



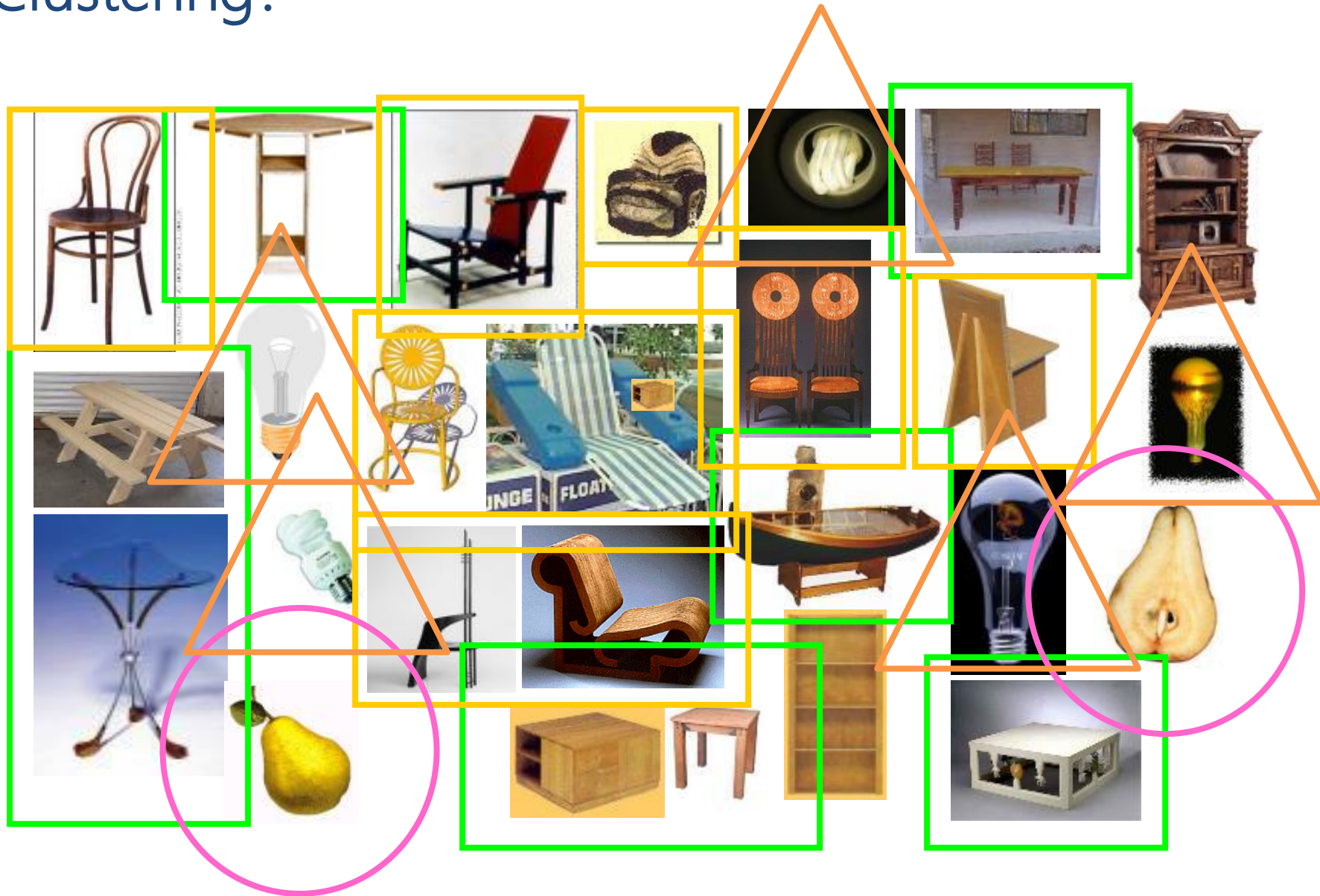
POLITECNICO
MILANO 1863

Data Analysis for Smart Agriculture

- Clustering -

Prof. Matteo Matteucci – matteo.matteucci@polimi.it

What is Clustering?



Clustering Definitions

"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

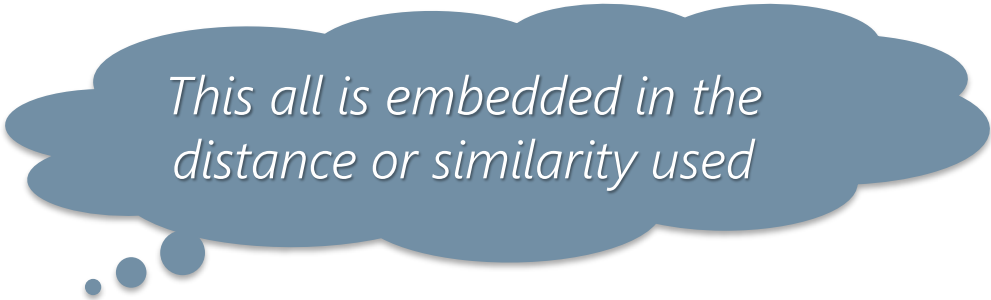
Brief insights about Clustering

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)

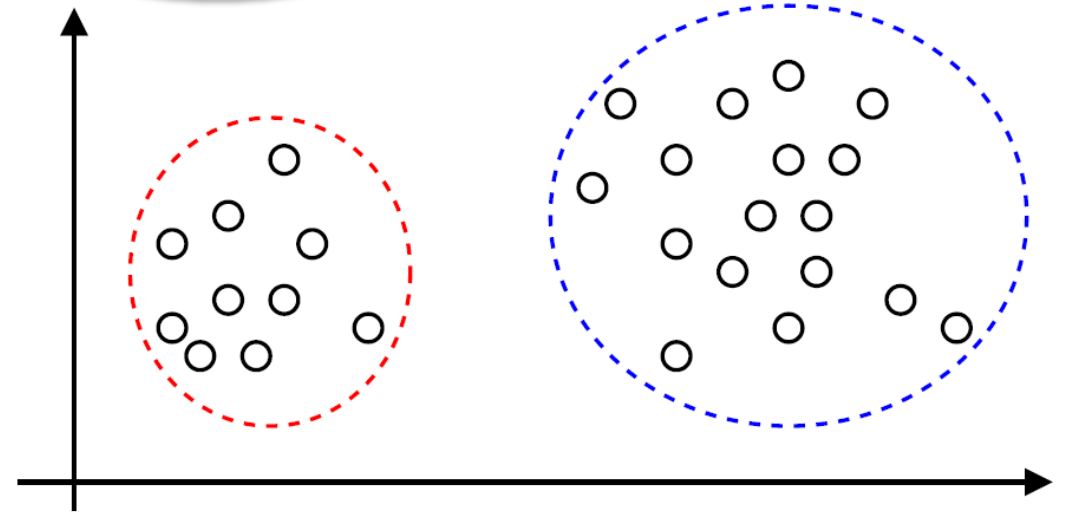
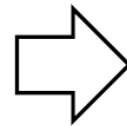
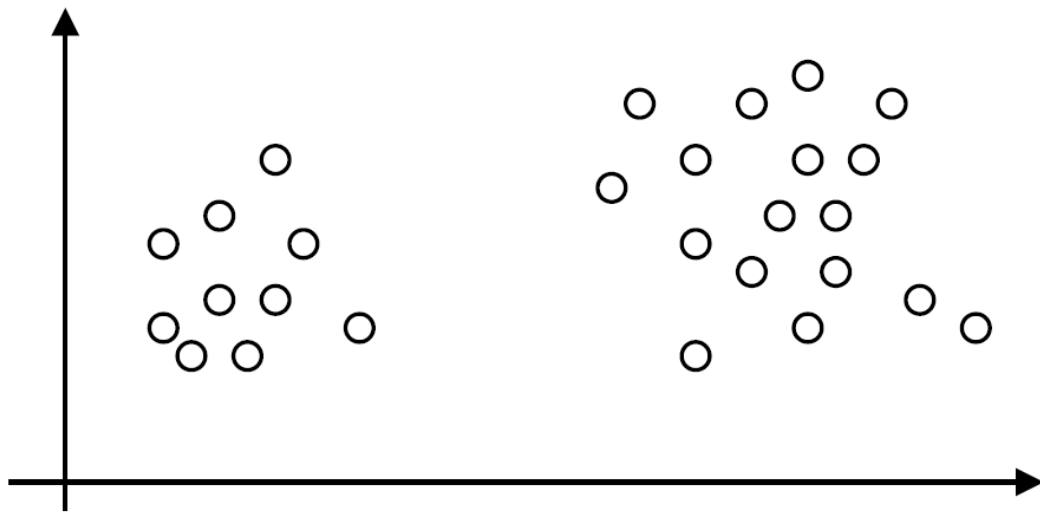
Clustering ... what are we looking for?

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

*Cosine similarity,
Pearson correlation,
Levenstain distance ...*

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

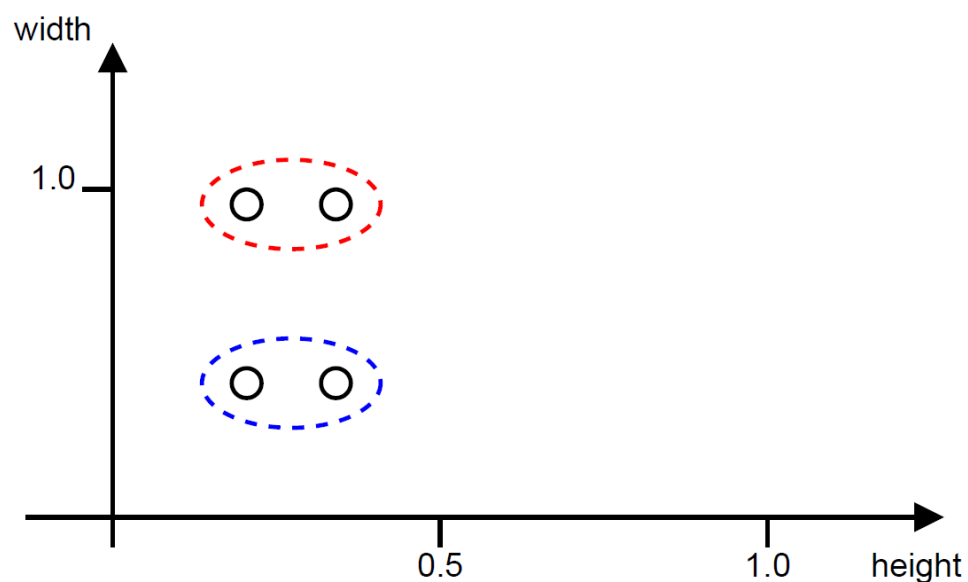
*Attributes might be
weighted too ...*

The clustering result depends on a similarity measure.

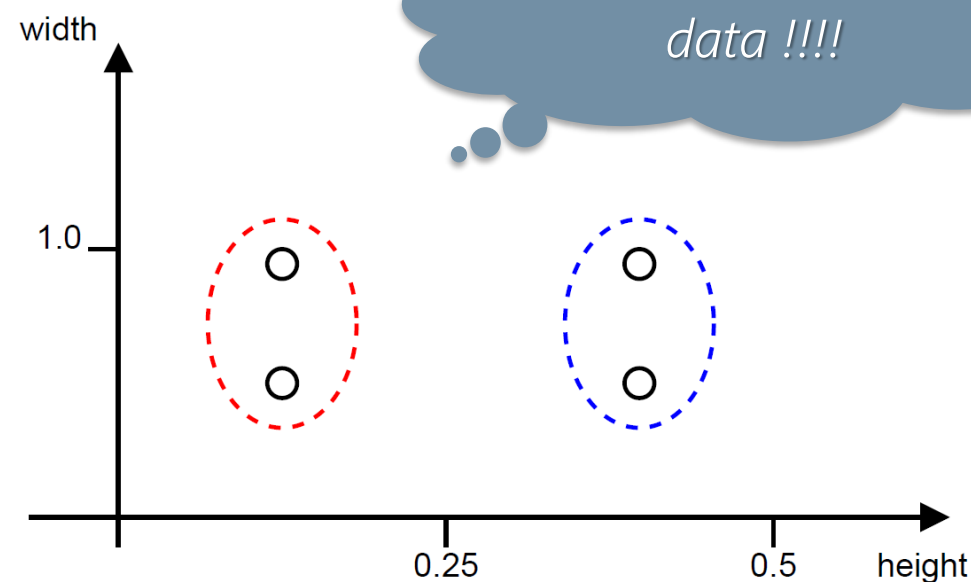
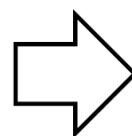
- One numeric attribute A -> $\text{Distance}(X,Y) = A(X) - A(Y)$
- Several numeric attributes -> $\text{Distance}(X,Y) = \text{Euclidean distance } \|X - Y\|_2$
- Nominal attributes -> $\text{Hamming distance } (X,Y) = \text{if different } 1, \text{ else } 0$
- ...

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.



Before Scaling



After Scaling

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 insurance, 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/>)



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:

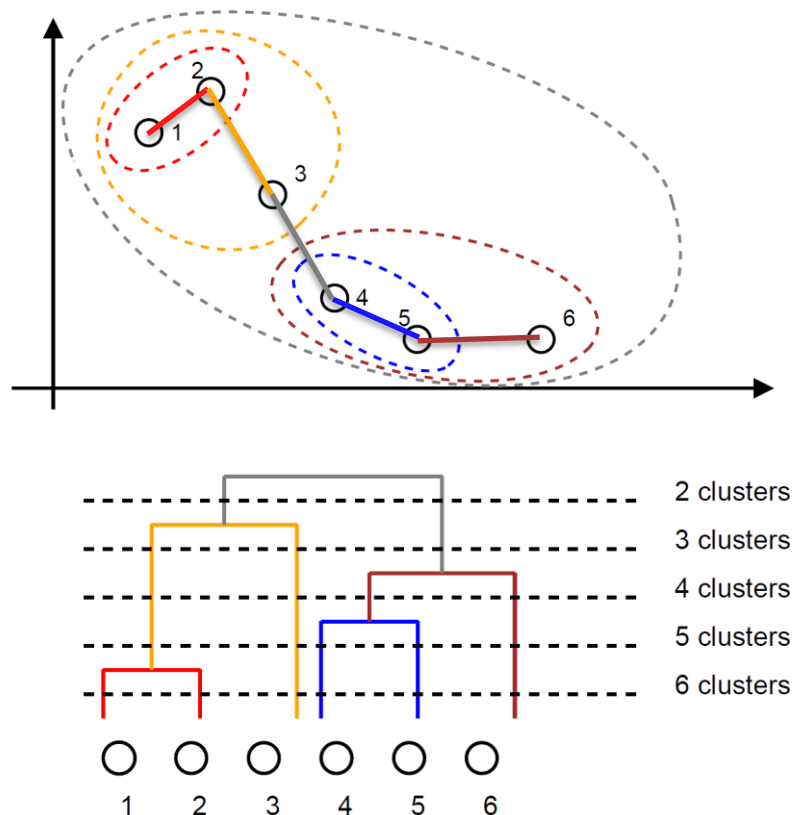
- *Exclusive Clusters*: examples belong to only one cluster
- *Overlapping Clusters*: examples may belong to more clusters
- *Probabilistic Clusters*: examples belong to one cluster with certain probability

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):

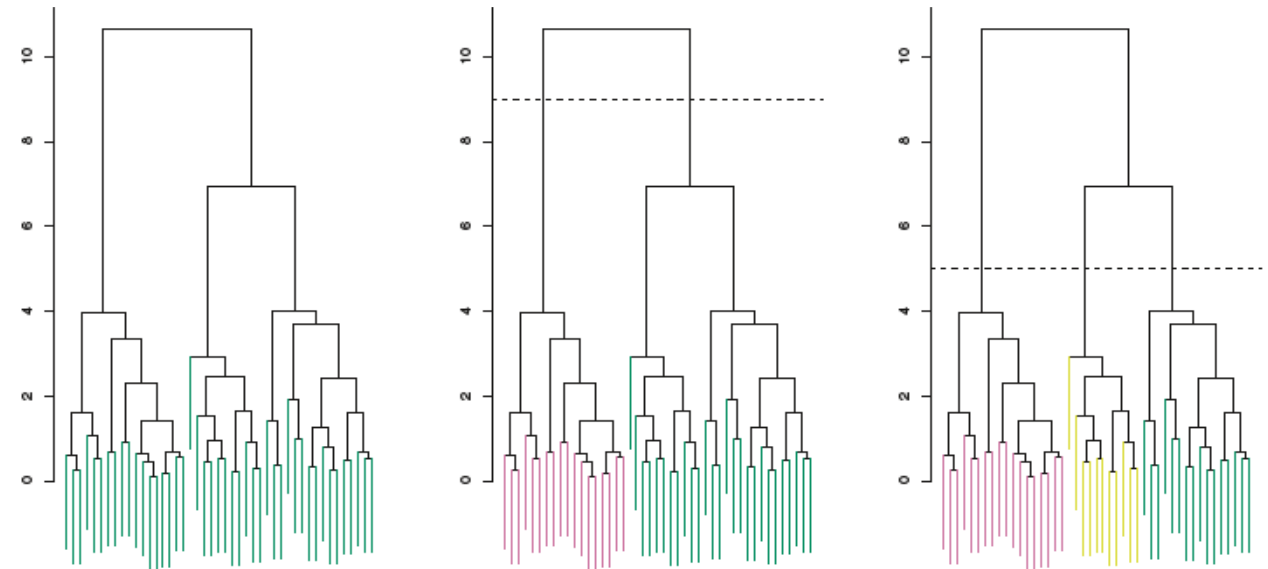
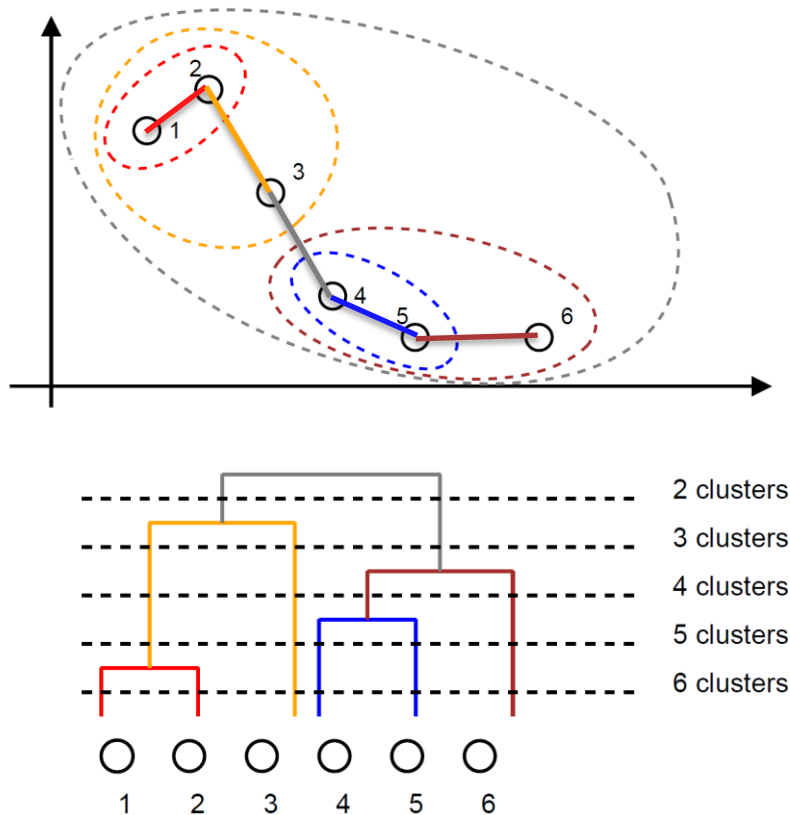
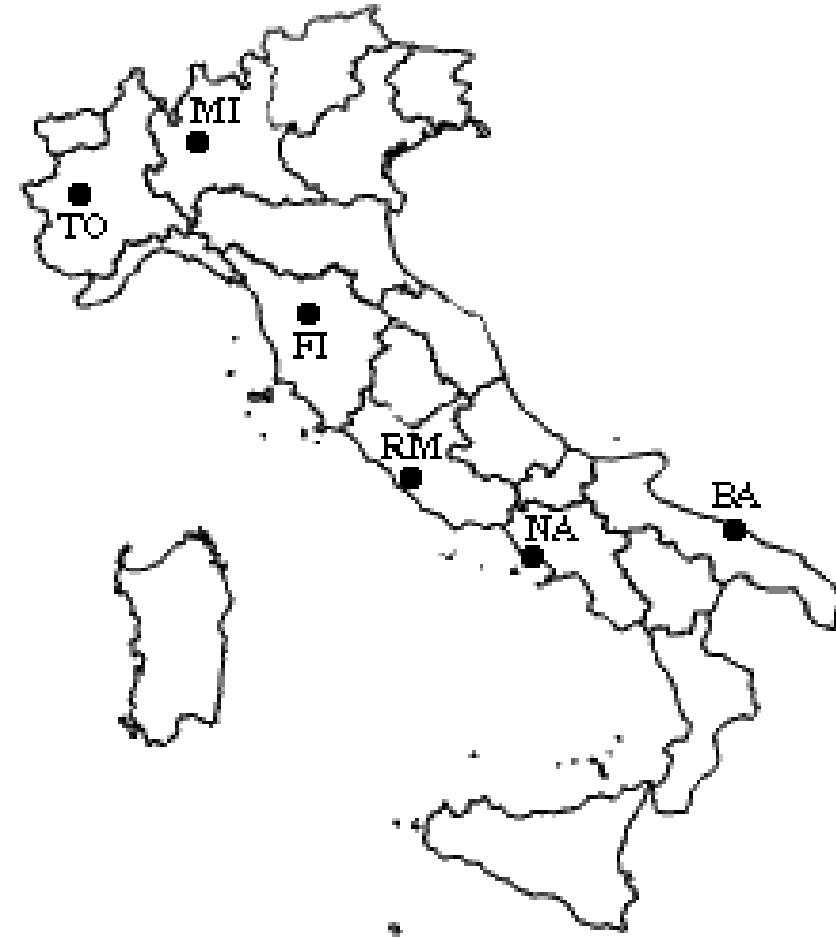


FIGURE 10.9. Left: dendrogram obtained from hierarchically clustering the data from Figure 10.8 with complete linkage and Euclidean distance. Center: the dendrogram from the left-hand panel, cut at a height of nine (indicated by the dashed line). This cut results in two distinct clusters, shown in different 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.

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



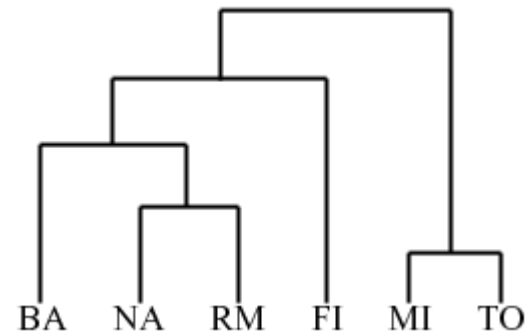
Hierarchical clustering example

	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



Hierarchical clustering example

<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.



K-means algorithm

One of the simplest unsupervised learning algorithms

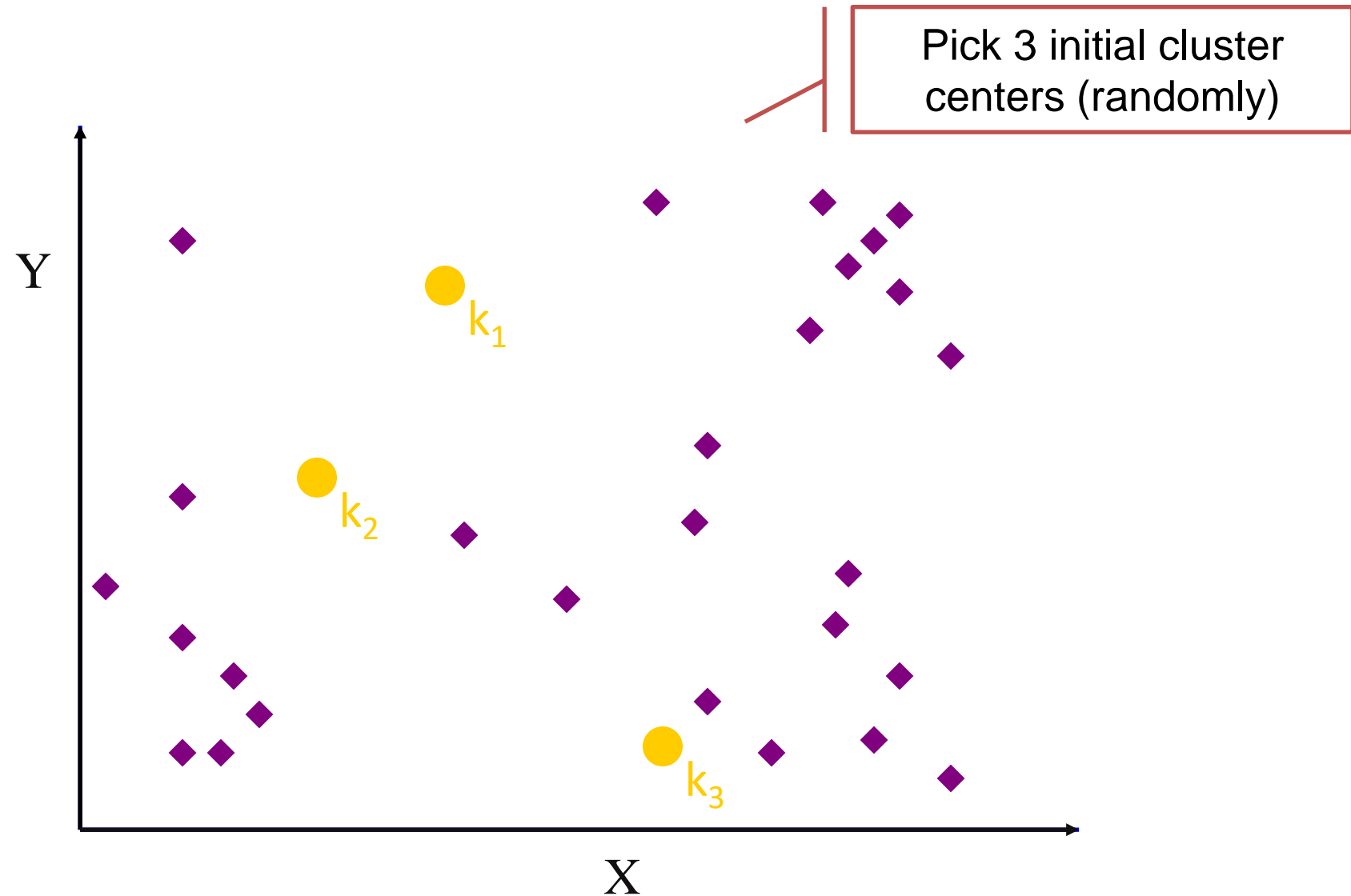
- Assumes an Euclidean space
- Number of clusters K is known/fixed a priori

*Random centroids
initialization*

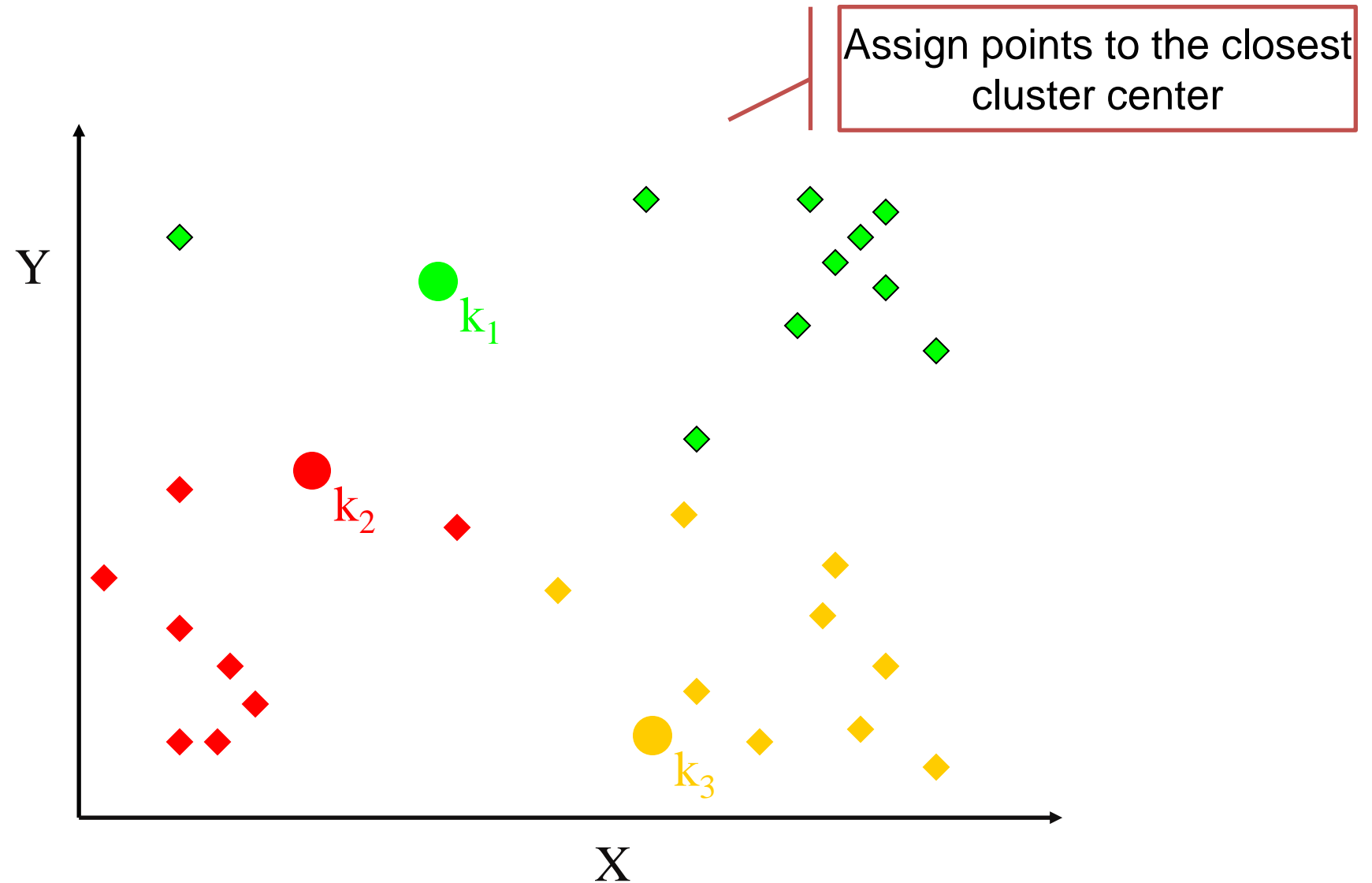
Algorithm 10.1 *K-Means Clustering*

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).

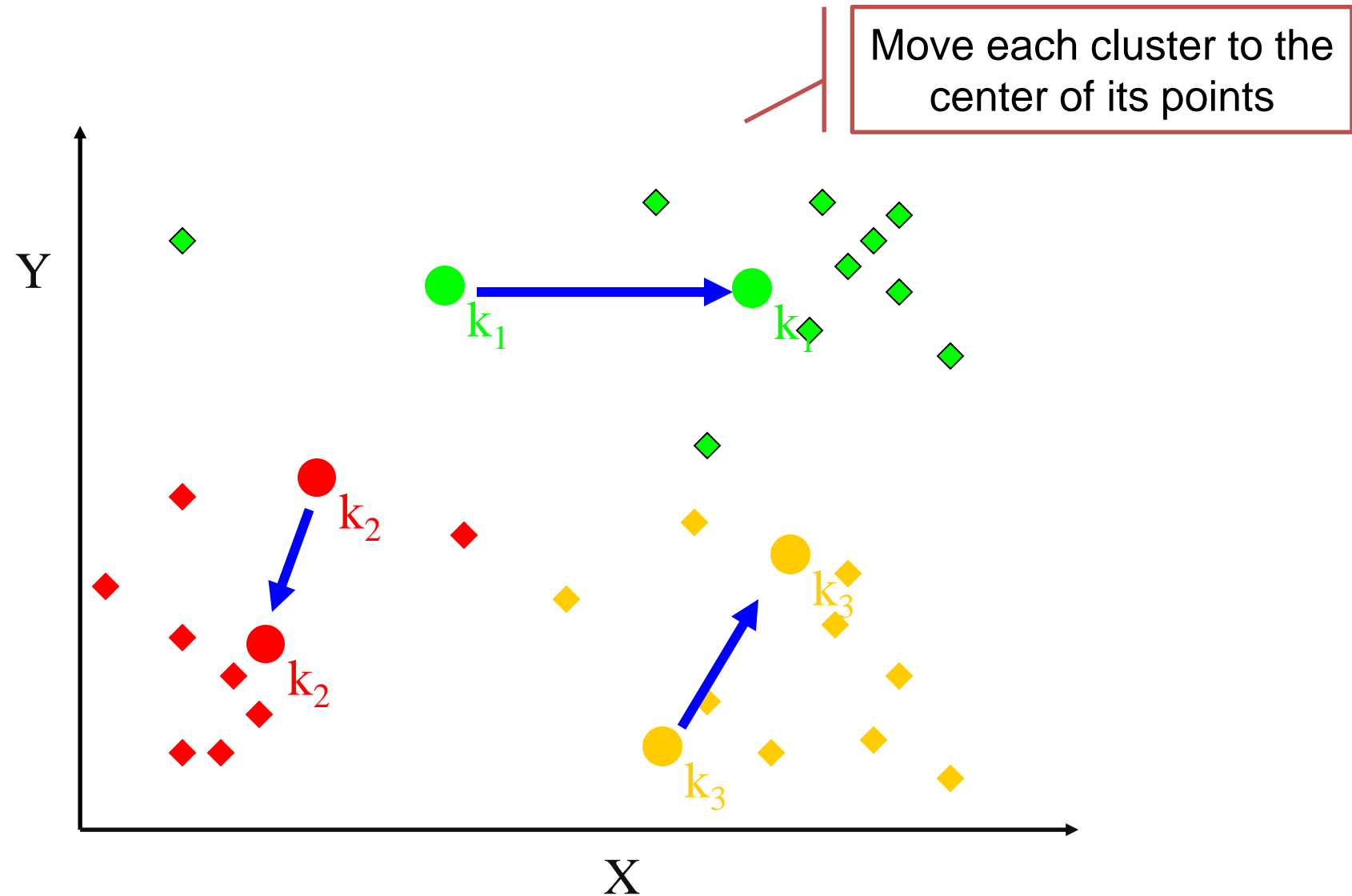
K-means example (1)



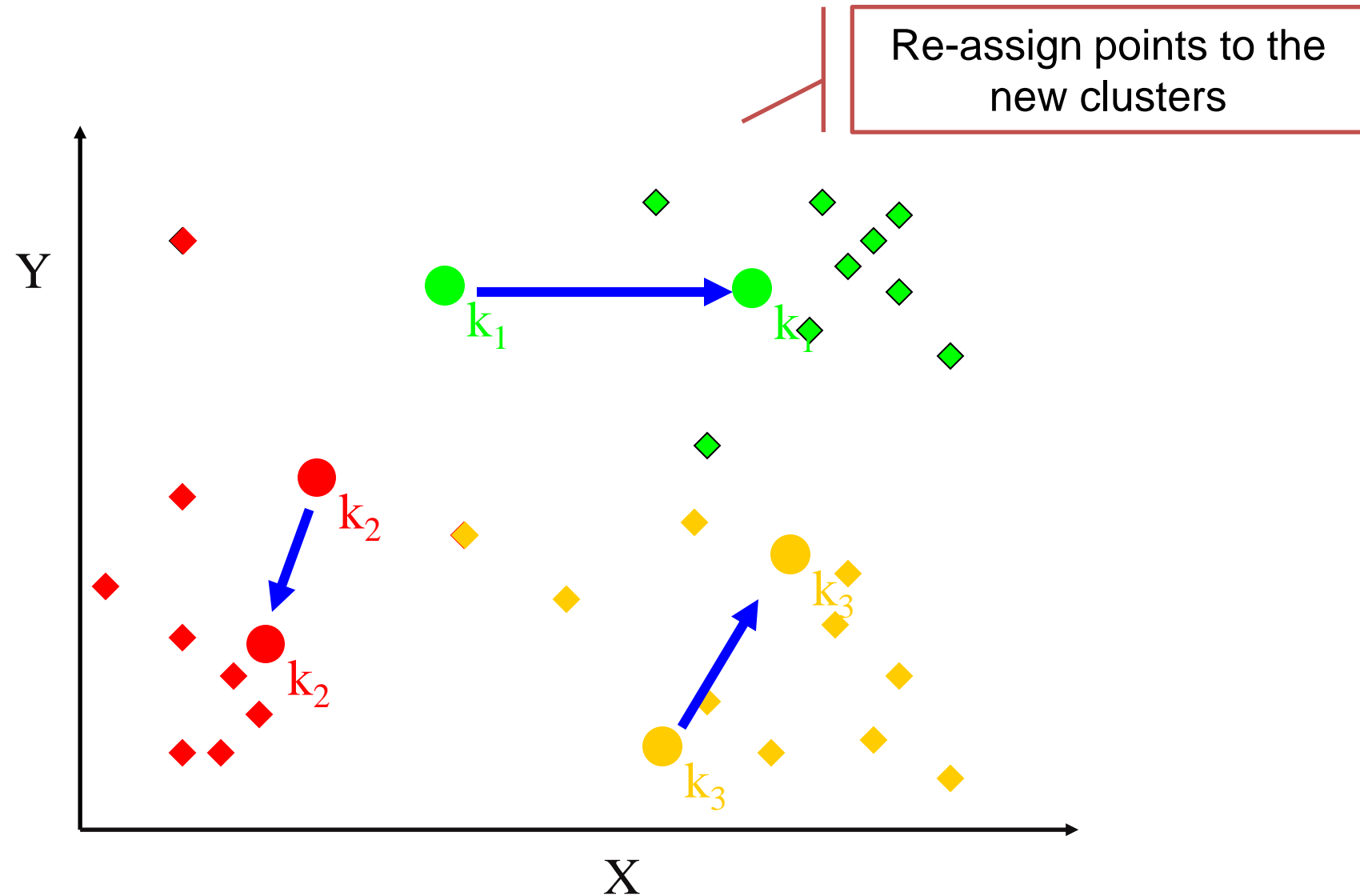
K-means example (2)



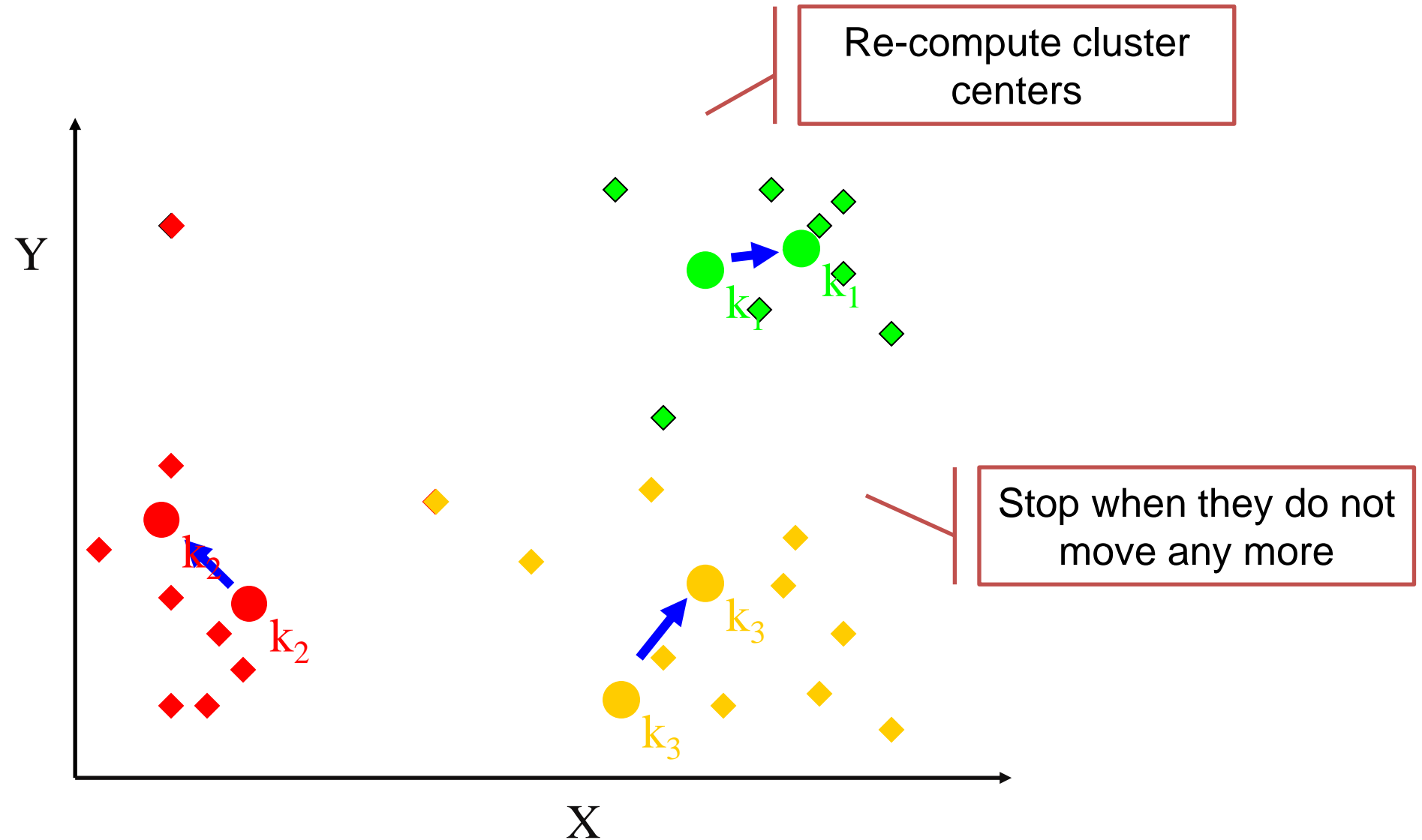
K-means example (3)



K-means example (4)



K-means example (5)



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 ($K, t \ll N$).

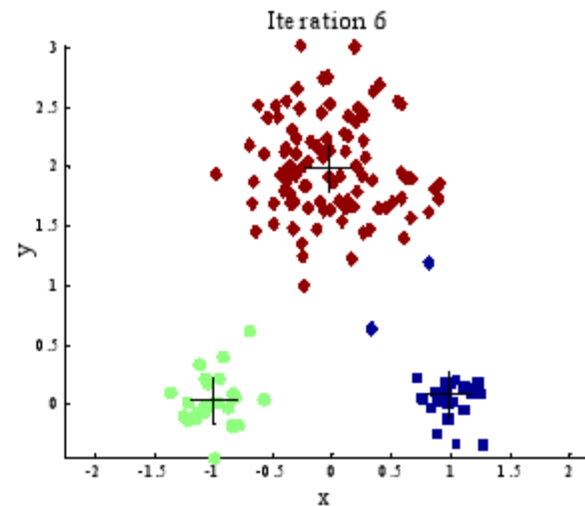
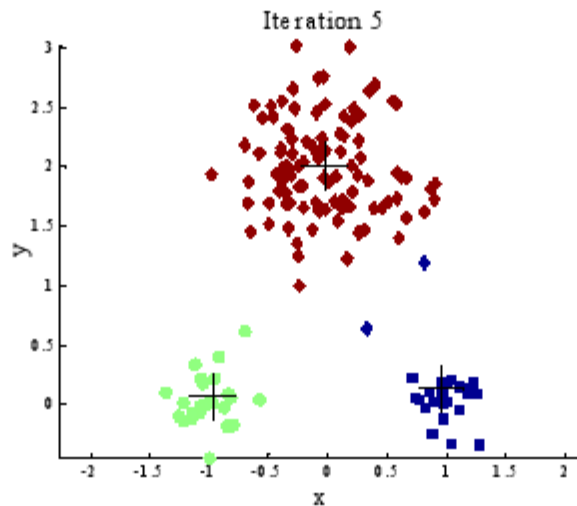
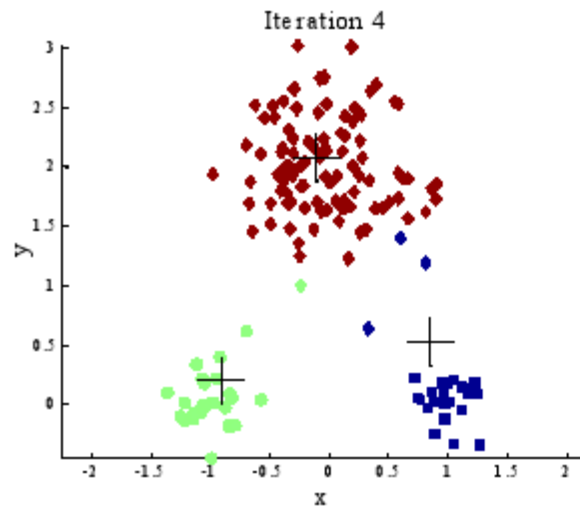
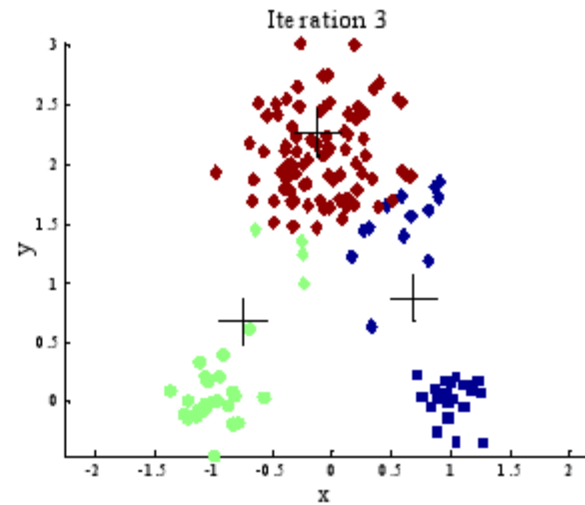
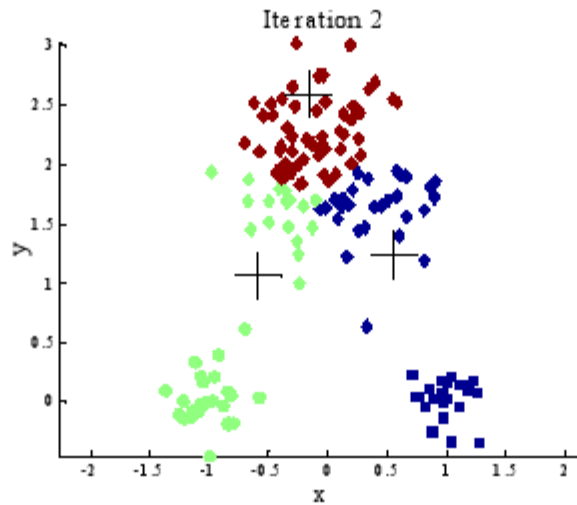
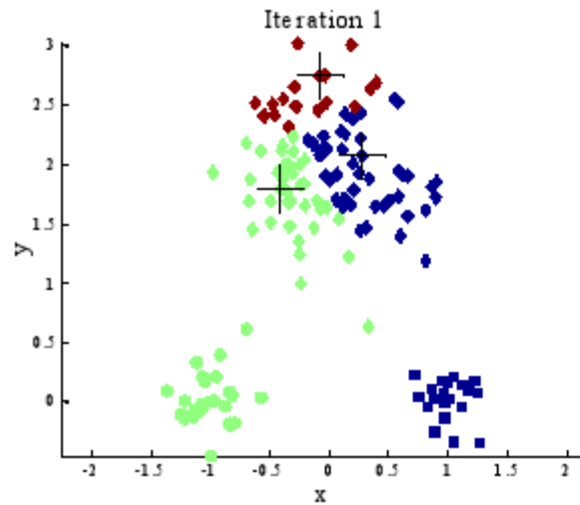
Disadvantages

- 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

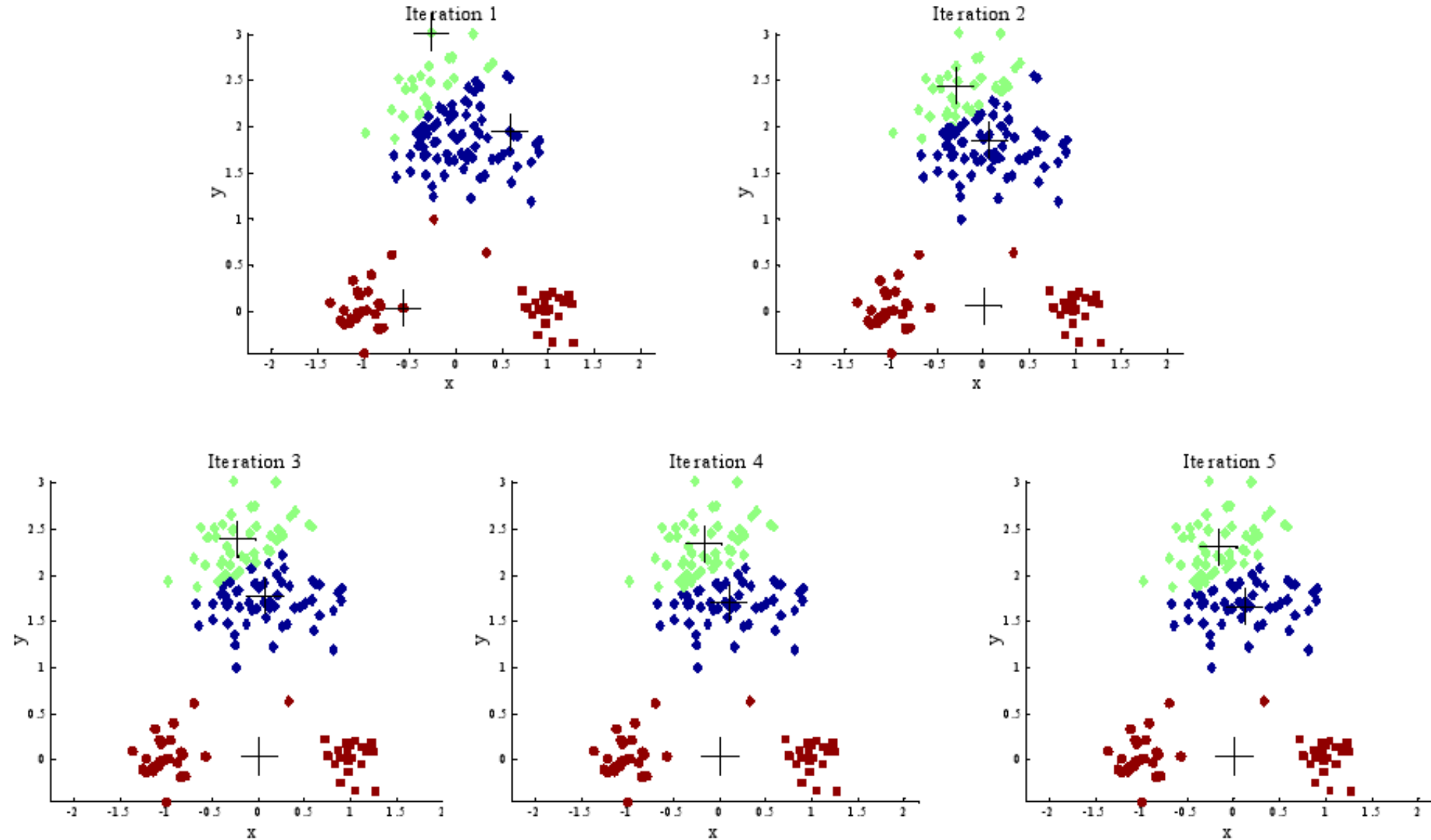
*Some heuristics to
do this exists ...*

Fixed by medoids

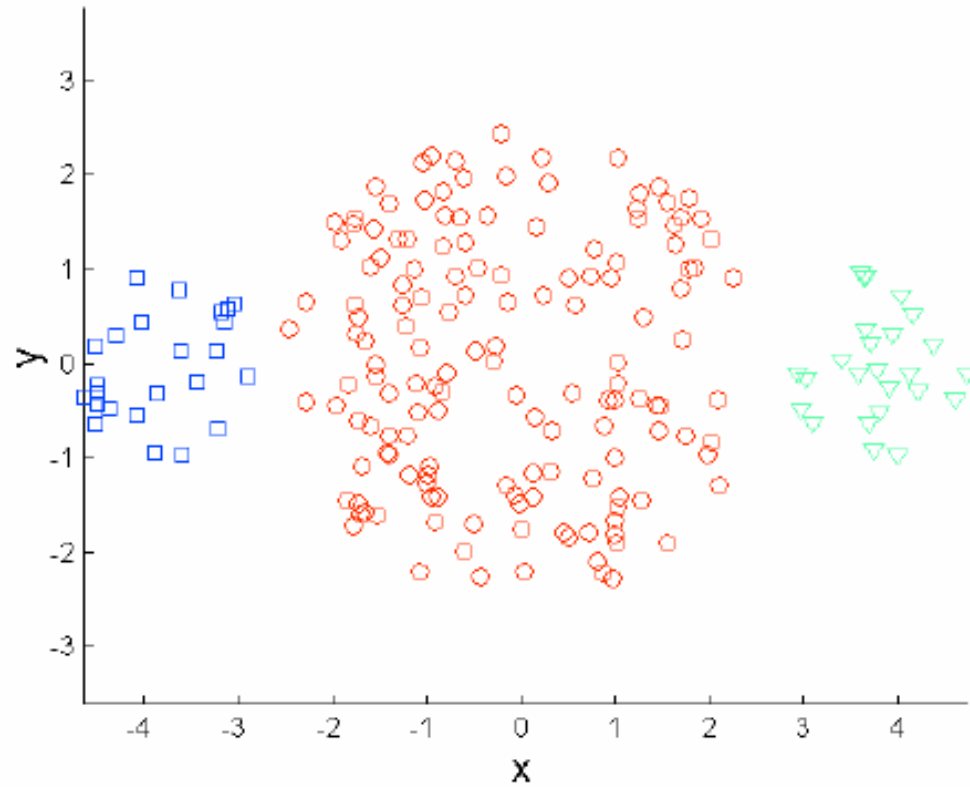
K-Means Initialization



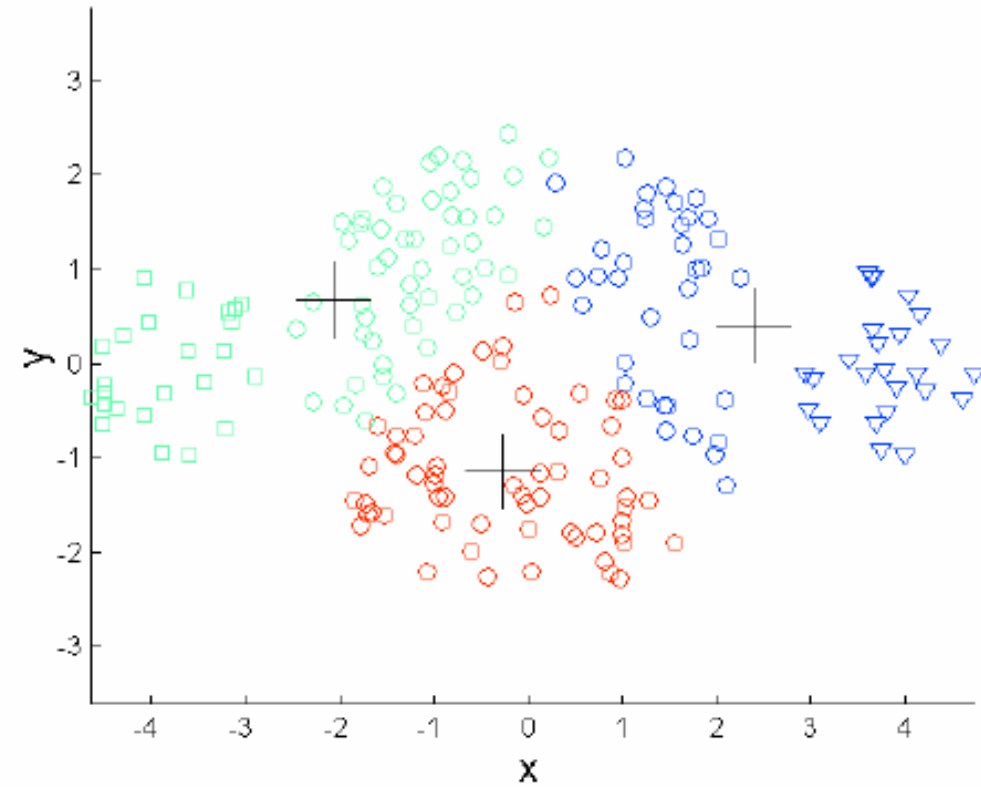
K-Means Initialization



K-Means Different Sizes

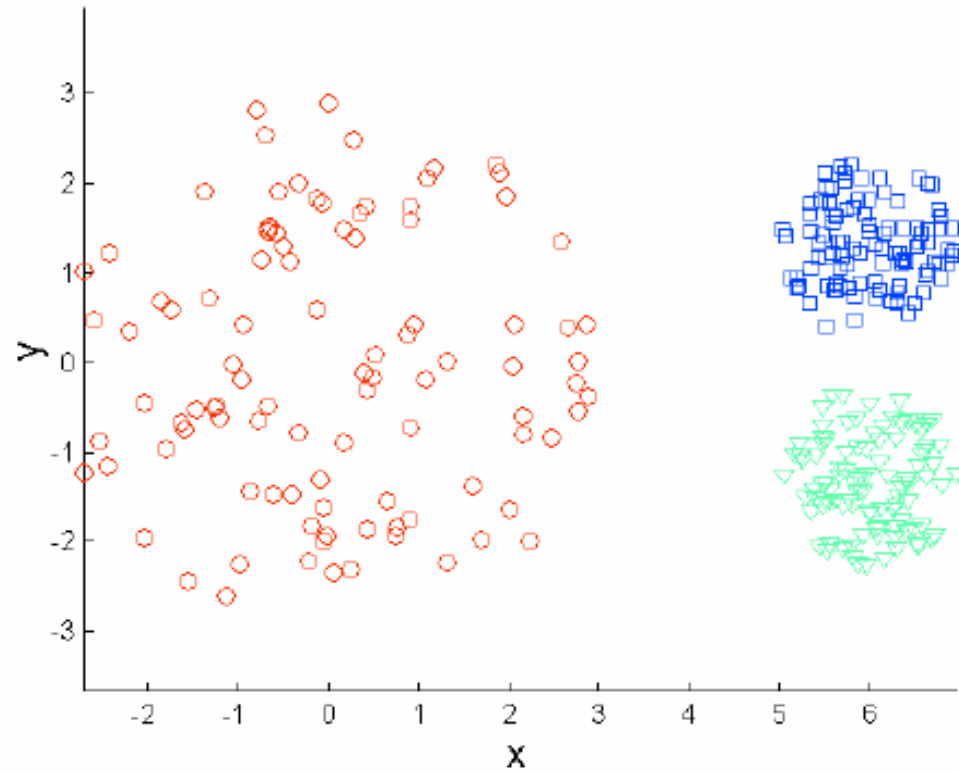


Original Points

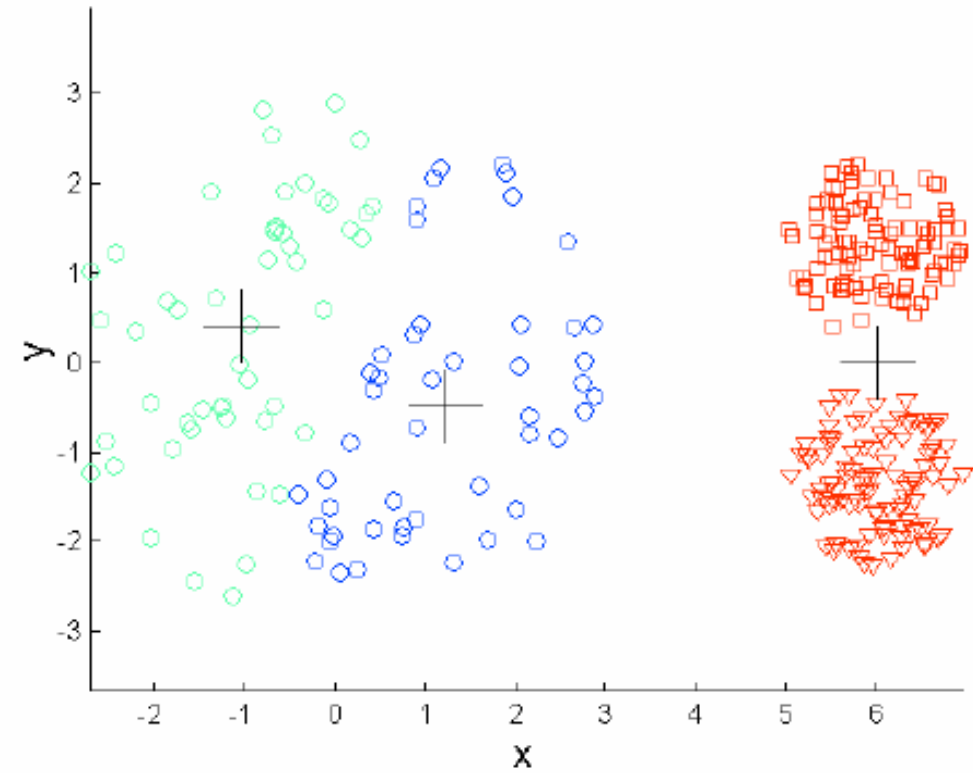


K-means Clusters

K-Means Different Densities

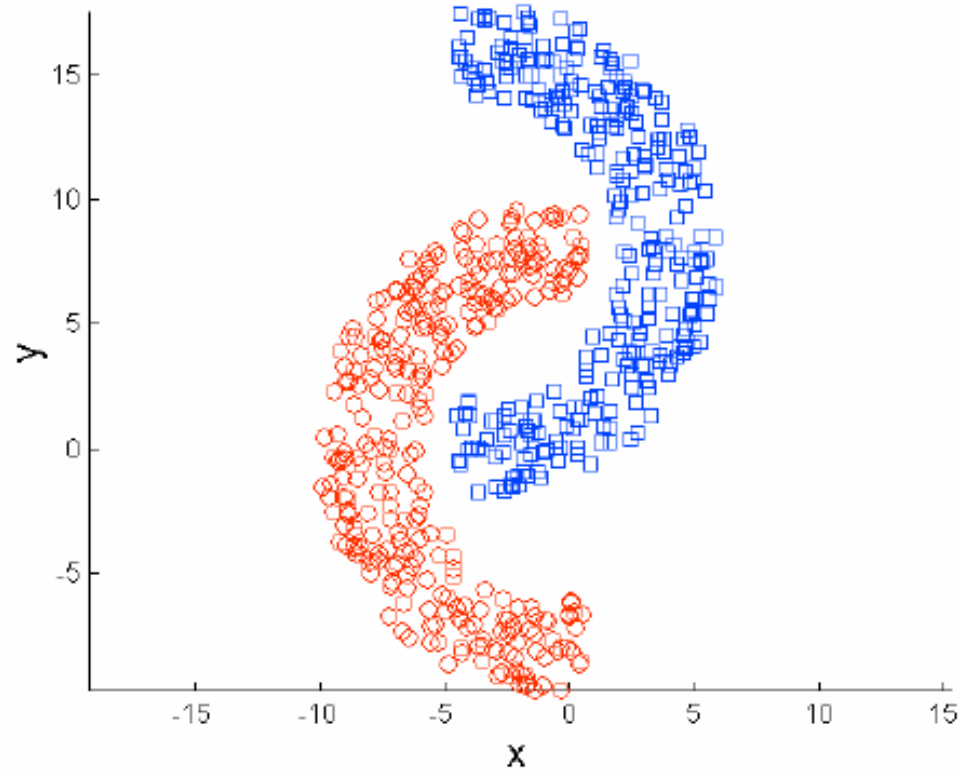


Original Points

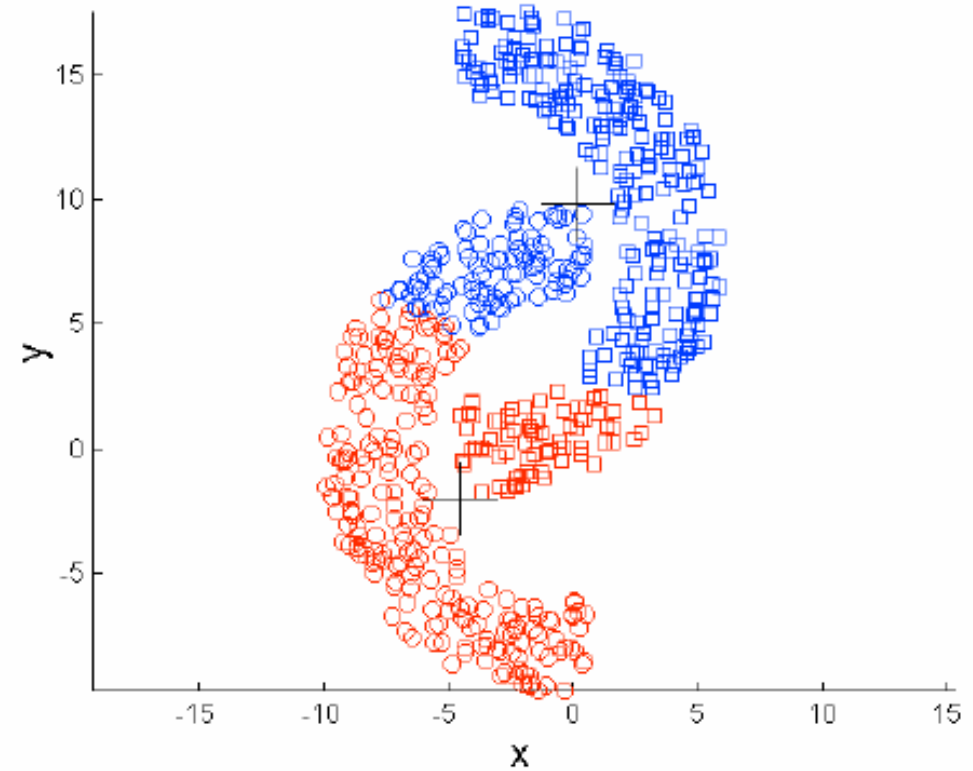


K-means Clusters

K-Means Non Globular Shapes

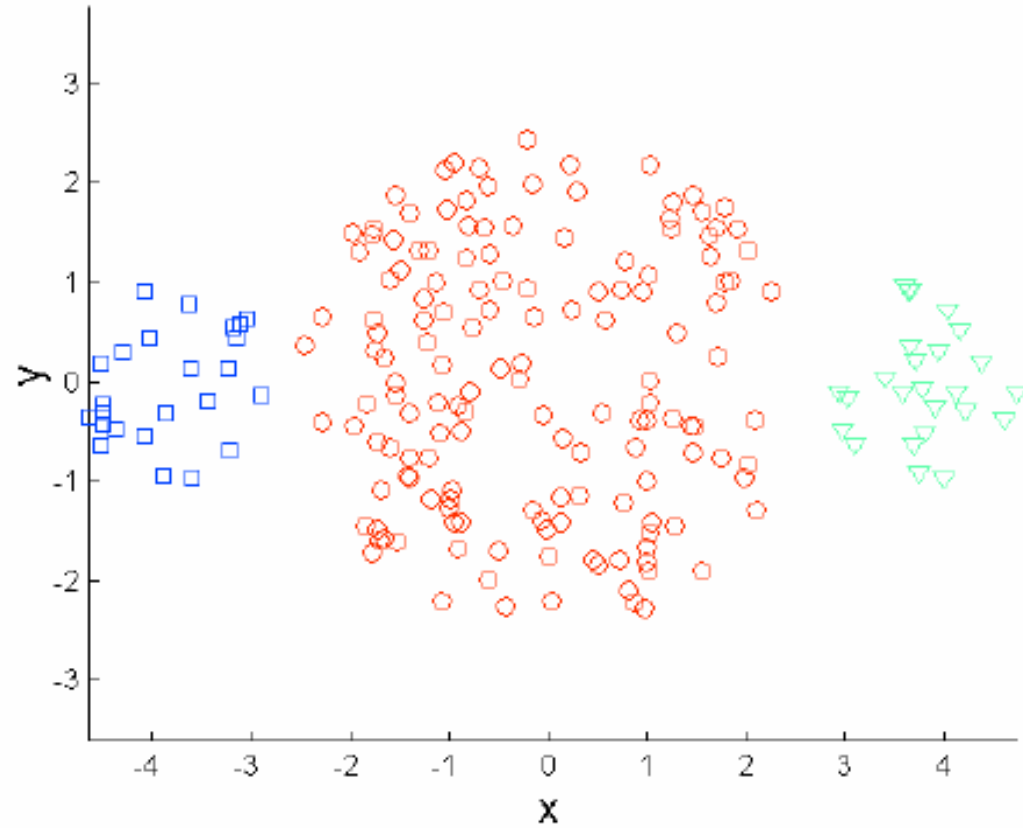


Original Points

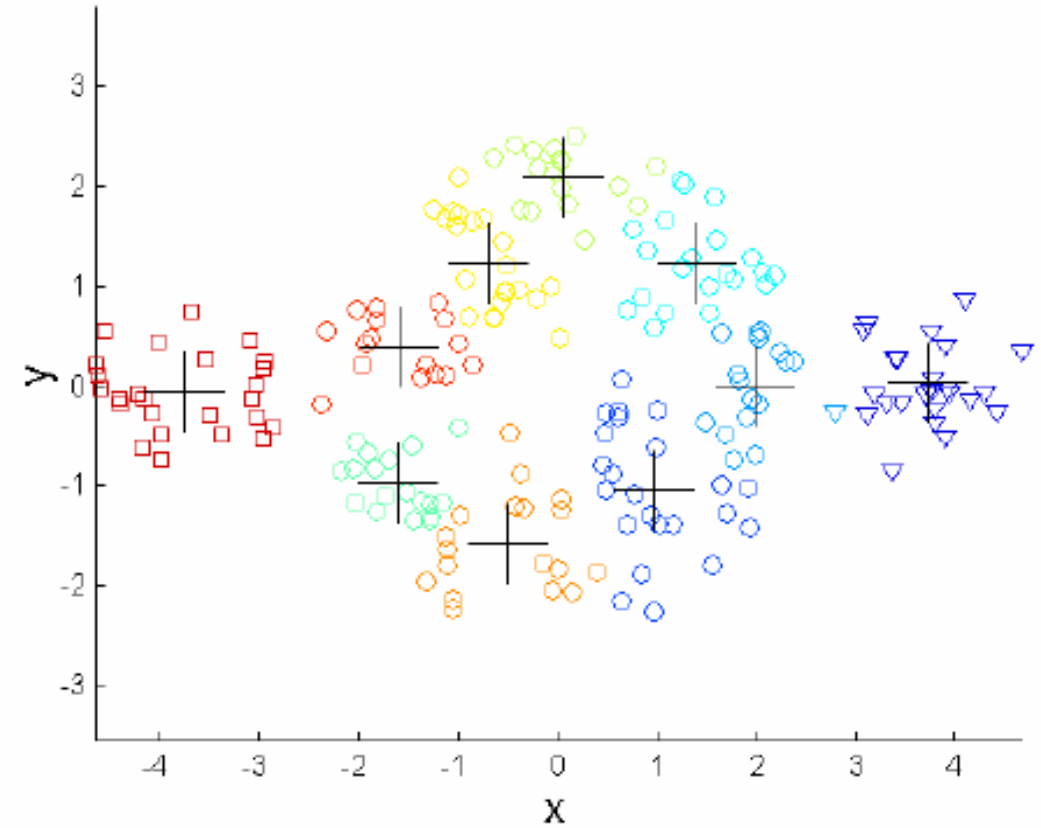


K-means Clusters

K-Means Increasing K

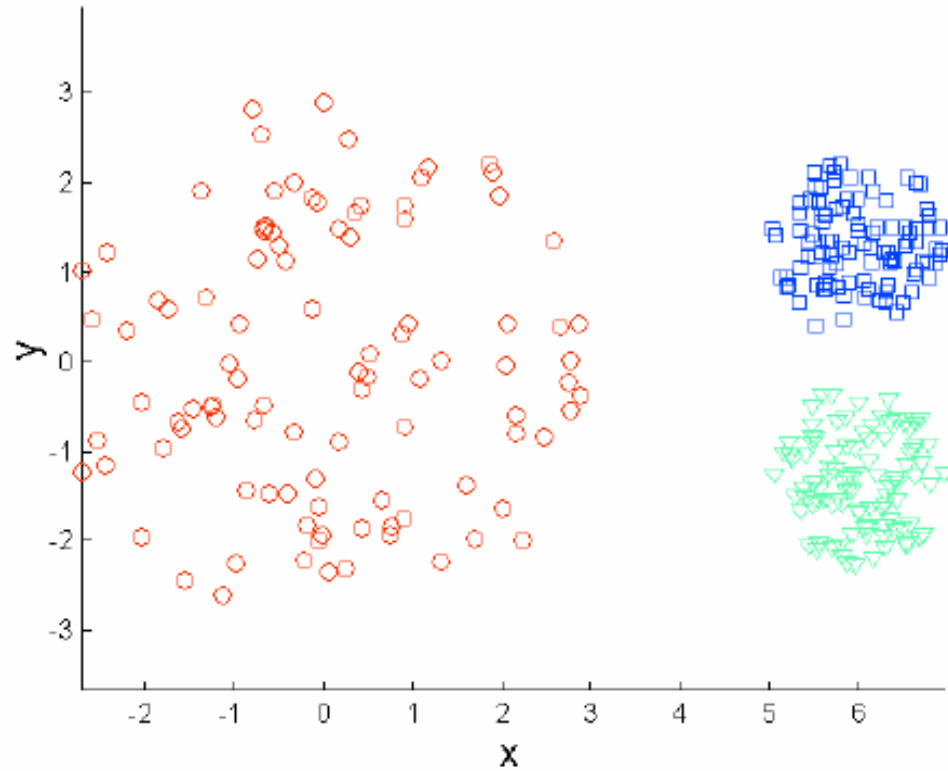


Original Points

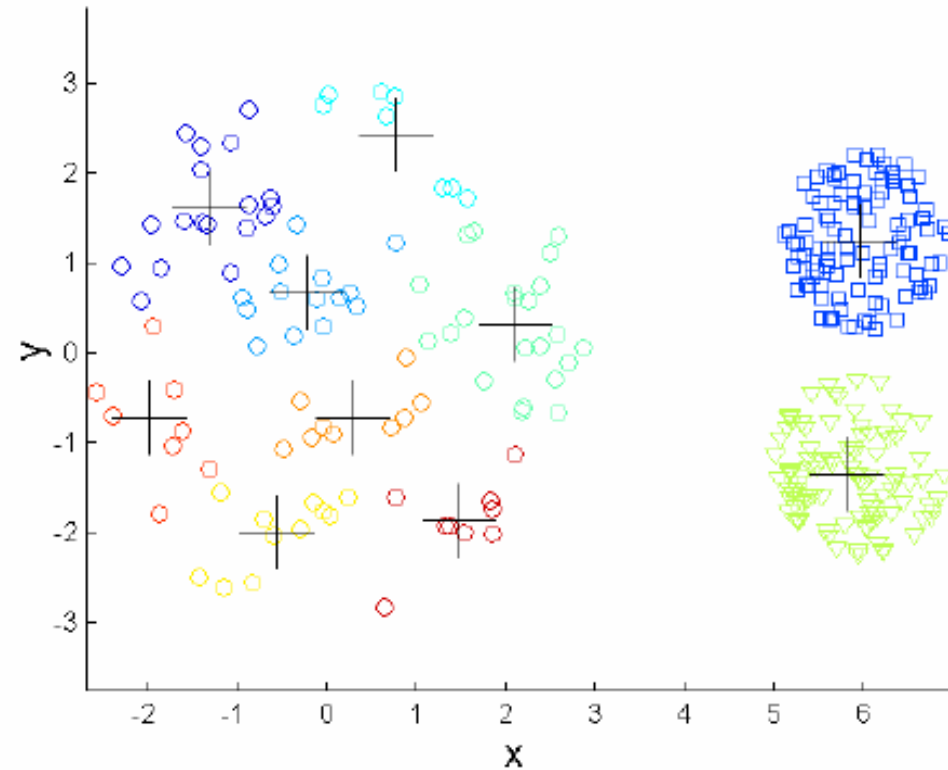


K-means Clusters

K-Means Increasing K

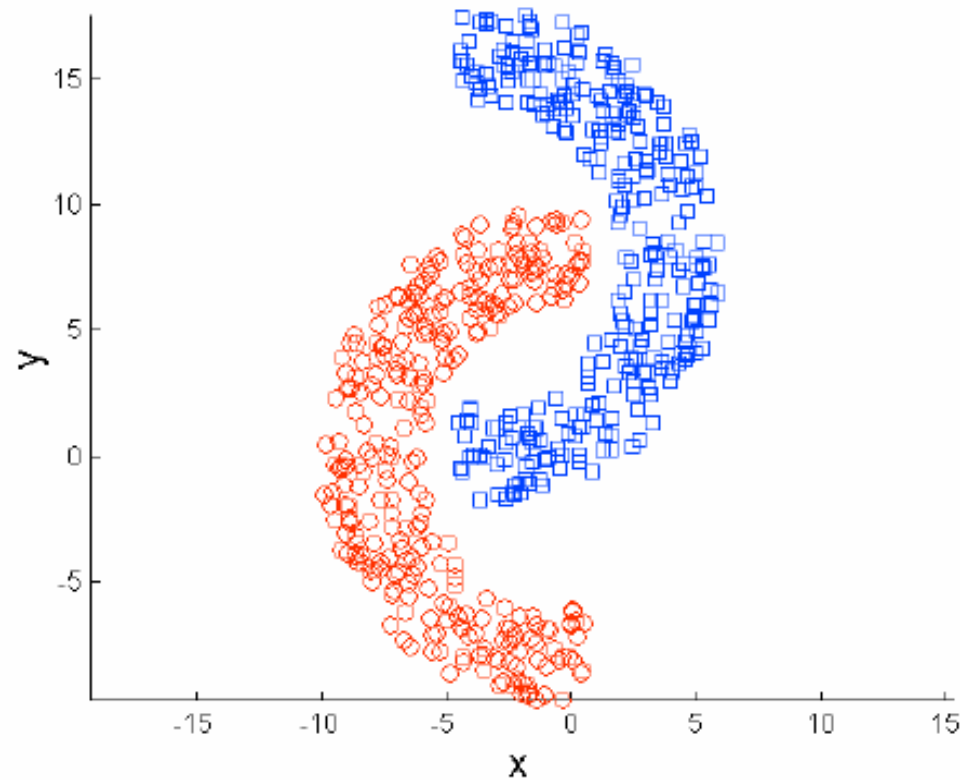


Original Points

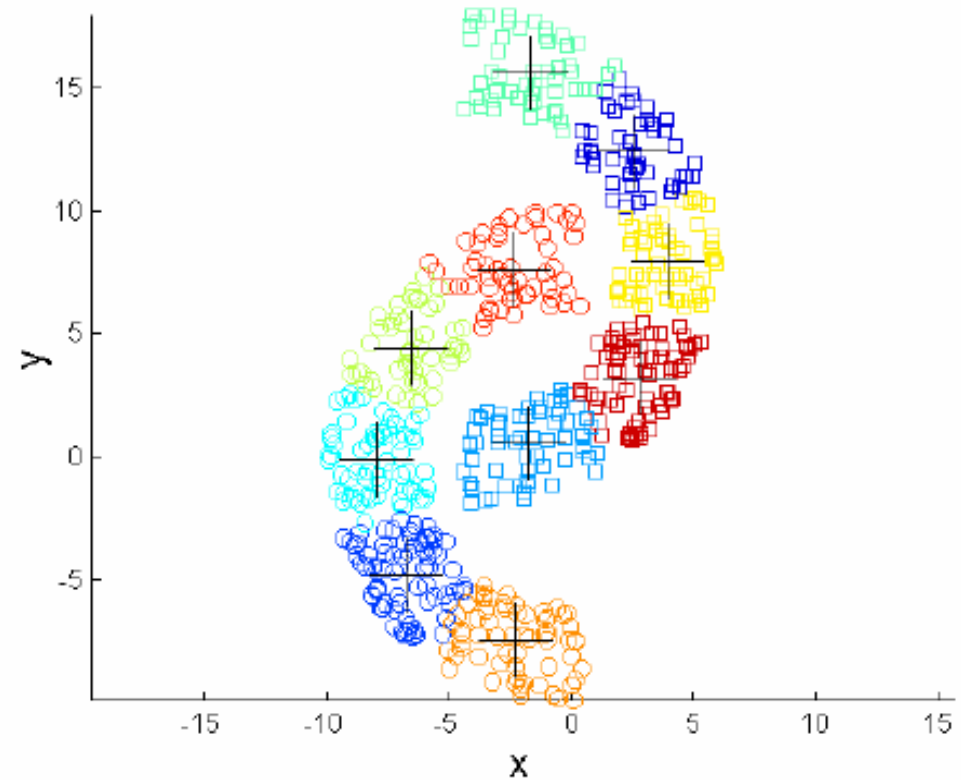


K-means Clusters

K-Means Increasing K



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

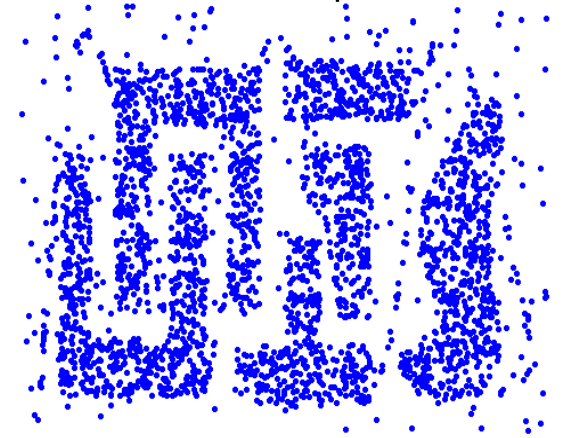
Post-processing:

- 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

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



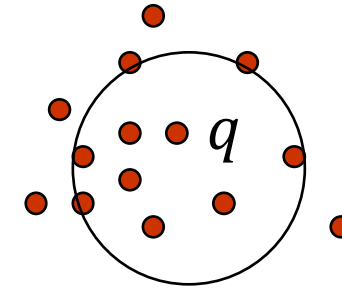
Several algorithms proposed:

- 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 ε of an object is called the ε -neighborhood, if it contains at least **MinPts** objects, then the object is a core object

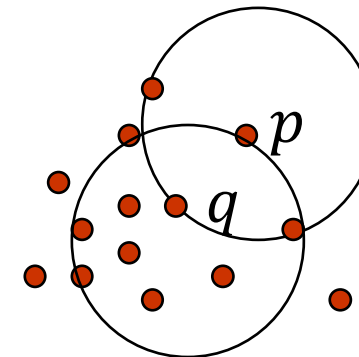
- *Eps*: Maximum radius of the neighbourhood
- *MinPts*: Minimum number of points in an Eps-neighbourhood of that point



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

Directly density-reachable

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

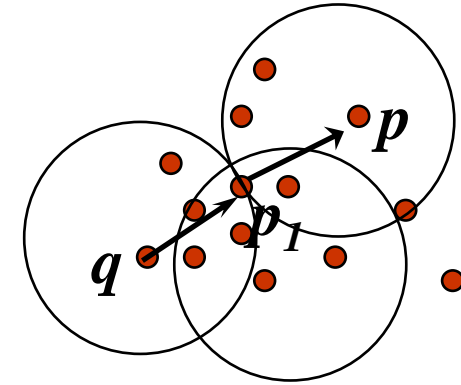


$$\begin{aligned} \text{MinPts} &= 5 \\ \text{Eps} &= 1 \text{ cm} \end{aligned}$$

Density Based Clustering Ideas (2/2)

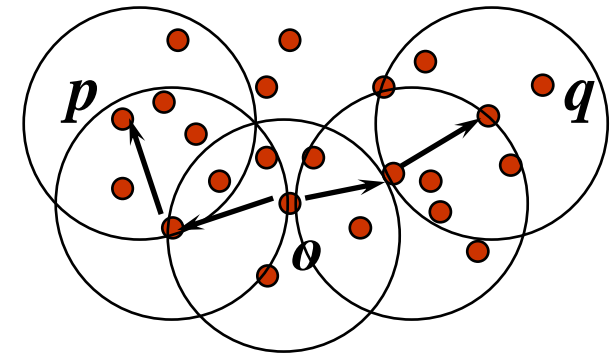
Density-reachable:

- 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

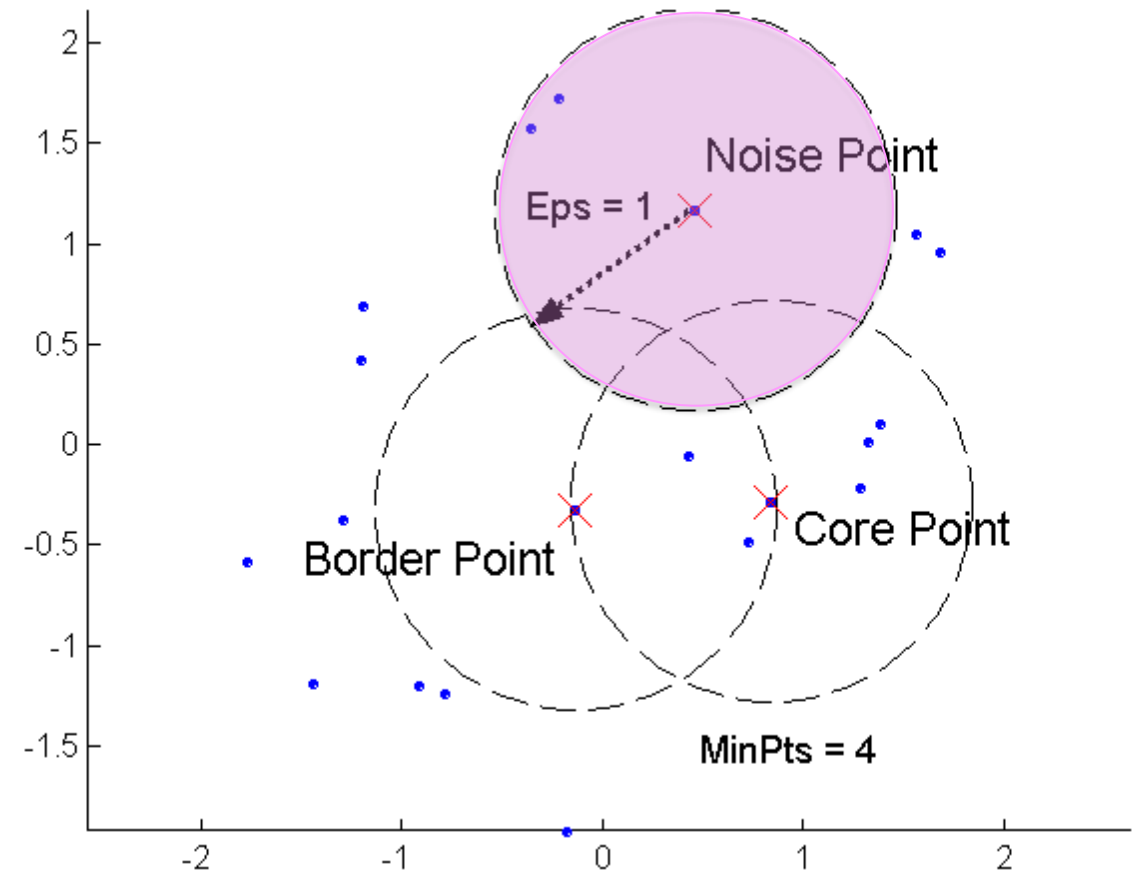
- A point p is density-connected to a point q w.r.t. Eps , $MinPts$ if there is a point o such that both, p and q are density-reachable from o w.r.t. Eps and $MinPts$.



DBSCAN Definitions

DBSCAN extract clusters as a set of density connected objects

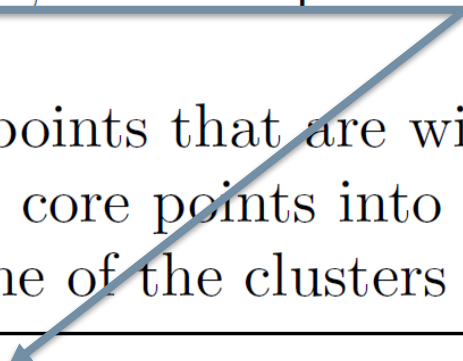
- A **density-based cluster** is a set of density-connected objects that is maximal w.r.t. density-reachability
- A **border point** has fewer than MinPts within Eps , but it is in the neighborhood of a core point
- A **noise point** is any point that is not a core point or a border point



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.
-

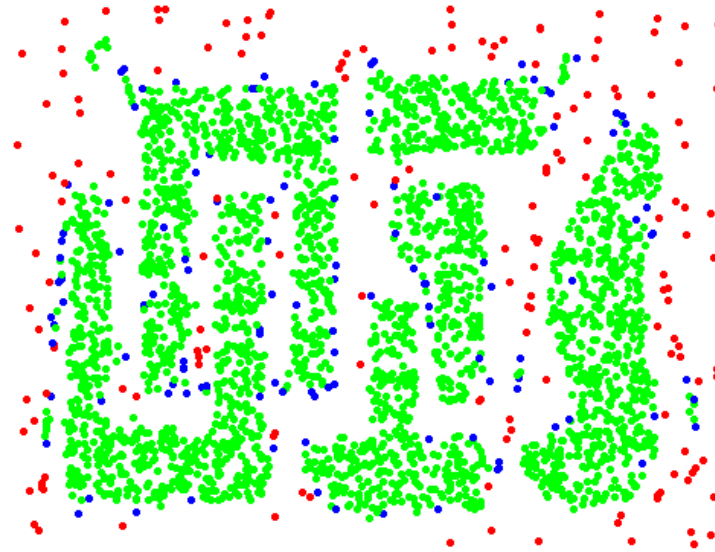
- 
1. Arbitrary select a point p
 2. Retrieve all points density-reachable from p given Eps and $MinPts$.
 1. If p is a core point, a cluster is formed.
 2. If p is a border point, no points are density-reachable from p and DBSCAN visits the next point of the database
 3. Continue the process until all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

DBSCAN: Core, Border and Noise Points

Eps = 10, MinPts = 4



Original Points



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

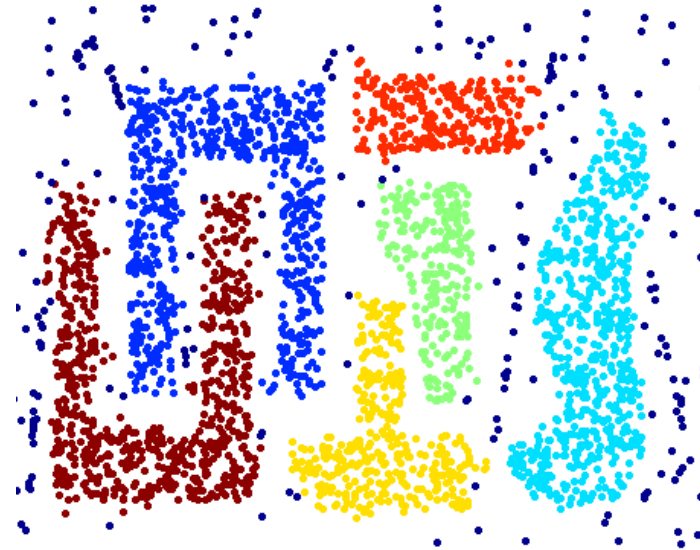
DBSCAN: Clustering

Can handle clusters of different shape

Very Robust to Noise

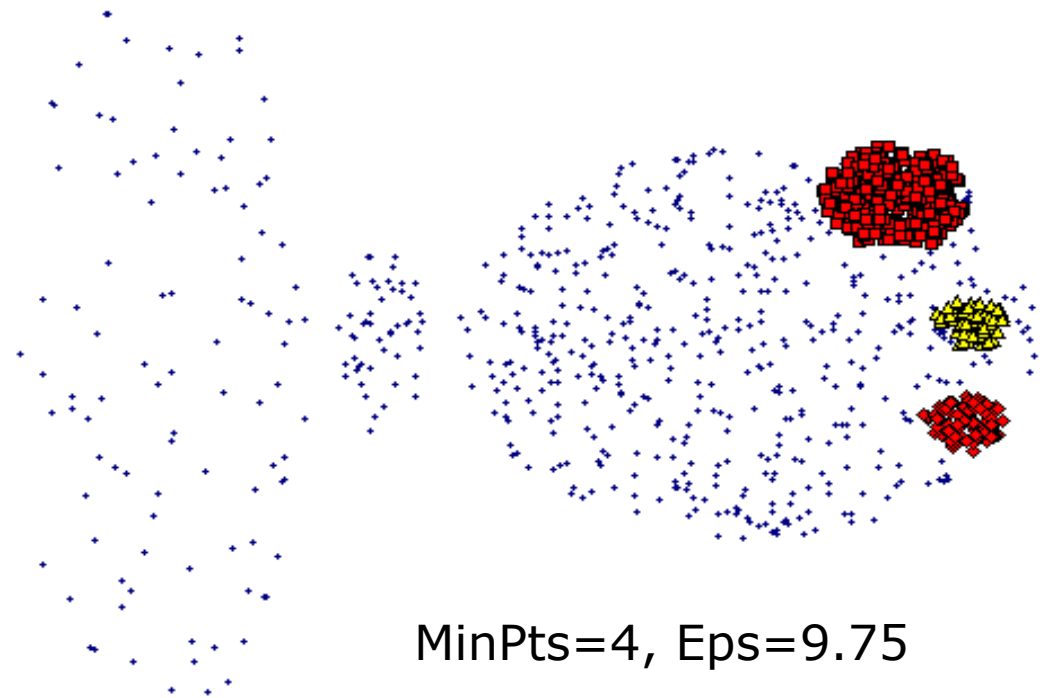
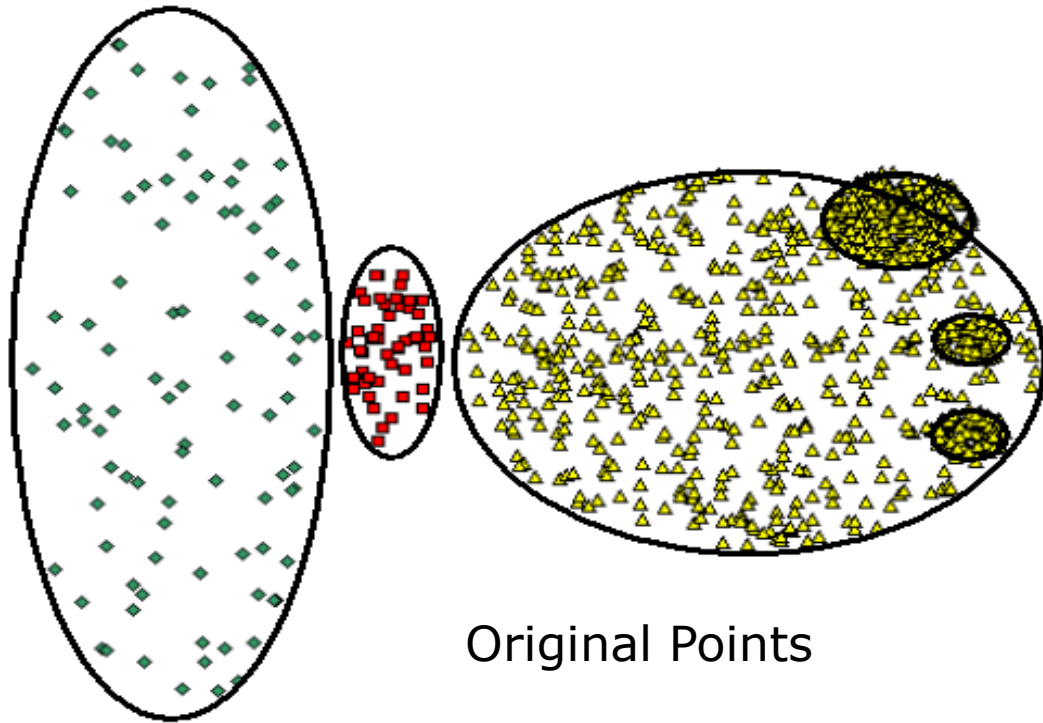


Original Points

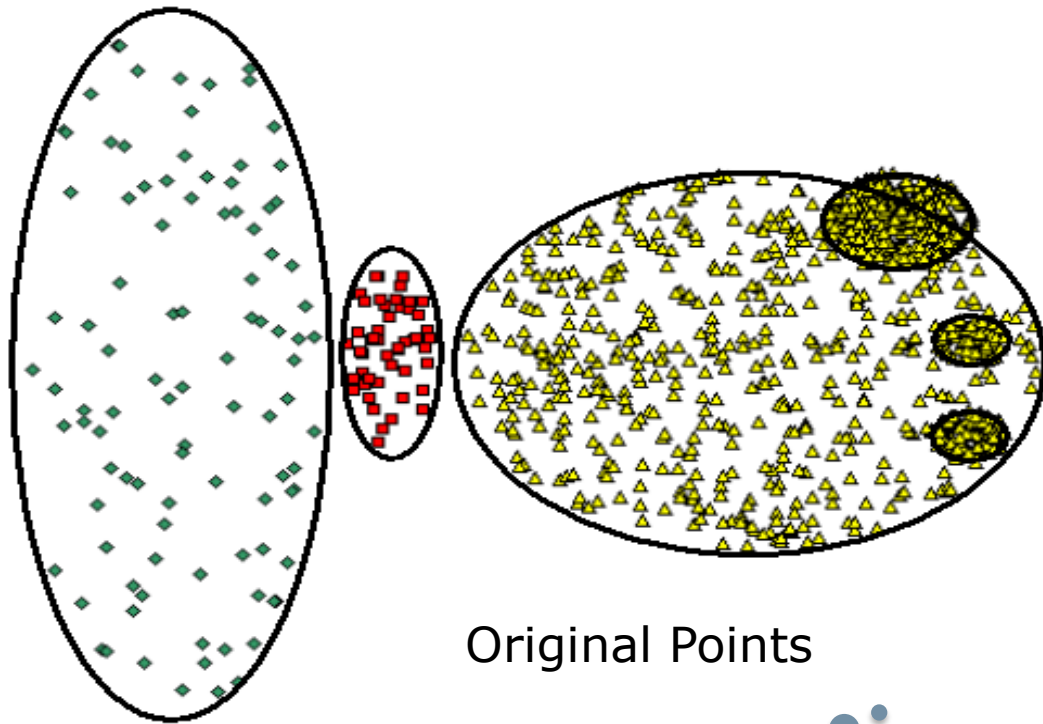


Clusters

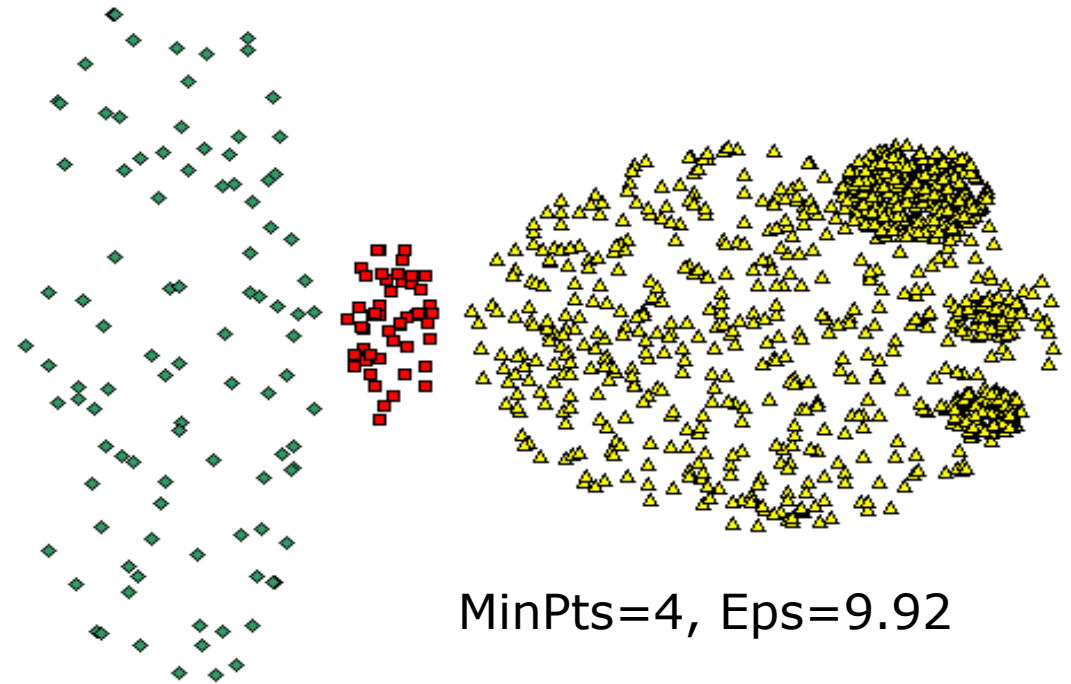
DBSCAN: When it fails ...



DBSCAN: When it fails ...



*Issues with varying
densities*

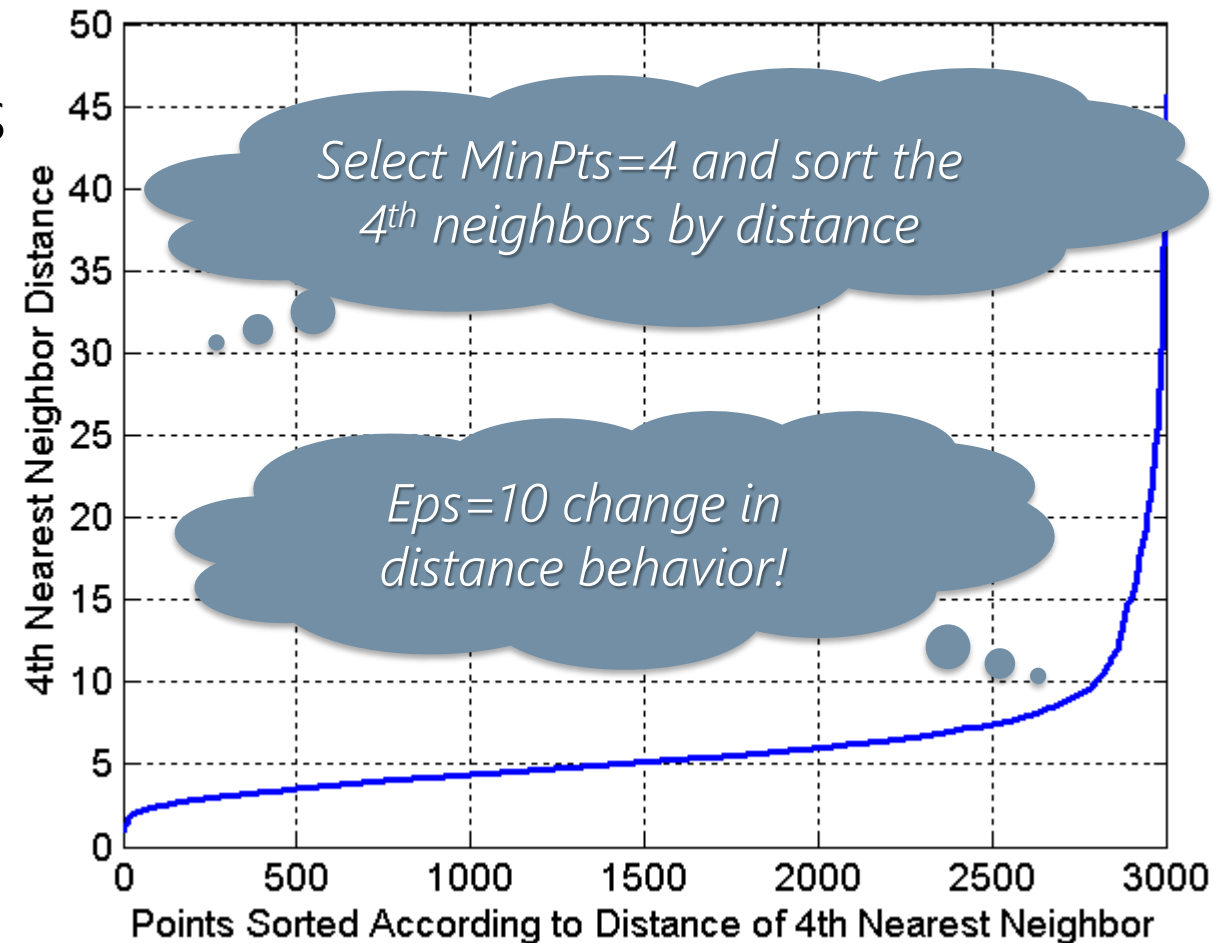


*Issues with high
dimensional data*

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.

Measuring Clusters Validity

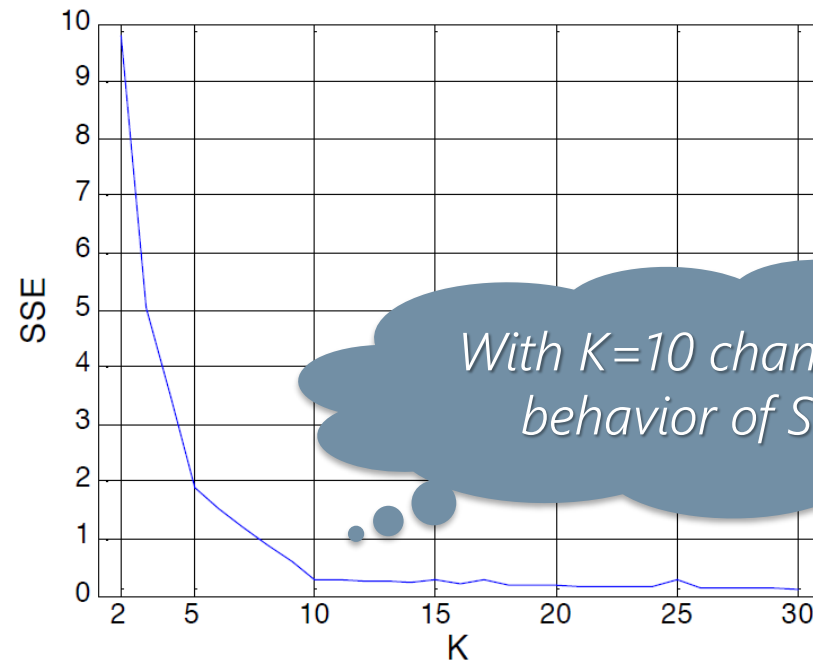
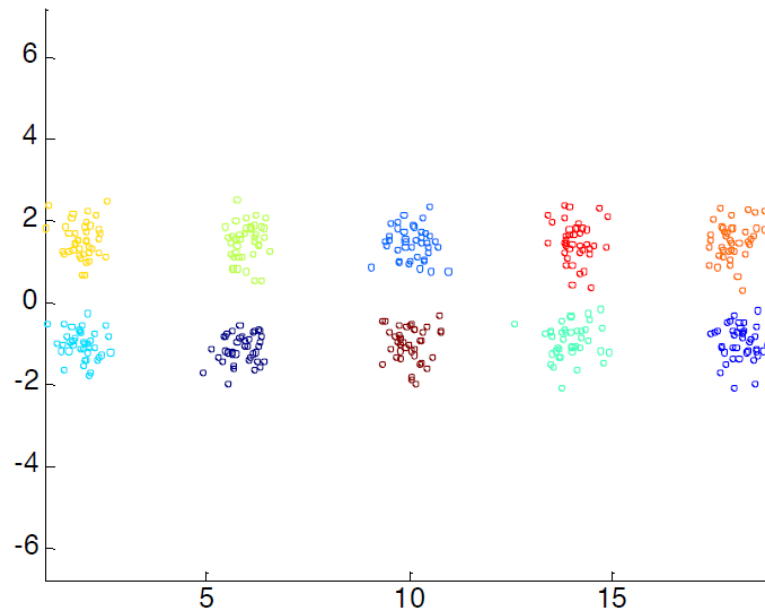
Metrics to evaluate clustering can be classified into

- **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

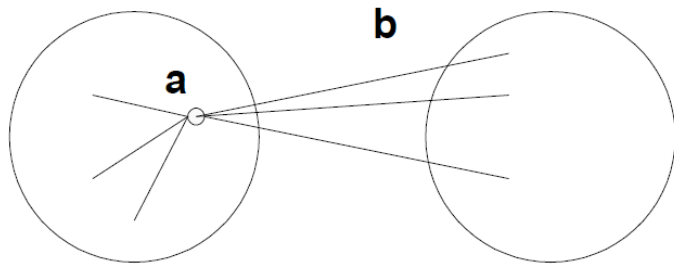


Internal Indexes: Silhouette Coefficient

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)



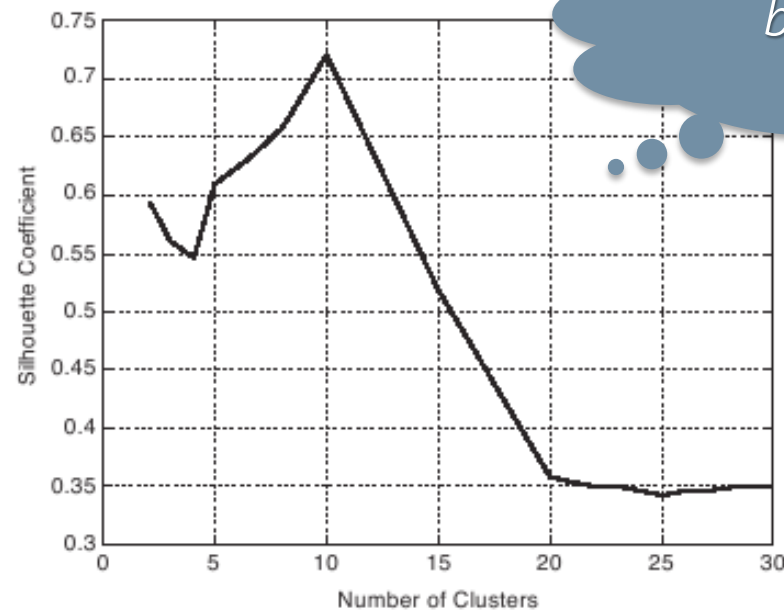
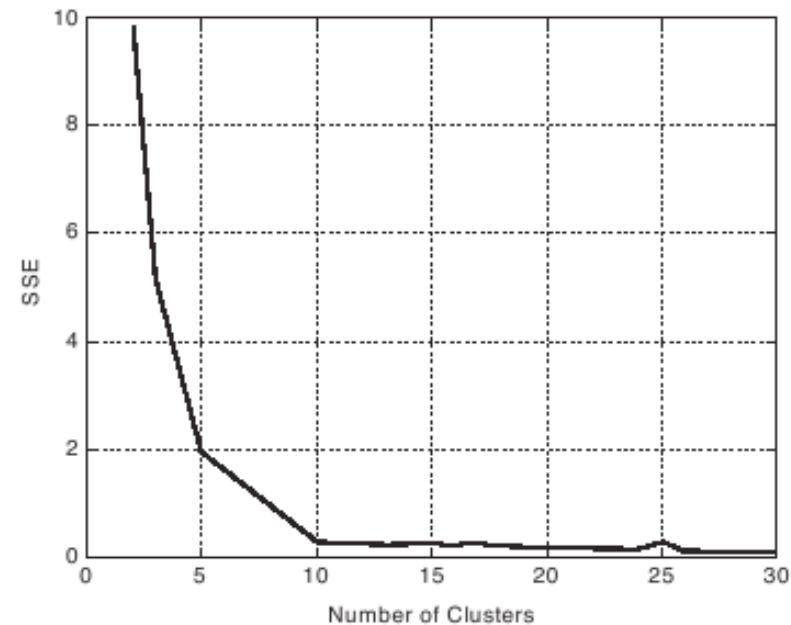
$$s(x_n) = 1 - \frac{a}{b} \text{ if } a < b \text{ (or } \frac{b}{a} - 1 \text{ if } a \geq b)$$

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

