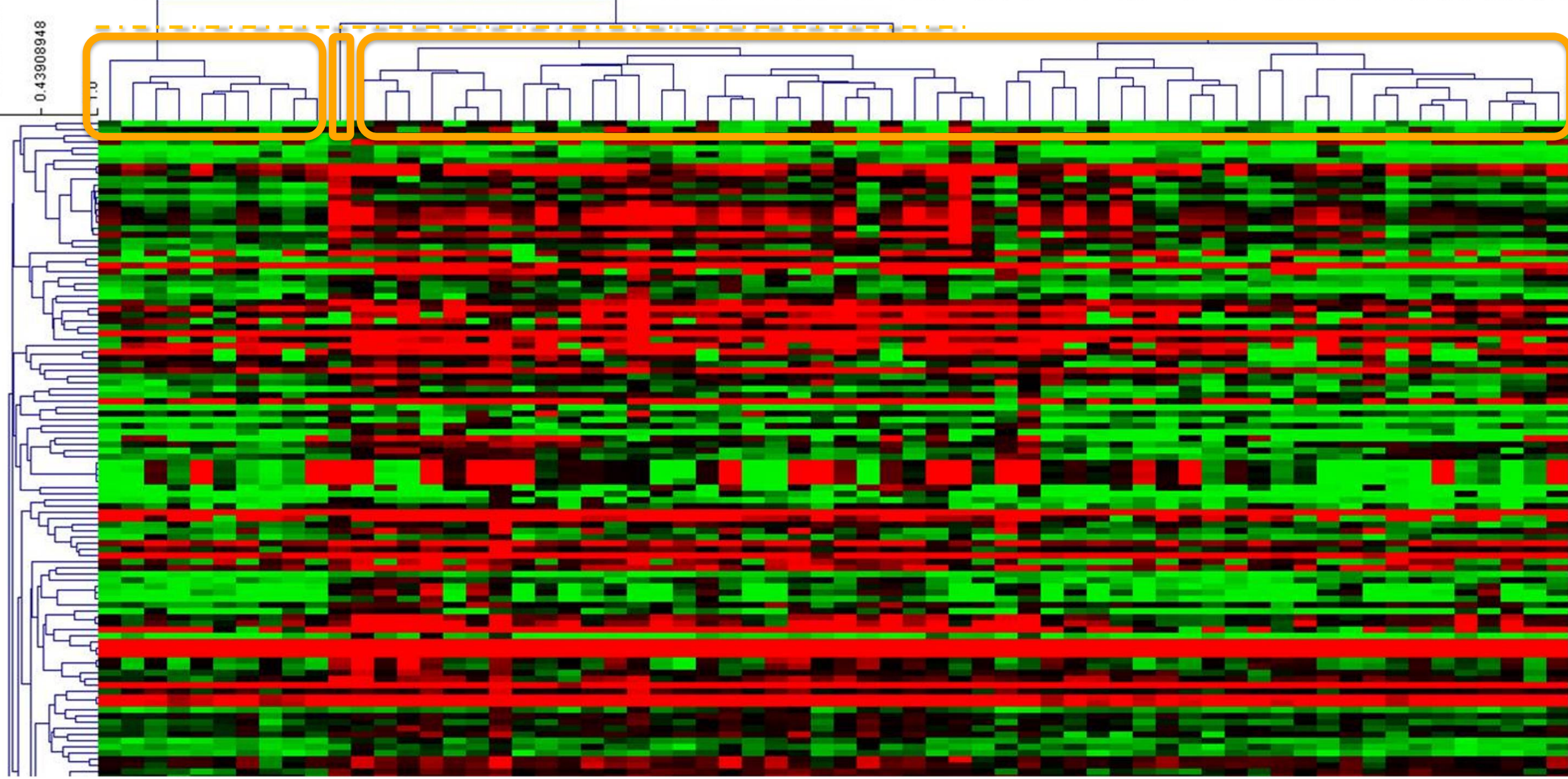
FII R/  
FII R/  
FII R/  
FIII F  
FII R/  
MII R/  
MII R/  
MII R/  
MII R/  
FII R/  
MIII F  
MIII F  
MIII F

MIV R/  
MIII F  
MIII F  
MIII F  
MIV R/  
MIII F  
MIII F  
MIII F

MIV R/  
MIII F  
MIII F  
MIII F  
MIV R/  
MIII F  
MIII F  
MIII F

60.37278

doi: <https://doi.org/10.1371/journal.pone.0092202.g005>



-0.121821046

-0.43908948

FIII RA.0

FIII RA.0

FIV RA

FIV RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

FIII RA

I RE

MII RE

MII RE

MII RE

MII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

MIII RE

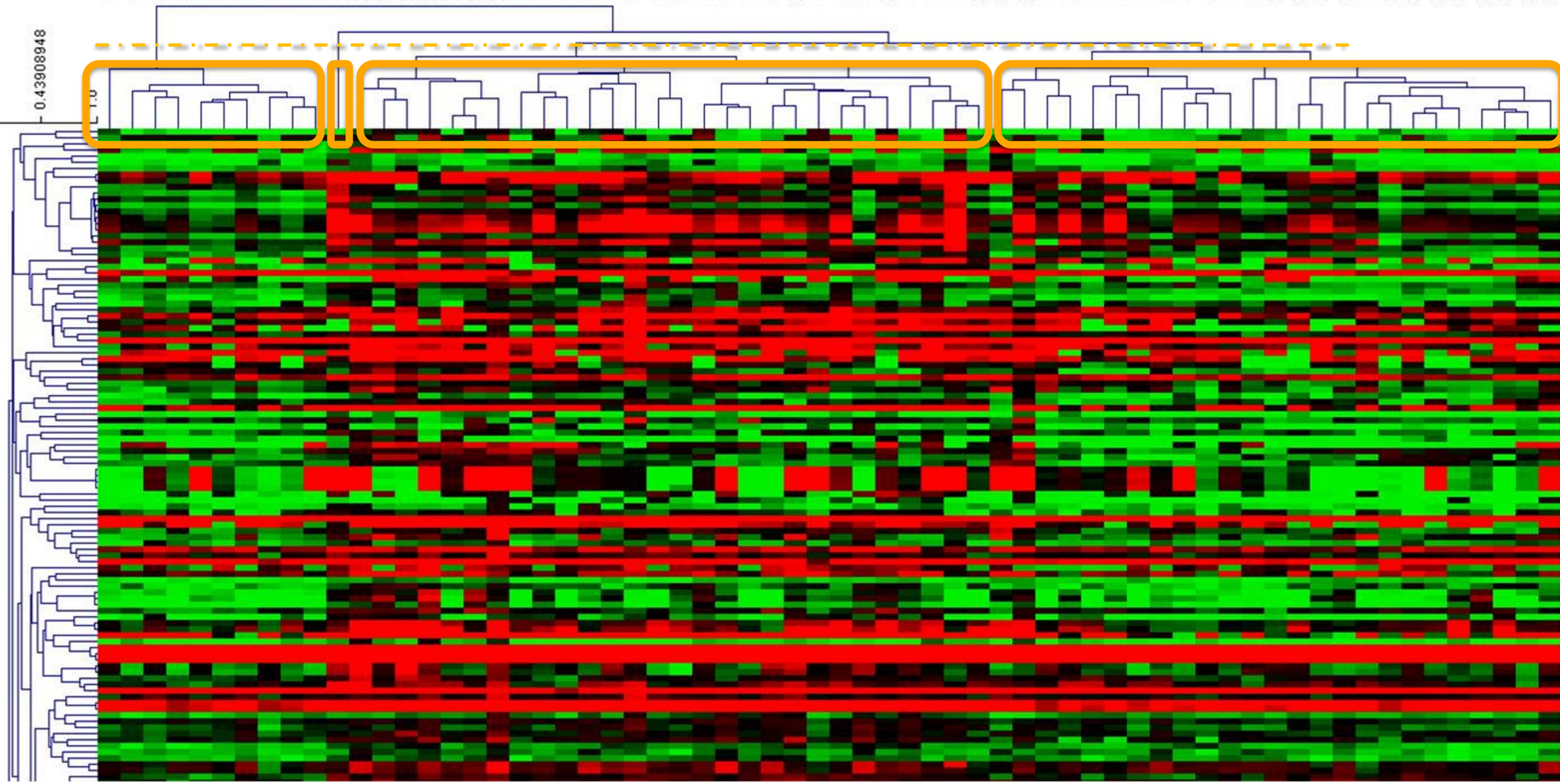
MIII RE

MIII RE

MIII RE

60.37278

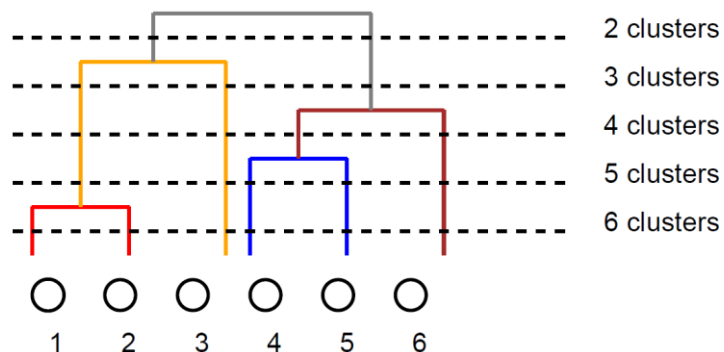
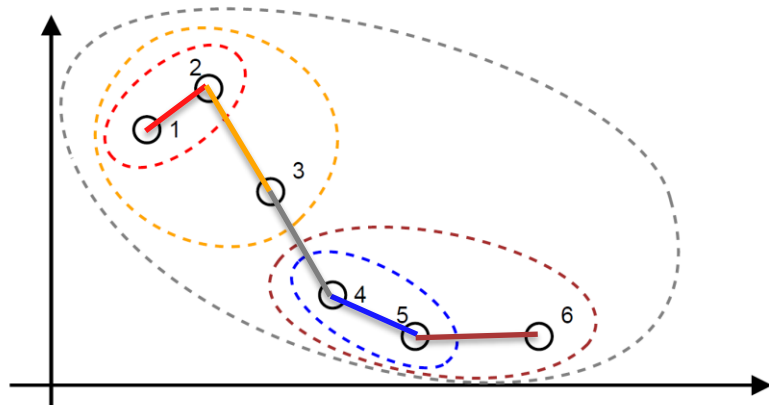
doi: <https://doi.org/10.1371/journal.pone.0092202.g005>





# Bottom up approach

Given  $N$  items to be clustered, and an  $N \times N$  distance (or similarity) matrix, follow the following algorithm (S.C. Johnson 1967):



---

## Algorithm 10.2 Hierarchical Clustering

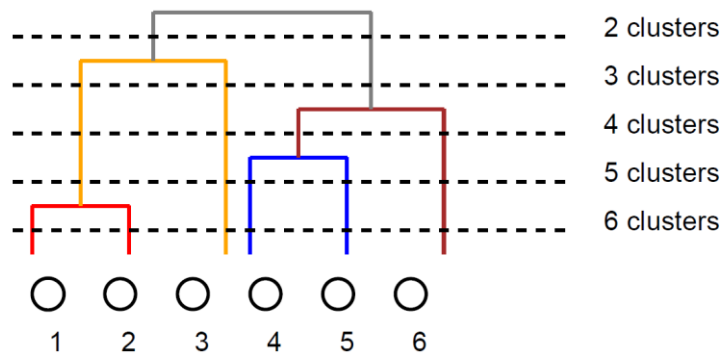
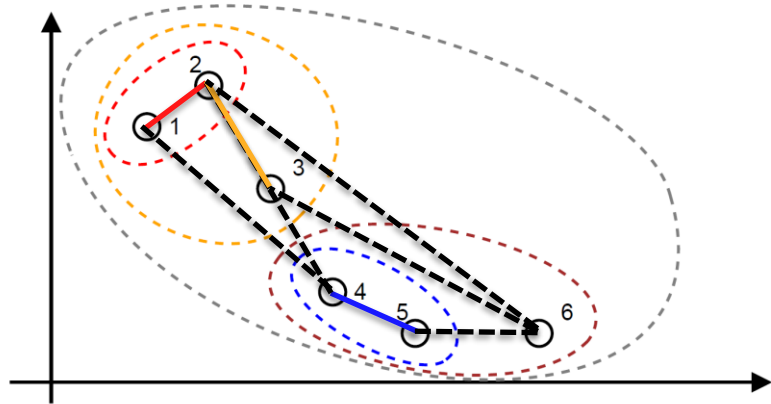
---

1. Begin with  $n$  observations and a measure (such as Euclidean distance) of all the  $\binom{n}{2} = n(n-1)/2$  pairwise dissimilarities. Treat each observation as its own cluster.
2. For  $i = n, n-1, \dots, 2$ :
  - (a) Examine all pairwise inter-cluster dissimilarities among the  $i$  clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.
  - (b) Compute the new pairwise inter-cluster dissimilarities among the  $i-1$  remaining clusters.



# Bottom up approach

Given  $N$  items to be clustered, given a distance matrix, follow the following

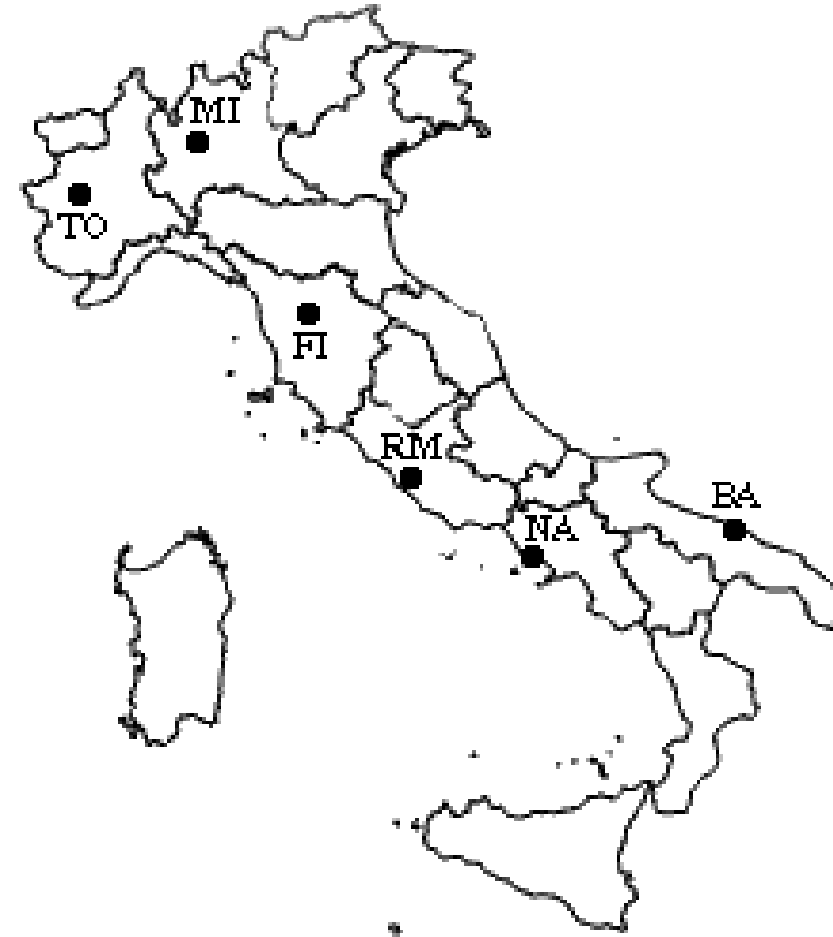


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# 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   |



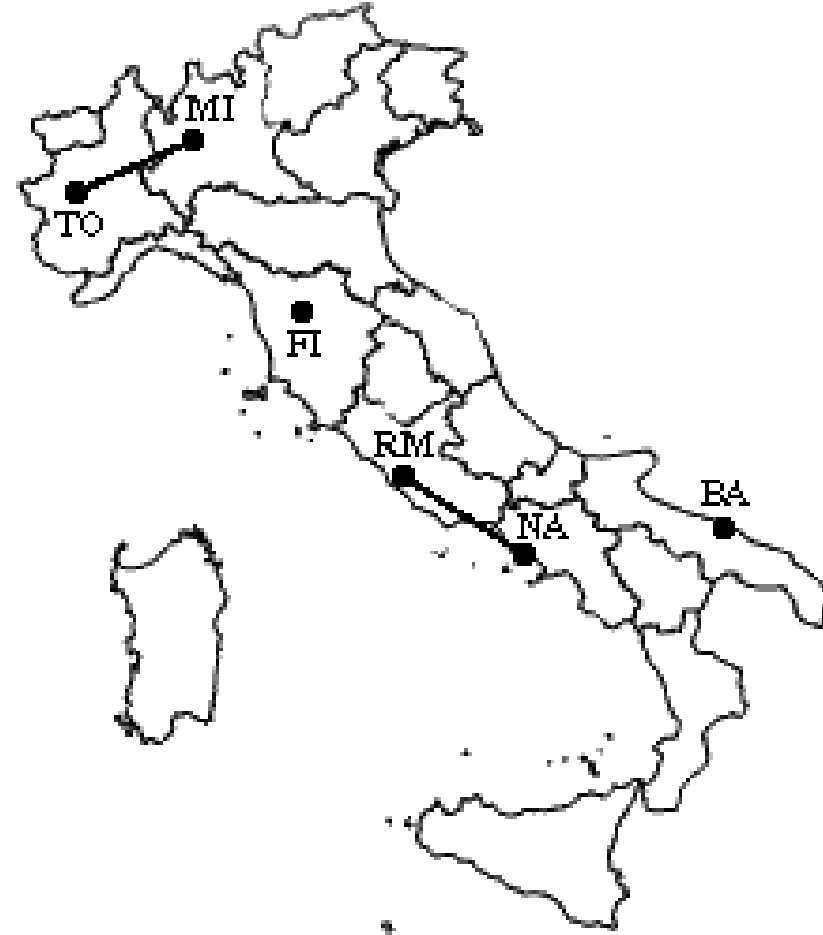
# Hierarchical clustering example

|       | BA  | FI  | MI/TO | NA  | RM  |
|-------|-----|-----|-------|-----|-----|
| BA    | 0   | 662 | 877   | 255 | 412 |
| FI    | 662 | 0   | 295   | 468 | 268 |
| MI/TO | 877 | 295 | 0     | 754 | 564 |
| NA    | 255 | 468 | 754   | 0   | 219 |
| RM    | 412 | 268 | 564   | 219 | 0   |



# Hierarchical clustering example

|       | BA  | FI  | MI/TO | NA/RM |
|-------|-----|-----|-------|-------|
| BA    | 0   | 662 | 877   | 255   |
| FI    | 662 | 0   | 295   | 268   |
| MI/TO | 877 | 295 | 0     | 564   |
| NA/RM | 255 | 268 | 564   | 0     |



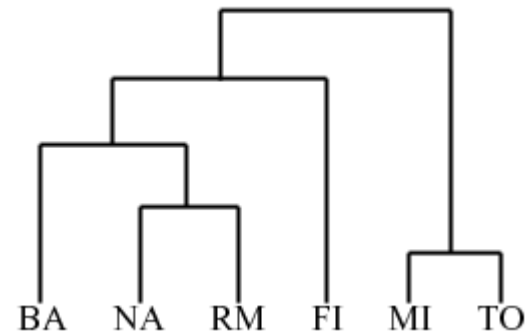
# Hierarchical clustering example

|          | BA/NA/RM | FI  | MI/TO |
|----------|----------|-----|-------|
| BA/NA/RM | 0        | 268 | 564   |
| FI       | 268      | 0   | 295   |
| MI/TO    | 564      | 295 | 0     |

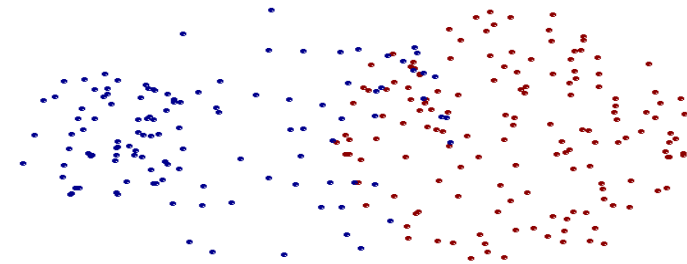
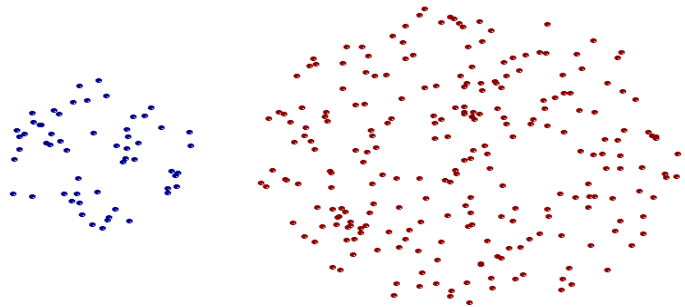
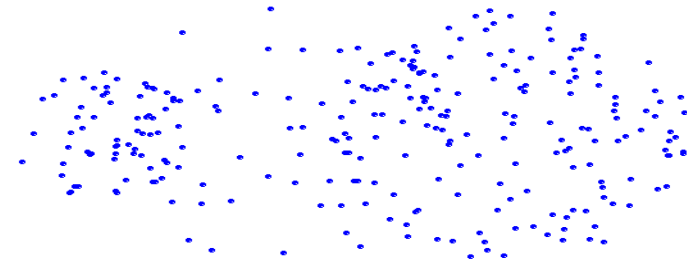
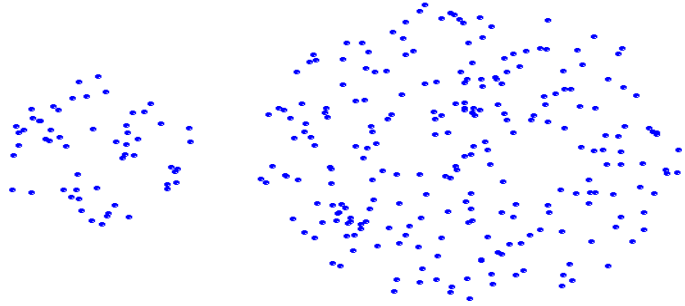


# Hierarchical clustering example

|             | BA/FI/NA/RM | MI/TO |
|-------------|-------------|-------|
| BA/FI/NA/RM | 0           | 295   |
| MI/TO       | 295         | 0     |



# Strengths and Weaknesses of Single Linkage



Pros:

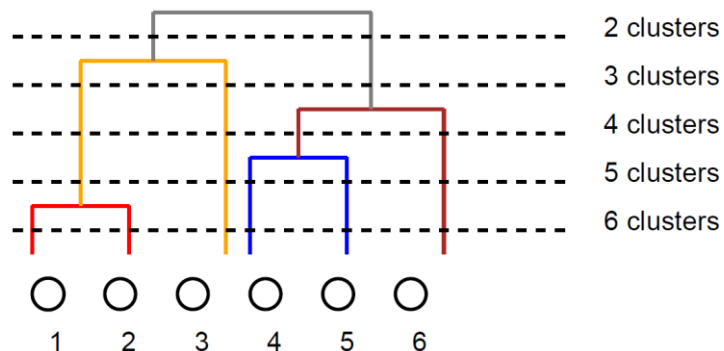
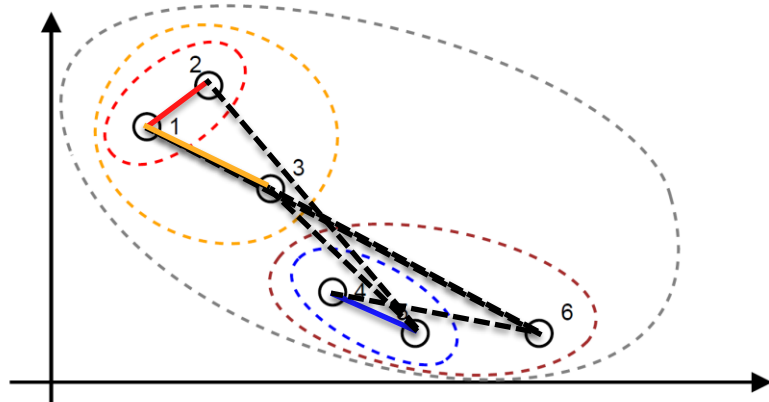
- Can handle non elliptical shapes

Cons:

- Sensitive to noise and outliers

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following

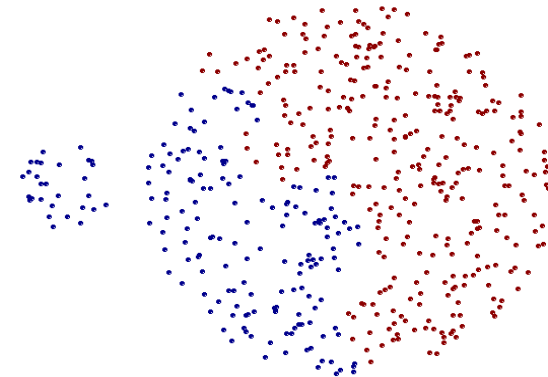
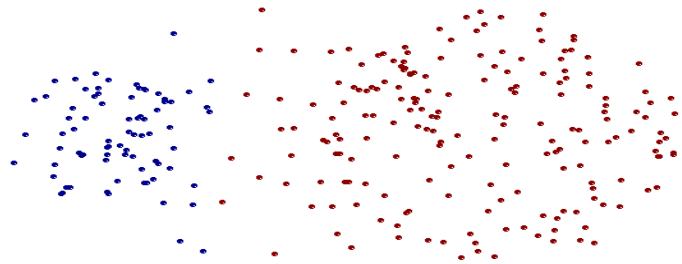
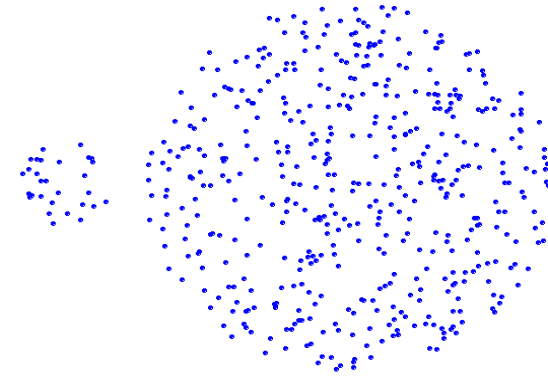
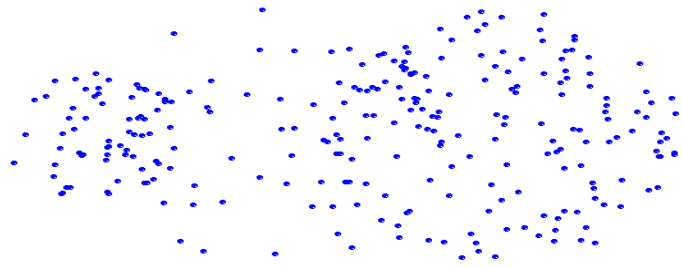


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.



# Strengths and Weaknesses of Complete Linkage



Pros:

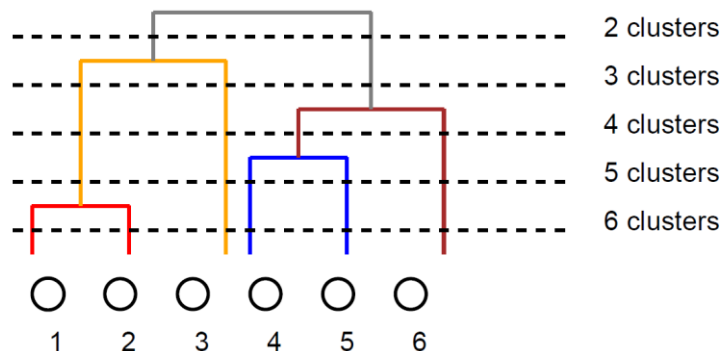
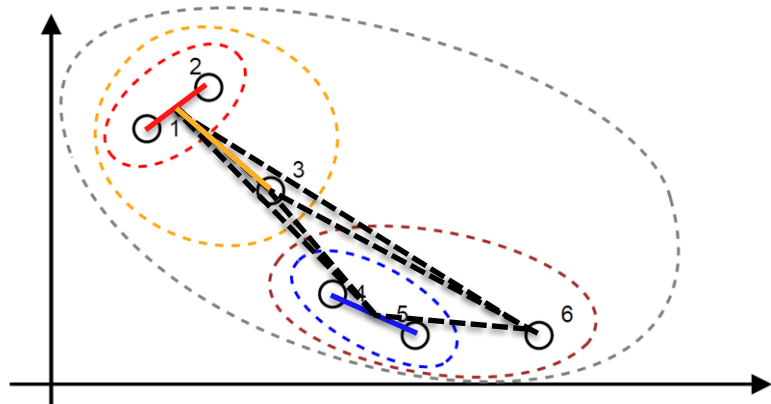
- Robust to noise and outliers

Cons:

- Breaks large clusters
- Biased to globular clusters

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following

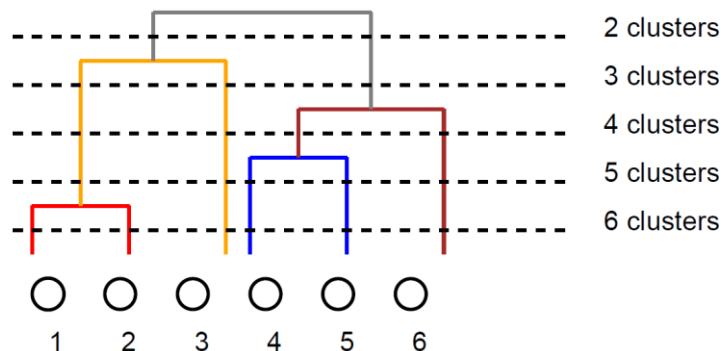
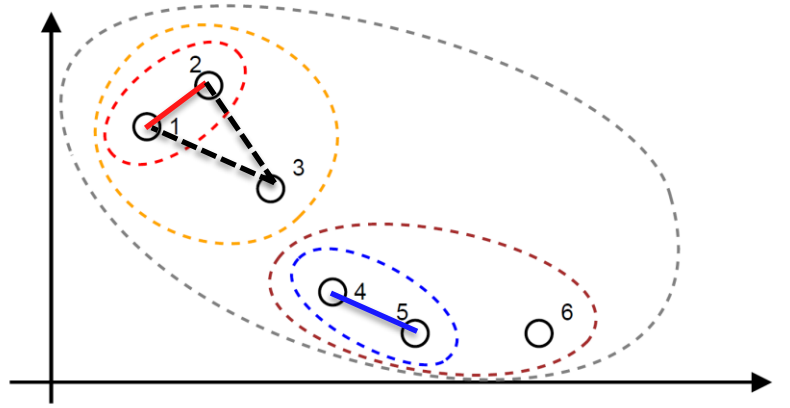


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following

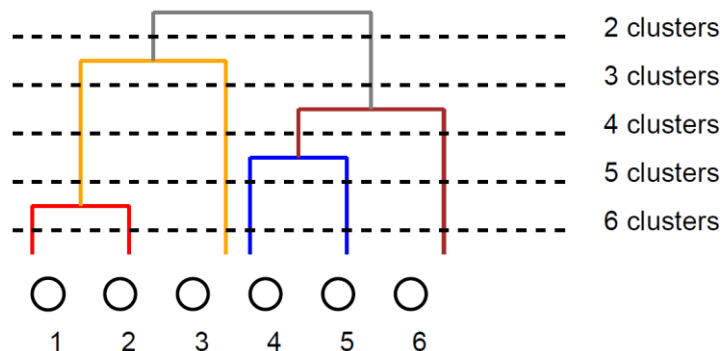
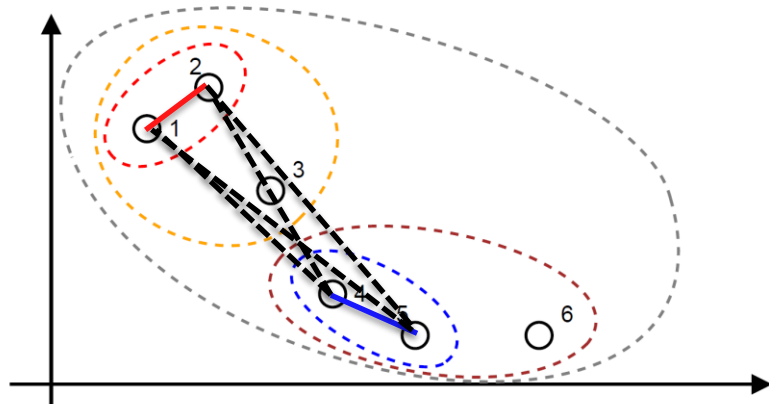


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered, given a distance matrix, follow the following

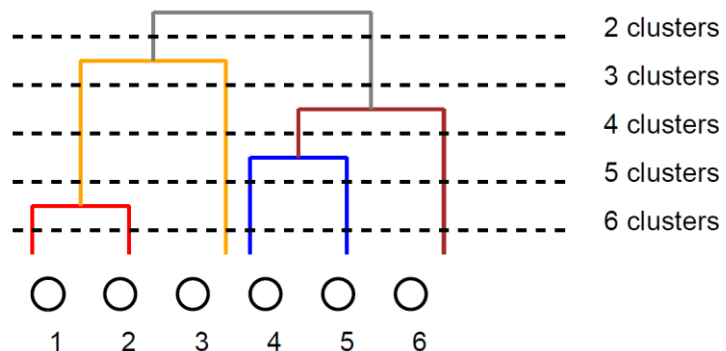
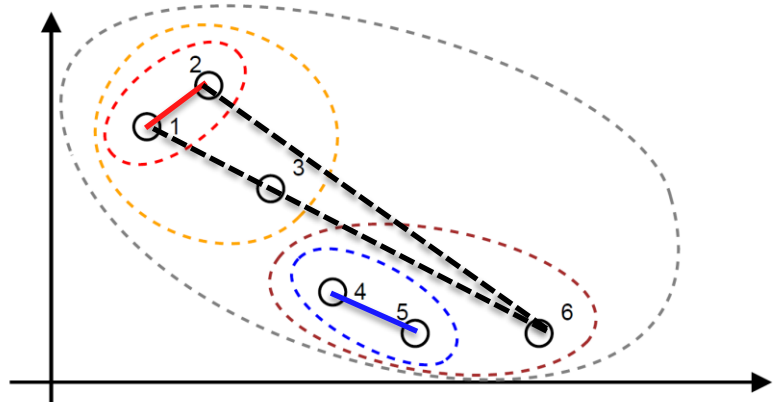


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered, given a distance matrix, follow the following

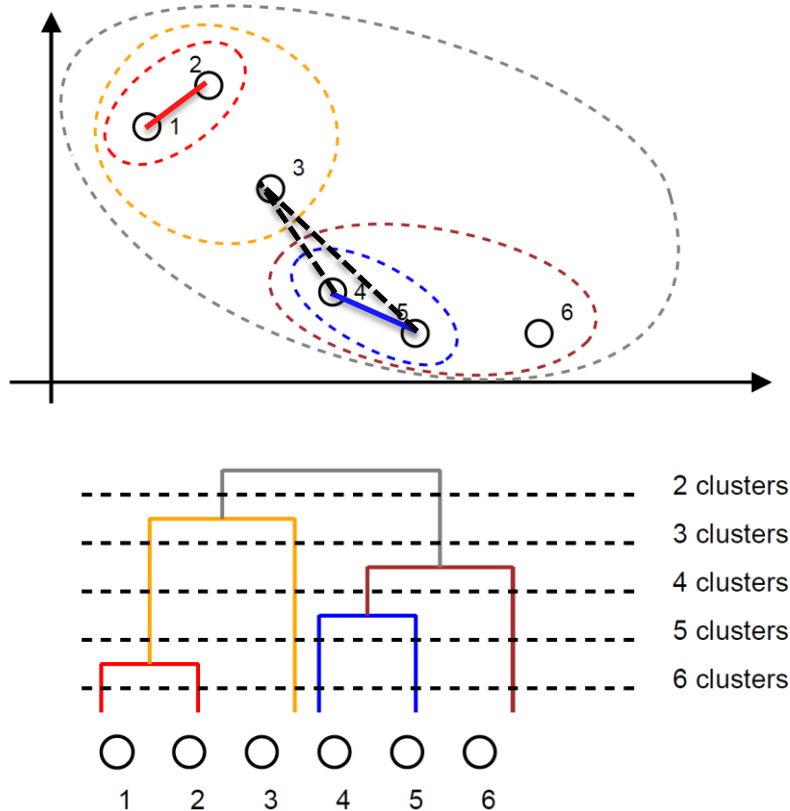


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following

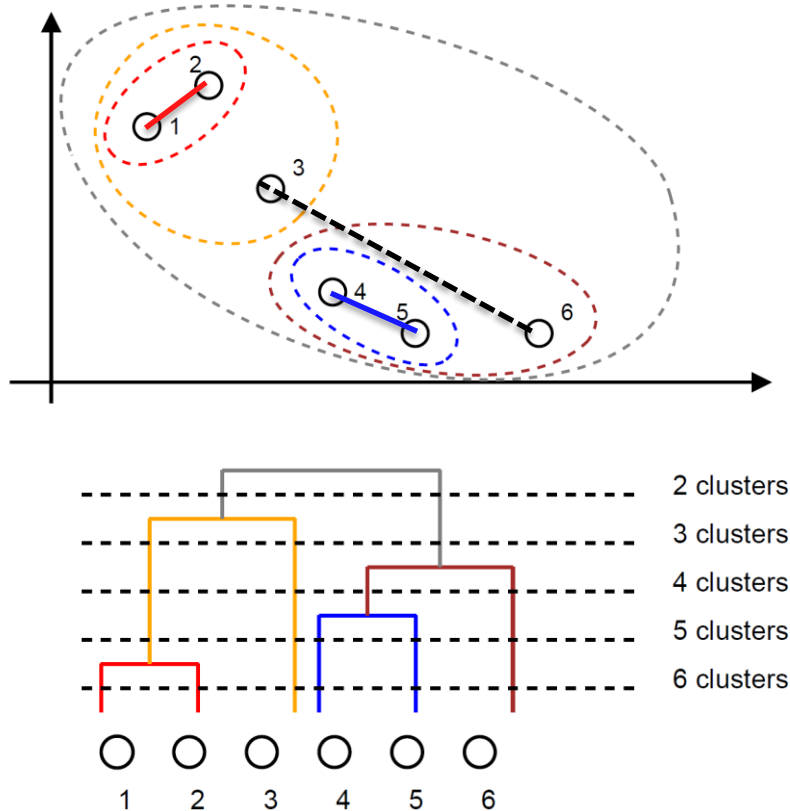


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following

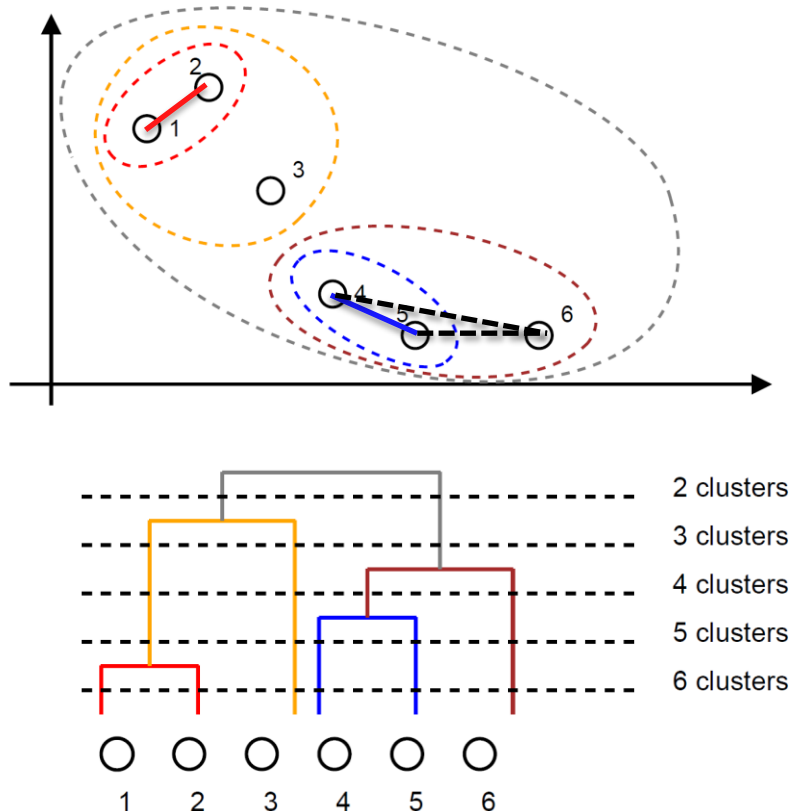


| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following



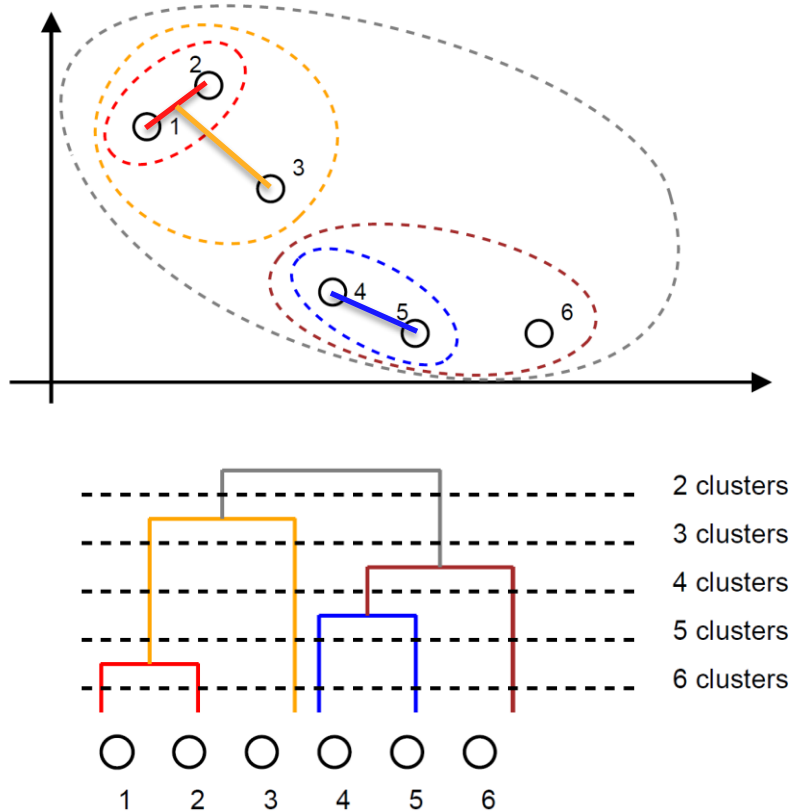
| <i>Linkage</i> | <i>Description</i>  |
|----------------|---|
| Complete       | Maximal intercluster dissimilarity. Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the <i>largest</i> of these dissimilarities.   |
| Single   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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.



# Bottom up approach

Given  $N$  items to be clustered  
matrix, follow the following



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

**TABLE 10.2.** A summary of the four most commonly-used types of linkage in hierarchical clustering.

# Hierarchical clustering wrap-up (and alternative)

## Bottom up Hierarchical Clustering

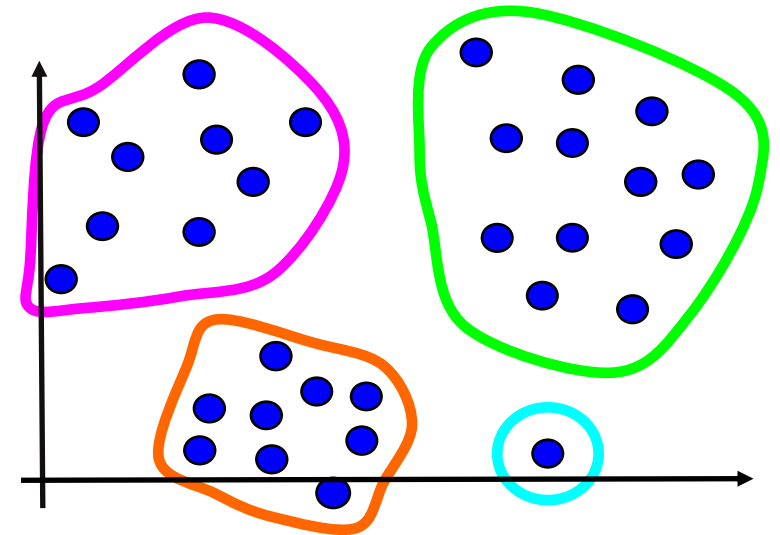
- Start with single-instance clusters and, at each step, join two closest clusters
- Need to decide about distance between clusters
- They do not scale well, complexity up to  $O(N^3)$
- They can never undo what was done before
- No clear objective function is optimized

## Top down Hierarchical Clustering

- Start with one universal cluster, find two clusters, proceed recursively
- No need to decide about distance between clusters
- They can be fast (parallel recursion)  $O(N^2 \log N)$
- They rely on *Partitioning Clustering*

# Partitioning Clustering

Partitioning methods construct a partition of  $N$  objects into  $K$  clusters,  $K$  is known, and the partition optimizes a partitioning criterion



*Global optimal*: exhaustively enumerate all partitions select the best one

- Unfeasible due to the high number of possible partitions:  $\binom{N}{K}$

*Heuristic methods*: select cluster representatives and optimize w.r.t. those

- k-means (MacQueen'67): clusters are represented by the points' centroids
- k-medoids or PAM (Partition around medoids) (Kaufman & Rousseeuw'87): each cluster is represented by one of the objects in the cluster

# K-means algorithm

One of the simplest unsupervised learning algorithms

- Assumes an Euclidean space
- Number of clusters  $K$  is known/fixed a priori

*Random centroids  
initialization*

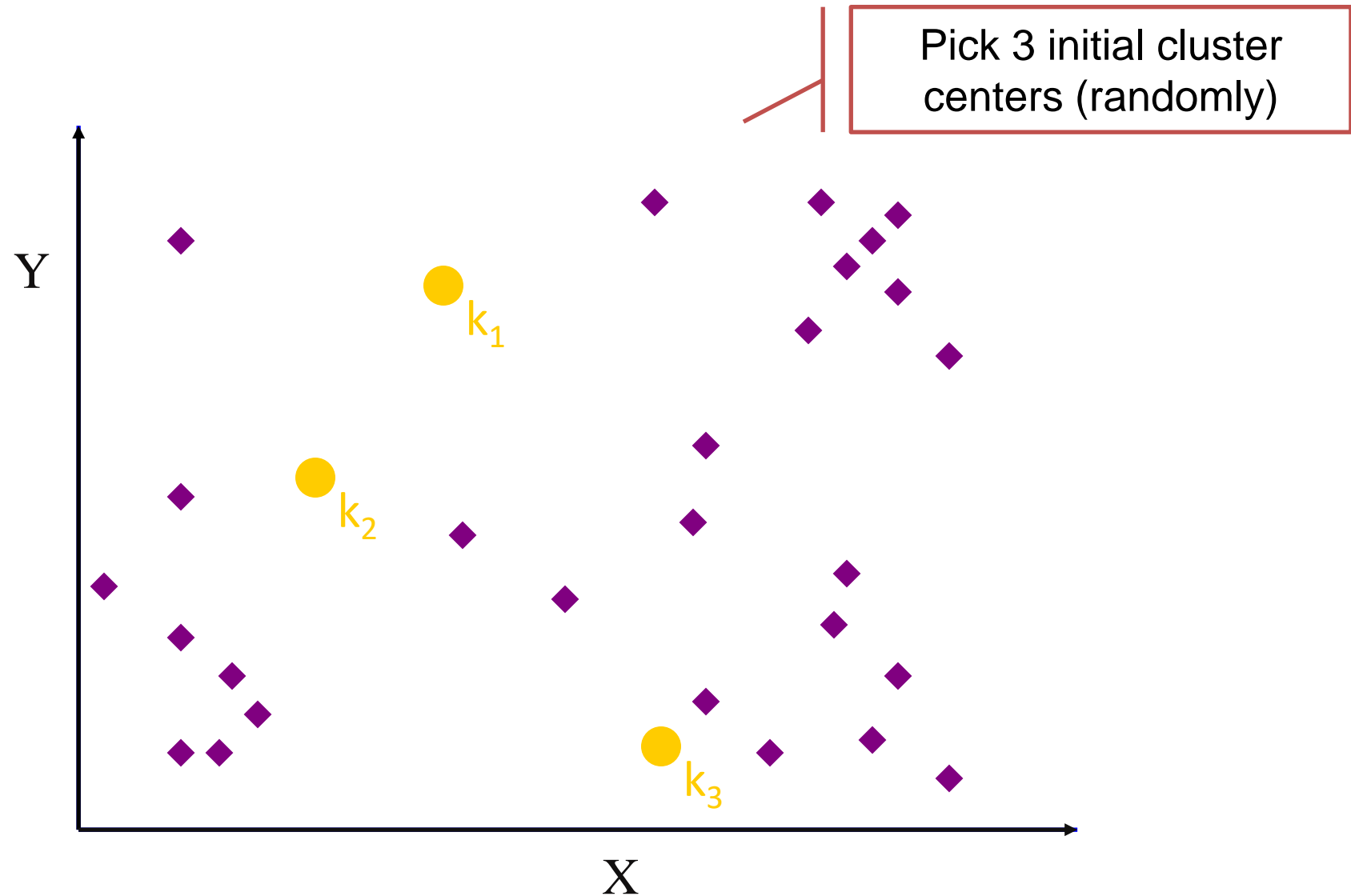
---

## **Algorithm 10.1** *K-Means Clustering*

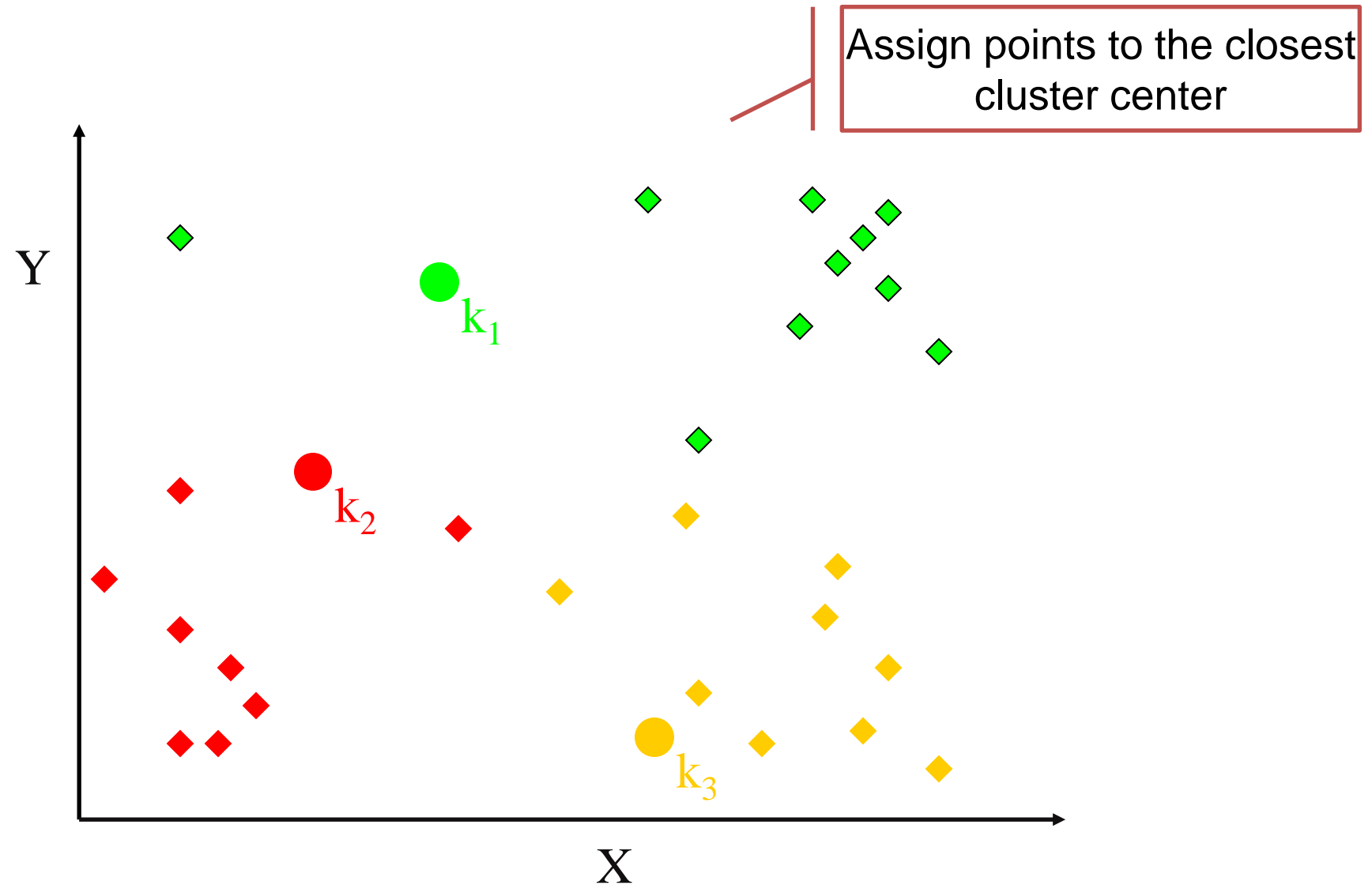
---

1. Randomly assign a number, from 1 to  $K$ , to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the cluster assignments stop changing:
  - (a) For each of the  $K$  clusters, compute the cluster *centroid*. The  $k$ th cluster centroid is the vector of the  $p$  feature means for the observations in the  $k$ th cluster.
  - (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

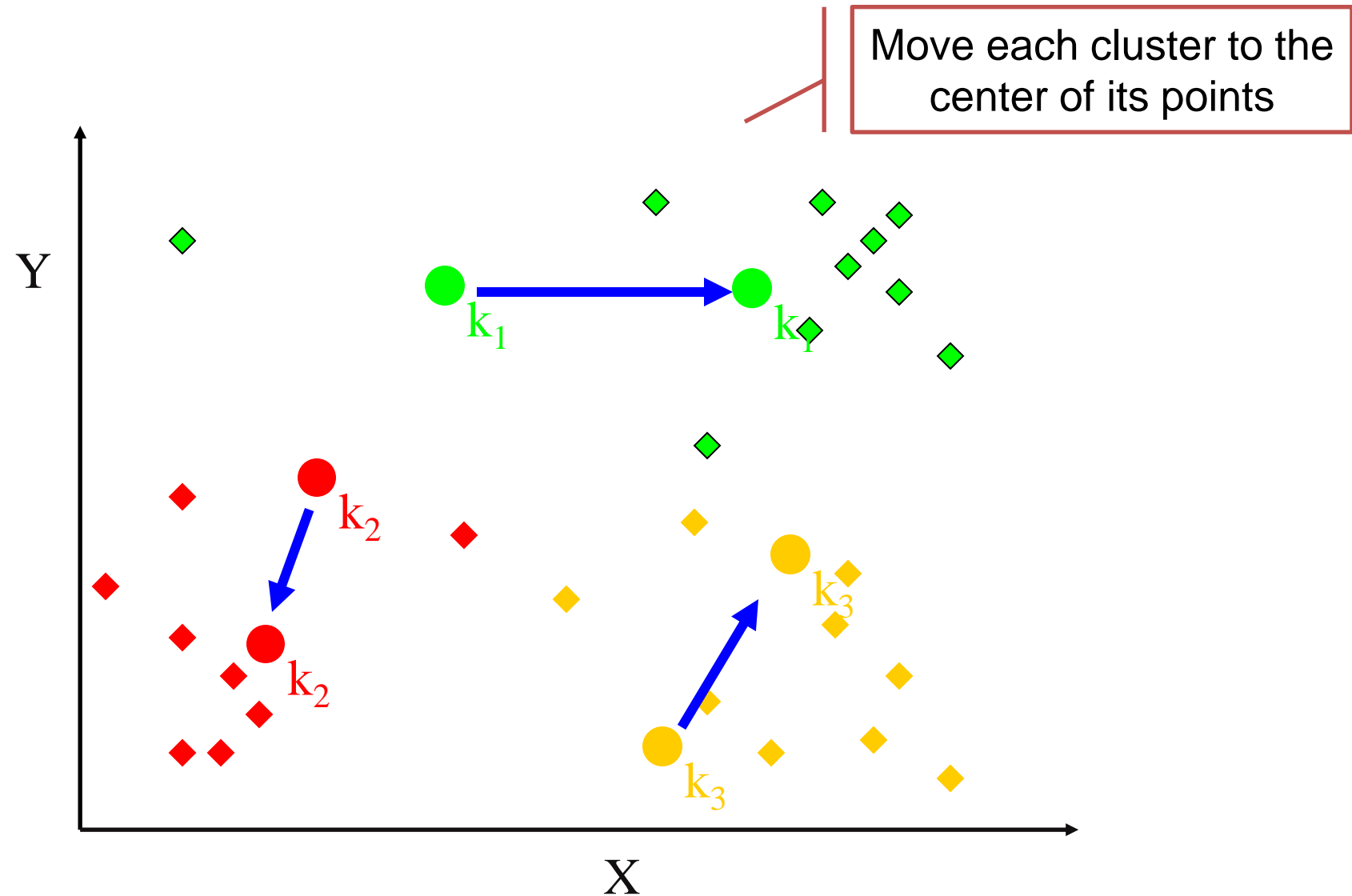
# K-means example (1)



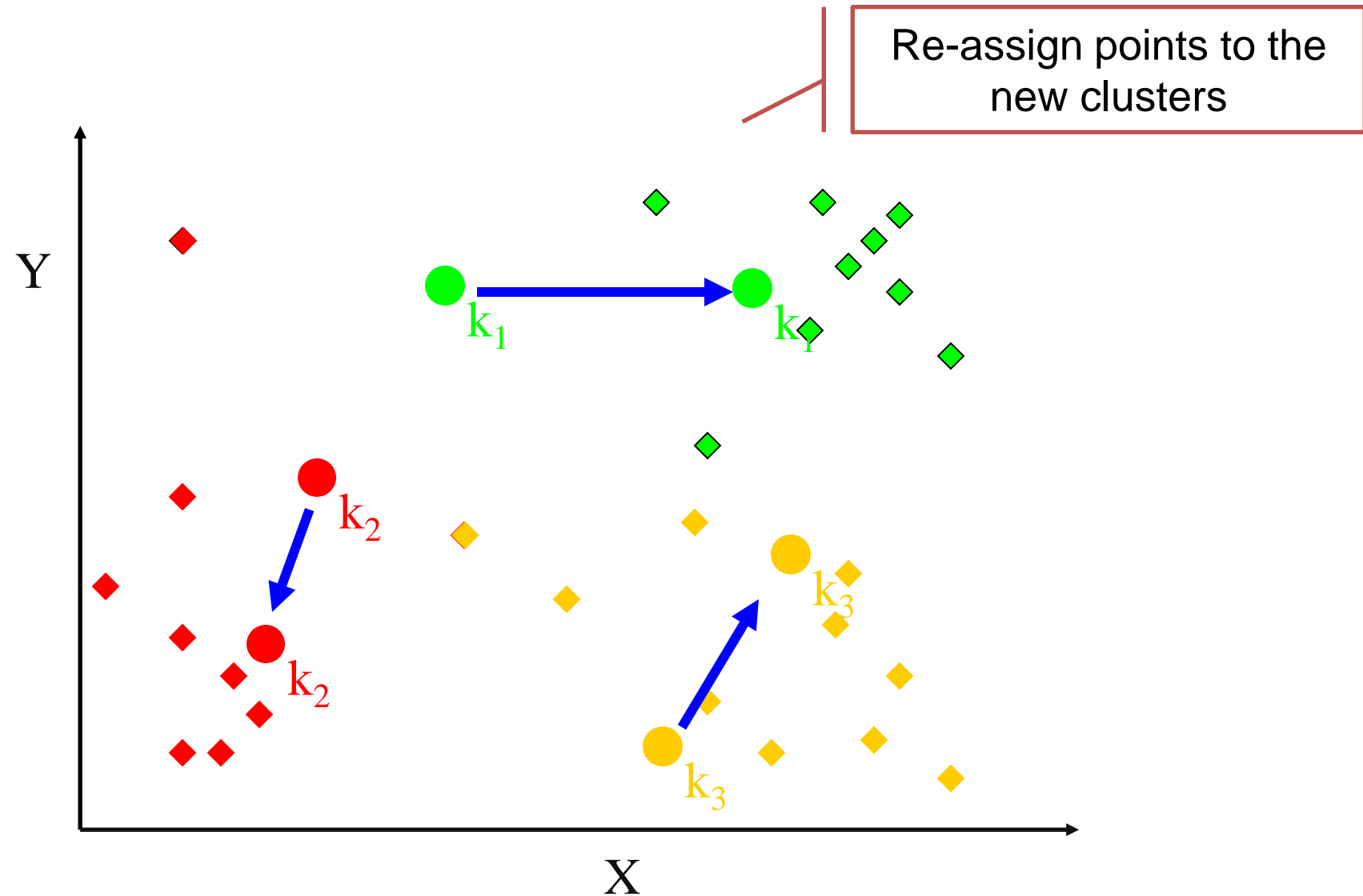
# K-means example (2)



# K-means example (3)

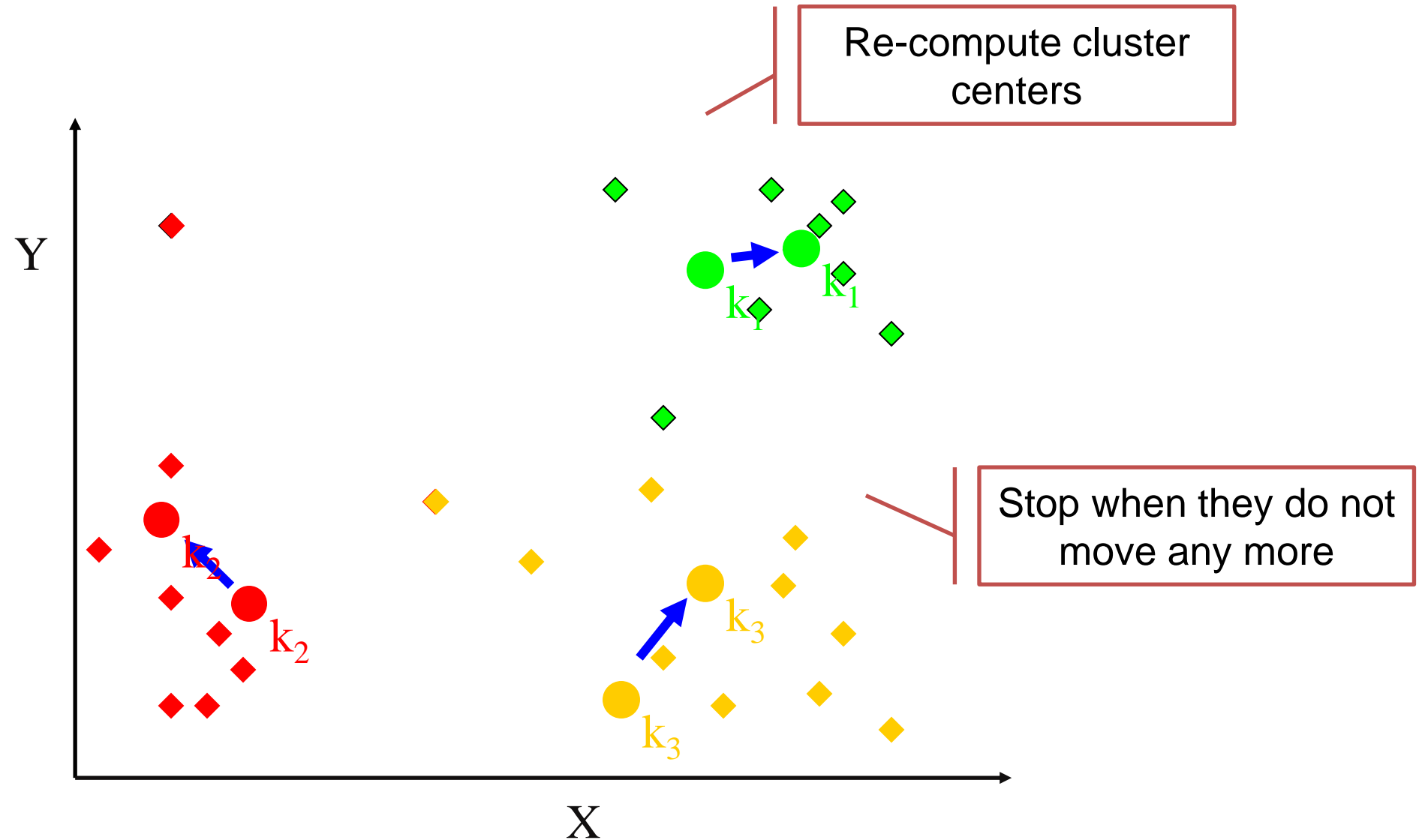


# K-means example (4)



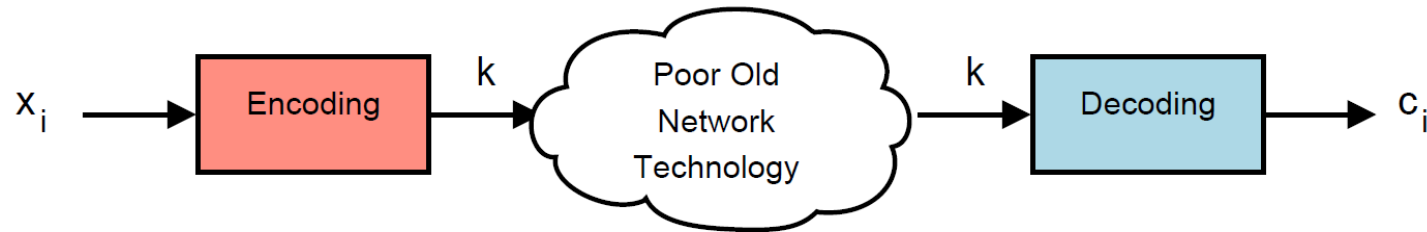


# K-means example (5)



# What's behind k-means?

Suppose we have a poor, old, fashioned network allowing  $K$  values. We want to transmit real-valued vectors with lossy compression



Given the two functions:

1. Encode :  $\mathfrak{R}^m \rightarrow [1 \dots K]$
2. Decode :  $[1 \dots K] \rightarrow \mathfrak{R}^m$

We can write  $D$  (a.k.a. Distortion) as:

$$D = \sum_n^N \left( x_n - \text{Decode}(\text{Encode}(x_n)) \right)^2 = \sum_n^N \left( x_n - c_{\text{Encode}(x_n)} \right)^2$$

*A.k.a. Vector Quantization!*

# What's behind k-means?

What properties must centers  $c_1, c_2, \dots, c_k$  have when  $D$  is minimized?

$$D = \sum_n^N (x_n - \text{Decode}(\text{Encode}(x_n)))^2 = \sum_n^N (x_n - c_{\text{Encode}(x_n)})^2$$

1.  $x_n$  must be encoded by its nearest center otherwise Distortion could be reduced by simply replacing  $\text{Encode}(x_n)$  by the nearest center
2. The partial derivative of Distortion with respect to centers must be 0

This translate into

1.  $c_{\text{Encode}(x_n)} = \text{argmin}_{c_k} (x_n - c_k)^2$

2.  $c_k = \frac{1}{|x_n \in C_k|} \sum_{x_n \in C_k} x_n$

Why?

Let's check this derivative.


# What's behind k-means?

What properties must centers  $c_1, c_2, \dots, c_k$  have when  $D$  is minimized?

$$D = \sum_n^N (x_n - \text{Decode}(\text{Encode}(x_n)))^2 = \sum_n^N (x_n - c_{\text{Encode}(x_n)})^2$$

$$\frac{\partial D}{\partial c_k} = \frac{\partial}{\partial c_k} \sum_n^N (x_n - c_{\text{Encode}(x_n)})^2 = \frac{\partial}{\partial c_k} \sum_k^K \sum_{x_n \in C_k} (x_n - c_{\text{Encode}(x_n)})^2 =$$

$$= \frac{\partial}{\partial c_k} \sum_{x_n \in C_k} (x_n - c_{\text{Encode}(x_n)})^2 = -2 \sum_{x_n \in C_k} (x_n - c_{\text{Encode}(x_n)}) = 0$$

  $c_{\text{Encode}(x_n)} = \frac{1}{|x_n \in C_k|} \sum_{x_n \in C_k} x_n$

# K-means pros and cons

## Advantages

- Simple, understandable, and fast! 😊
- Guaranteed to converge
- Items automatically assigned to clusters
- Relatively efficient:  $O(tKN)$ , where  $t$  is the number of iterations ( $K, t \ll N$ ).

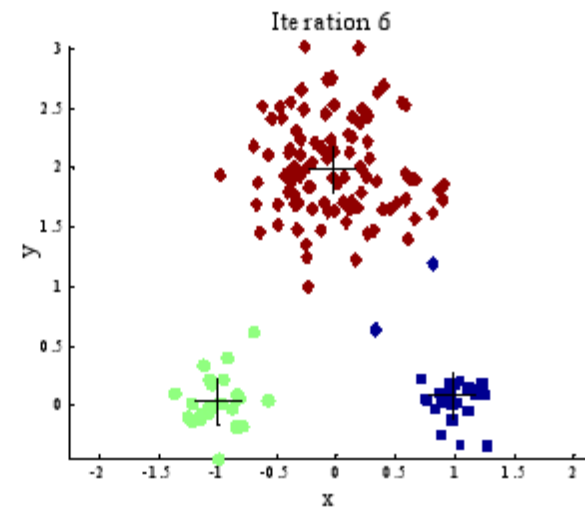
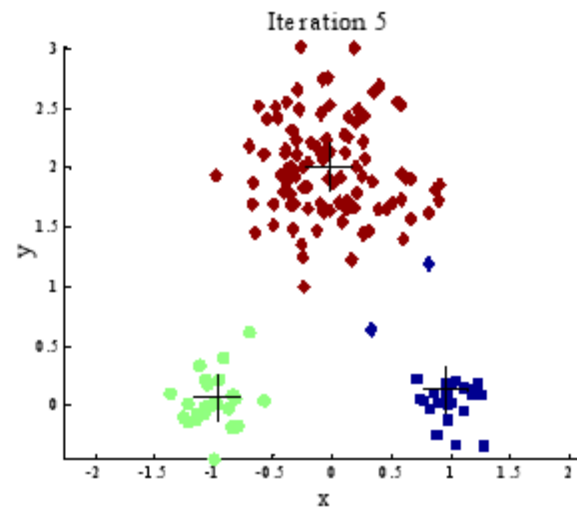
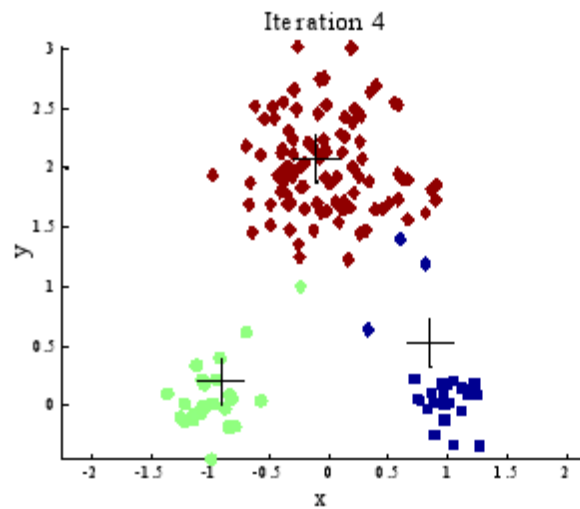
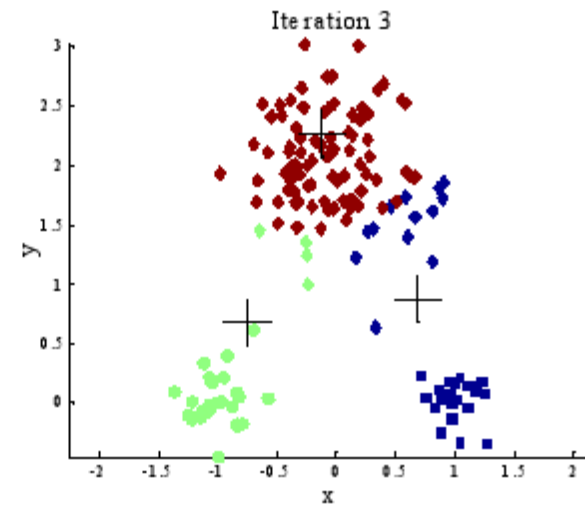
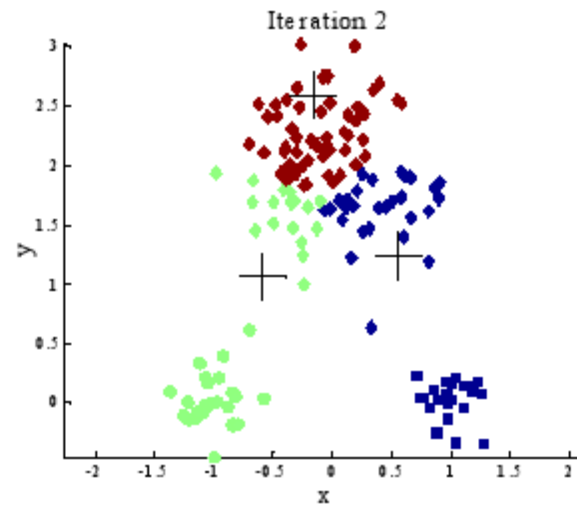
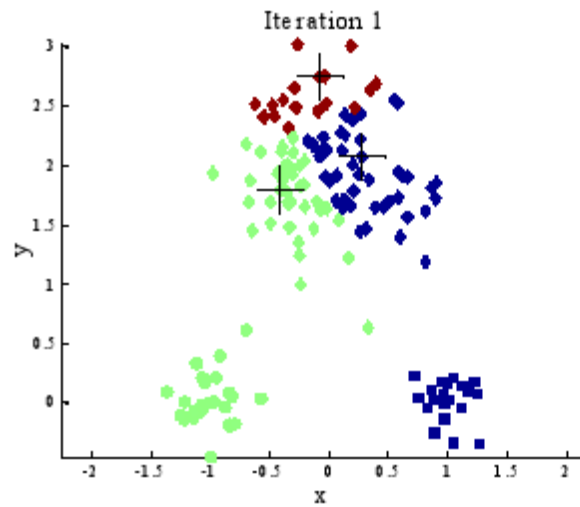
## Disadvantages

- Must pick number of clusters before hand
- Results can vary significantly depending on initial choice
- All items are forced into a cluster, including noise
- Applicable only when mean is defined (what about categorical data?)
- Not suitable with clusters with non-convex shapes

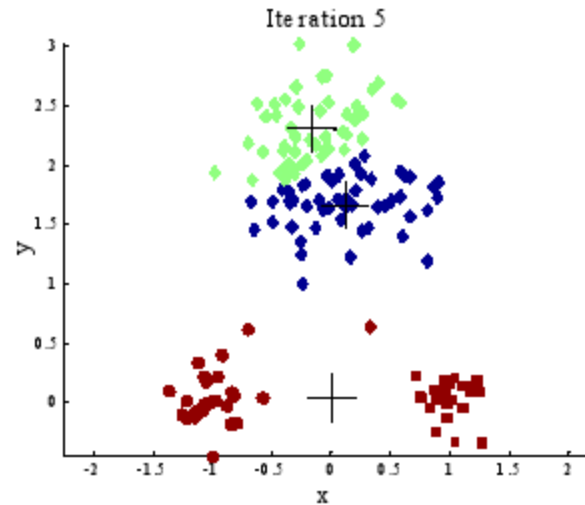
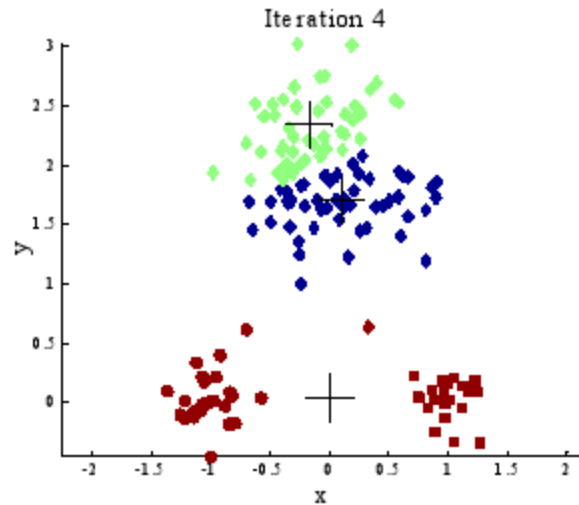
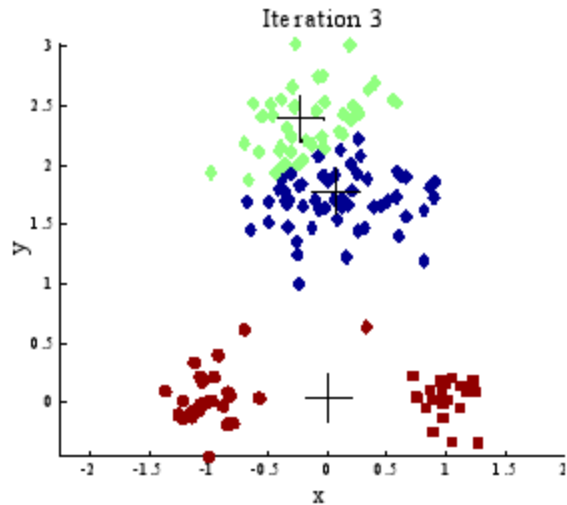
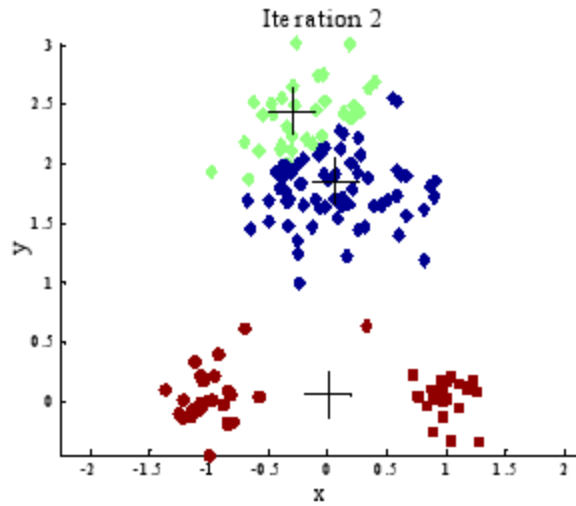
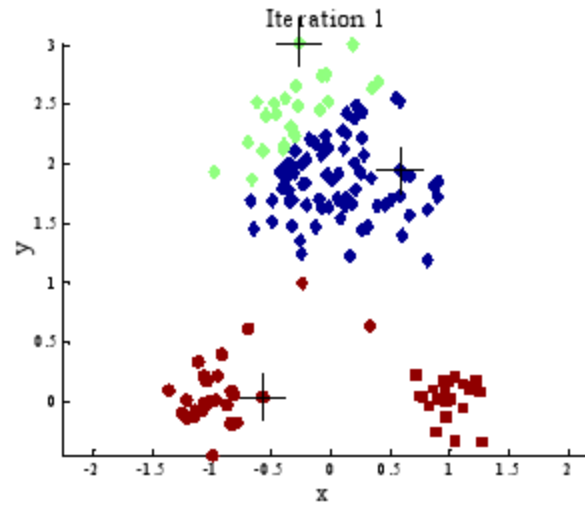
*Some heuristics to do this exists ...*

*Fixed by medoids*

# K-means: Initialization 1



# K-means: Initialization 2



# Selecting Initial Points

If there are  $K$  “real” clusters then chance of selecting one centroid from each cluster is small, e.g., with clusters of the same size

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K! n^K}{(Kn)^K} = \frac{K!}{K^K}$$

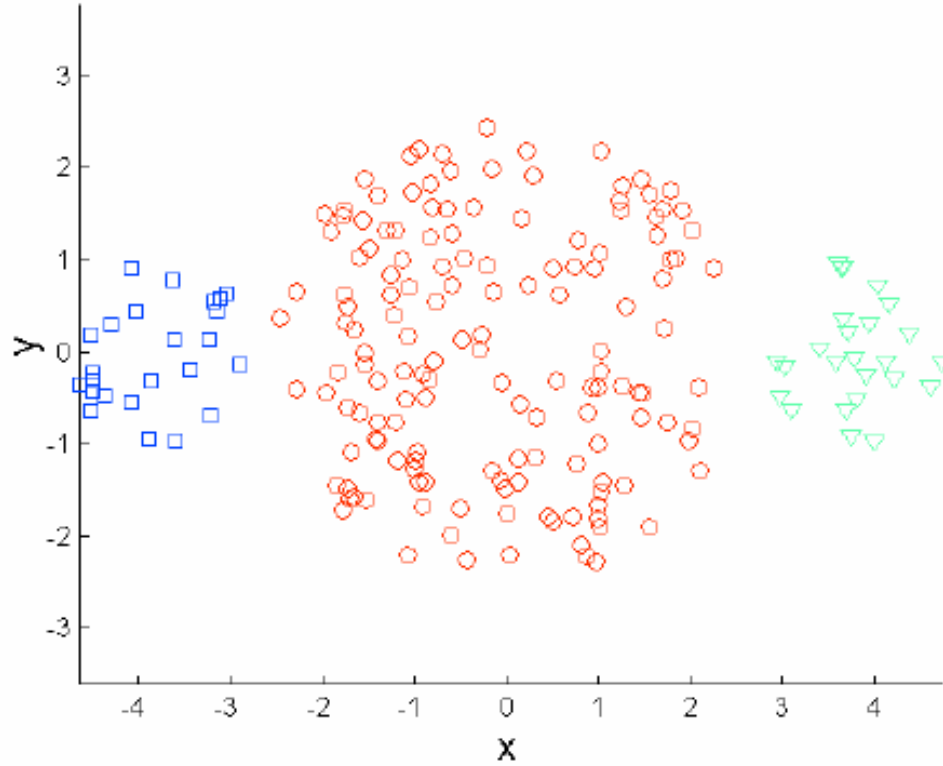
if  $K=10$ , then  $P = 10!/10^{10} = 0.00036$

## Possible solutions

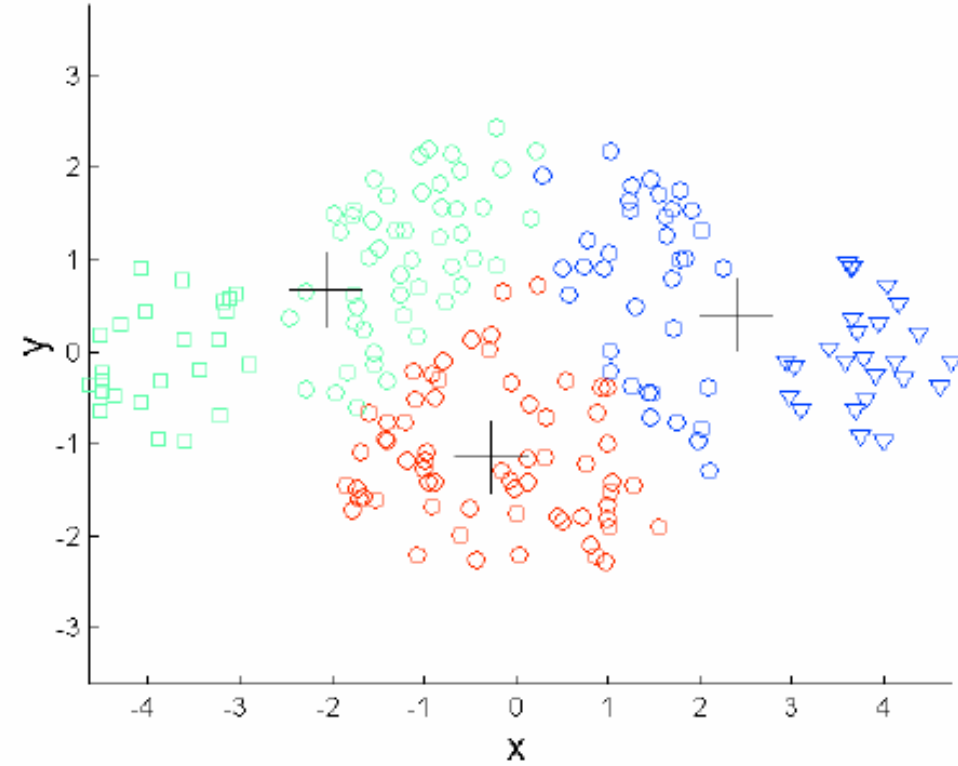
- Multiple runs helps, but probability is not on your side
- Initialize on far away points (serial initialization on far away points)
- Sample and use hierarchical clustering to determine initial centroids
- Select more than  $K$  initial centroids and then select among these initial centroids the most widely separated ones



# K-means: Different Sizes

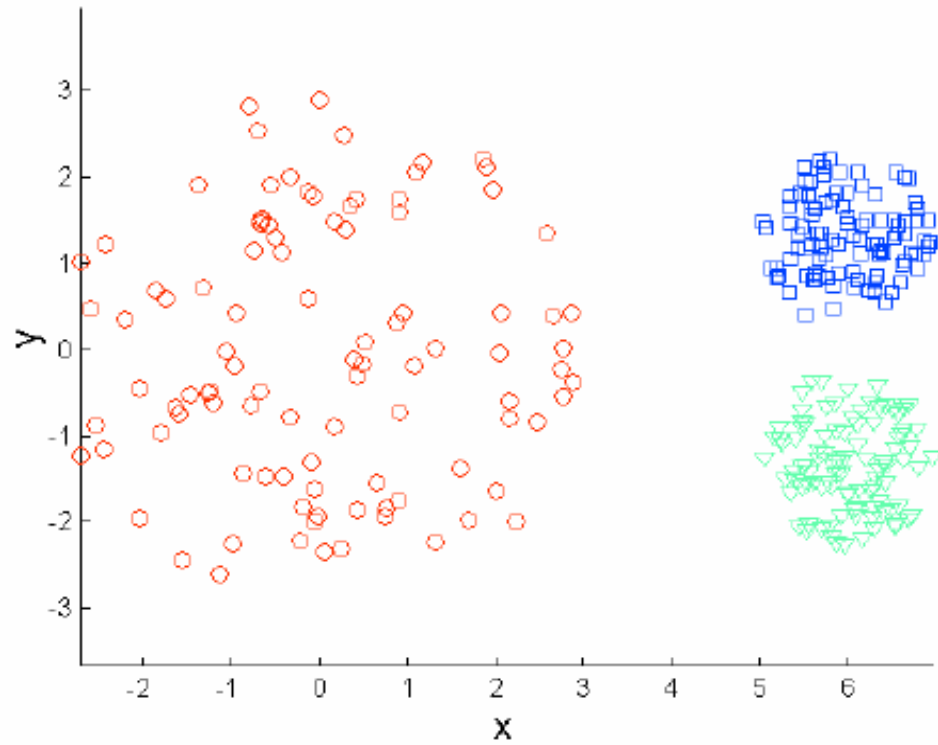


**Original Points**

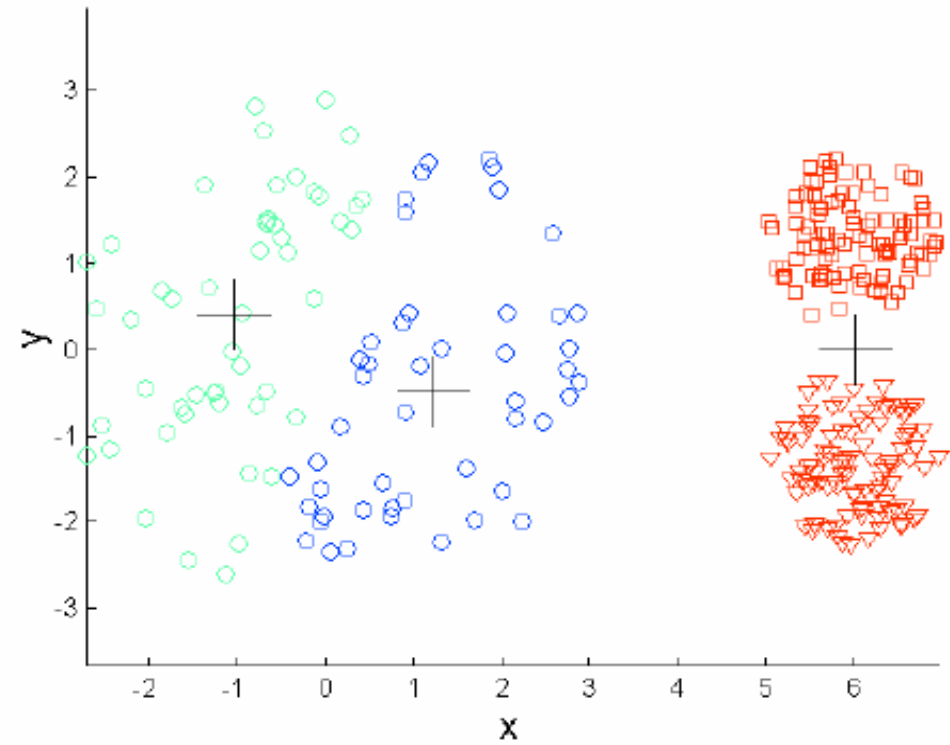


**K-means Clusters**

# K-means: Different Densities

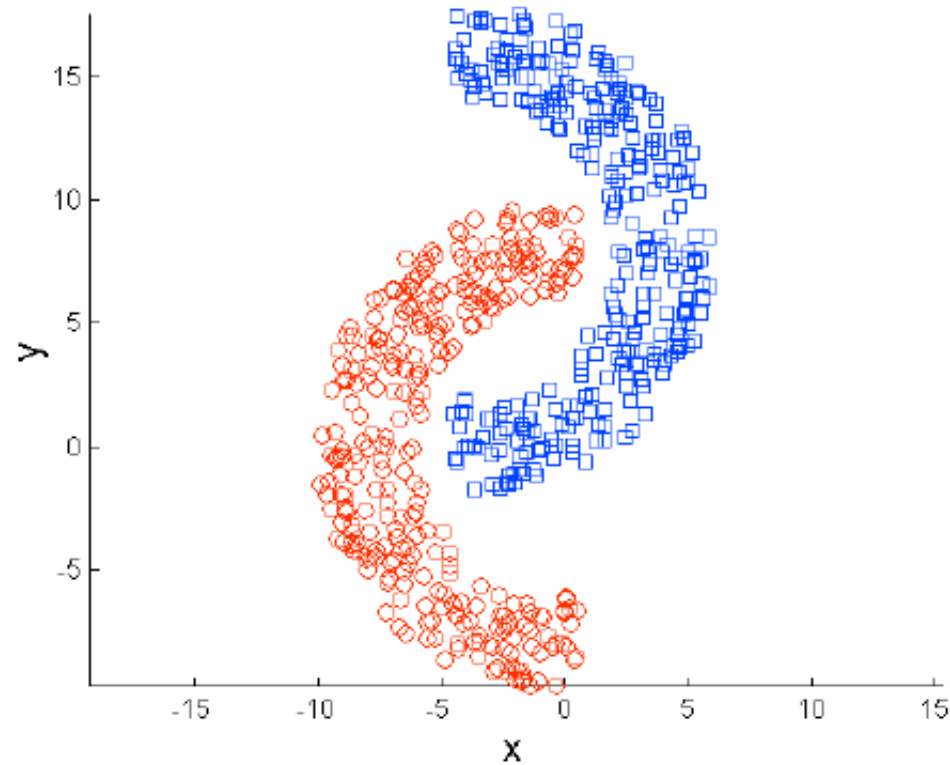


**Original Points**

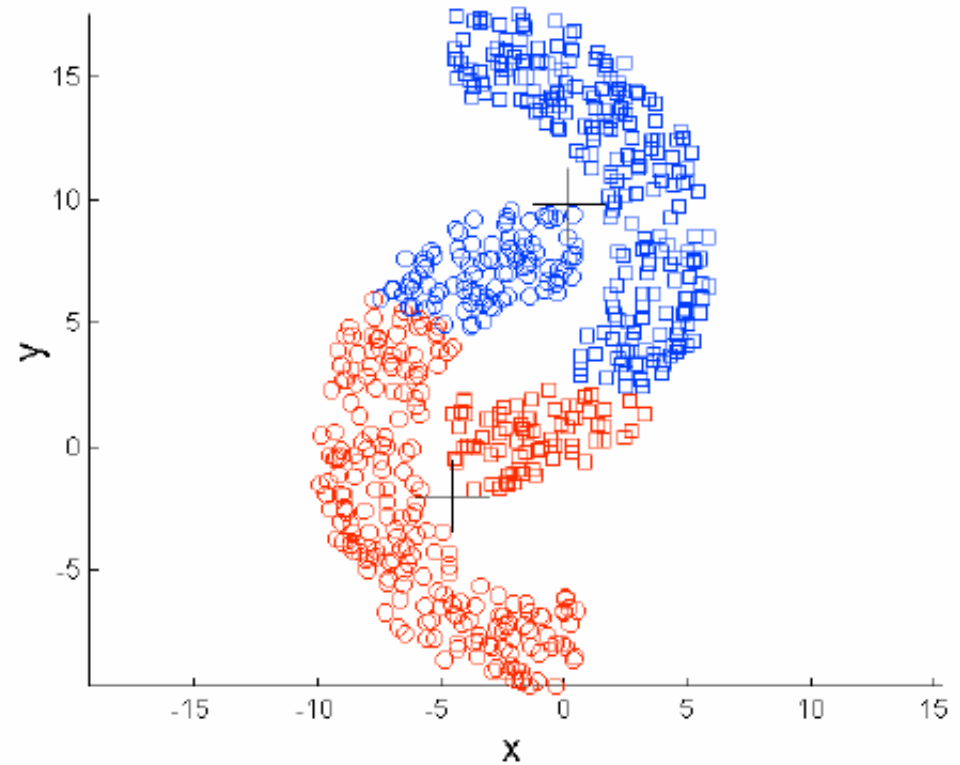


**K-means Clusters**

# K-means: Non-globular Shapes

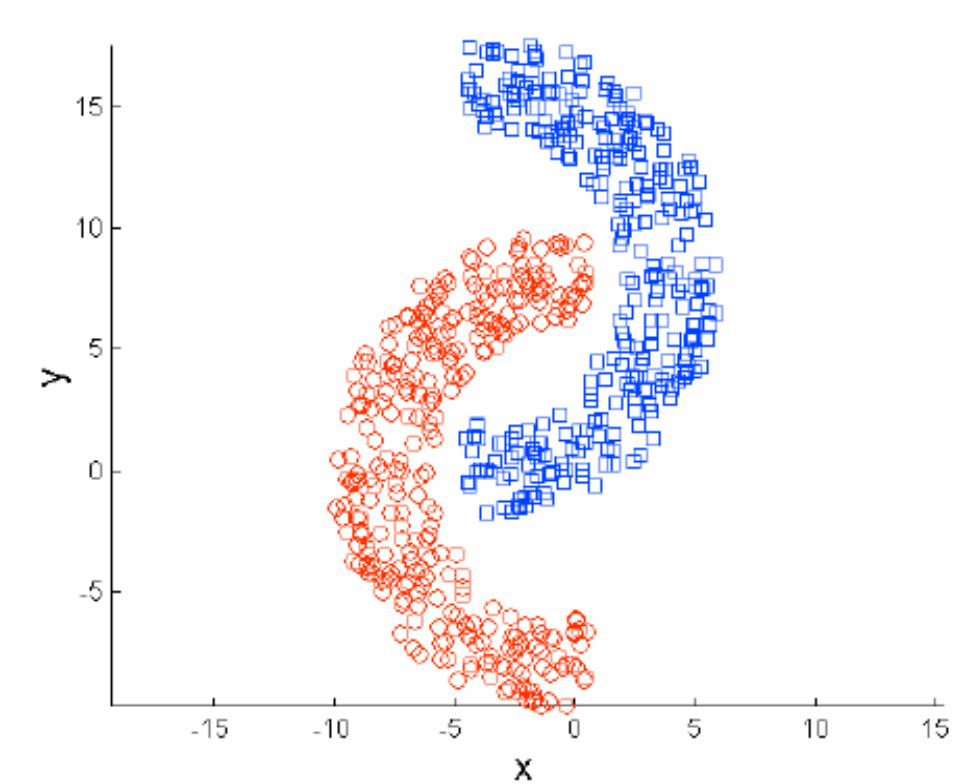


**Original Points**

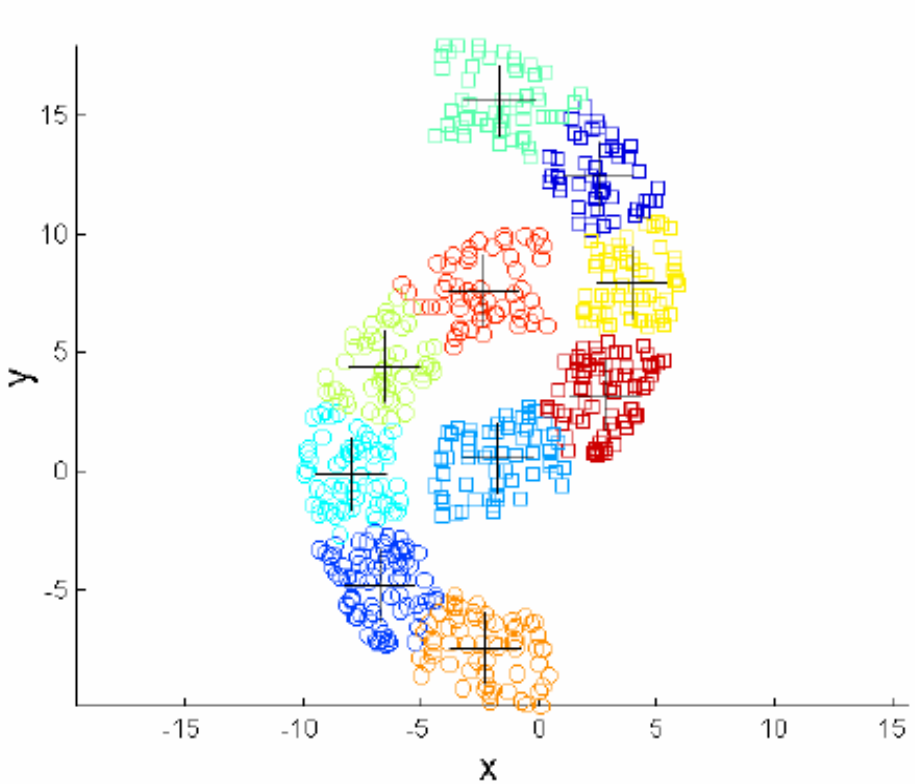


**K-means Clusters**

# What if we increase K?

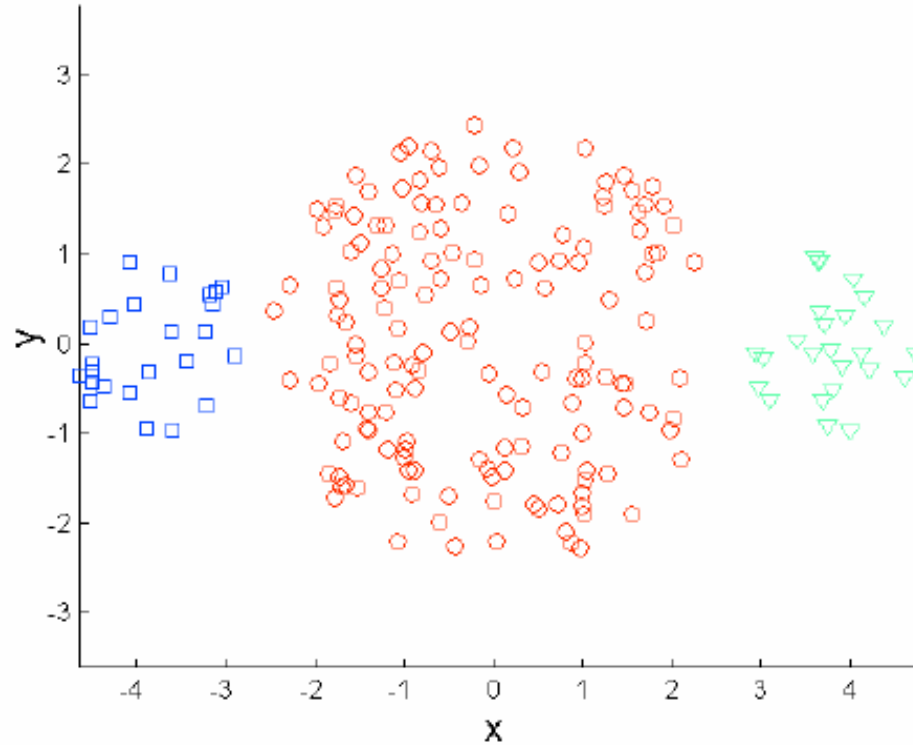


**Original Points**

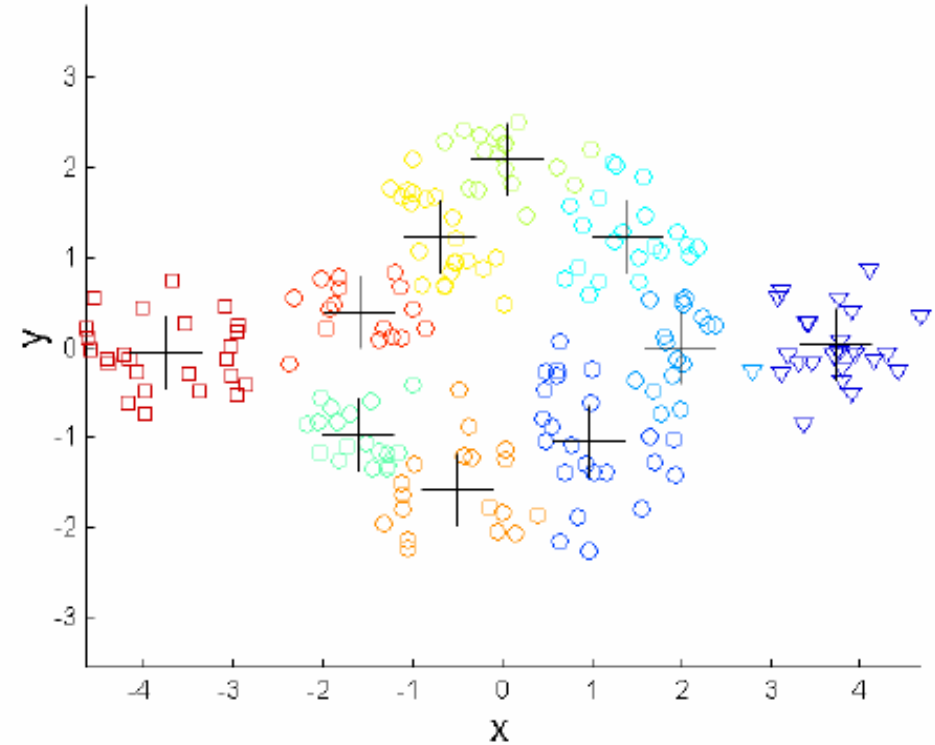


**K-means Clusters**

# What if we increase K?



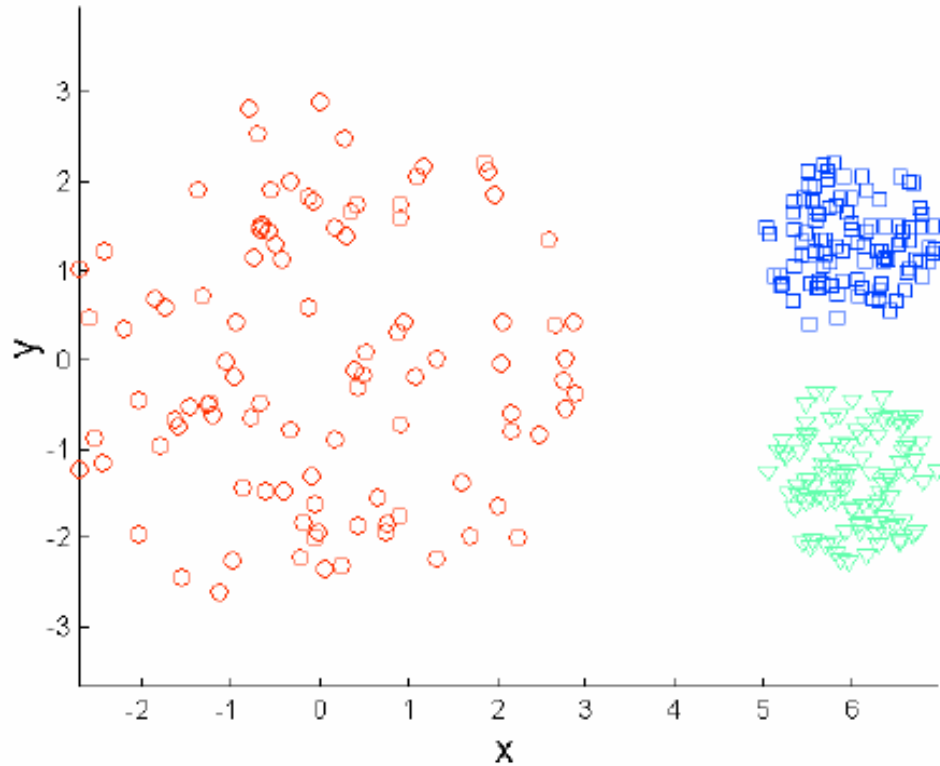
**Original Points**



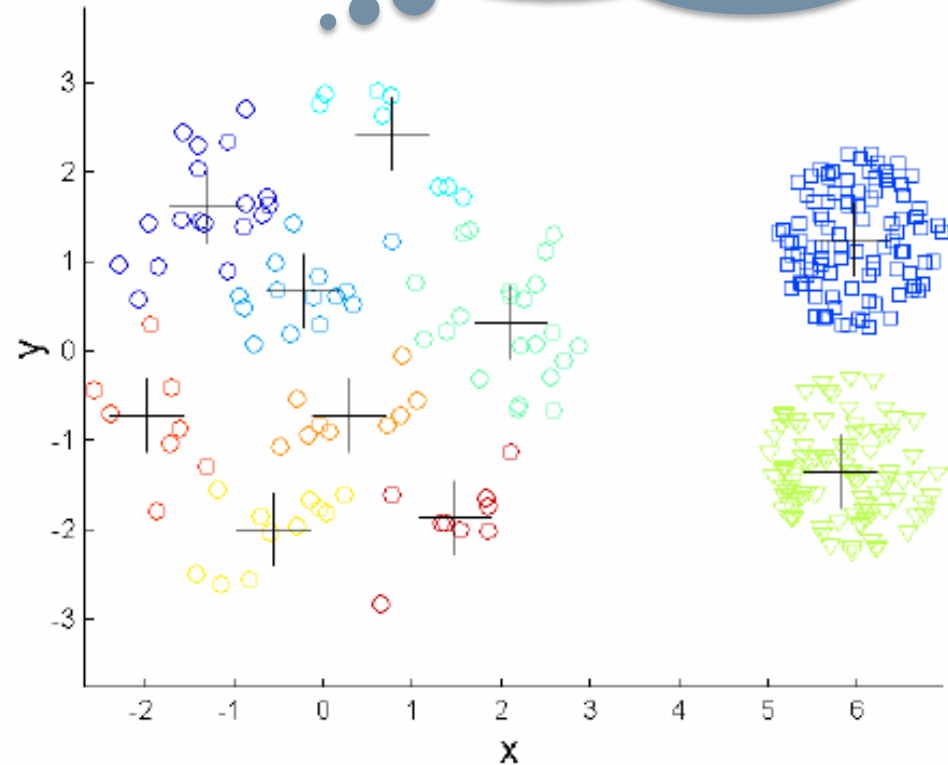
**K-means Clusters**

# What if we increase K?

Unfortunately, the higher  $K$  the lower the Distortion 😞



**Original Points**



**K-means Clusters**

# Tips and Tricks

Basic K-means can yield empty clusters in the initial assignment of points:

- Choose points contributing most to Distortion/SSE (farthest from centroids)
- Choose a point from the cluster with the highest Distortion/SSE
- If there are several empty clusters, the above can be repeated several times

Pre-processing:

- Normalize the data
- Eliminate outliers

Post-processing:

- Eliminate small clusters that may represent outliers
- Split 'loose' clusters, i.e., clusters with relatively high Distortion/SSE
- Merge clusters that are 'close' and that have relatively low Distortion/SSE

# K-medoids algorithm

k-means algorithm is too sensitive to outliers

- An object with an extremely large value may substantially distort the distribution of the data and thus the clustering result

k-medoid algorithm uses the most centrally located point in a cluster, as a representative point of the cluster

- Idea: use *medians*, instead of *means*, to describe the representative point
  - Mean 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
- A medoid is always a point inside a cluster, a centroid might not
- A medoid is defined also if the *mean* is not defined



# PAM: Partitioning around Medoids (1987)

1. Given  $K$
2. Randomly pick  $K$  instances as initial medoids  $m_k$
3. Assign each data point to the nearest medoid  $m_k$
4. Calculate the Distortion  $D$  as the sum of dissimilarities of all points to their nearest medoids (squared-error criterion)
5. For each non-medoid point  $x$ 
  1. Swap  $m_k$  and  $x$  and calculate the objective function
  2. Select the configuration with the lowest cost
6. Repeat (3-6) until no change

*Very robust results on small datasets, but it does not scale ...*

# Density Based Clustering

Clusters can be defined based on density via density-connected points

- Discover clusters of arbitrary shape
- Handle noise
- Single scan
- Need density parameters as termination condition



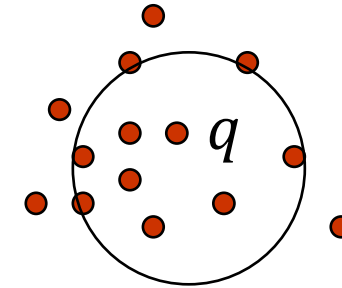
Several algorithms proposed:

- DBSCAN: Ester, et al. (KDD'96)
- OPTICS: Ankerst, et al (SIGMOD'99)
- DENCLUE: Hinneburg & D. Keim (KDD'98)
- CLIQUE: Agrawal, et al. (SIGMOD'98)

# Density Based Clustering Ideas (1/2)

The neighborhood of radius  $\varepsilon$  of an object is called the  $\varepsilon$ -neighborhood, if it contains at least **MinPts** objects, then the object is a core object

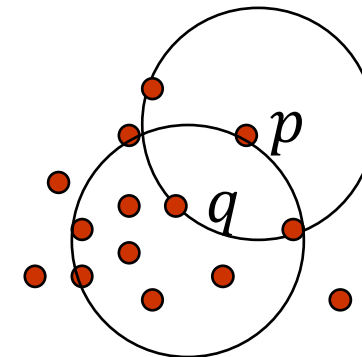
- *Eps*: Maximum radius of the neighbourhood
- *MinPts*: Minimum number of points in an Eps-neighbourhood of that point



$$\begin{aligned} \text{MinPts} &= 5 \\ \text{Eps} &= 1 \text{ cm} \end{aligned}$$

## Directly density-reachable

- An object  $p$  is directly density-reachable from object  $q$  if  $p$  is within the  $\varepsilon$ -neighborhood of  $q$  and  $q$  is a core object

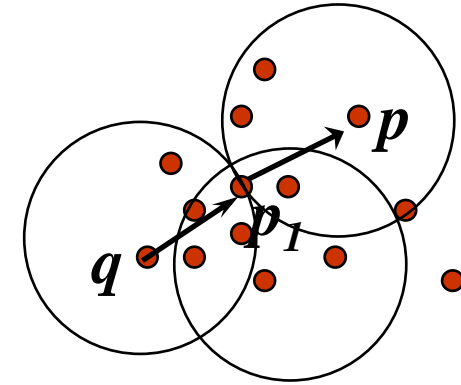


$$\begin{aligned} \text{MinPts} &= 5 \\ \text{Eps} &= 1 \text{ cm} \end{aligned}$$

# Density Based Clustering Ideas (2/2)

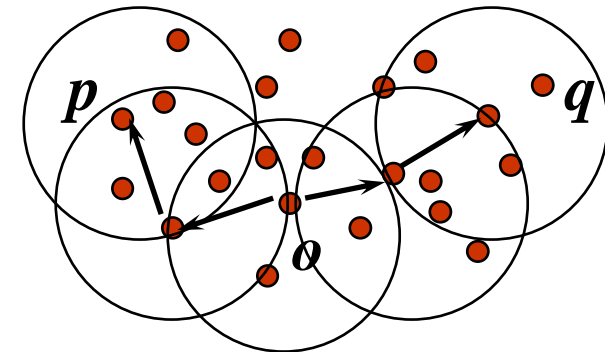
## Density-reachable:

- An object  $p$  is density-reachable from object  $q$  w.r.t.  $Eps, MinPts$  if there is a chain of points  $p_1, \dots, p_n$ , with  $p_1 = q$  and  $p_n = p$  such that  $p_{i+1}$  is directly density-reachable from  $p_i$



## Density-connected

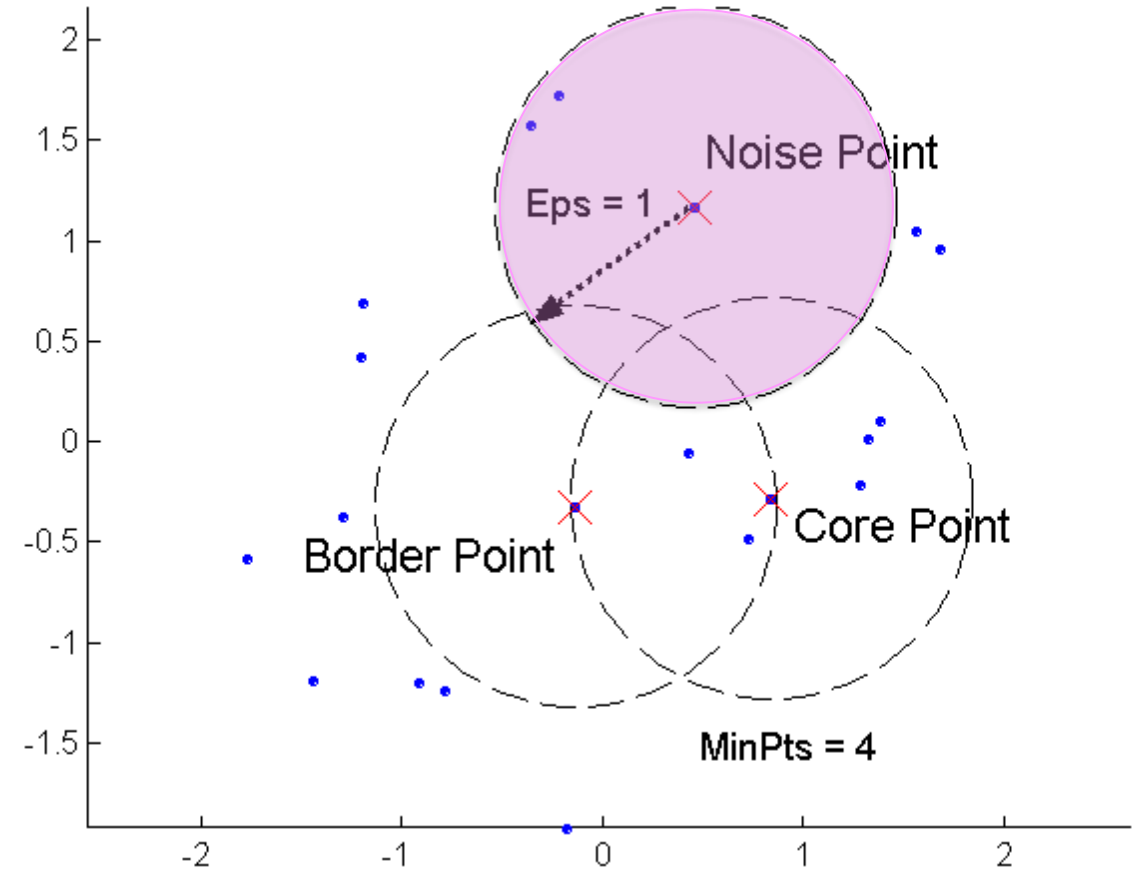
- A point  $p$  is density-connected to a point  $q$  w.r.t.  $Eps, MinPts$  if there is a point  $o$  such that both,  $p$  and  $q$  are density-reachable from  $o$  w.r.t.  $Eps$  and  $MinPts$ .



# DBSCAN Definitions

DBSCAN extract clusters as a set of density connected objects

- A **density-based cluster** is a set of density-connected objects that is maximal w.r.t. density-reachability
- A **border point** has fewer than  $\text{MinPts}$  within  $\text{Eps}$ , but it is in the neighborhood of a core point
- A **noise point** is any point that is not a core point or a border point



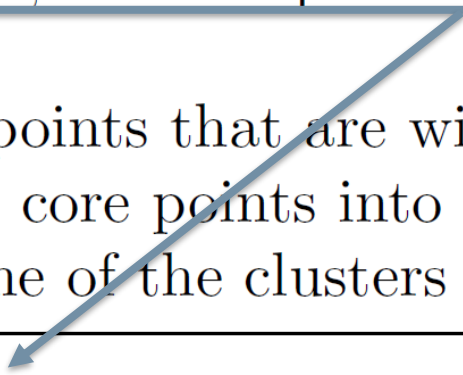
# DBSCAN Algorithm

---

## Algorithm 8.4 DBSCAN algorithm.

---

- 1: Label all points as core, border, or noise points.
  - 2: Eliminate 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.
- 

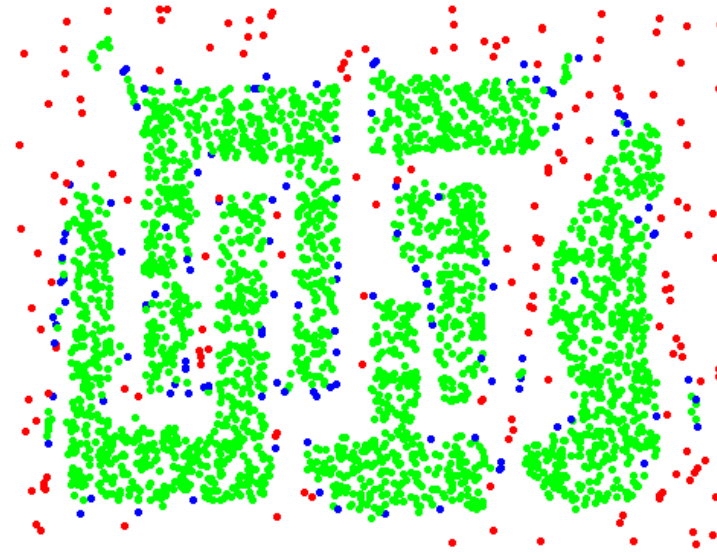
- 
1. Arbitrary select a point  $p$
  2. Retrieve all points density-reachable from  $p$  given  $Eps$  and  $MinPts$ .
    1. If  $p$  is a core point, a cluster is formed.
    2. If  $p$  is a border point, no points are density-reachable from  $p$  and DBSCAN visits the next point of the database
  3. Continue the process until all the points have been processed

# DBSCAN: Core, Border and Noise Points

Eps = 10, MinPts = 4



Original Points



Point types: core,  
border and noise

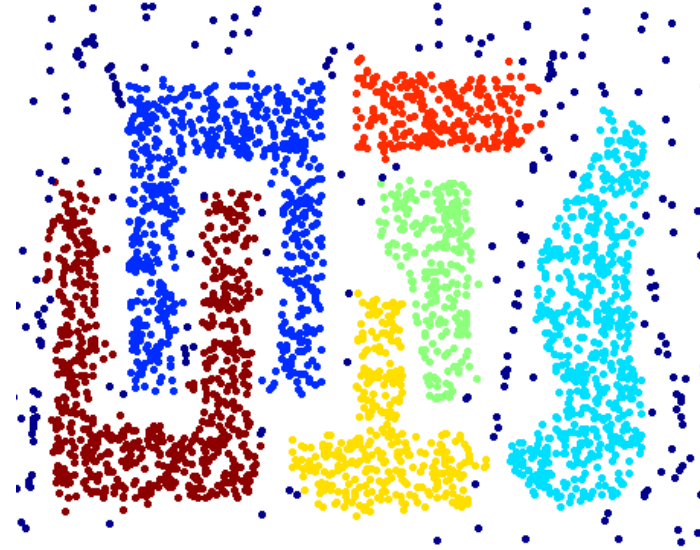
# DBSCAN: Clustering

*Can handle clusters of different shape*

*Very Robust to Noise*



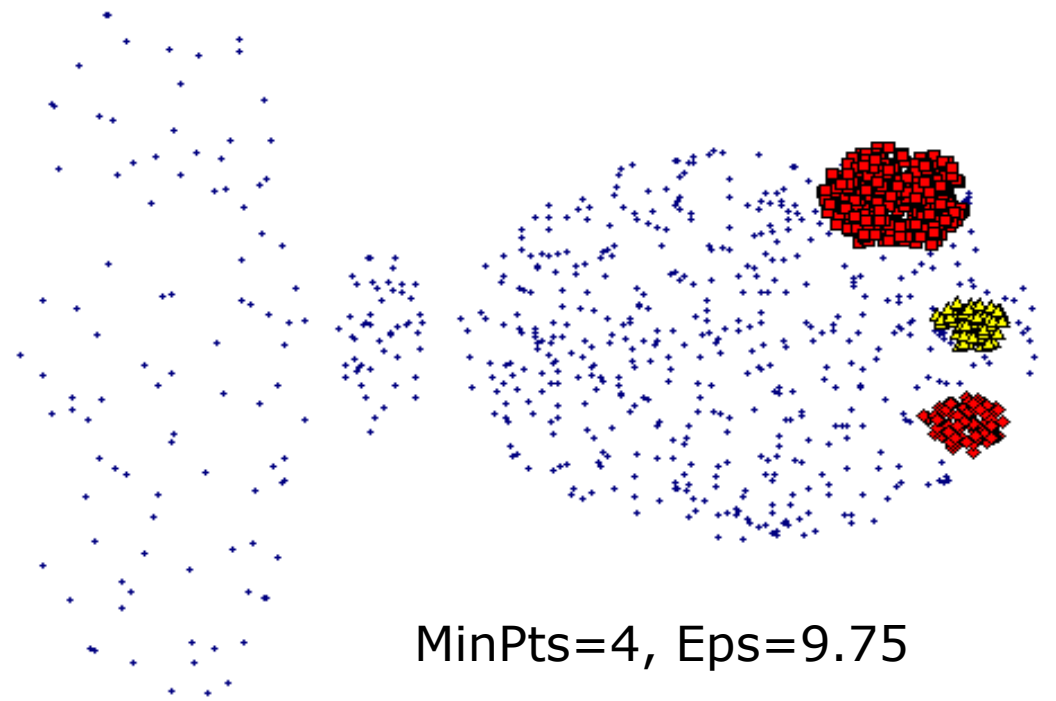
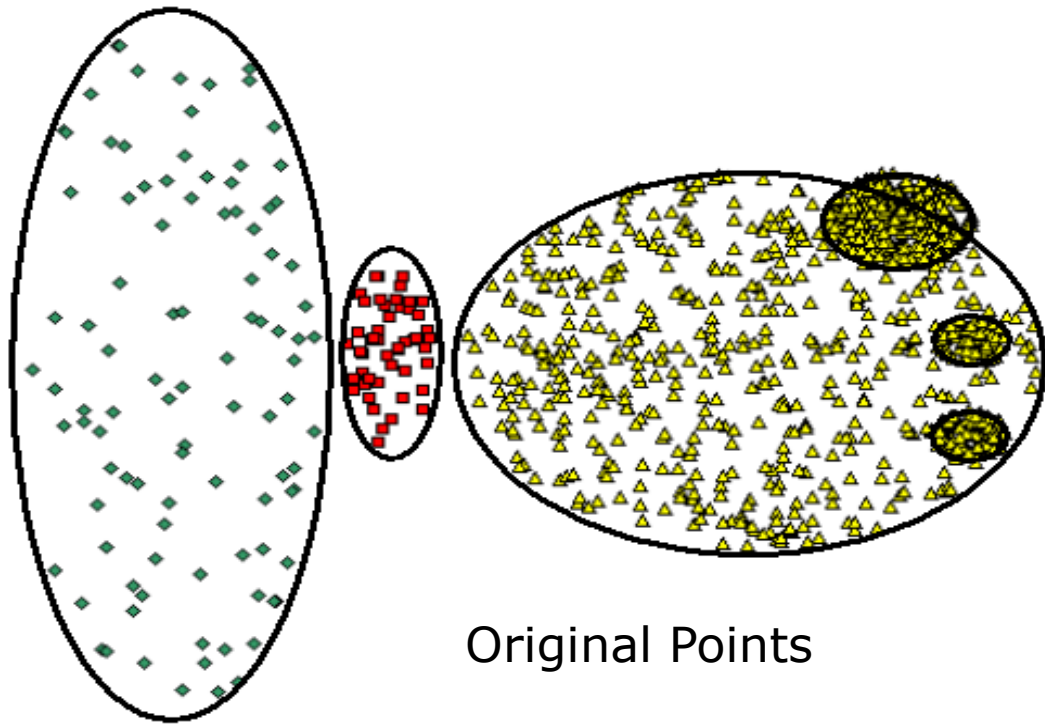
Original Points



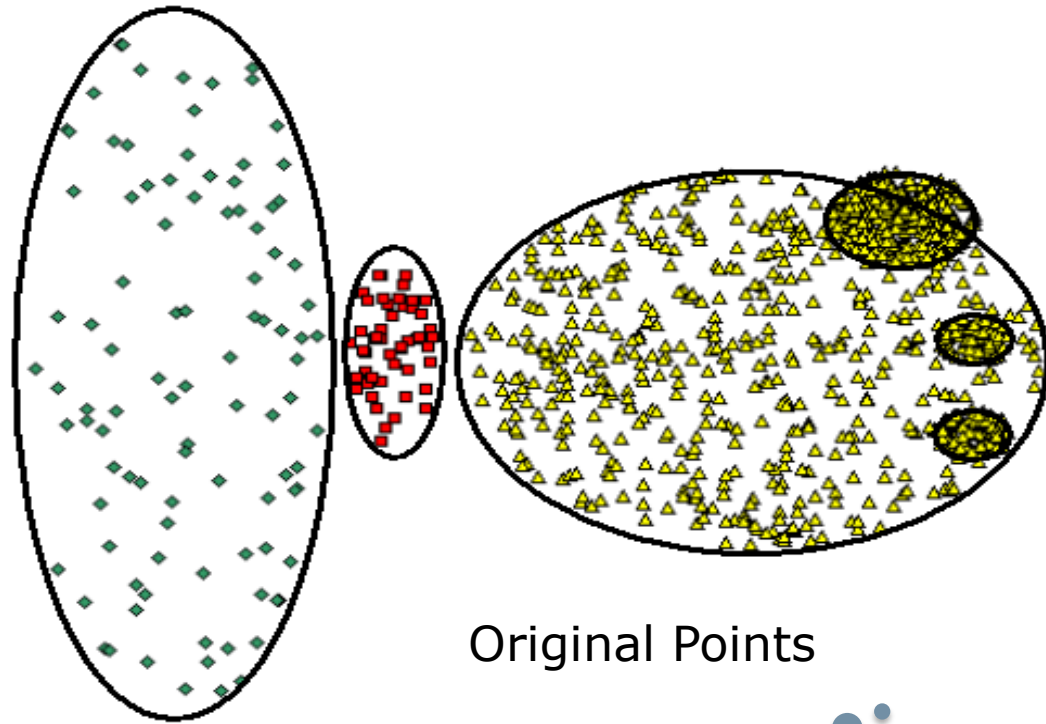
Clusters



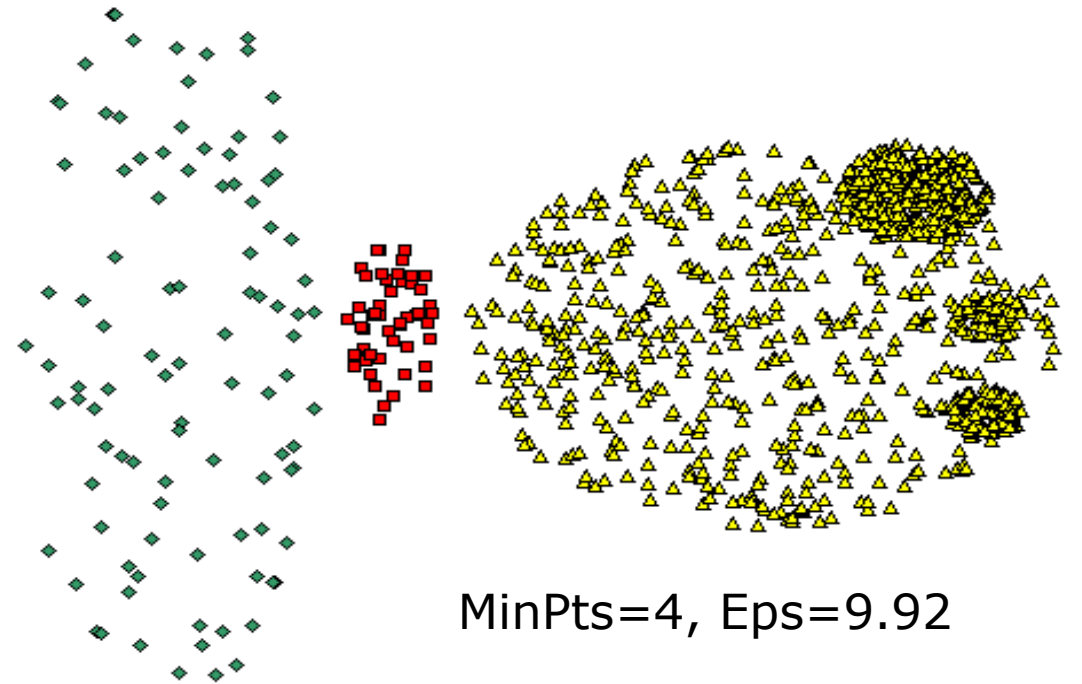
# DBSCAN: When it fails ...



# DBSCAN: When it fails ...



*Issues with varying densities*

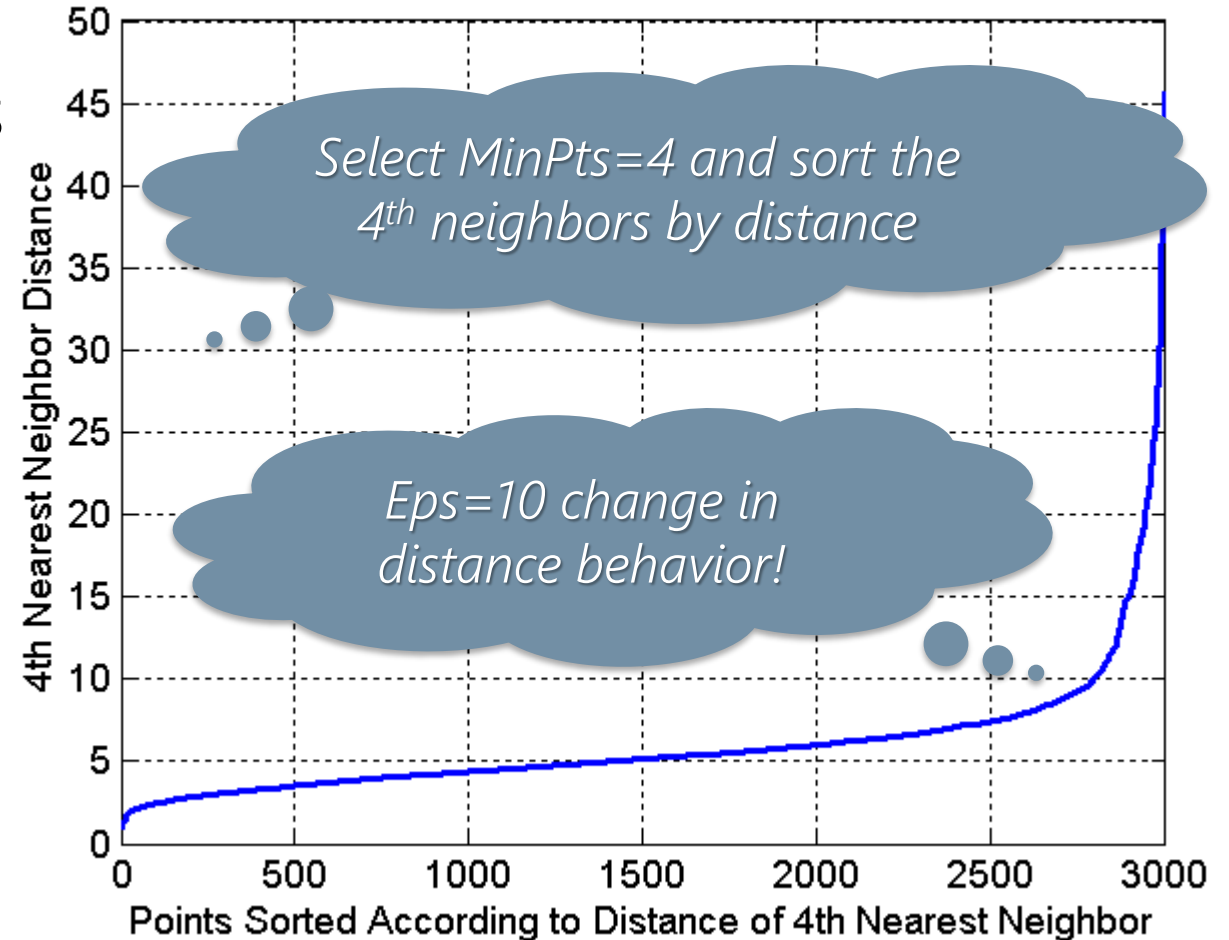


*Issues with high dimensional data*

# DBSCAN: Finding Eps and MinPts

Idea:  $k^{th}$  nearest neighbors of points within density-based clusters are at roughly the same distance

- Noise points have the  $k^{th}$  nearest neighbor at farther distance
- Select  $MinPts = k$  and plot sorted distance of every point to its  $k^{th}$  nearest neighbor



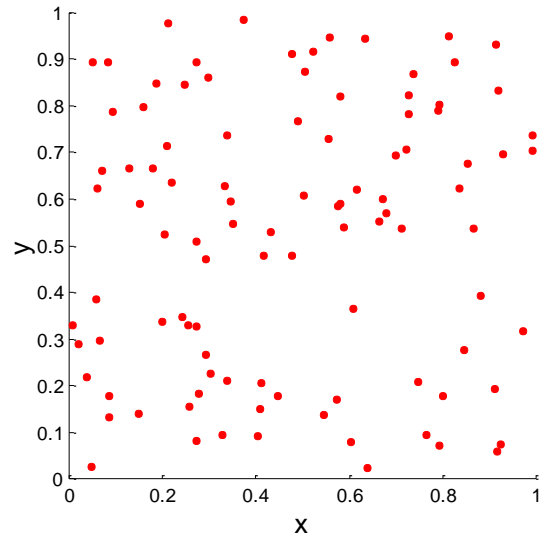
# How to evaluate clustering?

Clustering is an unsupervised learning method; how do we evaluate the “goodness” of the resulting clusters?

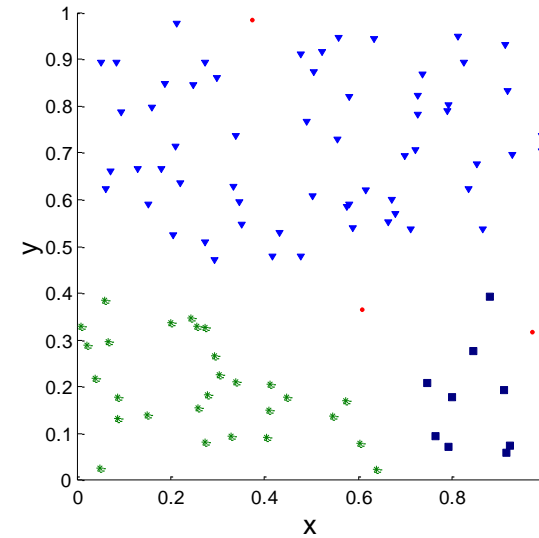
- To determine the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
- To compare the results of a cluster analysis to externally known results, e.g., to externally given class labels.
- To evaluate how well the results of a cluster analysis fit the data without reference to external information
- To compare the results of two different sets of cluster analyses
- To determine the ‘correct’ number of clusters.

# You can find clusters random data too ...

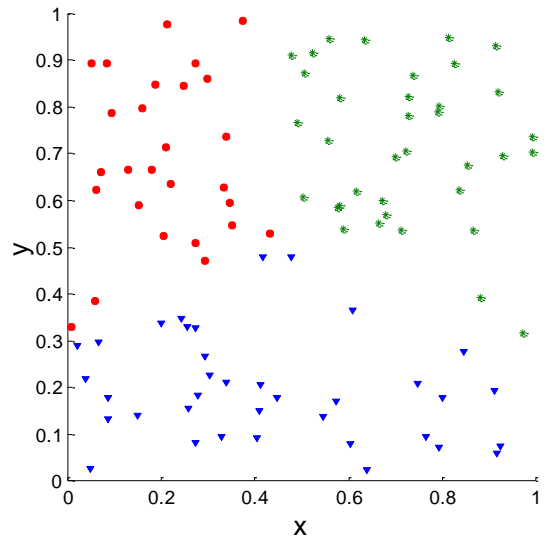
Random Points



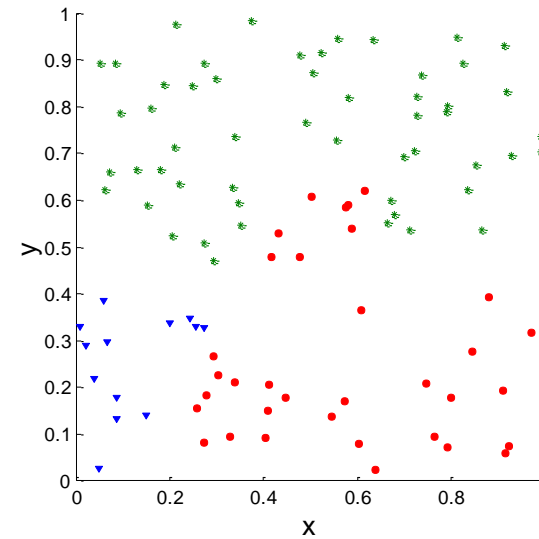
DBSCAN



K-means



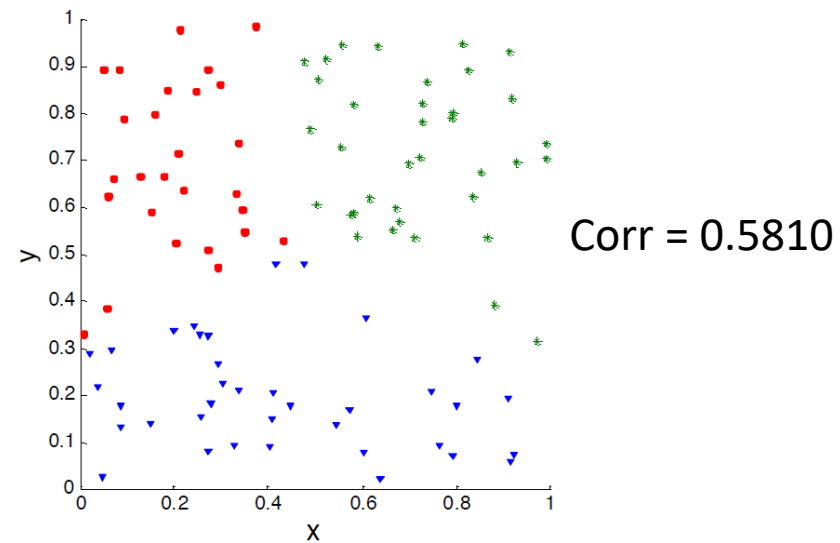
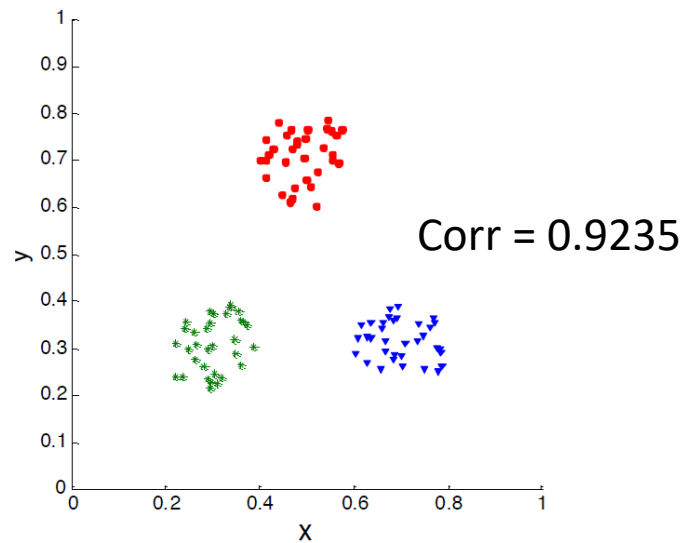
Complete Link



# Cluster validity via Correlation

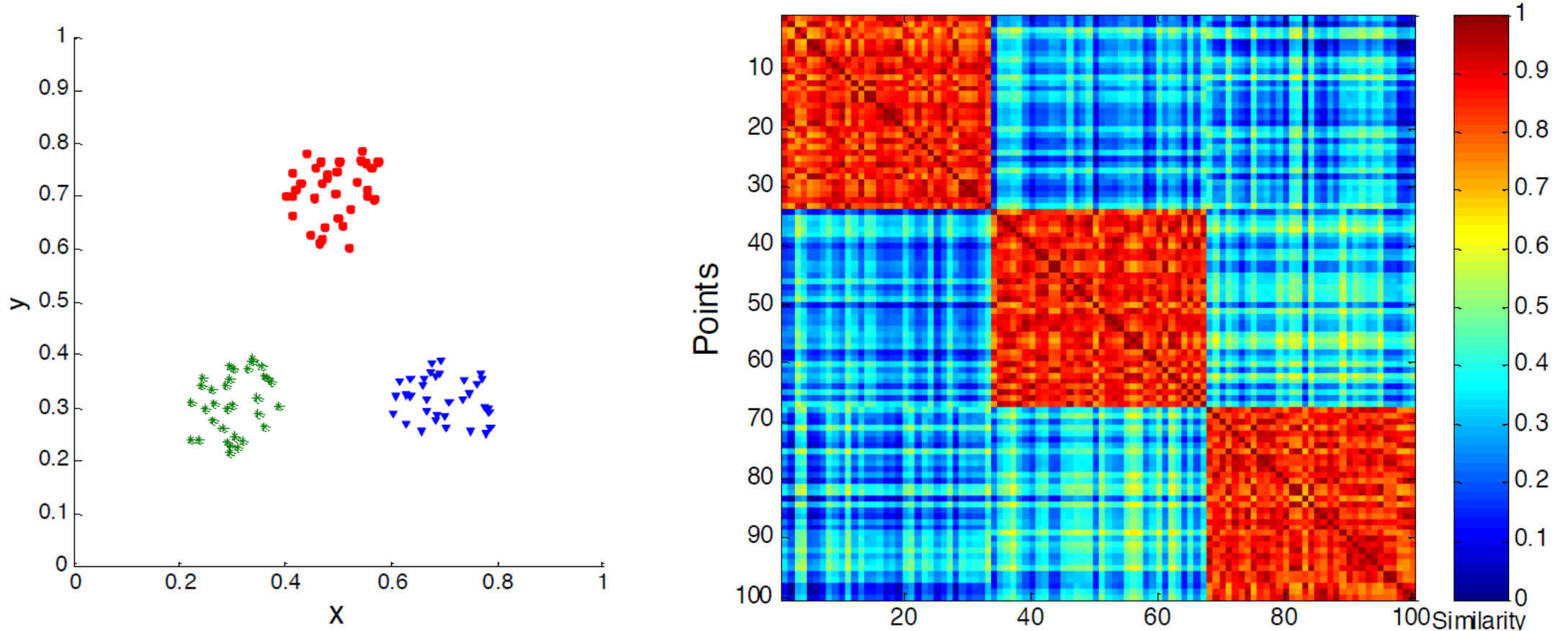
Compute the Correlation between the *Similarity Matrix* and the *Incidence Matrix*, which has a 1 if two points are in the same cluster 0 otherwise.

- Only the correlation between  $n(n-1) / 2$  entries needs to be calculated
- High correlation when points that belong to the same cluster are close
- Not a good measure for density-based clusters



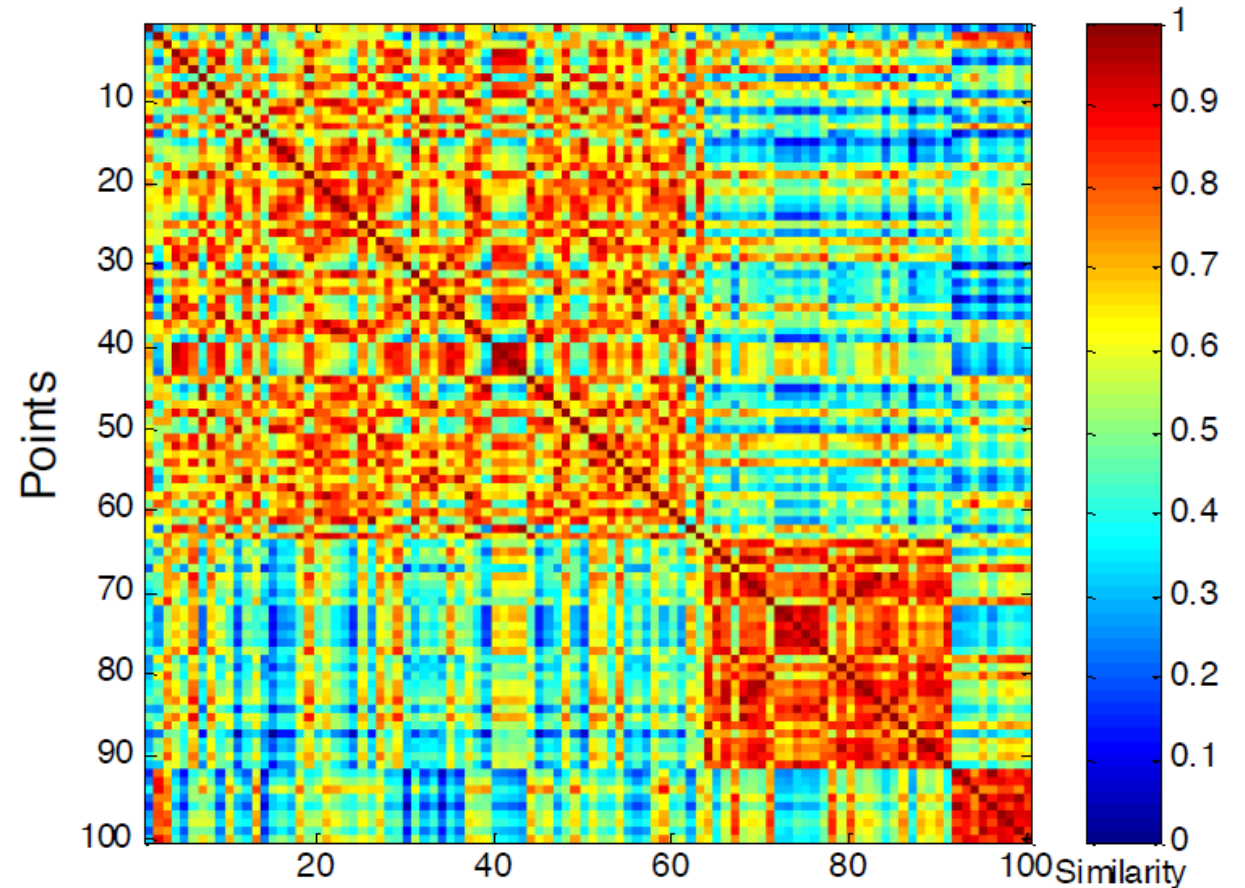
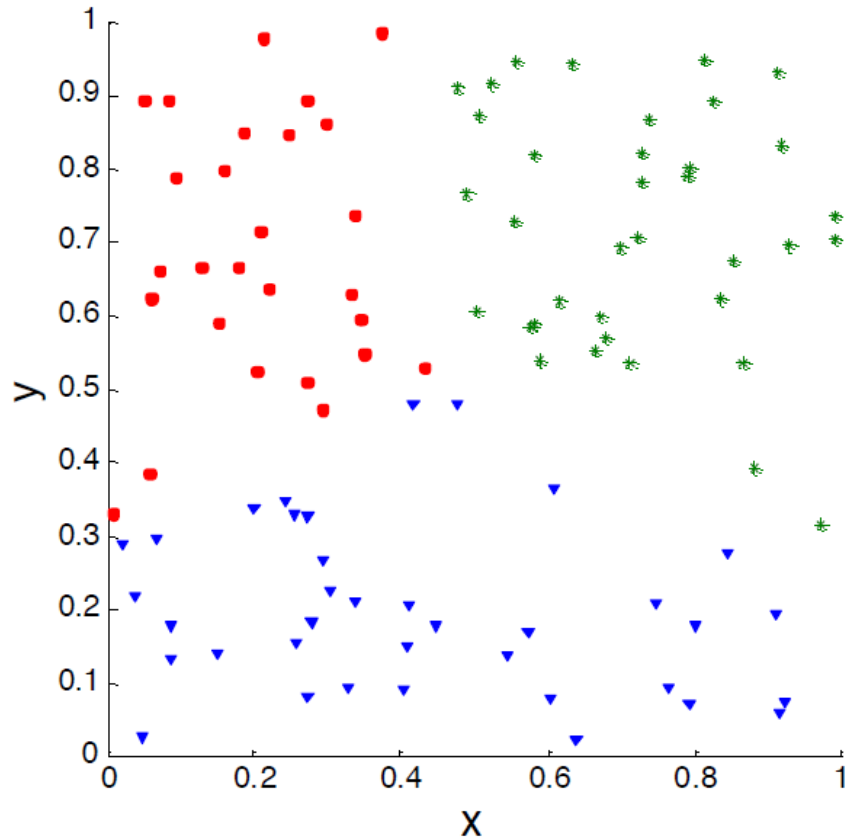
# Visual Inspection of Cluster Similarity Matrix

You can order the similarity matrix with respect to cluster labels



# Visual Inspection of Cluster Similarity Matrix

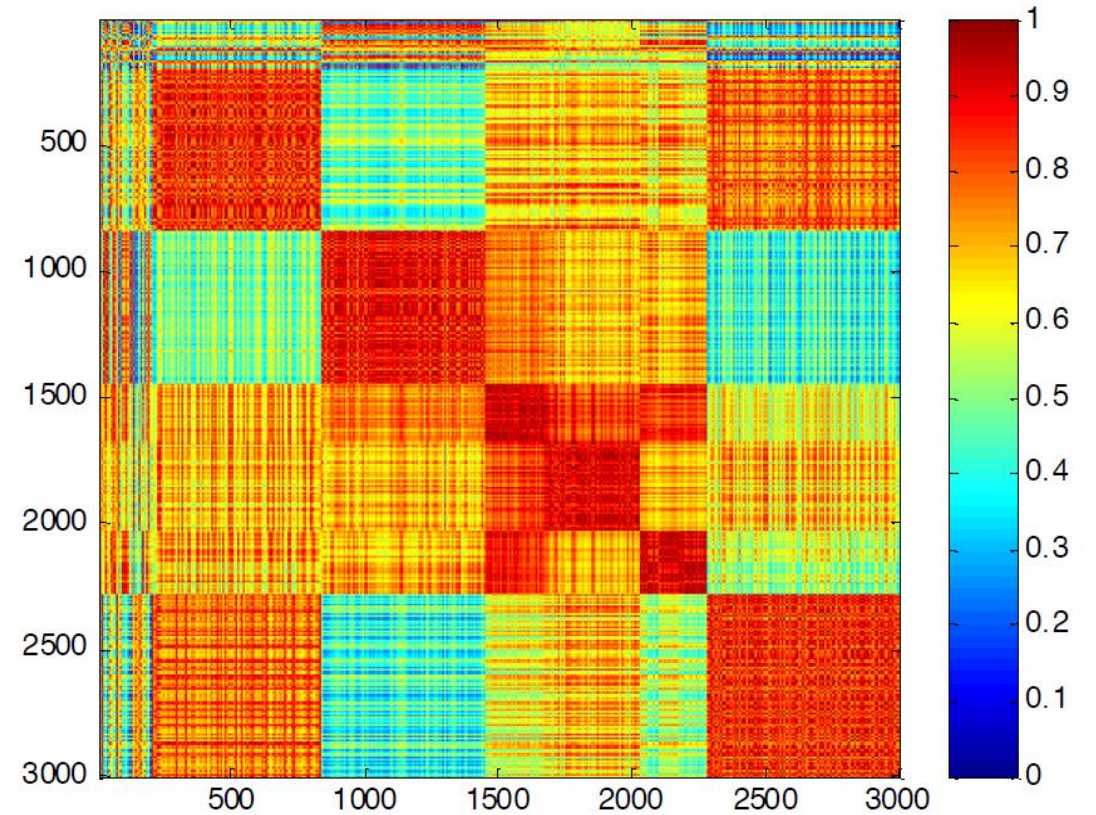
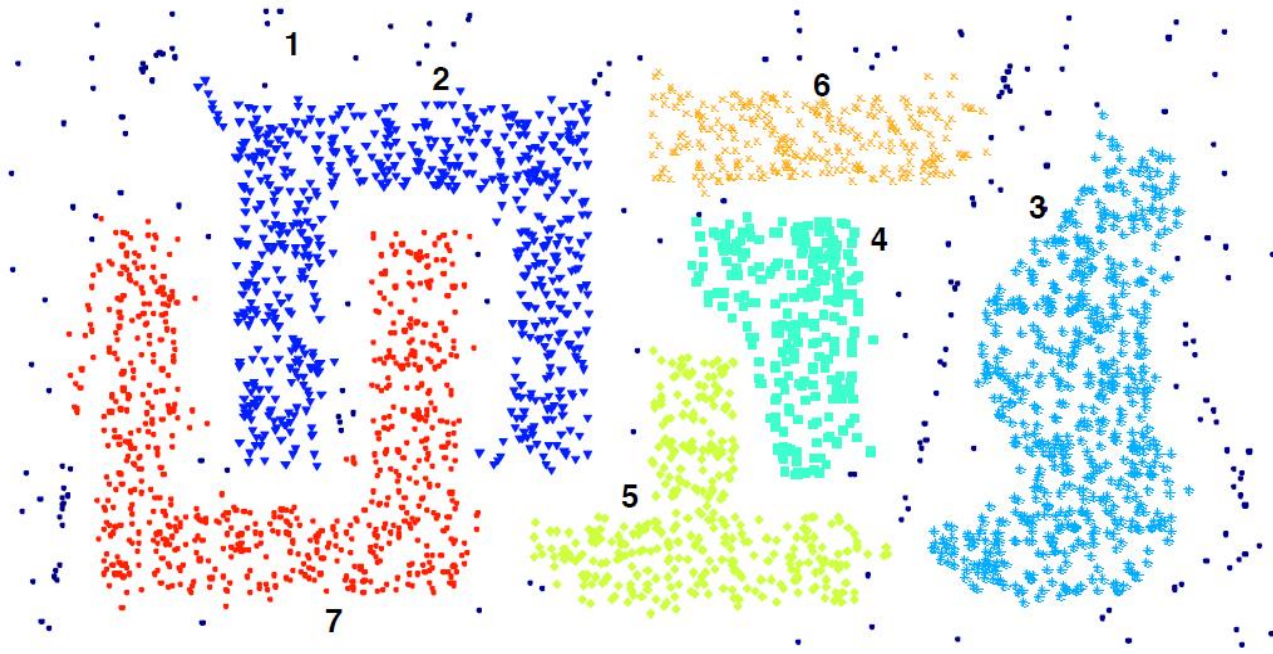
You can order the similarity matrix with respect to cluster labels





# Visual Inspection of Cluster Similarity Matrix

You can order the similarity matrix with respect to cluster labels



# Measuring Clusters Validity

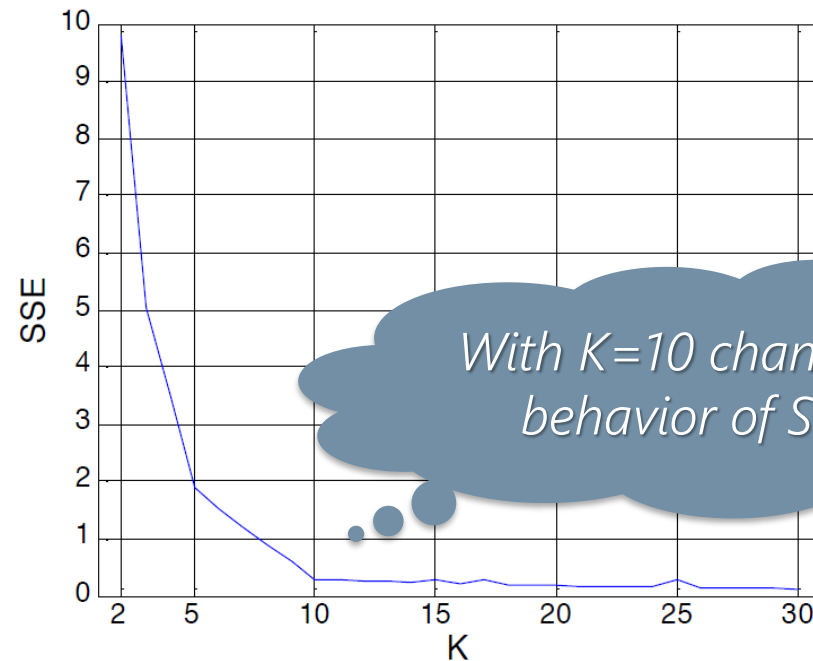
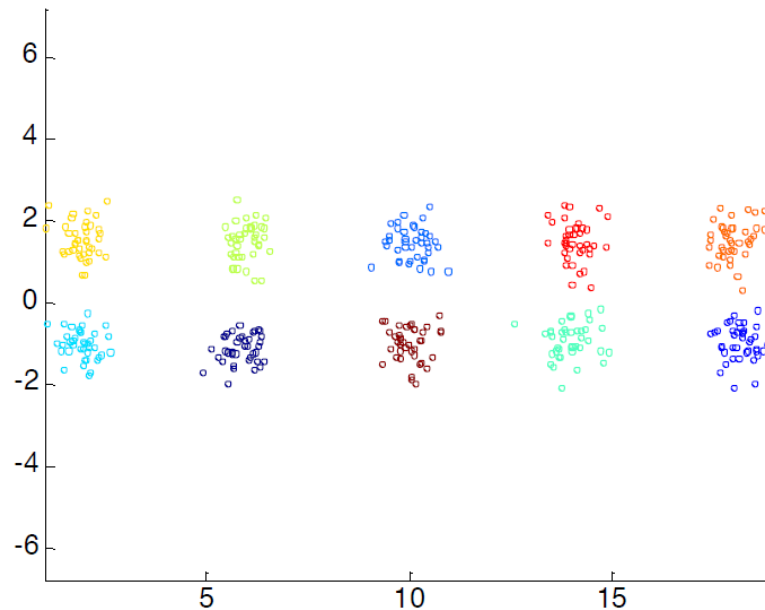
Metrics to evaluate clustering can be classified into

- **Internal Indexes:** Used to measure the goodness of a clustering structure without an external information, e.g., Sum of Squared Error (SSE)
- **External Indexes:** Used to measure the extent to which cluster labels match externally supplied class labels, e.g., Entropy.
- **Relative Indexes:** Used to compare two different algorithms or clusters

# Internal indexes: SSE

The Sum of Squared Errors (w.r.t. the centroid) can be used to measure the goodness of a clustering structure without external information

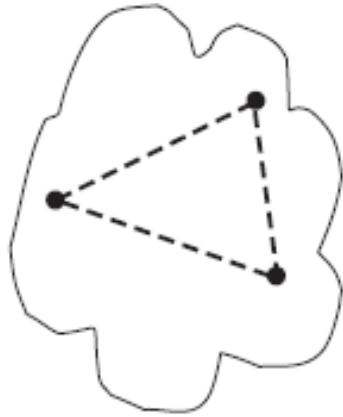
- Good for comparing two clusterings or two clusters (average SSE)
- Can also be used to estimate (*elbow method*) the number of clusters



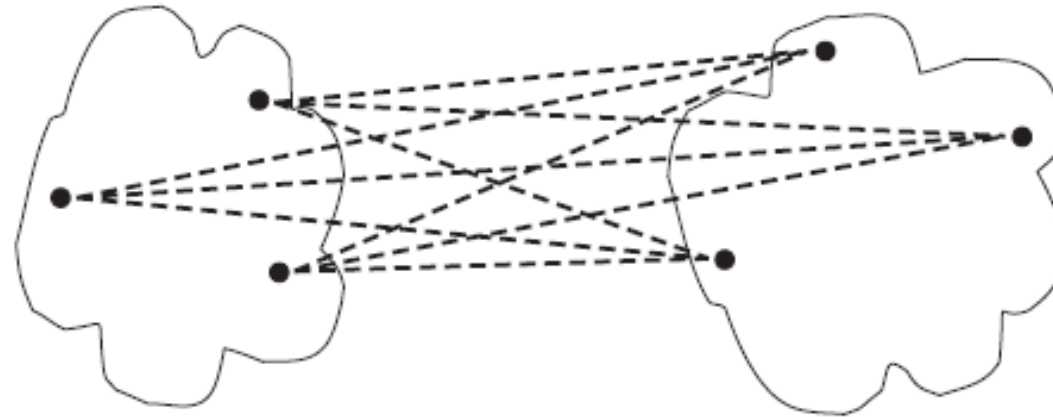
# Internal Indexes: Cohesion and Separation

**Cohesion:** Measures the affinity among all cluster objects

**Separation:** Measures how well-separated a cluster is from the others



(a) Cohesion.



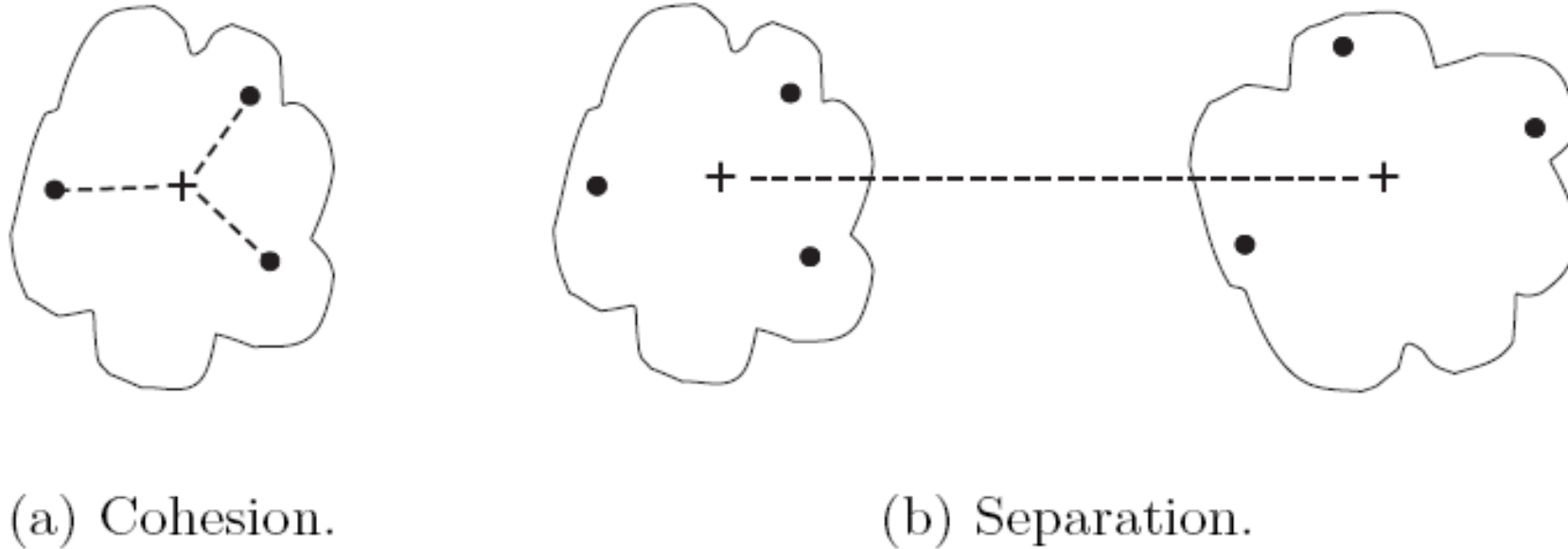
(b) Separation.

**Figure 8.27.** Graph-based view of cluster cohesion and separation.

# Internal Indexes: Cohesion and Separation

**Cohesion:** Measures the affinity among all cluster objects

**Separation:** Measures how well-separated a cluster is from the others



**Figure 8.28.** Prototype-based view of cluster cohesion and separation.

# Internal Indexes: Cohesion and Separation

**Cohesion:** Measures the affinity among all cluster objects

**Separation:** Measures how well-separated a cluster is from the others

Cohesion is measured by the within cluster SSE (WSS=SSE)

$$WSS = \sum_k \sum_{x_n \in C_k} (x_n - c_k)^2$$

Separation is measured by the between cluster SSE where  $|C_k|$  is the size of cluster  $k$  and  $\bar{c}$  is the mean of all centroids

$$BSS = \sum_k |C_k| (\bar{c} - c_k)^2$$

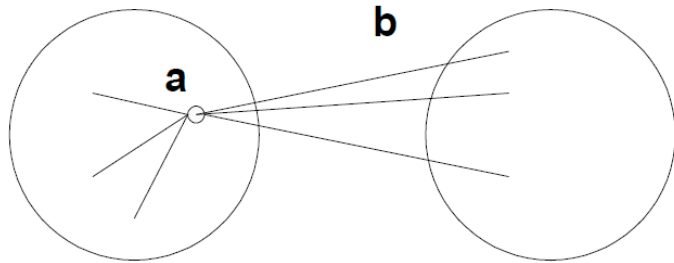
*The sum WSS+BSS is constant*

# Internal Indexes: Silhouette Coefficient

*Silhouette Coefficient* combine ideas of both cohesion and separation

For an individual point  $x_n$

- Calculate  $a$  = average distance of  $x_n$  to the points in its cluster
- Calculate  $b$  = min (average distance of  $x_n$  to points in another cluster)



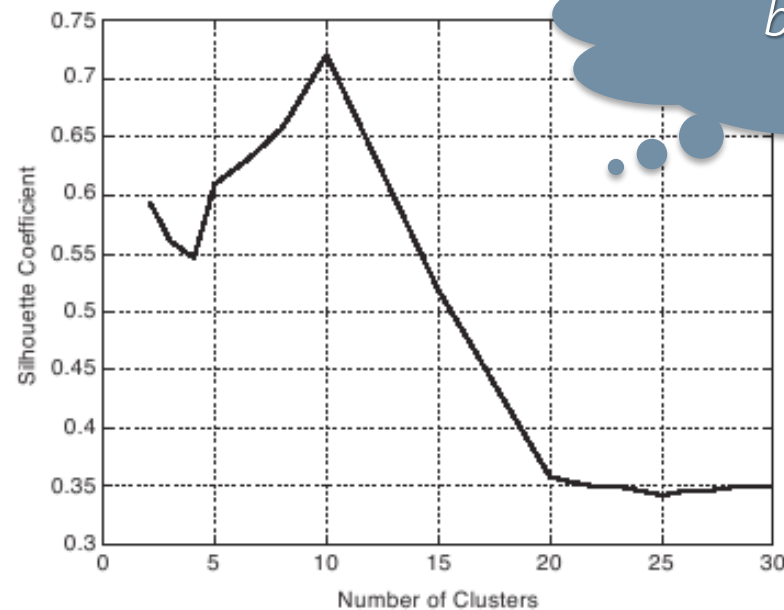
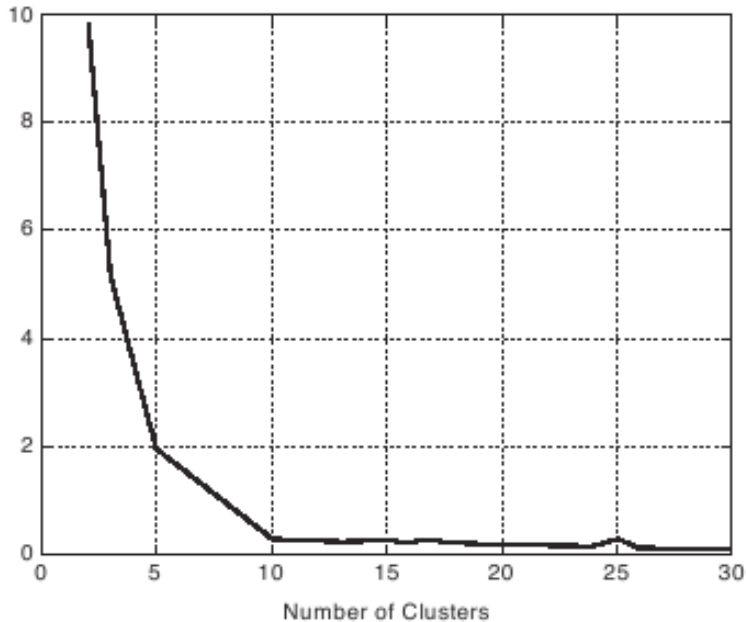
$$s(x_n) = 1 - \frac{a}{b} \text{ if } a < b \text{ (or } \frac{b}{a} - 1 \text{ if } a \geq b)$$

Typically between 0 and 1, the closer to 1 the better

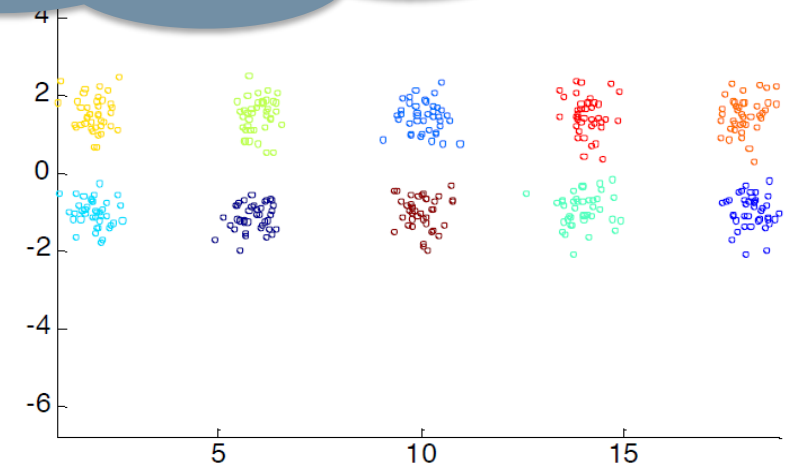
Can compute the *Average Silhouette* for a cluster or clustering algorithm

# Internal Indexes: Silhouette Coefficient

*Silhouette Coefficient* combine ideas of both cohesion and separation and it can be used to select the number of clusters



With  $K=10$  change in behavior of Average Silhouette





# Measuring Clusters Validity

Metrics to evaluate clustering can be classified into

- **Internal Indexes:** Used to measure the goodness of a clustering structure without an external information, e.g., Sum of Squared Error (SSE)
- **External Indexes:** Used to measure the extent to which cluster labels match externally supplied class labels, e.g., Entropy.
- **Relative Indexes:** Used to compare two different algorithms or clusters

# External Indexes: Precision and Recall (and Purity)

For each single cluster  $k$  w.r.t. class  $j$  use precision  $p_{kj}$  and recall  $r_{kj}$

- $N$  is the total number of elements to be clustered
- $N_k$  is the number of elements of cluster  $k$
- $N_j$  is the number of elements of class  $j$
- $N_{kj}$  is the number of elements of cluster  $k$  also belonging to class  $j$

$$\text{precision}(k, j) = p_{kj} = \frac{N_{kj}}{N_k} \quad \text{recall}(k, j) = r_{kj} = \frac{N_{kj}}{N_j}$$

$$F = \frac{2}{1/p + 1/r} = \frac{2pr}{p + r}$$

Purity is similar to precision; the purity of a cluster is  $p_k = \max_j p_{kj}$

The purity of a clustering is defined as  $p = \sum_{k=1}^K \frac{N_k}{N} p_k$

# External Indexes: Entropy

Degree to which each cluster consists of elements of the same class

- Let  $p_{kj} = N_{kj}/N_k$  the probability that a member of cluster  $k$  belong to class  $j$
- Let  $L$  the number of classes we define cluster entropy as

$$e_k = - \sum_{j=1}^L p_{kj} \log_2 p_{kj}$$

*0 if all members of cluster  $k$  belong to a single class*

- The total entropy of a clustering is a sum weighted by the size of each cluster

$$e = - \sum_{k=1}^K \frac{N_k}{N} e_k$$

# External Indexes: Jaccard Similarity

Ideal cluster similarity matrix ( $n \times n$ ) has entries equal to 1 if the two objects belong to the same cluster and 0 otherwise

Ideal class similarity matrix ( $n \times n$ ) has entries equal to 1 if the two objects belong to the same class and 0 otherwise

We can use the Jaccard similarity between binary vectors

- $f_{00}$  number of object pairs having different class and different clusters
- $f_{01}$  number of object pairs having different class and same clusters
- $f_{10}$  number of object pairs having same class and different clusters
- $f_{11}$  number of object pairs having same class and same clusters

$$Jaccard = \frac{f_{11}}{f_{01} + f_{10} + f_{11}}$$

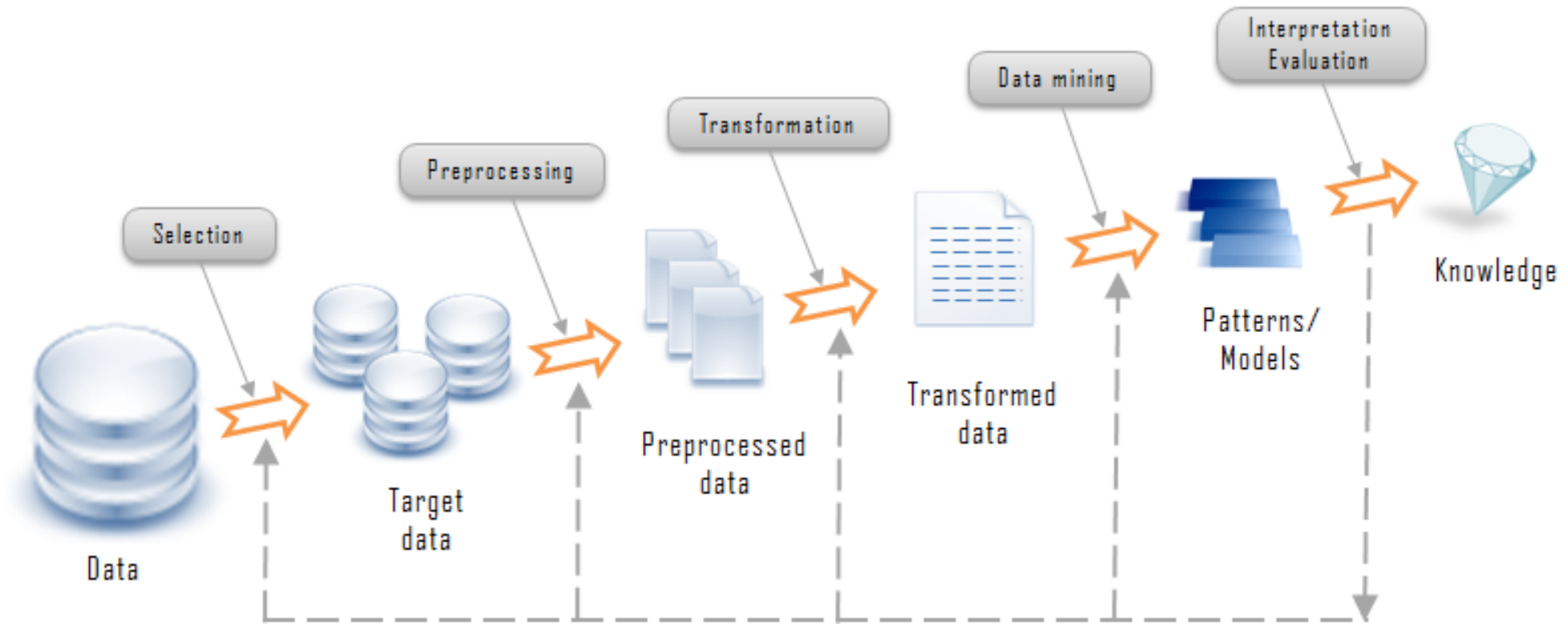
# Clustering evaluation ...

*"The validation of clustering structures is the most difficult and frustrating part of cluster analysis. Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage."*

Algorithms for Clustering Data, Jain and Dubes

*Resort to the domain expert  
or plan for validation  
experiments!*

# Behind the curtain (4/4)



## Validation / Interpretation

- How good are the results? Can we improve on those?
- Are the patterns we discovered sound? According to what criteria?
- Are the criteria sound? Can we explain the result?