Data Mining
Cluster Analysis: Basic Concepts and Algorithms

Lecture Notes for Chapter 7

Introduction to Data Mining, 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

What is Cluster Analysis?

Qual o agrupamento natural destes dados?

What is Cluster Analysis?

Esportivos

Não esportivos

What is Cluster Analysis?

Europeus

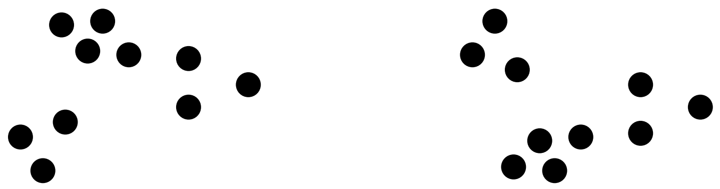
Asiáticos

Americanos

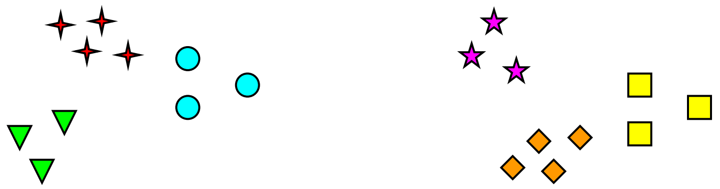
Processo subjetivo...

Notion of a Cluster can be Ambiguous

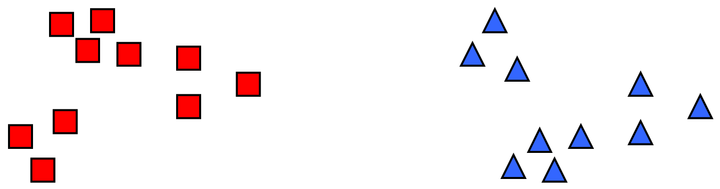
This figure illustrates that the definition of a cluster is imprecise and that the best definition depends on the nature of data and the desired results



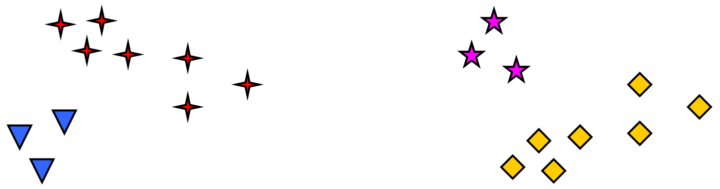
How many clusters?



Six Clusters



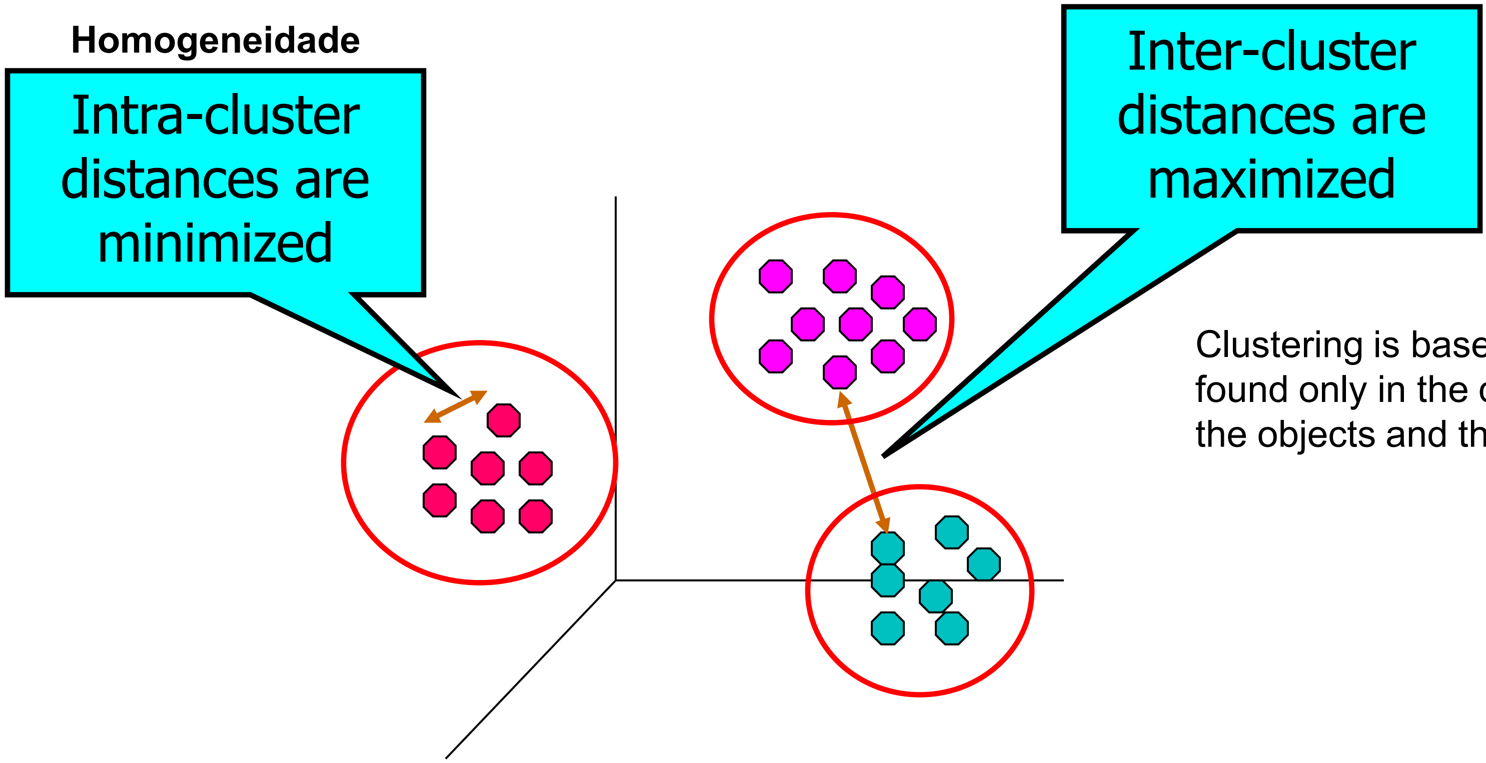
Two Clusters



Four Clusters

What is Cluster Analysis?

- Finding groups of objects such that the objects in a group will be similar (or related) to one another and different from (or unrelated to) the objects in other groups


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

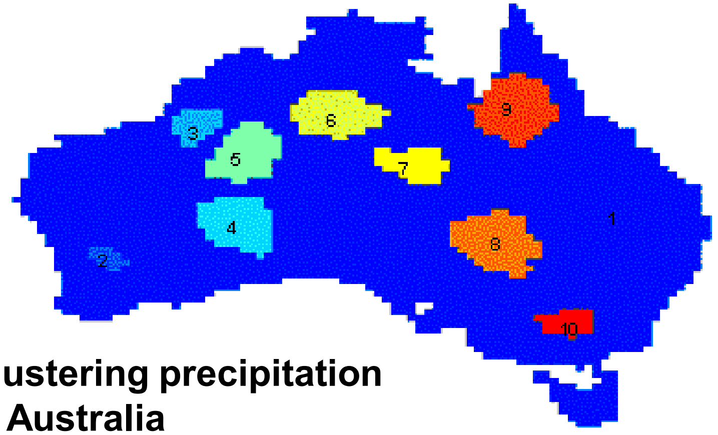
Clustering is based on information found only in the data that describes the objects and their relationships

Applications of Cluster Analysis

- **Understanding**
  - Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

	Discovered Clusters	*Industry Group*
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

- **Summarization**
  - Reduce the size of large data sets

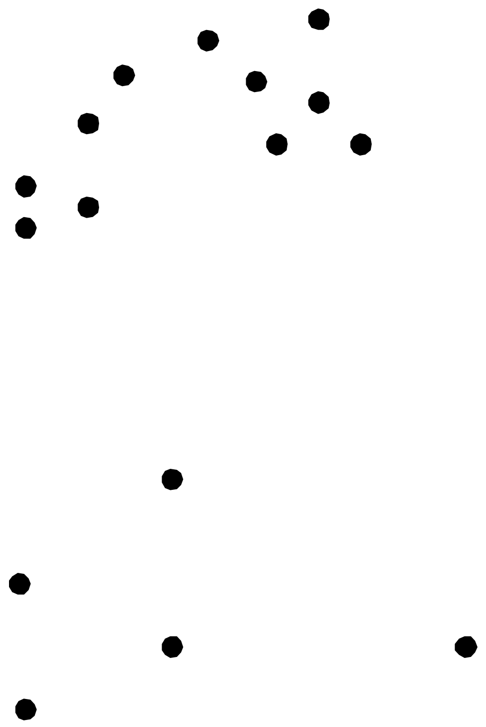


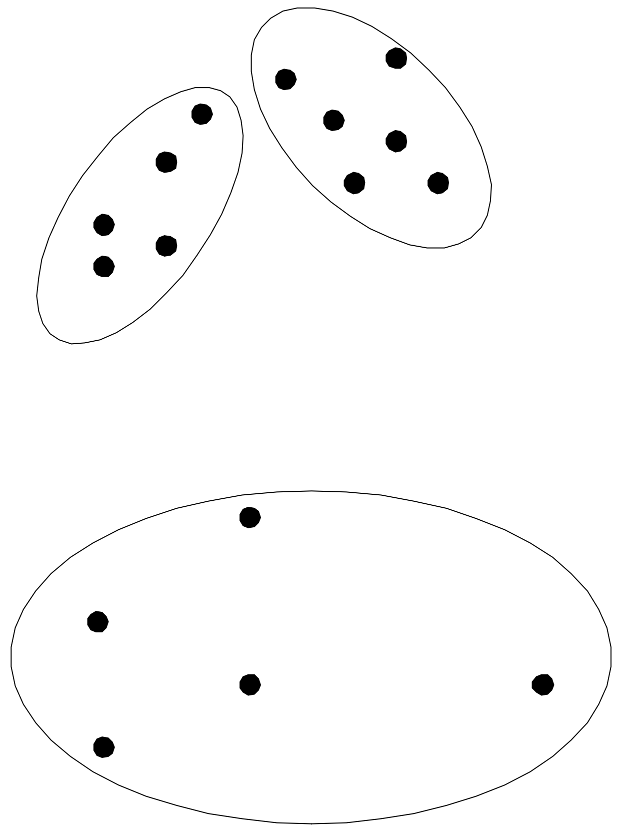
Clustering precipitation in Australia

Types of Clusterings

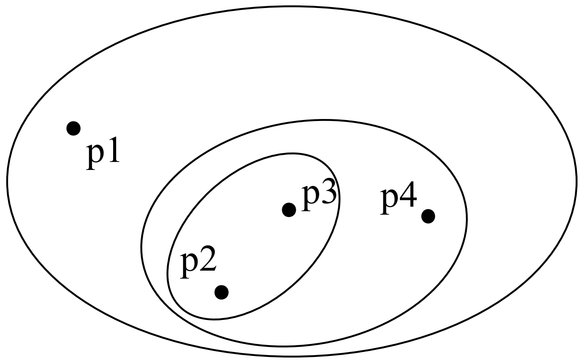
- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
- Hierarchical clustering
  - A set of nested clusters organized as a hierarchical tree

Partitional Clustering

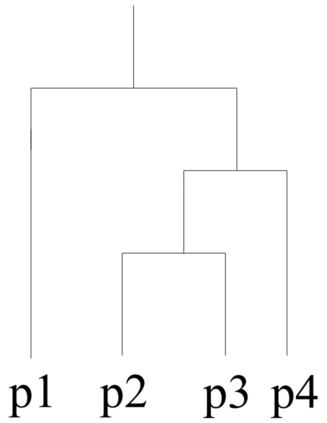

Original Points


A Partitional Clustering

Hierarchical Clustering



Traditional Hierarchical Clustering



Traditional Dendrogram

Other Distinctions Between Sets of Clusters

- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or ‘border’ points
- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to 1
  - Probabilistic clustering has similar characteristics
- Partial versus complete
  - In some cases, we only want to cluster some of the data
- Heterogeneous versus homogeneous
  - Clusters of widely different sizes, shapes, and densities

Types of Clusters

- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters

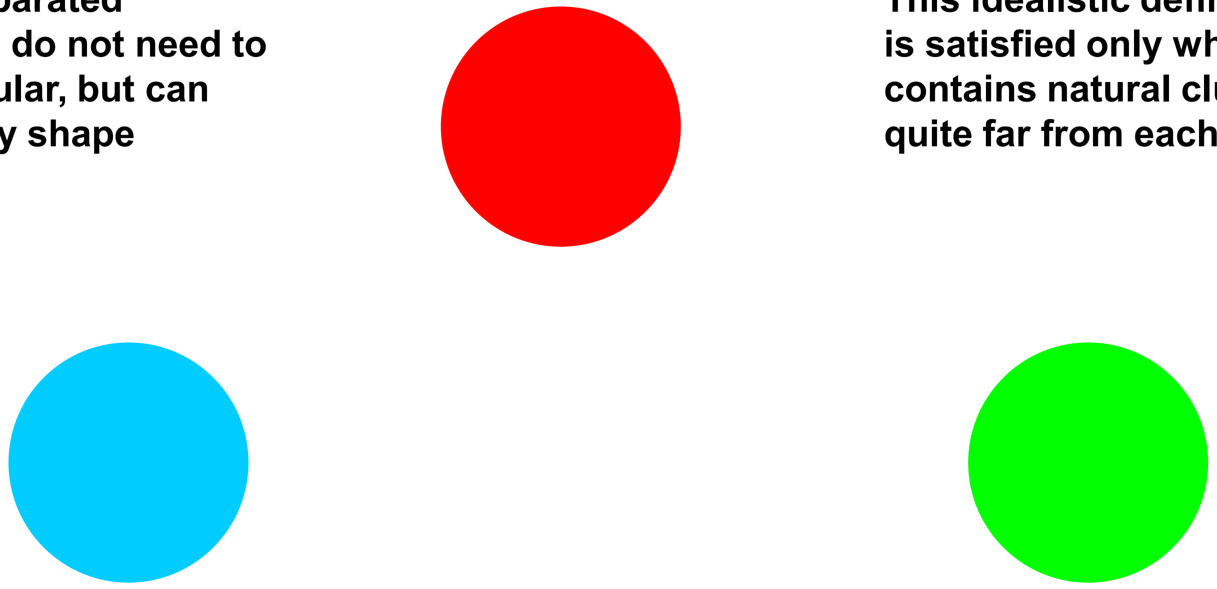
Clustering aims to find useful groups, where usefulness is defined by the goals of the data analysis. Therefore, several different notions of a cluster prove useful in practice

Types of Clusters: Well-Separated

- Well-Separated Clusters:
  – A cluster is a set of points such that each point is closer to all of the points in its cluster than to any point in another cluster.

Well-separated clusters do not need to be globular, but can have any shape

This idealistic definition of a cluster is satisfied only when the data contains natural clusters that are quite far from each other



3 well-separated clusters

Types of Clusters: Center-Based

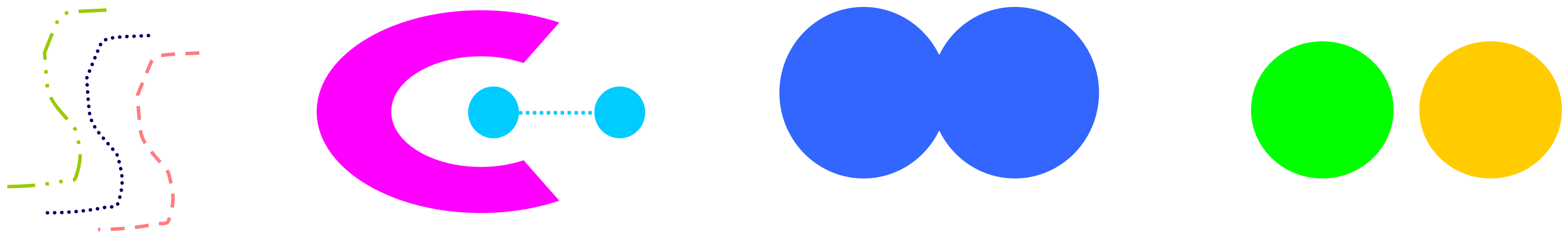
- Center-based
  - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster
  - The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster

Not surprisingly, such clusters tend to be globular

4 center-based clusters

Types of Clusters: Contiguity-Based

- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that each point is closer to at least one point in its cluster than to any point in another cluster.
  - Two objects are connected only if they are within a specified distance of each other – each object in a contiguity-based cluster is closer to some other object in the cluster than to any point in a different cluster

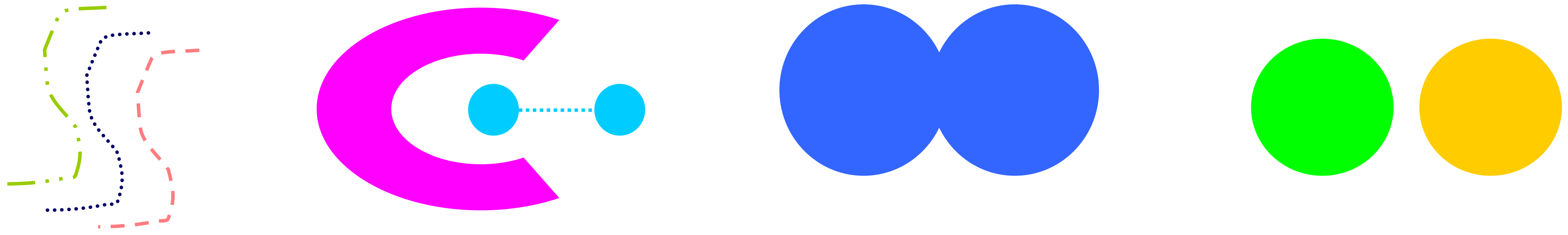


8 contiguous clusters

Graph-Based: seen as a connected component

Types of Clusters: Contiguity-Based

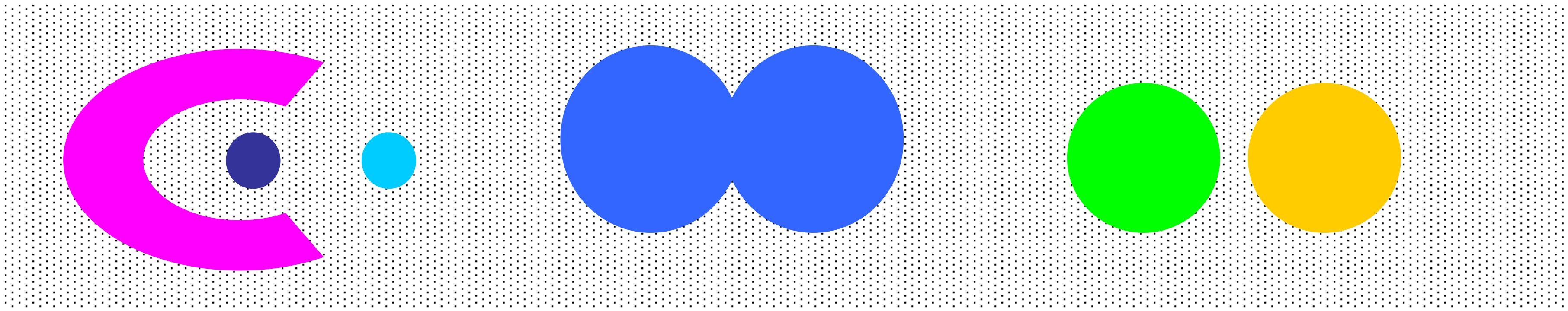
- Contiguous Cluster (Nearest neighbor or Transitive)
  - This definition of a cluster is useful when clusters are irregular or intertwined.
  - This approach can have trouble when noise is present.



8 contiguous clusters

Types of Clusters: Density-Based

- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Clustering Algorithms

- K-means and its variants
  - This is a prototype-based, partitional clustering technique
- Hierarchical clustering (Agglomerative)
  - Some of these techniques have a natural interpretation in terms of graph-based clustering, while others have an interpretation in terms of a prototype-based approach
- Density-based clustering (DBSCAN)
  - This is a partitional clustering technique
  - Points in low-density regions are classified as noise and omitted; thus, DBSCAN does not produce a complete clustering

K-means Clustering

The proximity function can be a similarity or a dissimilarity

- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

To assign a point to the closest centroid, we need a proximity measure that quantifies the notion of "closest" for the specific data under consideration

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

K-means Clustering

- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

Because most of the convergence occurs in the early steps, the condition can be replaced by a weaker condition

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

Only 1% of the points change clusters

K-means Clustering

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

The centroid can vary, depending on the proximity measure and the goal of the clustering

The goal of the clustering is typically expressed by an objective function that depends on the proximities of the points to one another or to the cluster centroids

The key point is this: after we have specified a proximity measure and an objective function, the centroid that we should choose can often be determined mathematically

Steps 3 and 4 of the K-means algorithm directly attempt to minimize/maximize the objective function

K-means: proximity, centroids, and objective functions

Table 7.2. K-means: Common choices for proximity, centroids, and objective functions.

Proximity Function	Centroid	Objective Function
Manhattan (L_1)	median	Minimize sum of the L_1 distance of an object to its cluster centroid **k-medians**
Squared Euclidean (L_2^2)	mean	Minimize sum of the squared L_2 distance of an object to its cluster centroid
cosine	mean	Maximize sum of the cosine similarity of an object to its cluster centroid

Step 3 forms clusters by assigning points to their nearest centroid, which minimizes the SSE for the given set of centroids

Step 4 recomputes the centroids so as to further minimize the SSE

However, Steps 3 and 4 are guaranteed to only find a local minimum with respect to the SSE because they are based on optimizing the SSE for specific choices of the centroids and clusters, rather than for all possible choices

K-means Clustering

VER REF. LIVRO

TABLE 8.1 Data Points for *k*-Means Example

a	*b*	*c*	*d*	*e*	*f*	*g*	*h*
(1,3)	(3,3)	(4,3)	(5,3)	(1,2)	(4,2)	(1,1)	(2,1)

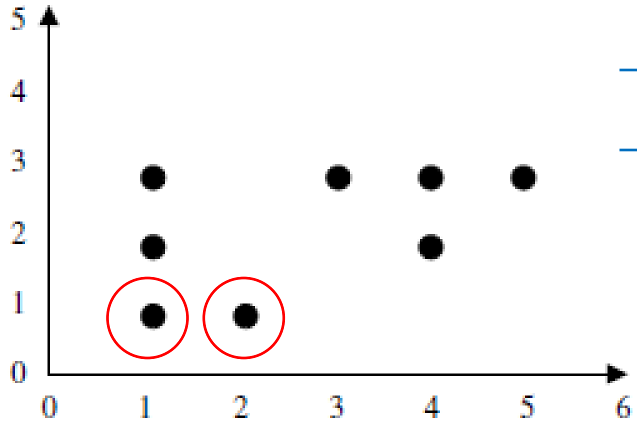
K=2

$m_1 = (1,1)$

$m_2 = (2,1)$

TABLE 8.2 Finding the Nearest Cluster Center for Each Record (First Pass)

Point	Distance from m_1	Distance from m_2	Cluster Membership
a	2.00	2.24	C_1
b	2.83	2.24	C_2
c	3.61	2.83	C_2
d	4.47	3.61	C_2
e	1.00	1.41	C_1
f	3.16	2.24	C_2
g	0.00	1.00	C_1
h	1.00	0.00	C_2

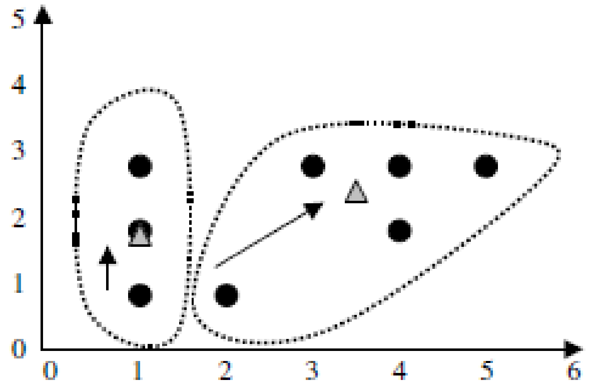


Cluster 1 contains points {a,e,g}
Cluster 2 contains points {b,c,d,f,h}

Cluster 1 is [(1 + 1 + 1) /3, (3 + 2 + 1) /3] = (1,2)
Cluster 2 is [(3 + 4 + 5 + 4 + 2) /5, (3 + 3 + 3 + 2 + 1) /5] = (3.6, 2.4)

K-means Clustering

VER REF. LIVRO



Clusters and centroids after first pass through *k*-means algorithm

K=2
m_1 = (1,2)
m_2 = (3.6, 2.4)

TABLE 8.3 Finding the Nearest Cluster Center for Each Record (Second Pass)

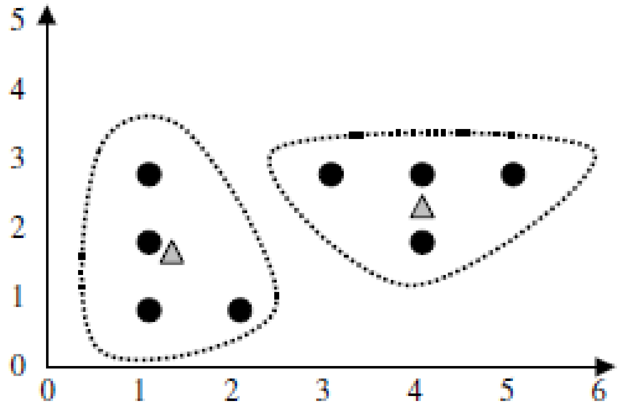
Point	Distance from m_1	Distance from m_2	Cluster Membership
a	1.00	2.67	C_1
b	2.24	0.85	C_2
c	3.16	0.72	C_2
d	4.12	1.52	C_2
e	0.00	2.63	C_1
f	3.00	0.57	C_2
g	1.00	2.95	C_1
h	1.41	2.13	C_1

Cluster 1 is {a,e,g,h}
Cluster 2 is {b,c,d,f}

Cluster 1 is [(1 + 1 + 1 + 2)/4, (3 + 2 + 1 + 1)/4] = (1.25, 1.75)
Cluster 2 is [(3 + 4 + 5 + 4)/4, (3 + 3 + 3 + 2)/4] = (4, 2.75)

K-means Clustering

VER REF. LIVRO



Clusters and centroids after second pass through *k*-means algorithm

K=2
m_1 = (1.25, 1.75)
m_2 = (4, 2.75)

TABLE 8.4 Finding the Nearest Cluster Center for Each Record (Third Pass)

Point	Distance from m_1	Distance from m_2	Cluster Membership
a	1.27	3.01	C_1
b	2.15	1.03	C_2
c	3.02	0.25	C_2
d	3.95	1.03	C_2
e	0.35	3.09	C_1
f	2.76	0.75	C_2
g	0.79	3.47	C_1
h	1.06	2.66	C_1

Note that no records have shifted cluster membership from the preceding pass

$$\text{SSE} = \sum_{i=1}^{k} \sum_{p \in C_i} d(p,m_i)^2 = 1.27^2 + 1.03^2 + 0.25^2 + 1.03^2 + 0.35^2 + 0.75^2$$
$$+0.79^2 + 1.06^2 = 6.25$$

Evaluating K-means Clusters

VER REF. LIVRO

Most common measure is Sum of Squared Error (SSE)

- → For each point, the error is the distance to the nearest cluster
- → To get SSE, we square these errors and sum them

Given two sets of clusters, we prefer the one with the smallest error

TABLE 8.4 Finding the Nearest Cluster Center for Each Record (Third Pass)

Point	Distance from m_1	Distance from m_2	Cluster Membership
a	1.27	3.01	C_1
b	2.15	1.03	C_2
c	3.02	0.25	C_2
d	3.95	1.03	C_2
e	0.35	3.09	C_1
f	2.76	0.75	C_2
g	0.79	3.47	C_1
h	1.06	2.66	C_1

$$\mathrm{SSE} = \sum_{i=1}^{k} \sum_{p \in C_i} d(p, m_i)^2 = 1.27^2 + 1.03^2 + 0.25^2 + 1.03^2 + 0.35^2 + 0.75^2$$
$$+0.79^2 + 1.06^2 = 6.25$$

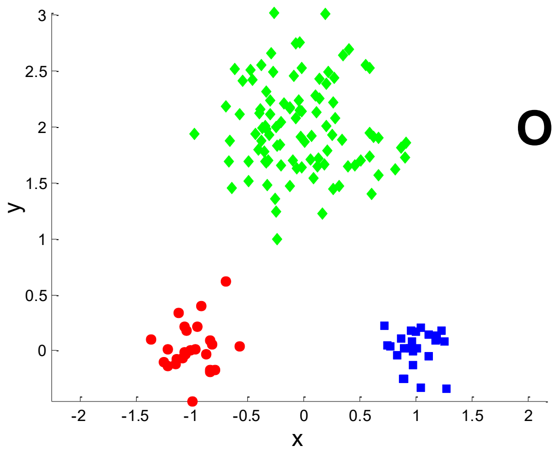
K-means Clustering – Details

- **Initial centroids are often chosen randomly.**
  - **Clusters produced vary from one run to another.**
- The centroid is (typically) the mean of the points in the cluster.
- 'Closeness' is measured by Euclidean distance, cosine similarity, correlation, etc.
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to 'Until relatively few points change clusters'
- Complexity is O(n * K * I * d)
  - n = number of points, K = number of clusters,
    I = number of iterations, d = number of attributes

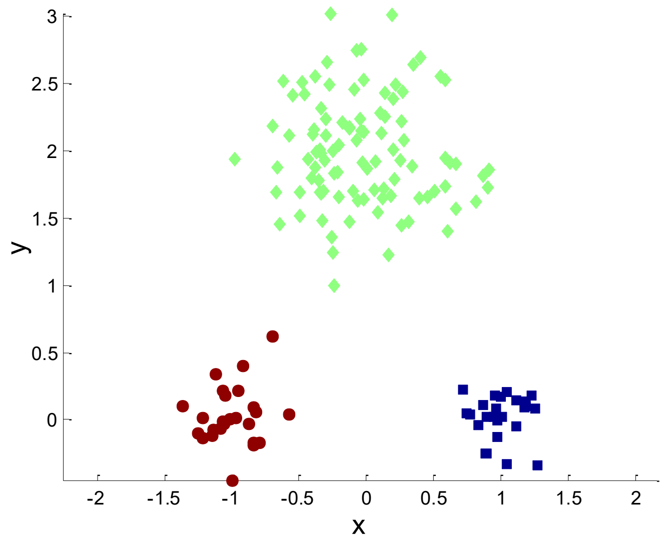
Two different K-means Clusterings

When random initialization of centroids is used, different runs of K-means typically produce different total SSEs

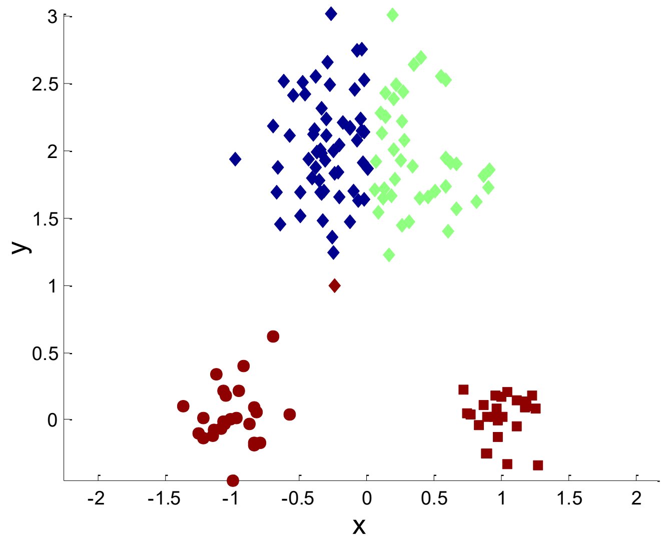
Choosing the proper initial centroids is the key step of the basic K-means procedure



Original Points

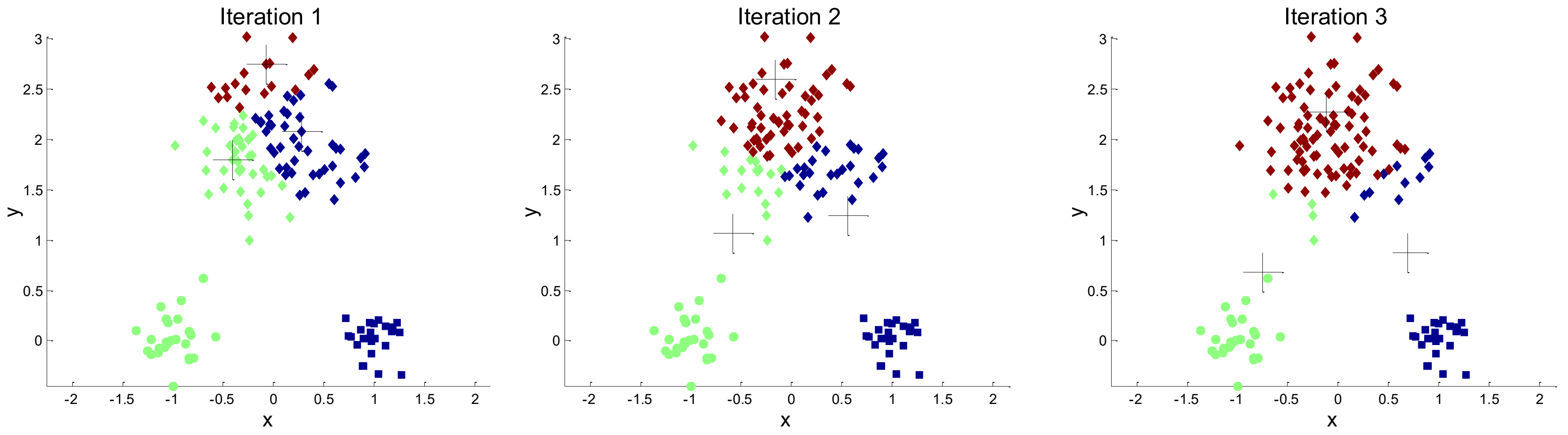


Optimal Clustering

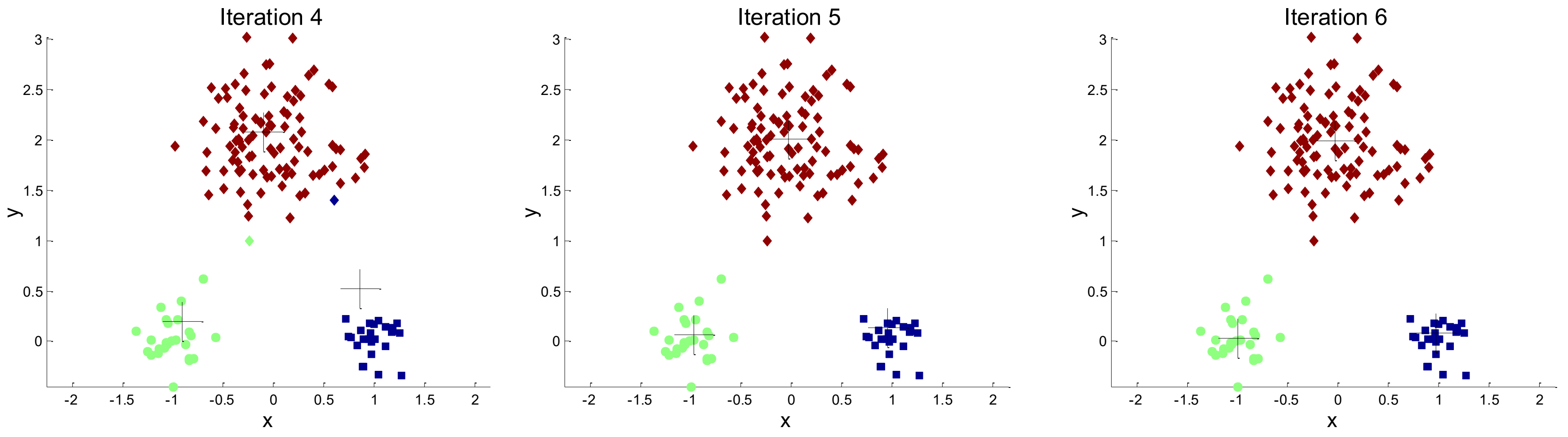


Sub-optimal Clustering

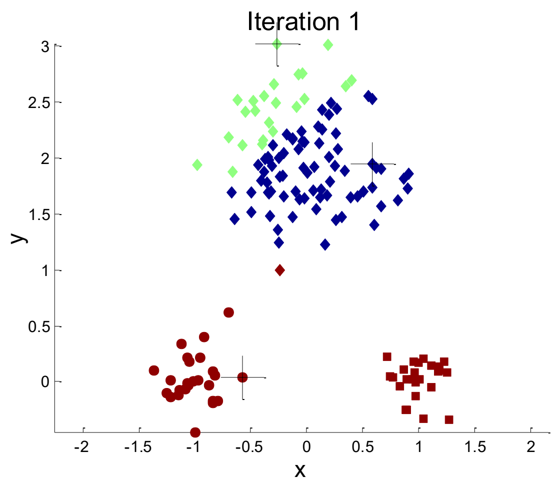
Importance of Choosing Initial Centroids

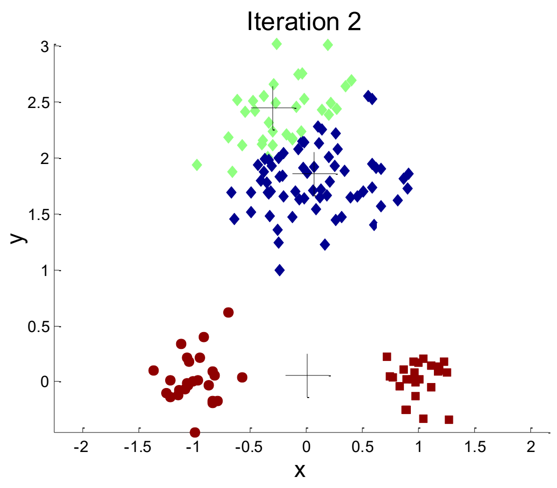


Optimal Clustering

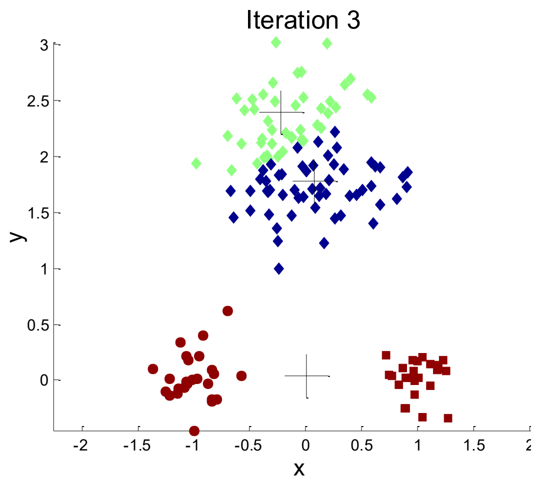


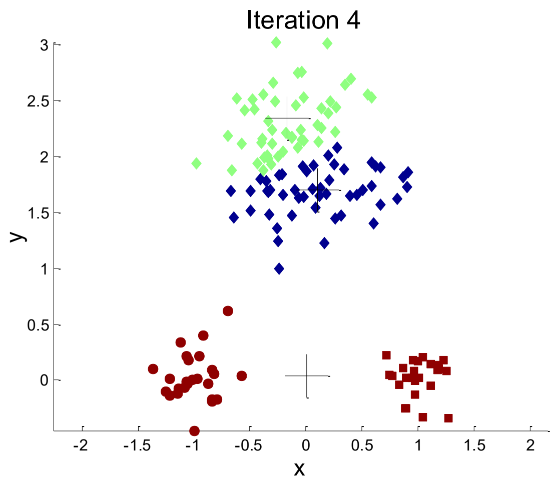
Importance of Choosing Initial Centroids …

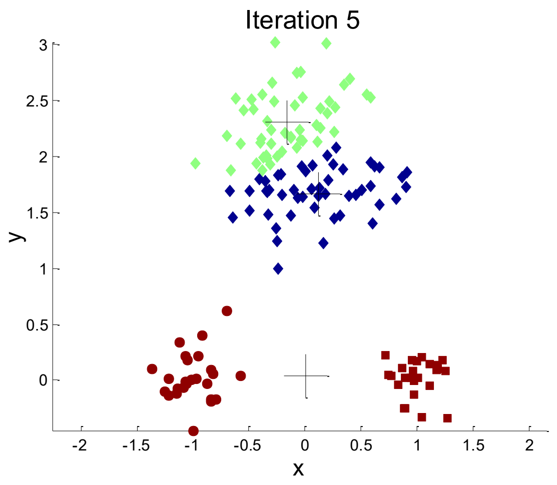




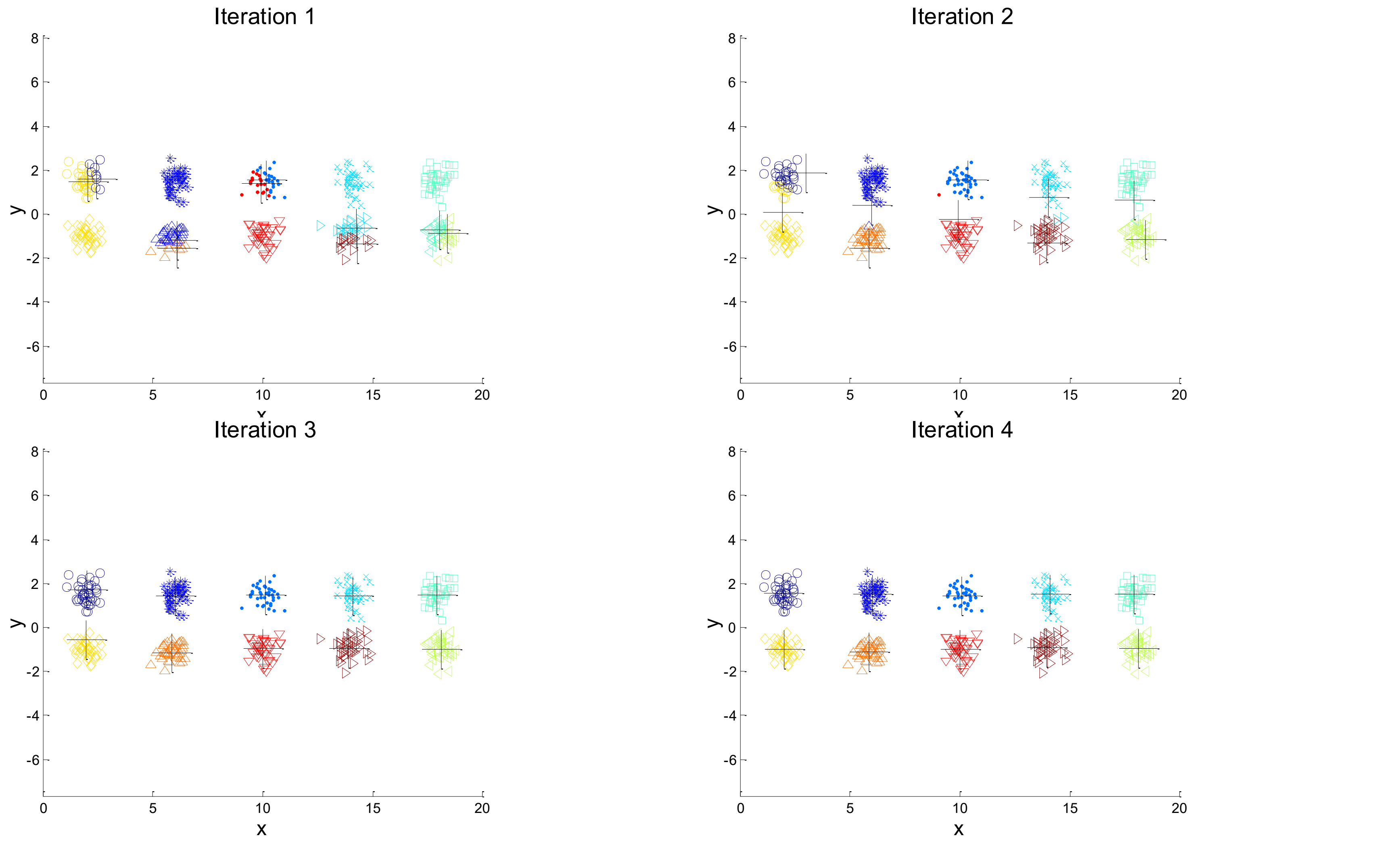
Sub-optimal Clustering





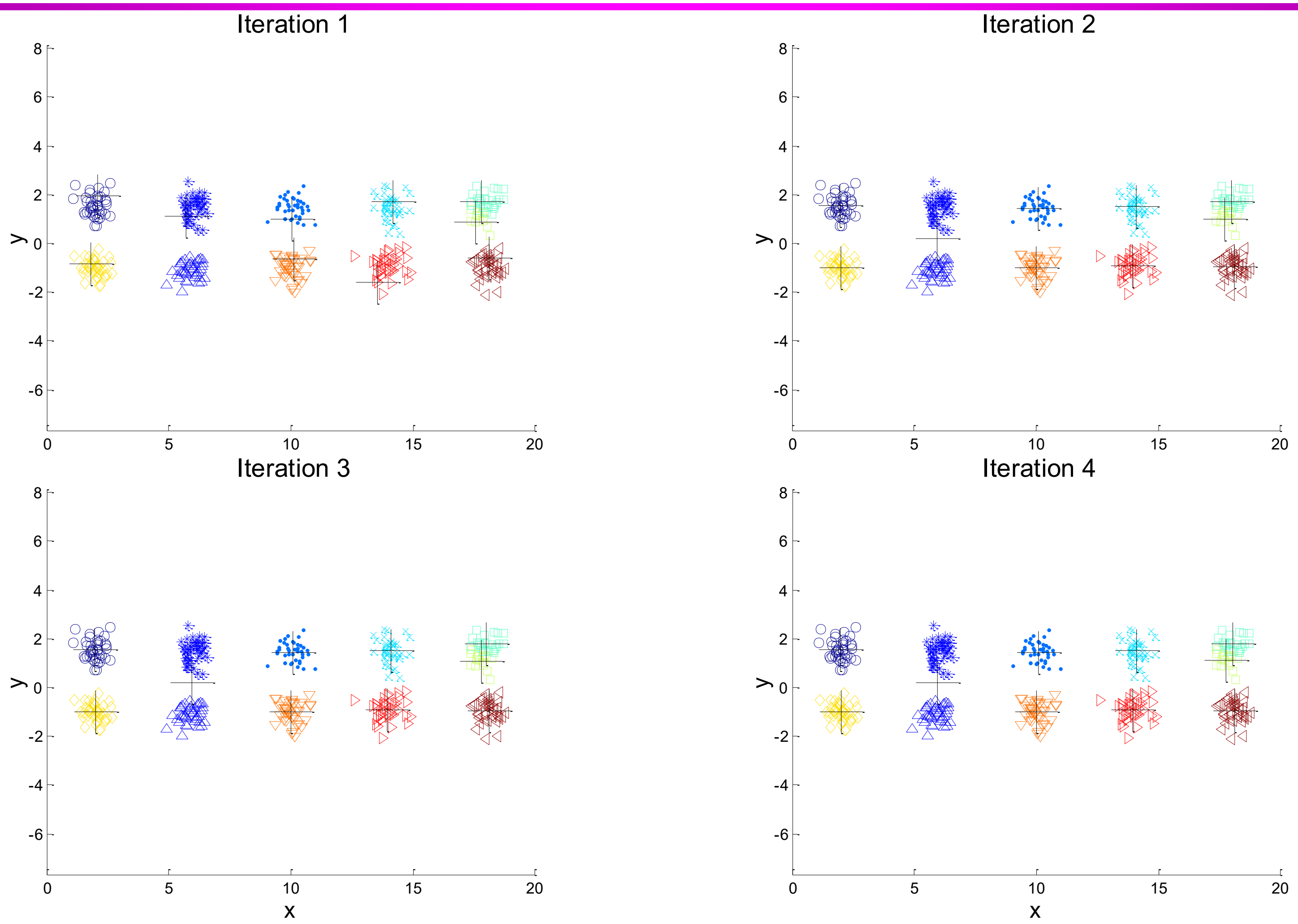


10 Clusters Example



Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



Starting with some pairs of clusters having three initial centroids, while other have only one

Solutions to Initial Centroids Problem

- Multiple runs
  - While simple, this strategy might not work very well, depending on the data set and the number of clusters sought
- Sample and use hierarchical clustering to determine initial centroids
  - *K* clusters are extracted from the hierarchical clustering, and the centroids of those clusters are used as the initial centroids
  - Practical only if (1) the sample is relatively small, e.g., a few hundred to a few thousand (hierarchical clustering is expensive), and (2) *K* is relatively small compared to the sample size
- Generate a larger number of clusters and then perform a hierarchical clustering
- K-means++, Bisecting K-means (examples)

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side

- Sample and use hierarchical clustering to determine initial centroids

- <u>Generate a larger number of clusters and then perform a hierarchical clustering</u>

- **K-means++, Bisecting K-means (examples)**

K-means++

- This approach can be slower than random initialization, but very consistently produces better results in terms of SSE
  - It might seem that this approach might tend to choose outliers for centroids, but because outliers are rare, by definition, this is unlikely

The intuition behind this approach is that spreading out the *k* initial cluster centers is a good thing

Select a set of initial centroids, *C*

Algorithm 7.2 K-means++ initialization algorithm.

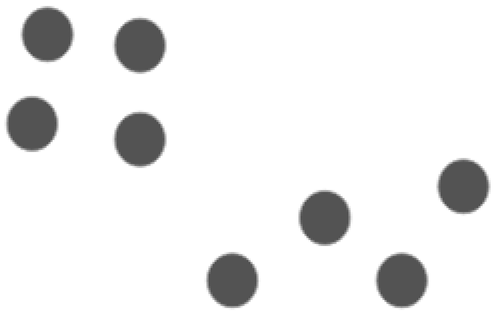
```
1: For the first centroid, pick one of the points at random.
2: for $i = 1$ to $number\ of\ trials$ do
3:     Compute the distance, $d(x)$, of each point to its closest centroid.
4:     Assign each point a probability proportional to each point's $d(x)^2$.
5:     Pick new centroid from the remaining points using the weighted probabilities.
6: end for
```

A point having maximum distance from the nearest centroid is most likely to be selected next as a centroid

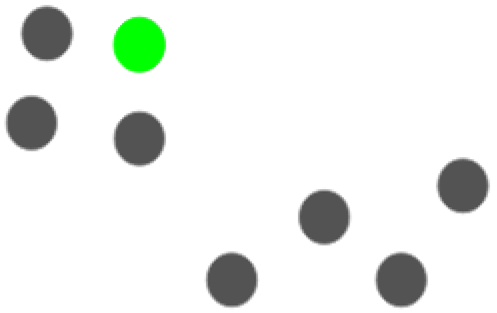
K-means++

https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

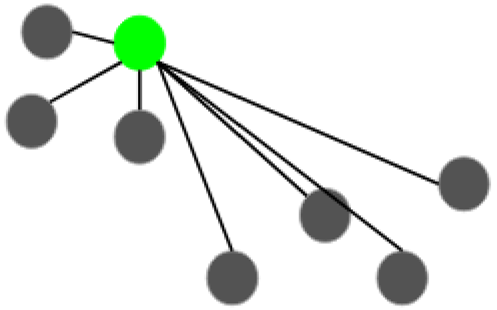
K=3



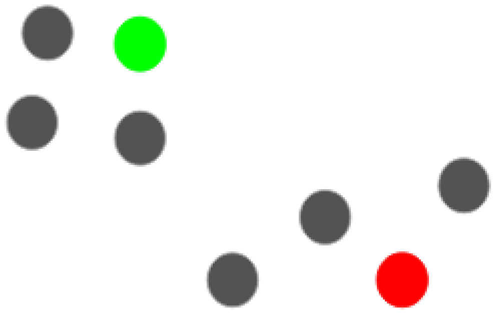
Randomly pick a data point as a cluster centroid



Calculate the distance d(x) of each data point with this centroid



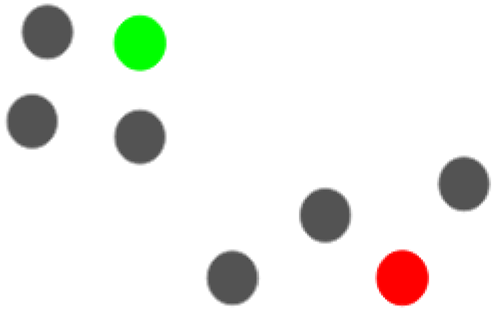
The next centroid will be the one whose squared distance $d(x)^2$ is the farthest from the current centroid

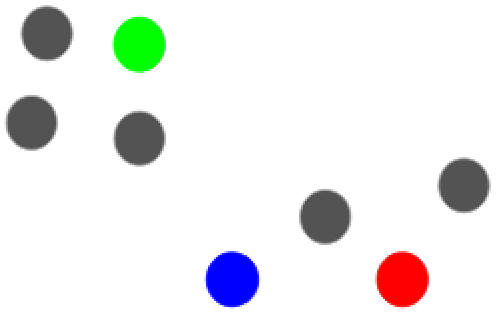


K-means++

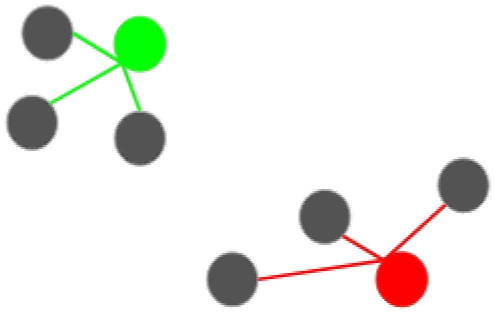
https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-k-means-clustering/

The next centroid will be the one whose squared distance $d(x)^2$ is the farthest from the current centroid





To select the last centroid, we will take the distance of each point from its closest centroid and the point having the largest squared distance will be selected as the next centroid



Bisecting K-means

- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering

```
1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3:    Select a cluster from the list of clusters
4:    **for** $i = 1$ to $number\_of\_iterations$ **do**
5:       Bisect the selected cluster using basic K-means
6:    **end for**
7:    Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains $K$ clusters
```

Less susceptible to initialization problems: it performs several trial bisections and takes the one with the lowest SSE, and because there are only two centroids at each step

Bisecting K-means

There are a number of different ways to choose which cluster to split

We can choose the largest cluster at each step, choose the one with the largest SSE, or use a criterion based on both size and SSE

Different choices result in different clusters

```
1: Initialize the list of clusters to contain the cluster containing all points.
2: repeat
3:     Select a cluster from the list of clusters
4:     for $i = 1$ to $number\_of\_iterations$ do
5:         Bisect the selected cluster using basic K-means
6:     end for
7:     Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: until Until the list of clusters contains $K$ clusters
```

Less susceptible to initialization problems: it performs several trial bisections and takes the one with the lowest SSE, and because there are only two centroids at each step

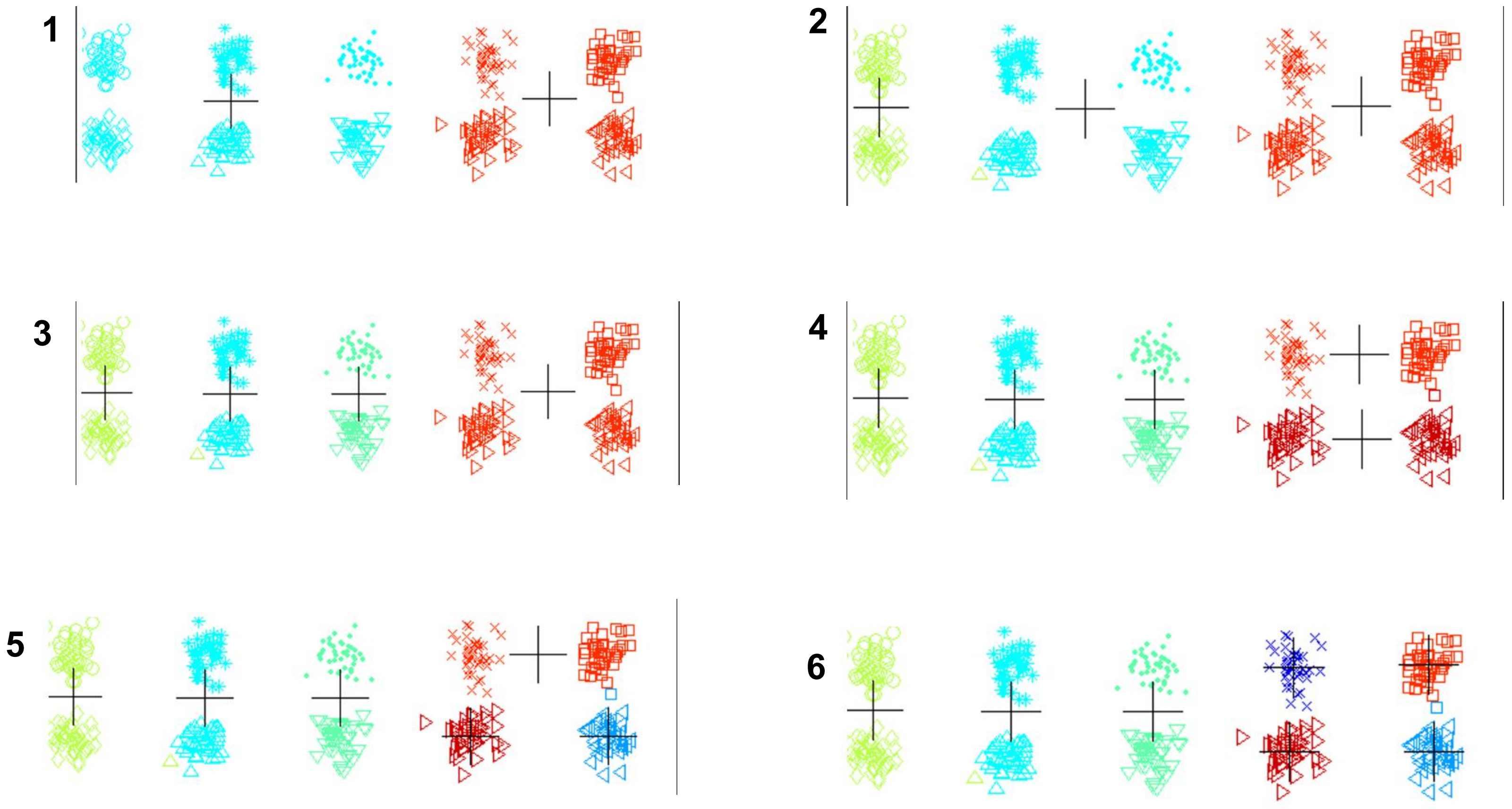
Bisecting K-means

- Because we are using the K-means algorithm “locally,” i.e., to bisect individual clusters, the final set of clusters does not represent a clustering that is a global minimum with respect to the total SSE

- Thus, we often refine the resulting clusters by using their cluster centroids as the initial centroids for the standard K-means algorithm

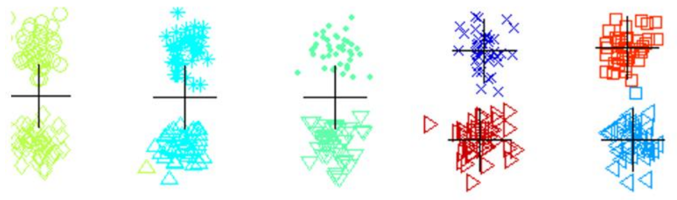
- Finally, by recording the sequence of clusterings produced as K-means bisects clusters, we can also use bisecting K-means to produce a hierarchical clustering

Bisecting K-means Example

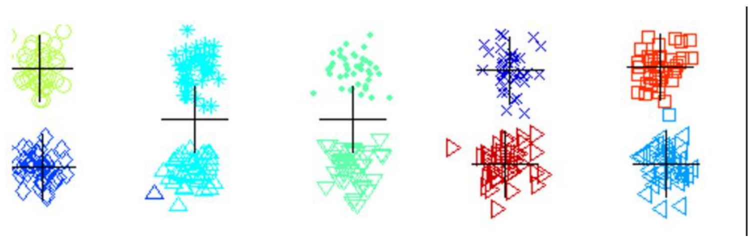


Bisecting K-means Example …

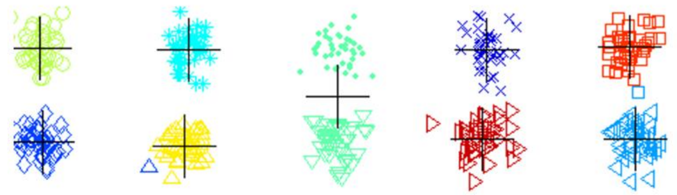
6



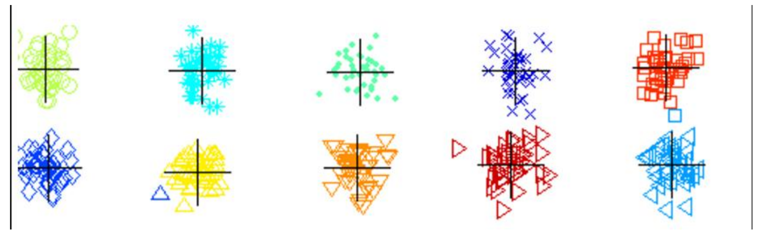
7



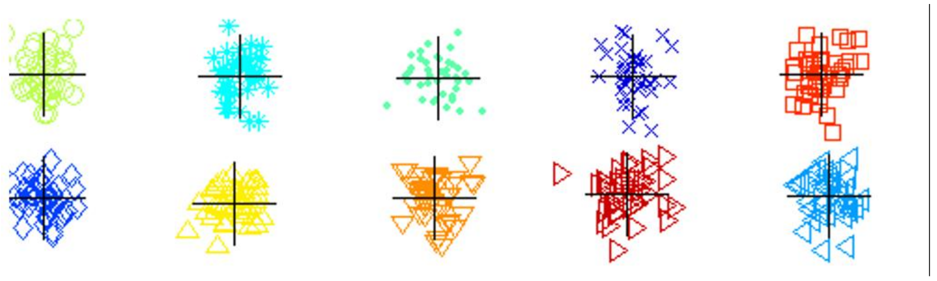
8



9



10



Strengths and Weaknesses

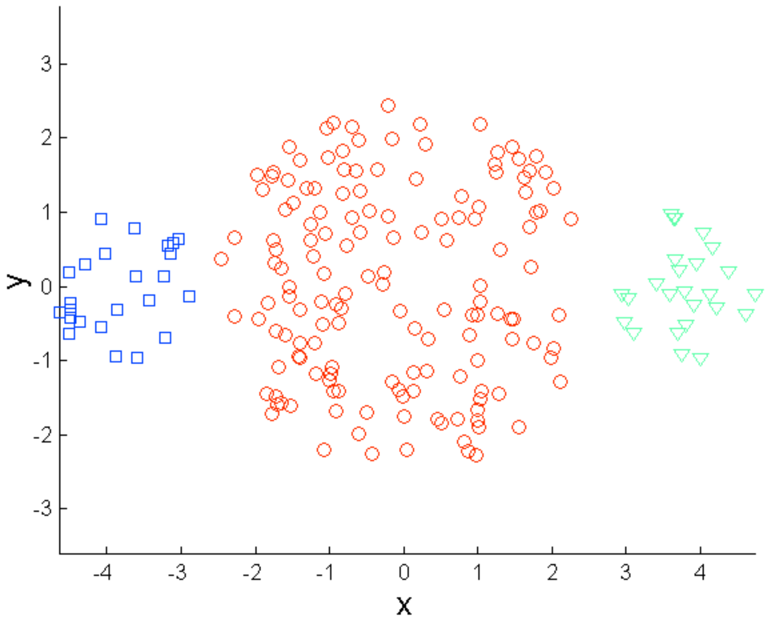
- K-means is simple and can be used for a wide variety of data types.
- It is also quite efficient, even though multiple runs are often performed.
- K-means is not suitable for all types of data.
  - It cannot handle non-globular clusters or clusters of different sizes and densities.
- K-means also has trouble clustering data that contains outliers.
  - Outlier detection and removal can help significantly in such situations.
- Finally, K-means is restricted to data for which there is a notion of a center (centroid).

Limitations of K-means

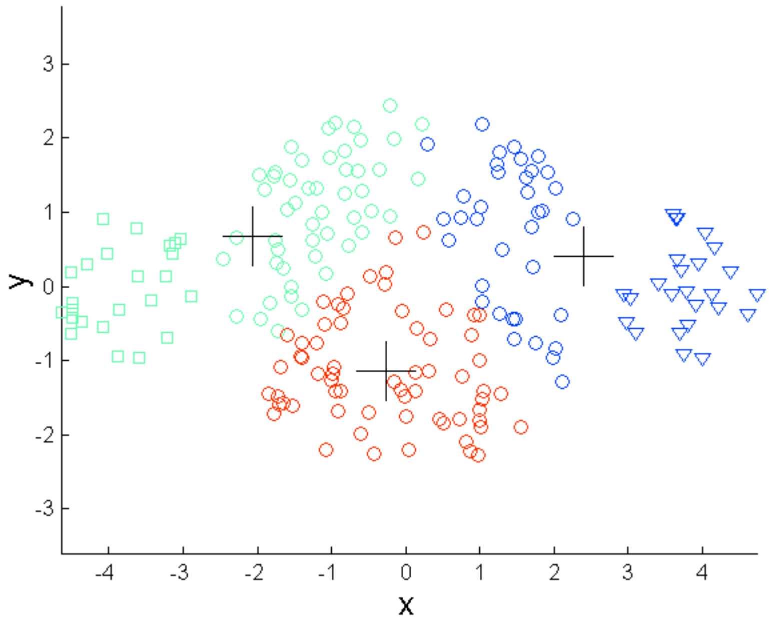
- K-means has problems when clusters are of differing
  - Sizes, Densities, Non-globular shapes
  - K-means objective function is a mismatch for these kinds of clusters because it is minimized by globular clusters of equal size and density or by clusters that are well separated

- K-means has problems when the data contains outliers

Limitations of K-means: Differing Sizes

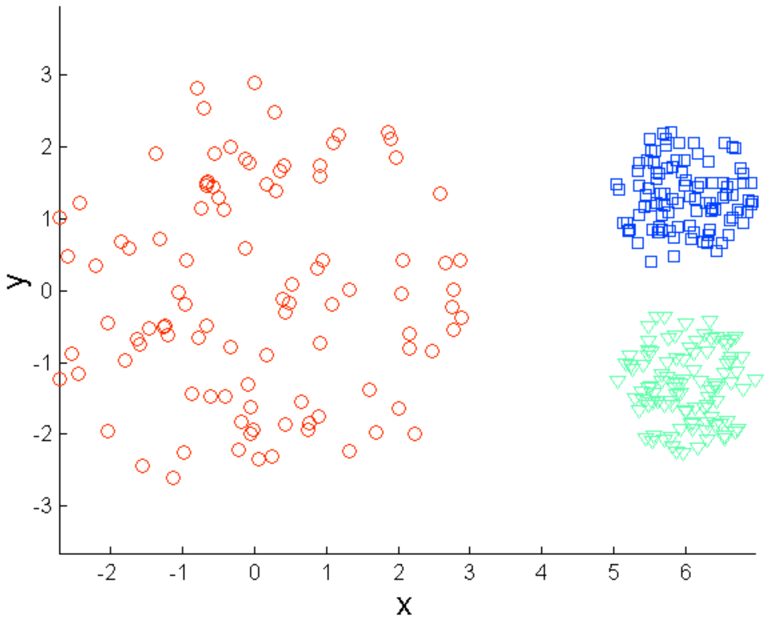


Original Points

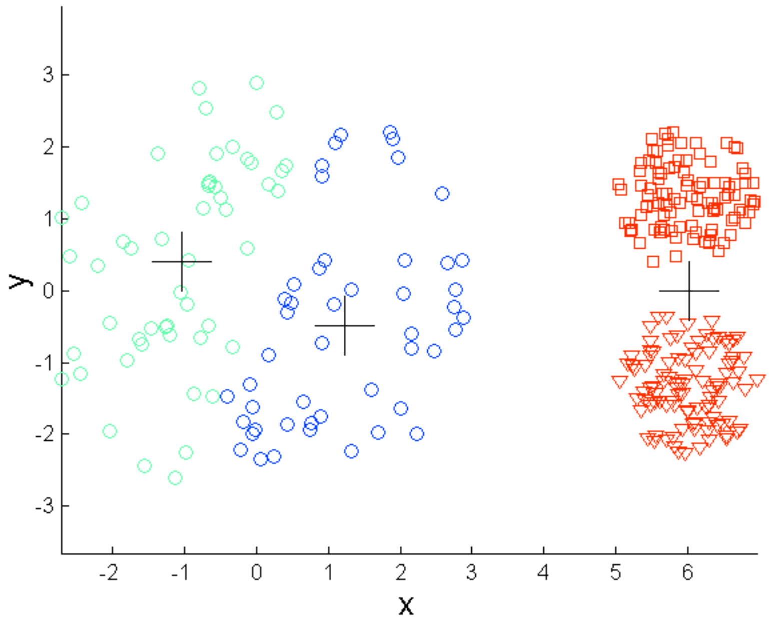


K-means (3 Clusters)

Limitations of K-means: Differing Density

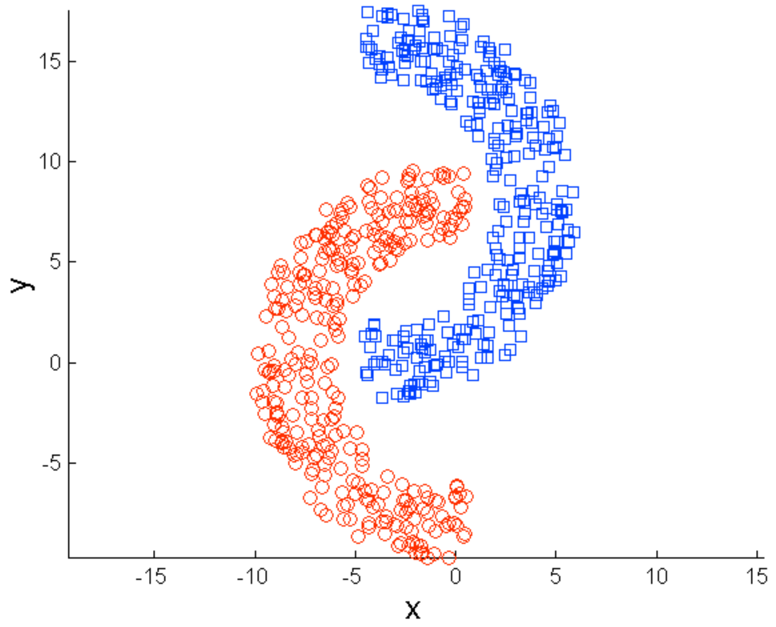


Original Points

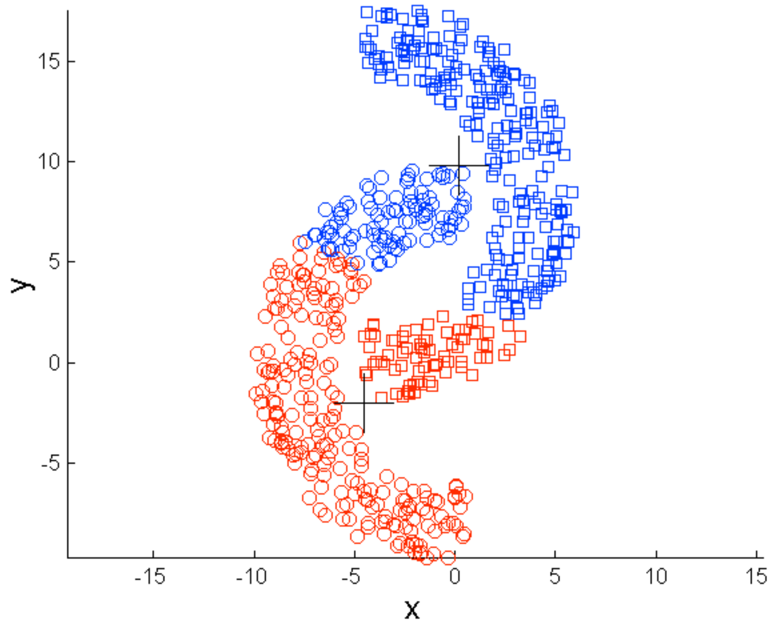


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes



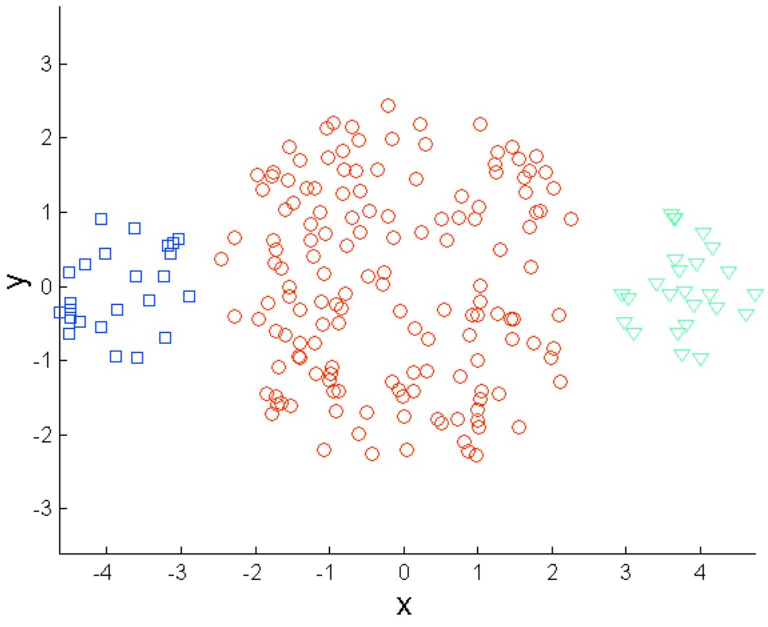
Original Points

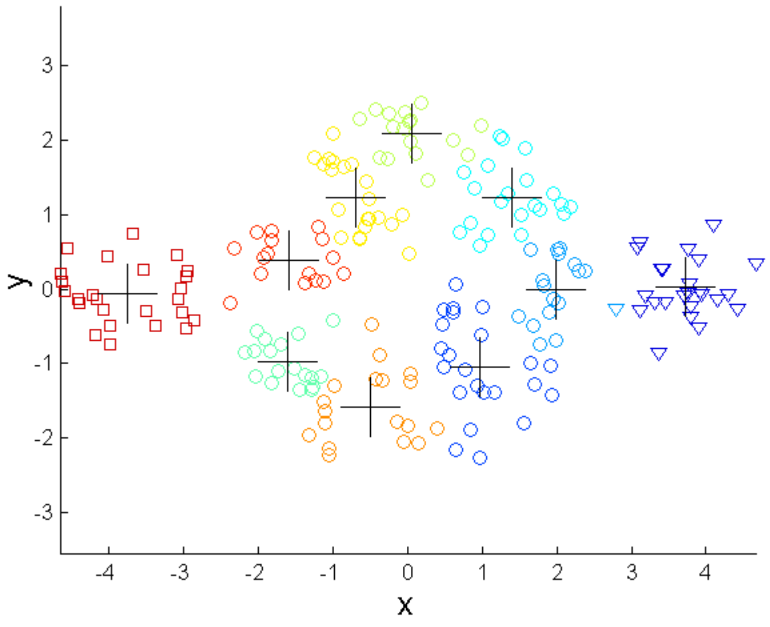


K-means (2 Clusters)

Overcoming K-means Limitations

Generate a larger number of clusters and then perform a hierarchical clustering





Original Points

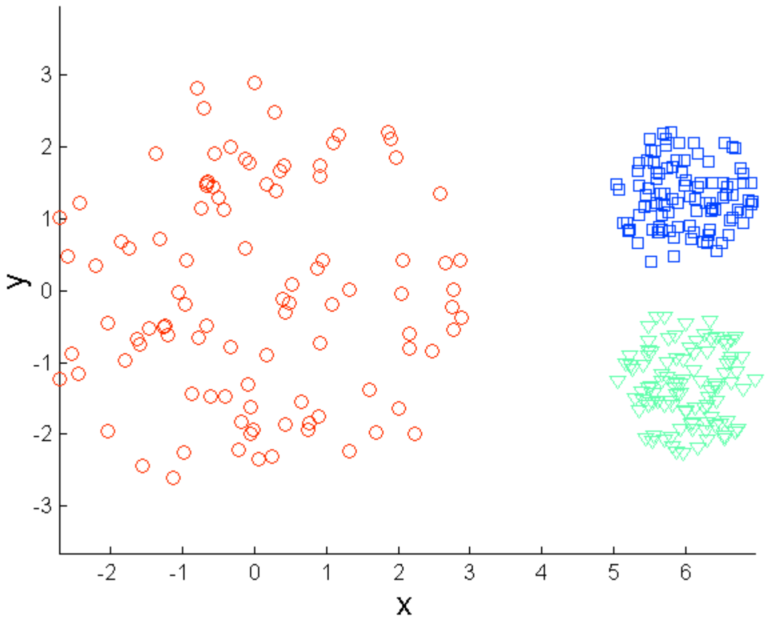
K-means Clusters

One solution is to use many clusters.

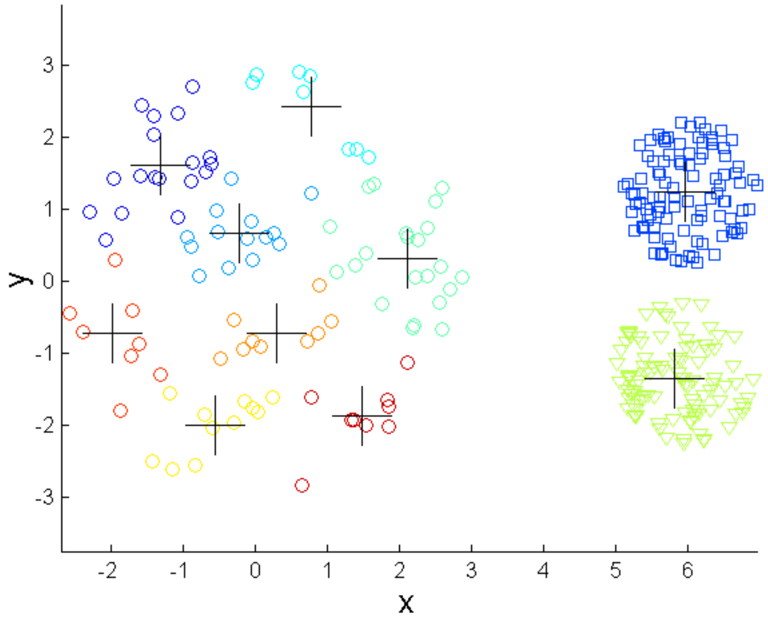
Find parts of clusters, but need to put together.

Overcoming K-means Limitations

<u>**Generate a larger number of clusters and then perform a hierarchical clustering**</u>



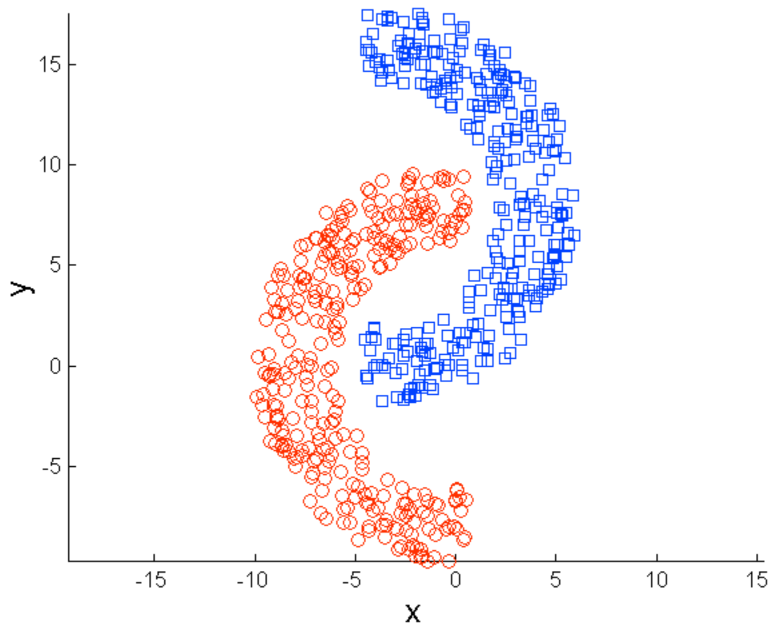
Original Points

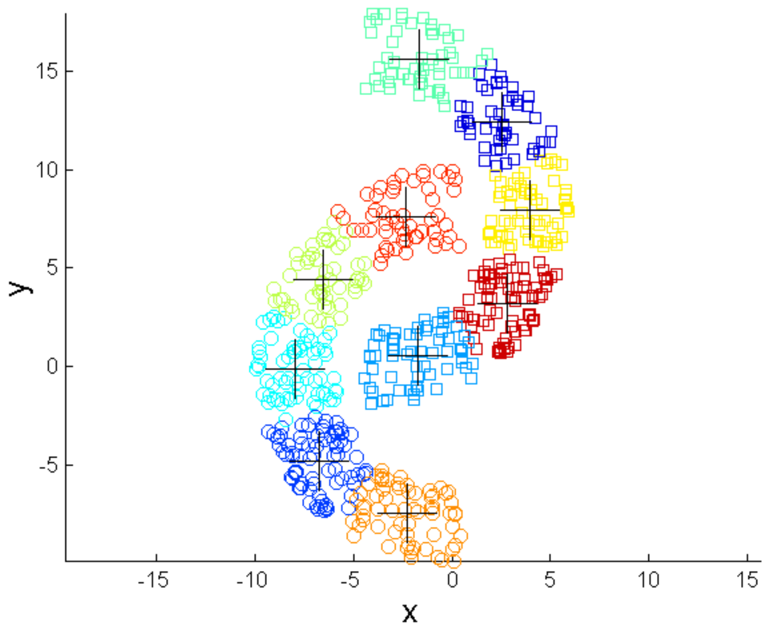


K-means Clusters

Overcoming K-means Limitations

<u>**Generate a larger number of clusters and then perform a hierarchical clustering**</u>



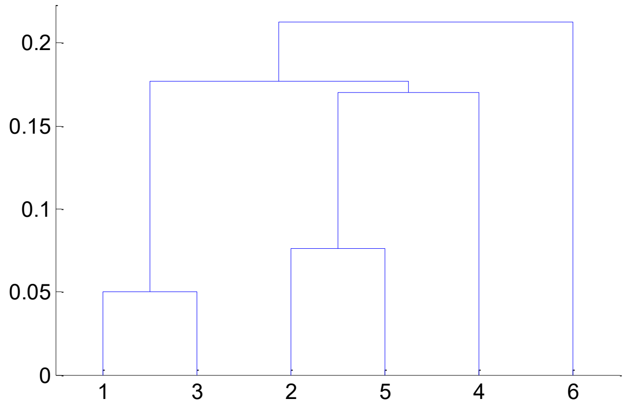


Original Points

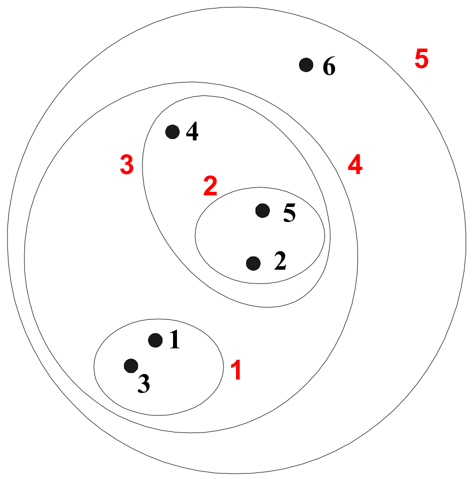
K-means Clusters

Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits



Dendrogram

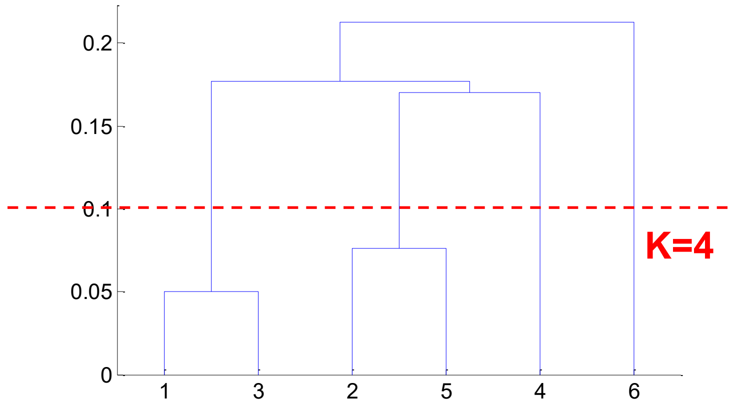


Nested cluster diagram

Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Some studies have suggested that these algorithms can produce better-quality clusters. However, they are expensive in terms of their computational and storage requirements



Hierarchical Clustering

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

Agglomerative Clustering Algorithm

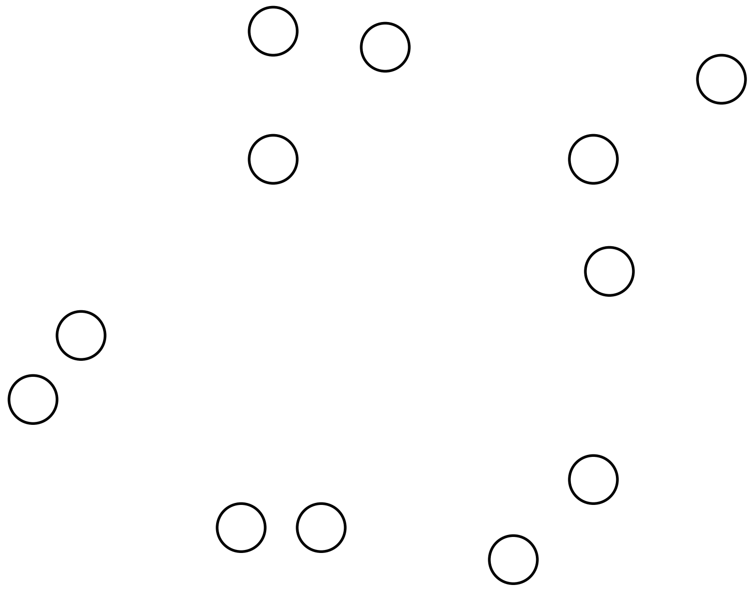
- Most popular hierarchical clustering technique
- Basic algorithm is straightforward

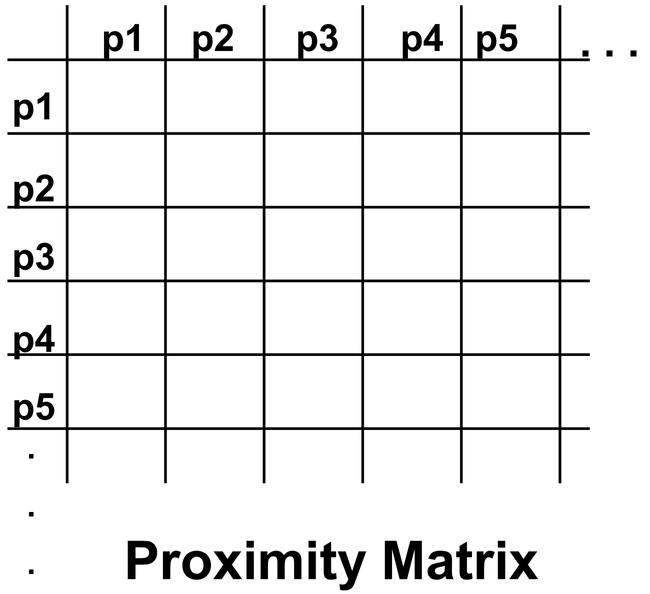
```
1. Compute the proximity matrix
2. Let each data point be a cluster
3. Repeat
4.         Merge the two closest clusters
5.         Update the proximity matrix
6. Until only a single cluster remains
```

- Key operation is the computation of the proximity of two clusters
  – Different approaches to defining the distance between clusters distinguish the different algorithms

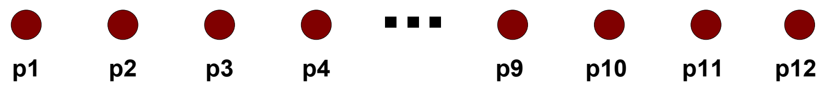
Starting Situation

- Start with clusters of individual points and a proximity matrix



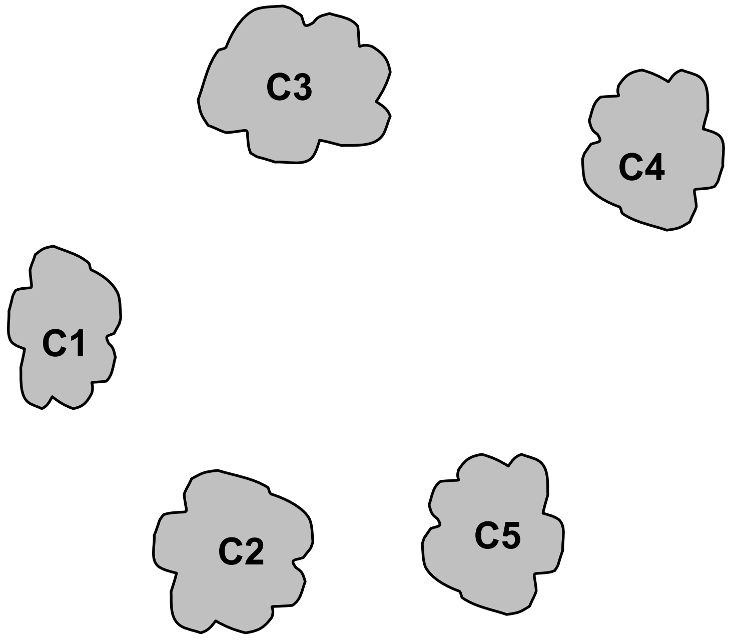


Proximity Matrix



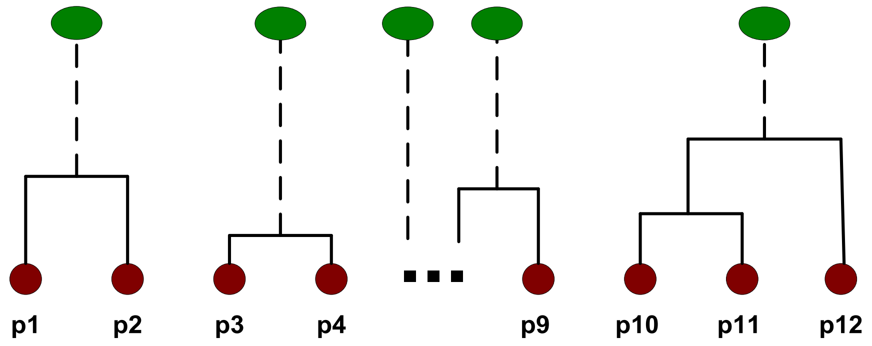
Intermediate Situation

- After some merging steps, we have some clusters



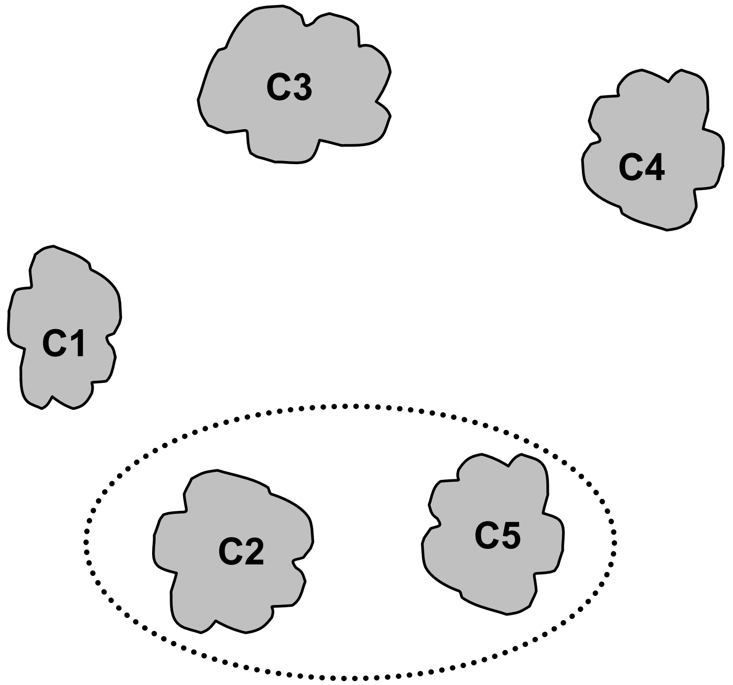
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

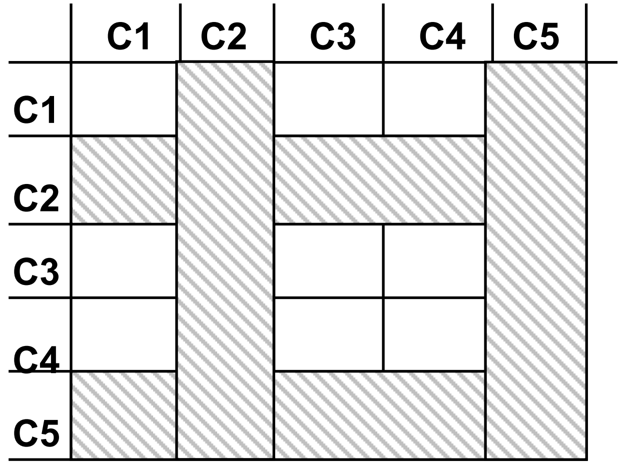
Proximity Matrix



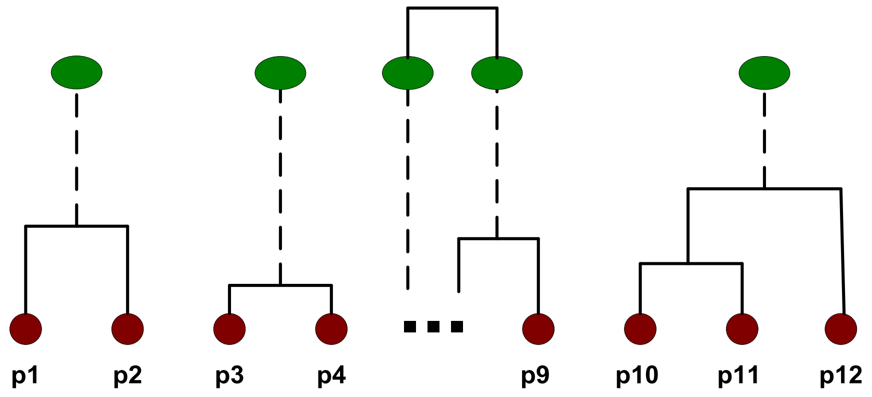
Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.



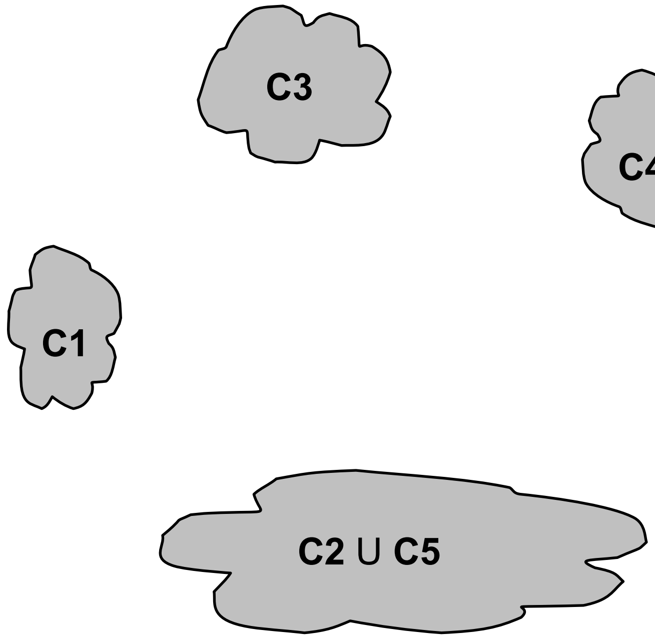


Proximity Matrix



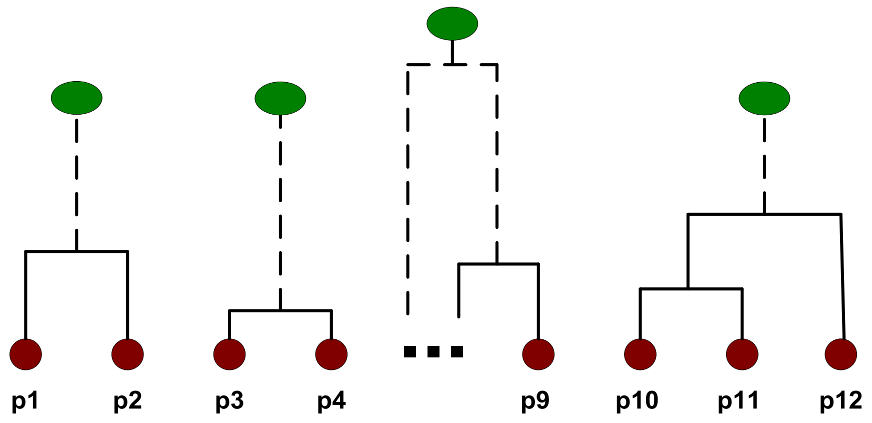
After Merging

- The question is “How do we update the proximity matrix?”

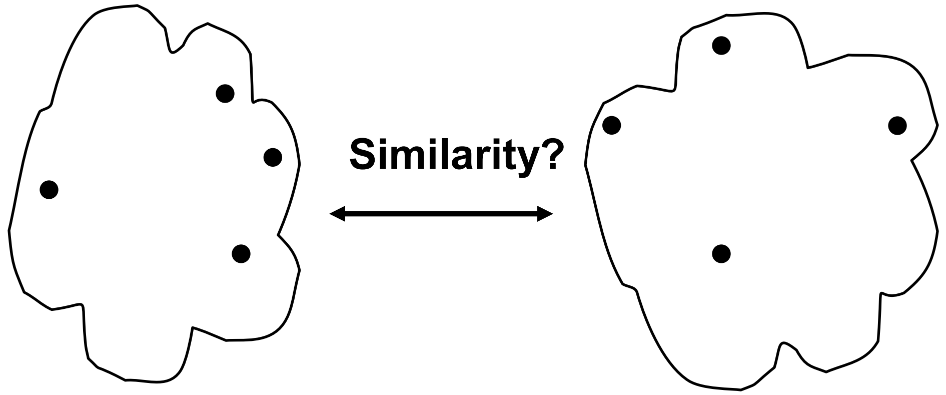


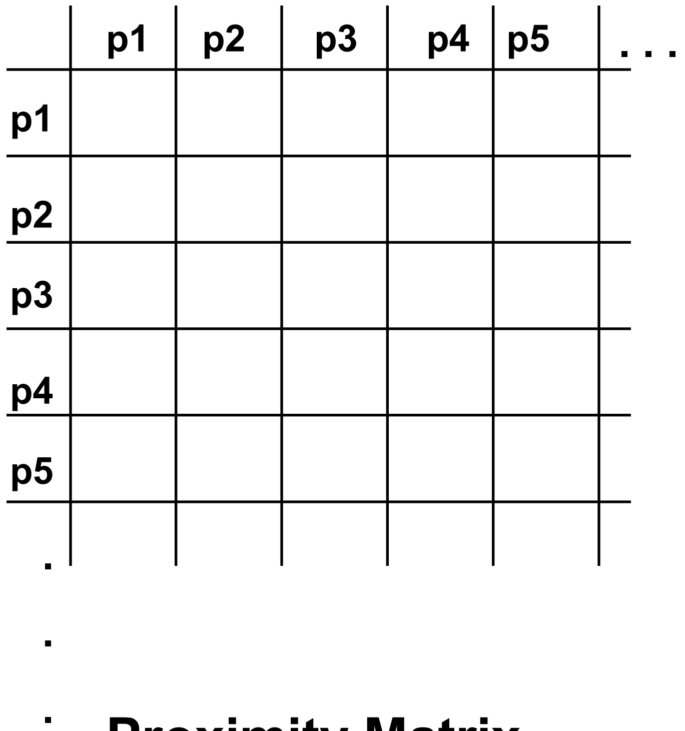
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

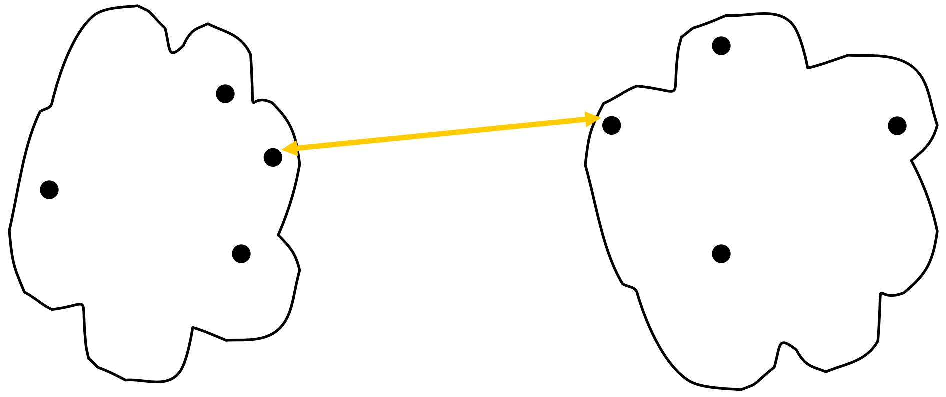




Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  – Ward's Method uses squared error

How to Define Inter-Cluster Similarity

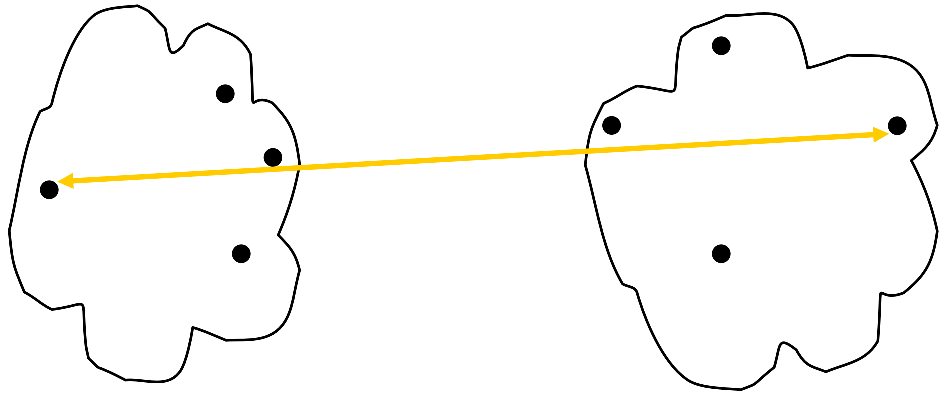


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  – Ward's Method uses squared error

How to Define Inter-Cluster Similarity

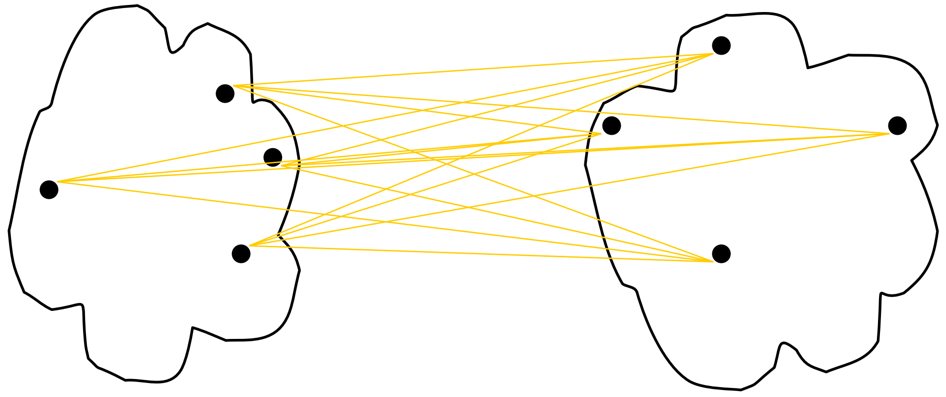


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

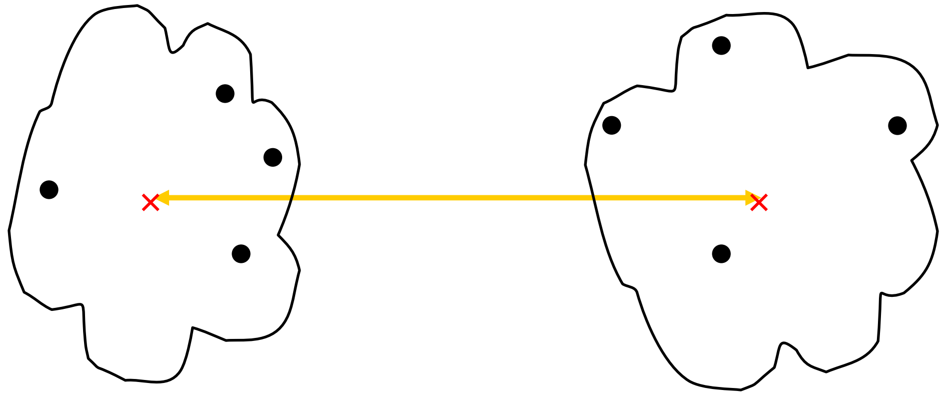


	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

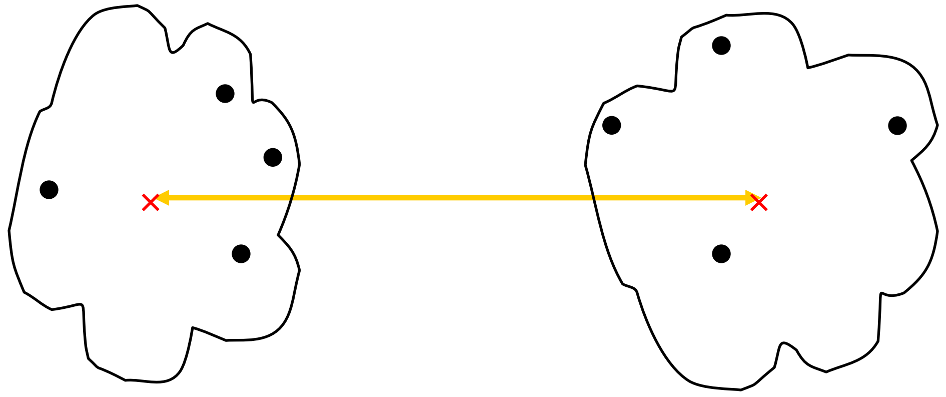


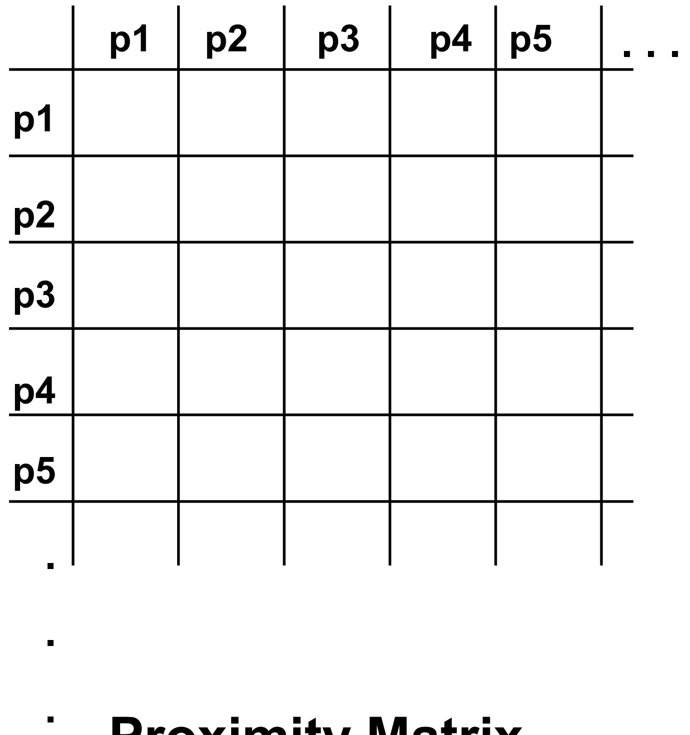
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

How to Define Inter-Cluster Similarity

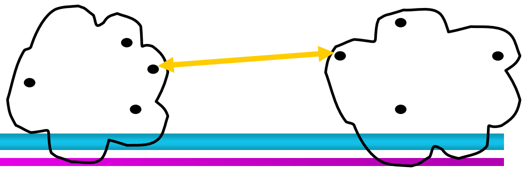




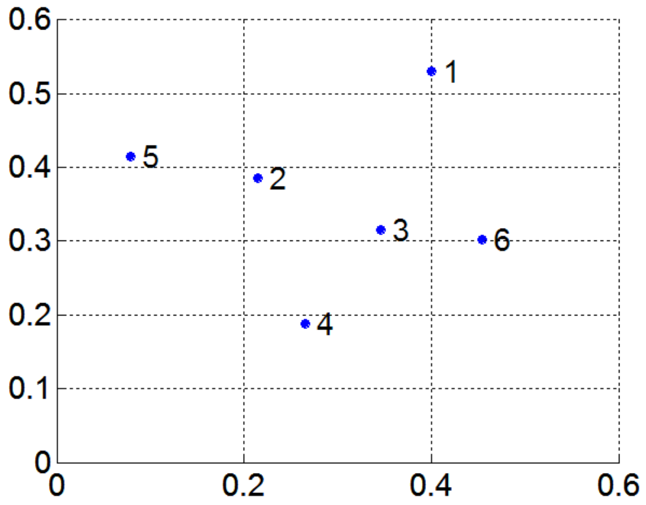
Proximity Matrix

- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

MIN or Single Link



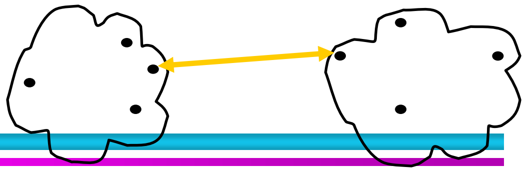
- Proximity of two clusters is based on the two closest points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph (the shortest edge between two nodes in different subsets of nodes)
- Example:



Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

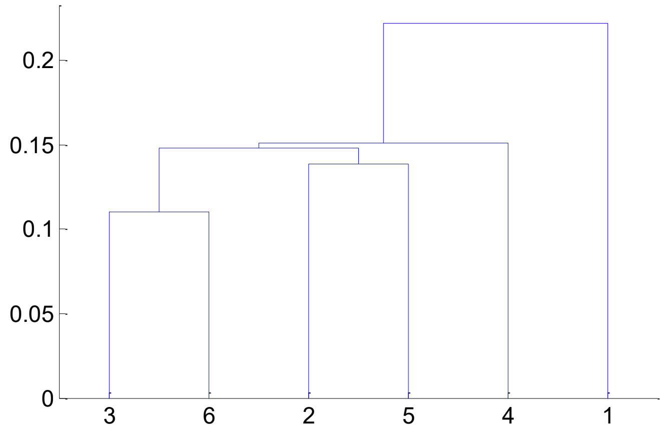
MIN or Single Link



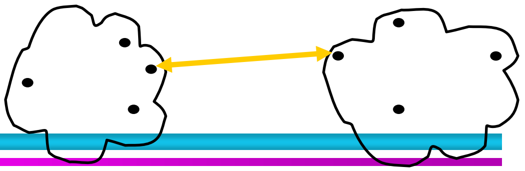
	p1	p2	p3	p4	p5	p6
p1	0.00					
p2	0.24	0.00				
p3	0.22	0.15	0.00			
p4	0.37	0.20	0.15	0.00		
p5	0.34	0.14	0.28	0.29	0.00	
p6	0.23	0.25	0.11	0.22	0.39	0.00

	p1	p2	p4	p5	p3+p6
p1	0.00				
p2	0.24	0.00			
p4	0.37	0.20	0.00		
p5	0.34	0.14	0.29	0.00	
p3+p6	0.22	0.15	0.15	0.28	0.11

dist({3,6}, {1}) = min(dist(3,1), dist(6,1)) =
min(0.22,0.23) = 0.22
…



MIN or Single Link

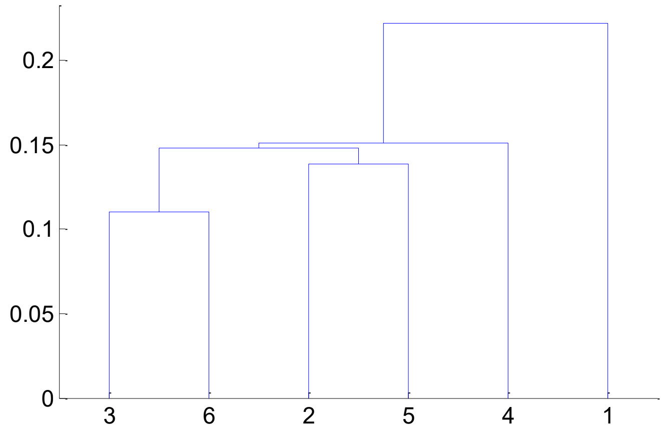


	p1	p2	p4	p5	p3+p6
p1	0.00				
p2	0.24	0.00			
p4	0.37	0.20	0.00		
p5	0.34	0.14	0.29	0.00	
p3+p6	0.22	0.15	0.15	0.28	0.11

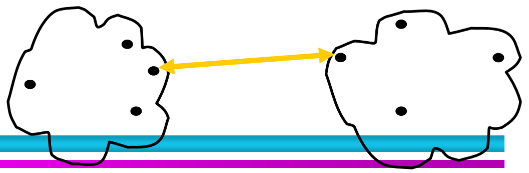
	p1	p4	p3+p6	p2+p5
p1	0.00			
p4	0.37	0.00		
p3+p6	0.22	0.15	0.11	
p2+p5	0.24	0.20	0.15	0.14

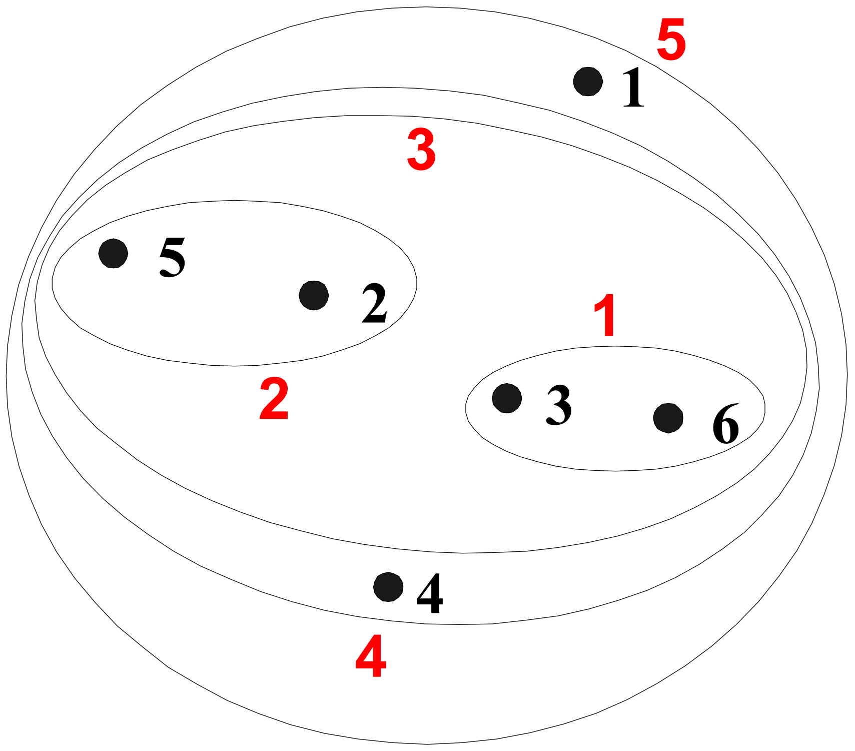
... e assim sucessivamente...

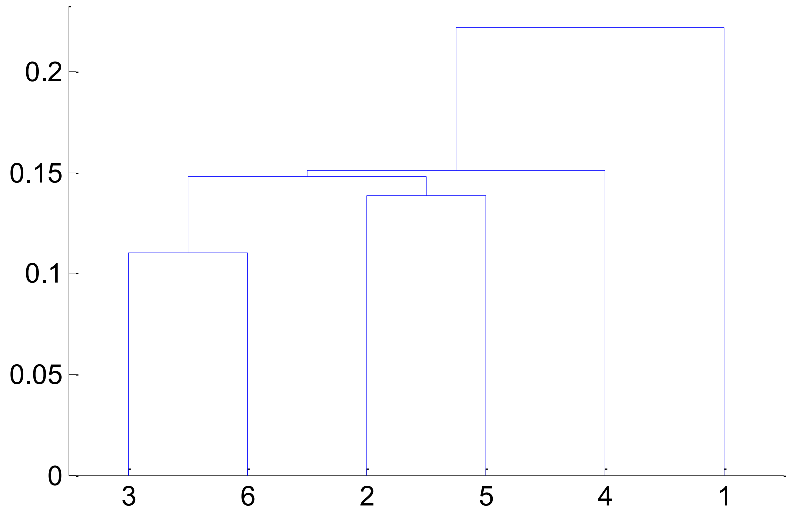
$$\begin{aligned} dist(\{3,6\},\{2,5\}) &= \min(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\ &= \min(0.15, 0.25, 0.28, 0.39) \\ &= 0.15. \end{aligned}$$



Hierarchical Clustering: MIN



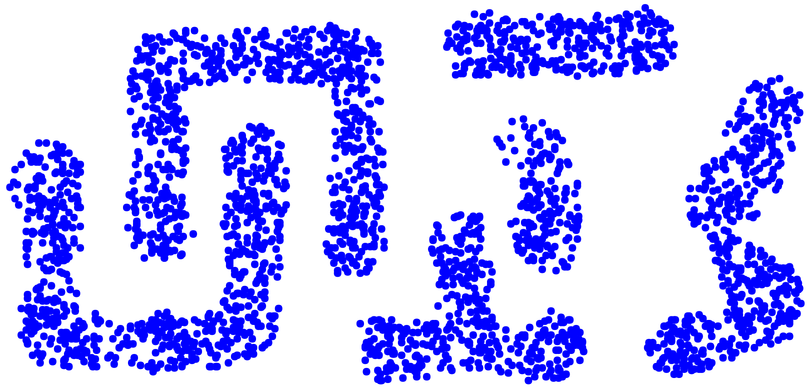


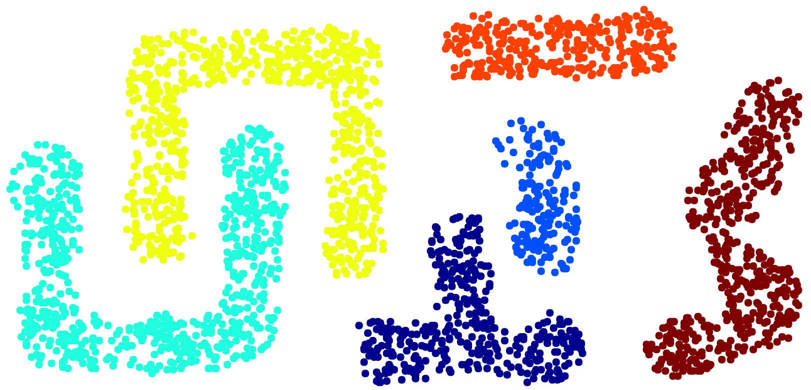


Nested Clusters

Dendrogram

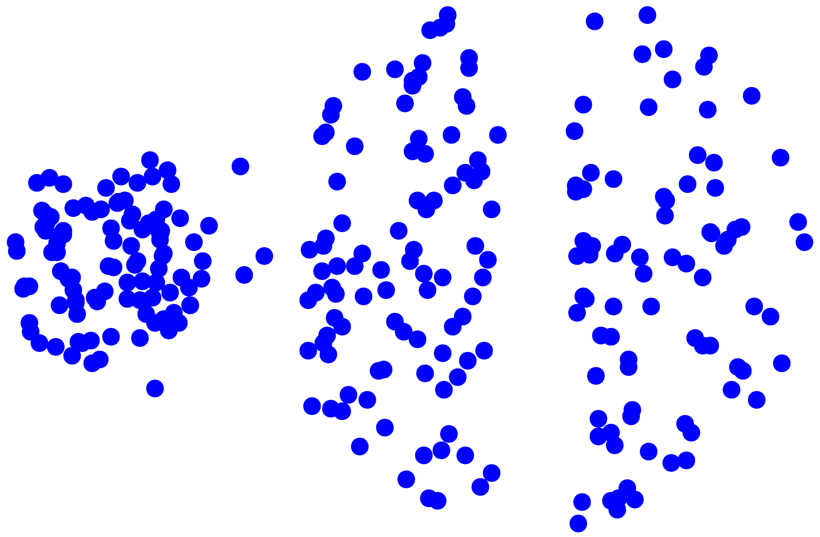
Strength of MIN

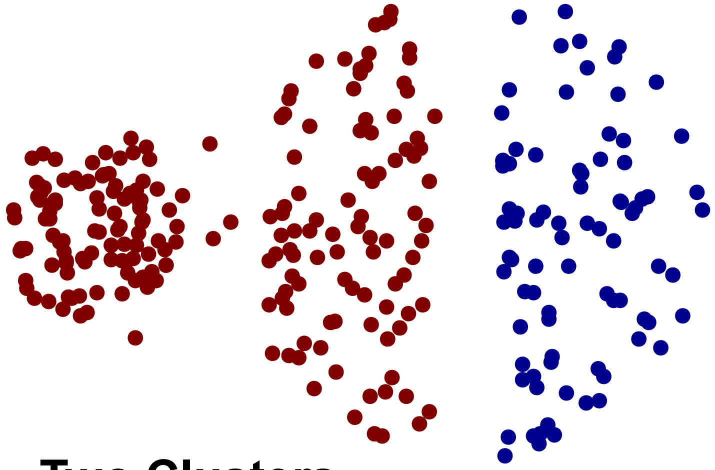

Original Points

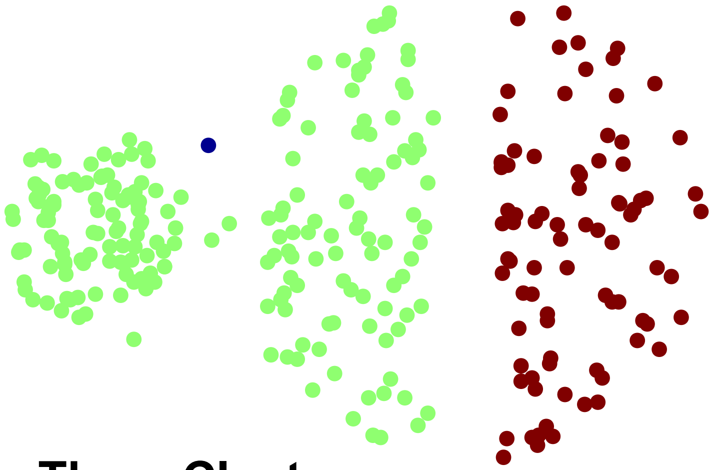

Six Clusters

- Can handle non-elliptical shapes

Limitations of MIN

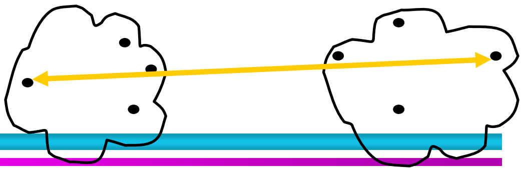

Original Points


Two Clusters

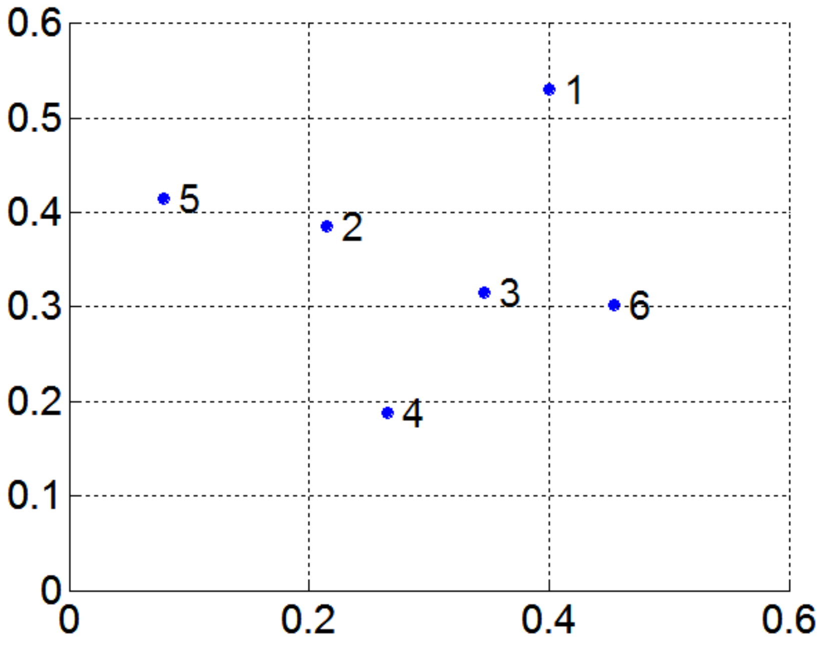

Three Clusters

- Sensitive to noise and outliers

MAX or Complete Linkage



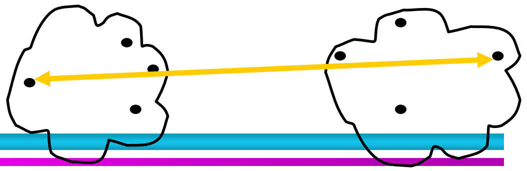
- Proximity of two clusters is based on the two most distant points in the different clusters
  - Determined by all pairs of points in the two clusters (the longest edge between two nodes in different subsets of nodes)



Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

MAX or Complete Linkage



	p1	p2	p3	p4	p5	p6
p1	0.00					
p2	0.24	0.00				
p3	0.22	0.15	0.00			
p4	0.37	0.20	0.15	0.00		
p5	0.34	0.14	0.28	0.29	0.00	
p6	0.23	0.25	0.11	0.22	0.39	0.00

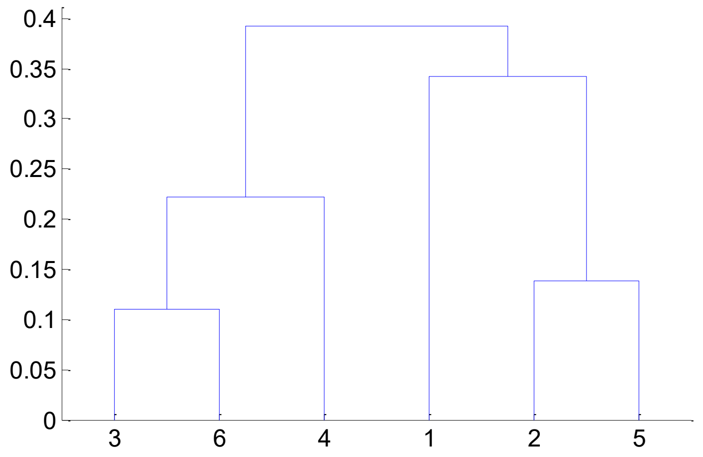
	p1	p2	p4	p5	p3+p6
p1	0.00				
p2	0.24	0.00			
p4	0.37	0.20	0.00		
p5	0.34	0.14	0.29	0.00	
p3+p6	0.23	0.25	0.22	0.39	0.11

$$\begin{aligned} dist(\{3,6\},\{1\}) &= \max(dist(3,1), dist(6,1)) \\ &= \max(0.22, 0.23) \\ &= 0.23. \end{aligned}$$

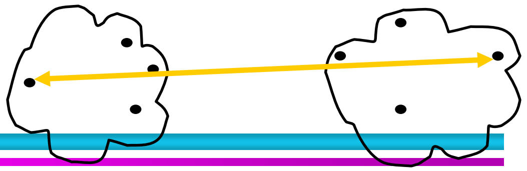
...

$$\begin{aligned} dist(\{3,6\},\{4\}) &= \max(dist(3,4), dist(6,4)) \\ &= \max(0.15, 0.22) \\ &= 0.22. \end{aligned}$$

...



MAX or Complete Linkage

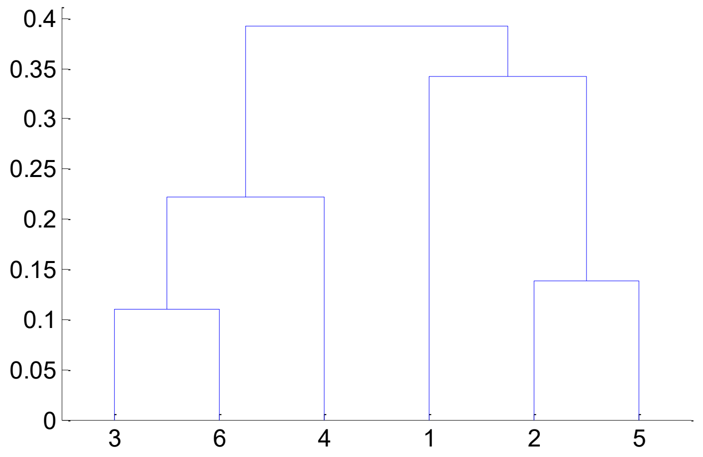


	p1	p2	p4	p5	p3+p6
p1	0.00				
p2	0.24	0.00			
p4	0.37	0.20	0.00		
p5	0.34	0.14	0.29	0.00	
p3+p6	0.23	0.25	0.22	0.39	0.11

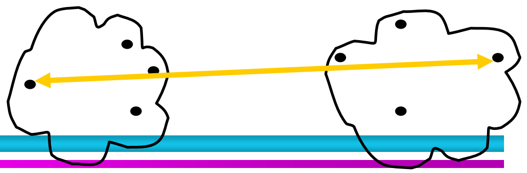
	p1	p4	p3+p6	p2+p5
p1	0.00			
p4	0.37	0.00		
p3+p6	0.22	0.15	0.11	
p2+p5	0.34	0.29	0.39	0.14

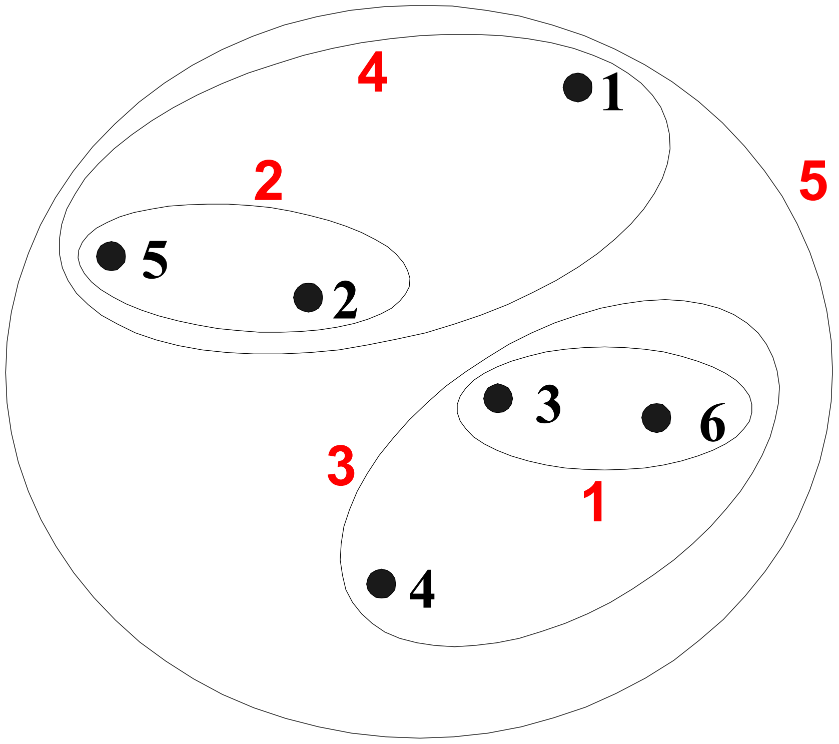
... e assim sucessivamente...

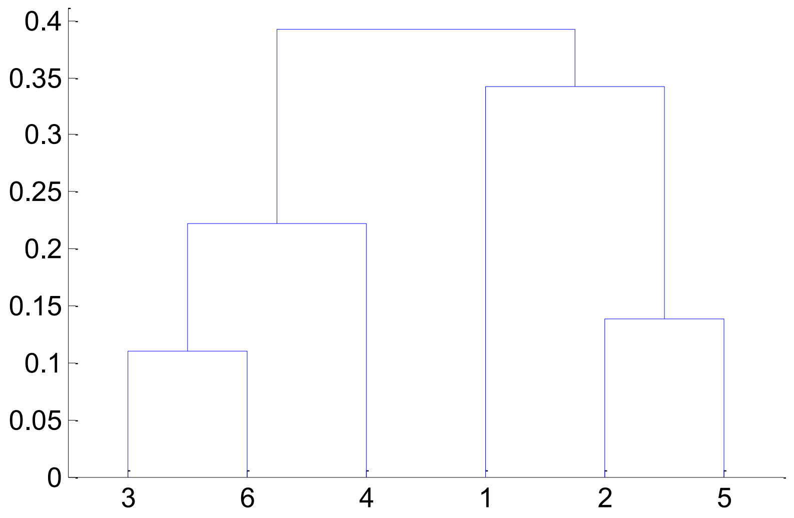
$$\begin{aligned} dist(\{3,6\},\{2,5\}) &= \max(dist(3,2), dist(6,2), dist(3,5), dist(6,5)) \\ &= \max(0.15, 0.25, 0.28, 0.39) \\ &= 0.39. \end{aligned}$$



Hierarchical Clustering: MAX

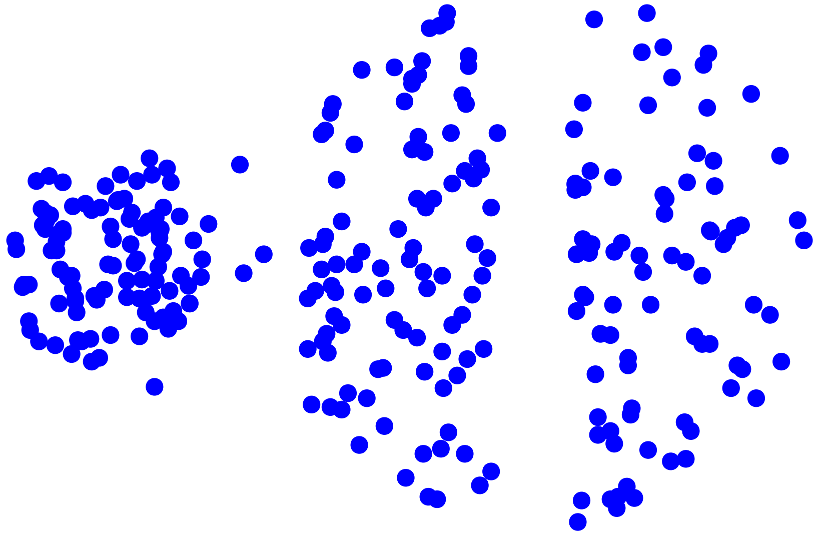


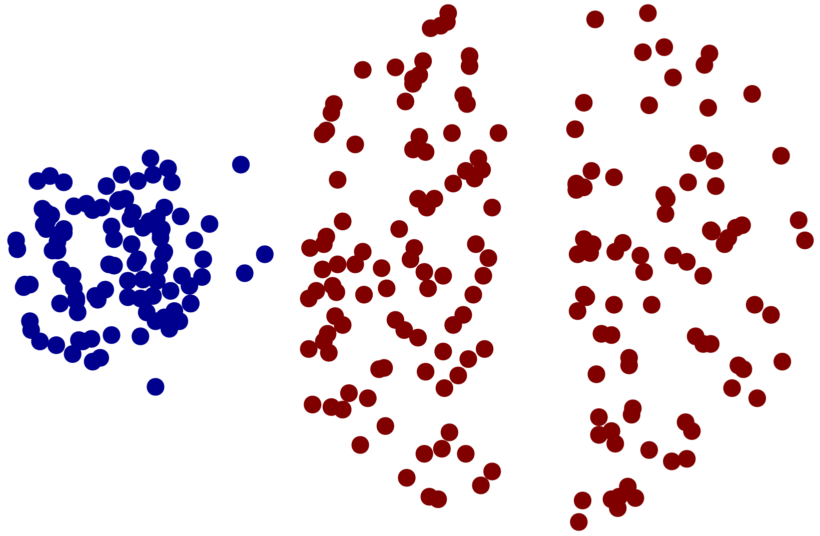




Nested Clusters **Dendrogram**

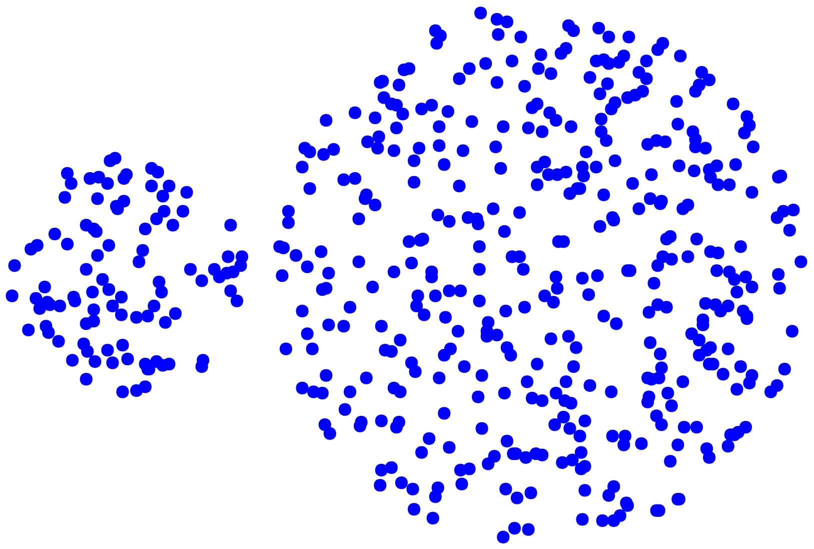
Strength of MAX

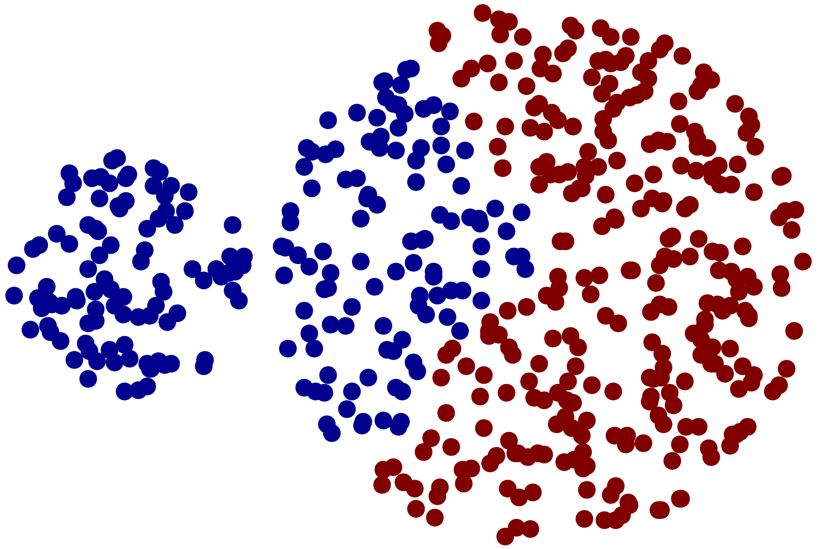

Original Points


Two Clusters

- Less susceptible to noise and outliers

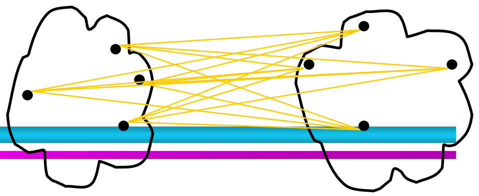
Limitations of MAX


Original Points


Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

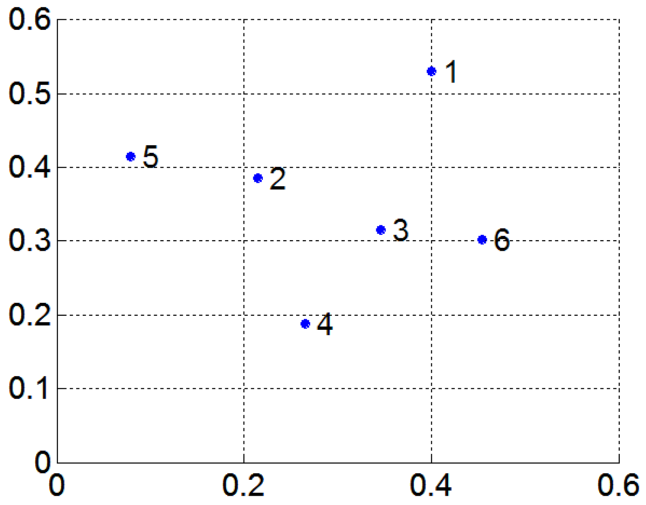
Group Average



- Proximity of two clusters is the average of pairwise proximity between points in the two clusters (average length of edges).

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum\limits_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| \times |\text{Cluster}_j|}$$

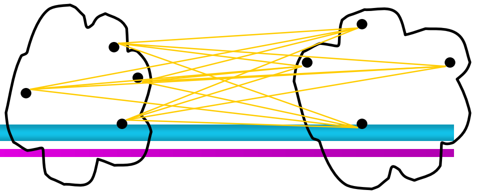
- Need to use average connectivity for scalability since total proximity favors large clusters



Distance Matrix:

	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Group Average



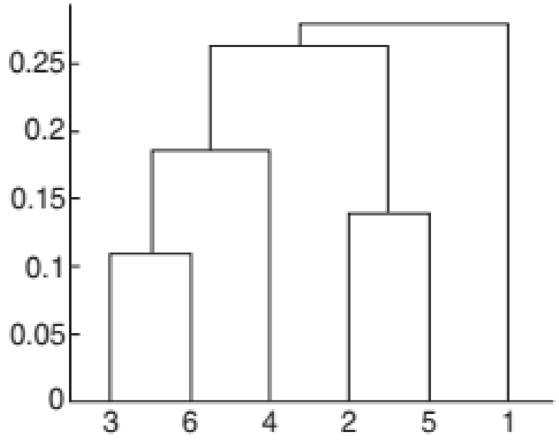
	p1	p2	p3	p4	p5	p6
p1	0.00					
p2	0.24	0.00				
p3	0.22	0.15	0.00			
p4	0.37	0.20	0.15	0.00		
p5	0.34	0.14	0.28	0.29	0.00	
p6	0.23	0.25	0.11	0.22	0.39	0.00

	p1	p2	p4	p5	p3+p6
p1	0.00				
p2	0.24	0.00			
p4	0.37	0.20	0.00		
p5	0.34	0.14	0.29	0.00	
p3+p6	?	?	?	?	0.11

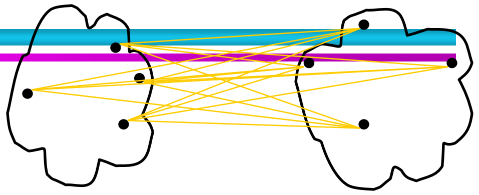
... e assim sucessivamente...

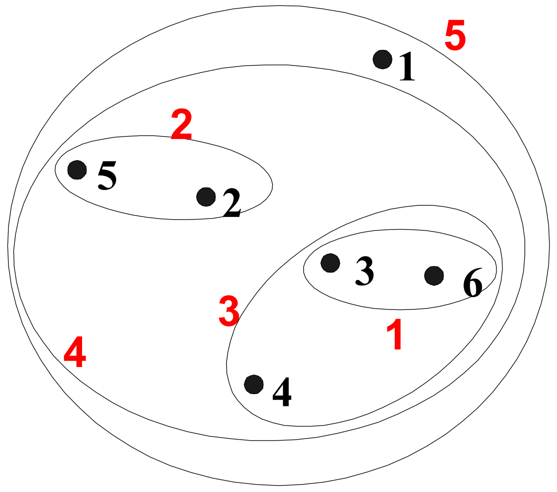
Alguns exemplos de cálculos...

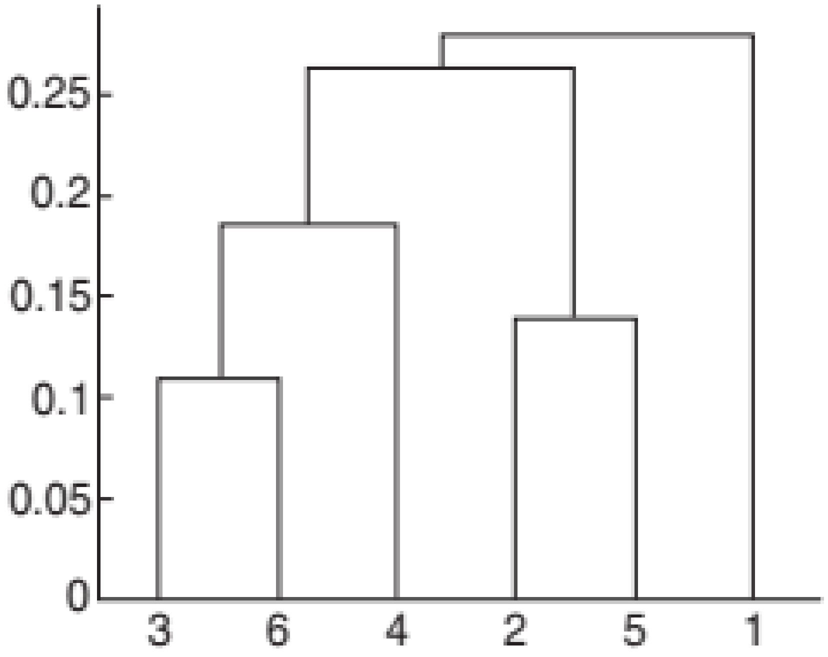
$$\begin{aligned} dist(\{3,6,4\},\{1\}) &= (0.22+0.37+0.23)/(3\times 1) \\ &= 0.28 \\ dist(\{2,5\},\{1\}) &= (0.24+0.34)/(2\times 1) \\ &= 0.29 \\ dist(\{3,6,4\},\{2,5\}) &= (0.15+0.28+0.25+0.39+0.20+0.29)/(3\times 2) \\ &= 0.26 \end{aligned}$$



Hierarchical Clustering: Group Average







Nested Clusters

Dendrogram

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link

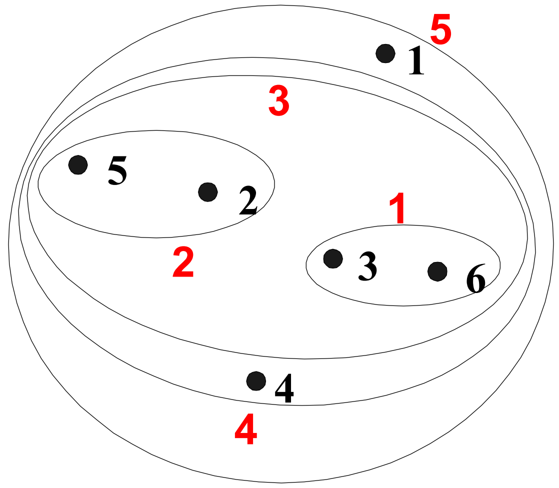
- Strengths
  - Less susceptible to noise and outliers

- Limitations
  - Biased towards globular clusters

Cluster Similarity: Ward's Method

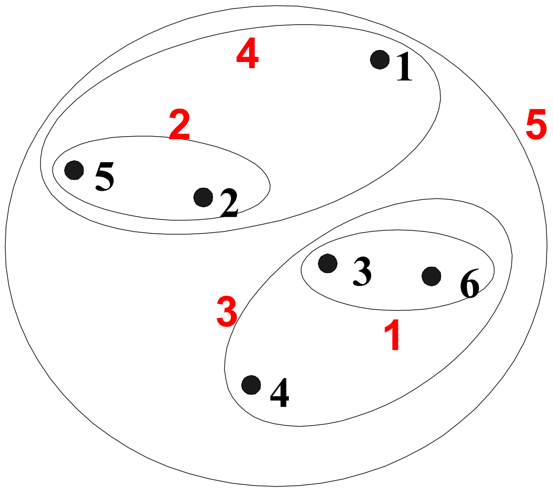
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

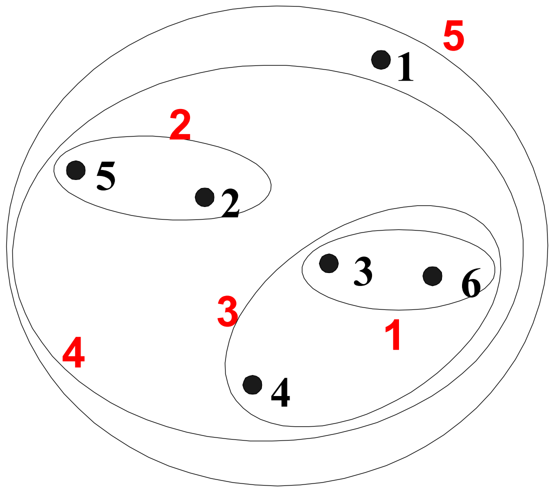
Hierarchical Clustering: Comparison



MIN

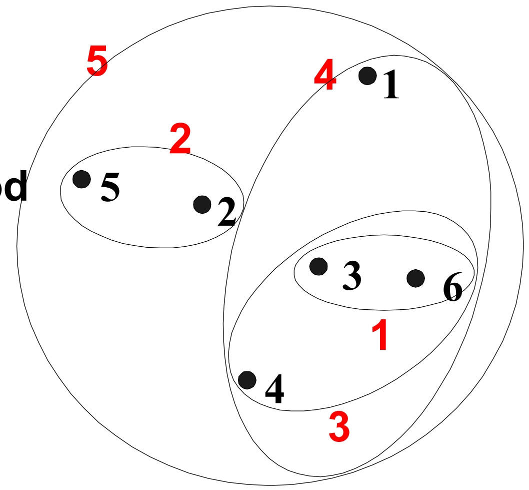
MAX





Group Average

Ward's Method



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  – N is the number of points.

- $O(N^3)$ time in many cases
  – There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  – Complexity can be reduced to $O(N^2 \log(N))$ time with some cleverness

Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone

- No global objective function is directly minimized

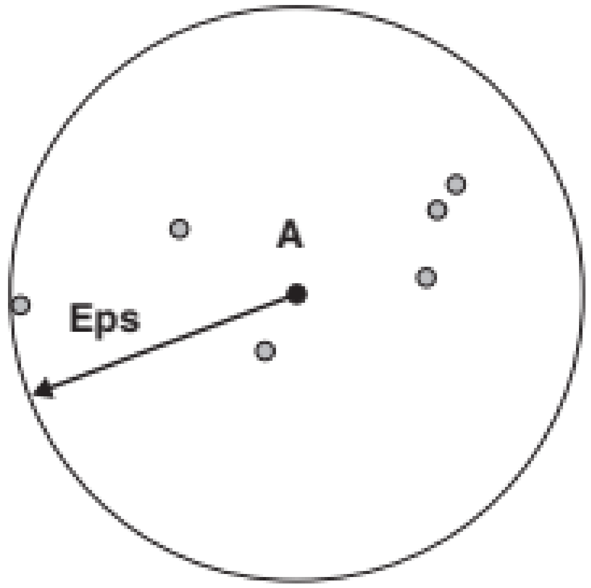
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters

DBSCAN

- DBSCAN is a density-based algorithm.
  - Density-based clustering locates regions of high density that are separated from one another by regions of low density.
  - Although there are not as many approaches for defining density, as there are for defining similarity, there are several distinct methods.
  - DBSCAN is based in the center-based approach.

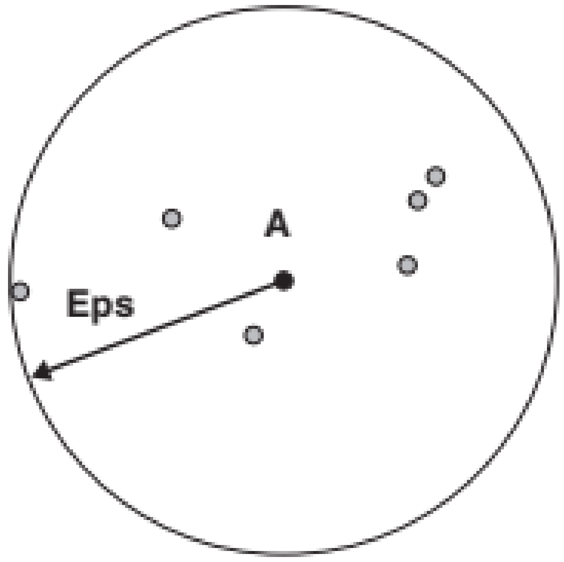
DBSCAN

- DBSCAN is a density-based algorithm.
  - In the center-based approach, density is estimated for a particular point in the data set by counting the number of points (MinPts) within a specified radius (Eps) of that point
  - Density = number of points (MinPts) within a specified radius (Eps) (raio/distância)



DBSCAN

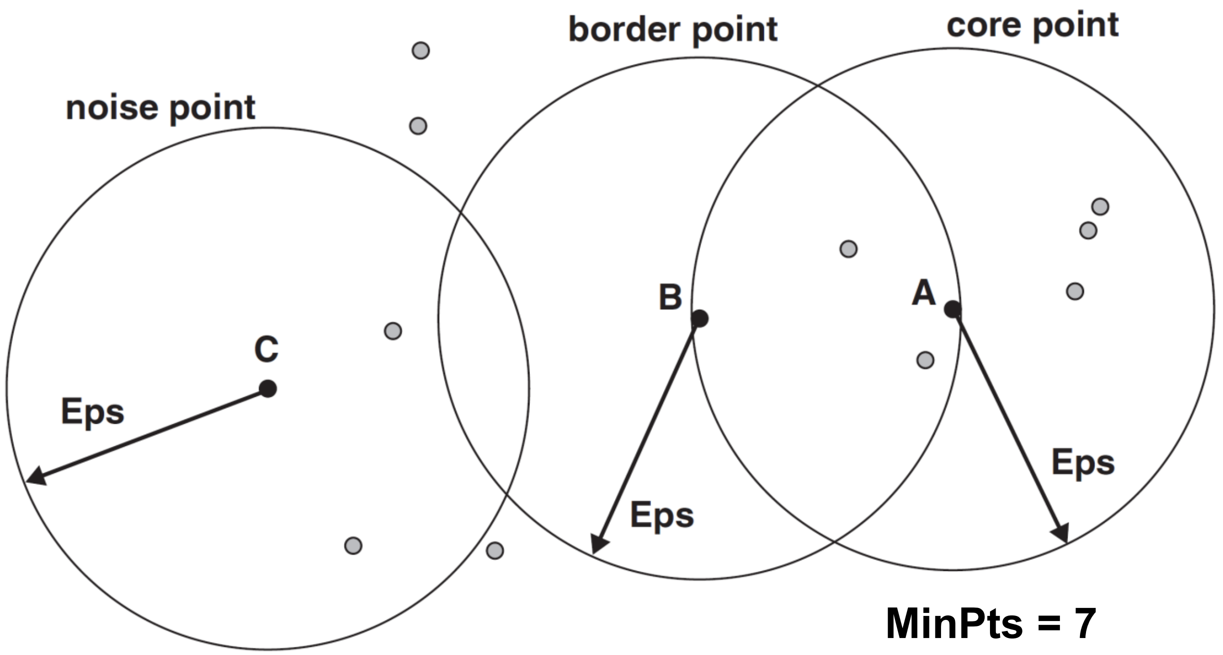
- DBSCAN is a density-based algorithm.
  - A point is a core point if it has at least a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
    - Counts the point itself



MinPts = 7

DBSCAN

- DBSCAN is a density-based algorithm.
  - A border point is not a core point, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point

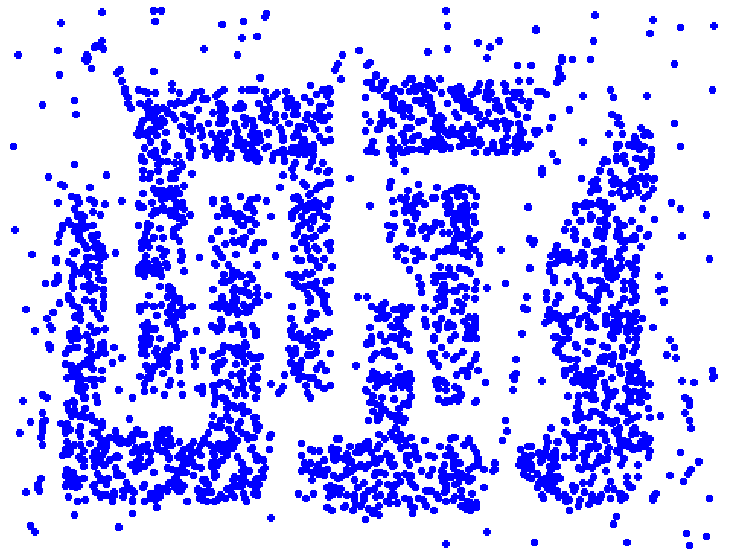


DBSCAN Algorithm

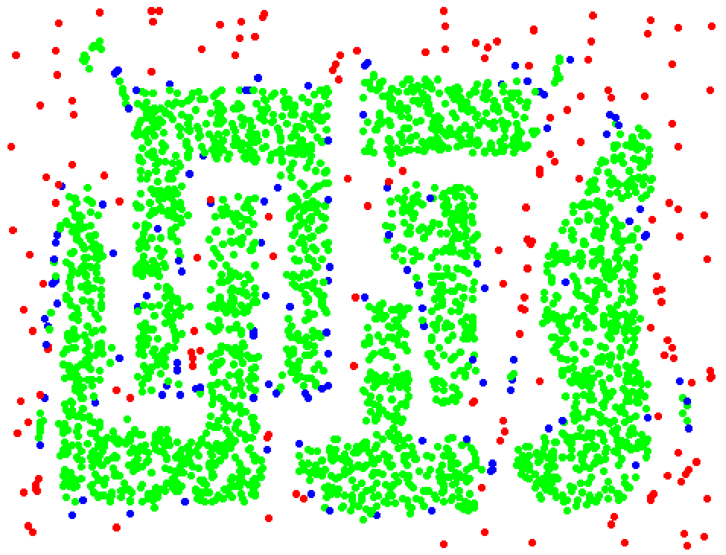
Optimized for simplicity, not efficiency

```
Label all points as core, border, or noise points
Eliminate noise points
$current\_cluster\_label \leftarrow 1$
for all core points do
  if the core point has no cluster label then
    $current\_cluster\_label \leftarrow current\_cluster\_label + 1$
    Label the current core point with cluster label $current\_cluster\_label$
  end if
  for all points in the $Eps$-neighborhood, except $i^{th}$ the point itself do
    if the point does not have a cluster label then
      Label the point with cluster label $current\_cluster\_label$
    end if
  end for
end for
```

DBSCAN: Core, Border and Noise Points



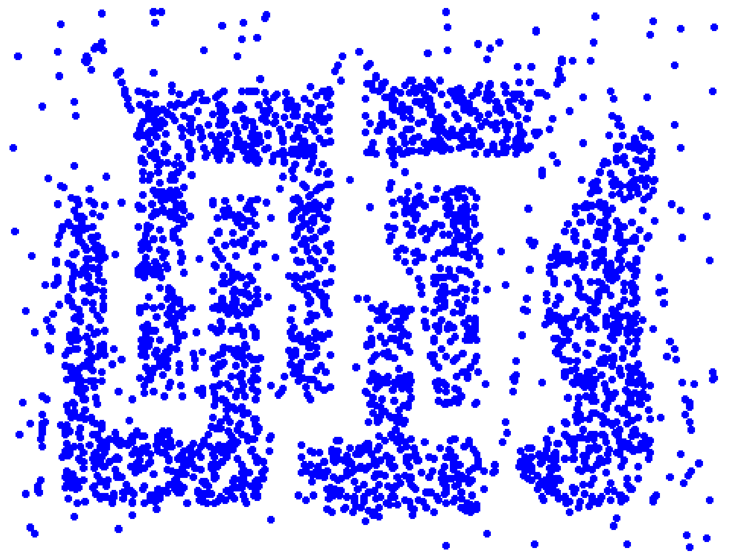
Original Points



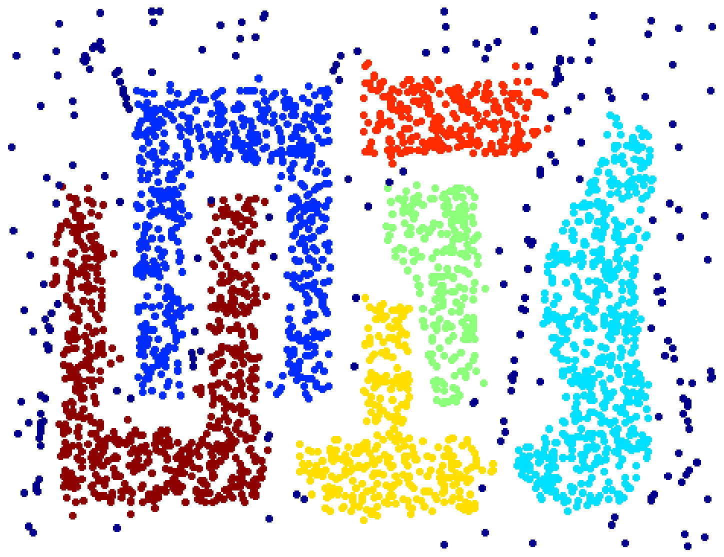
Point types: core, border and noise

Eps = 10, MinPts = 4

When DBSCAN Works Well



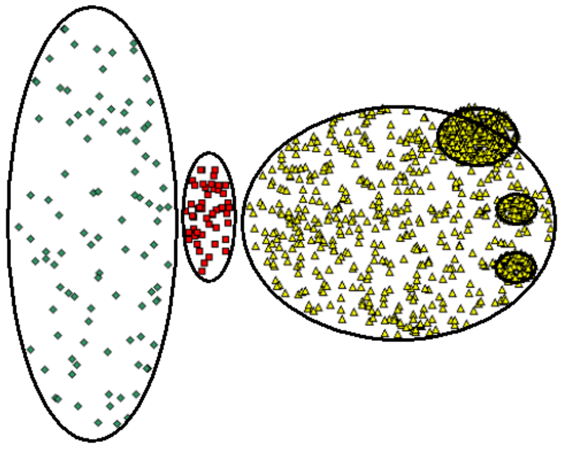
Original Points



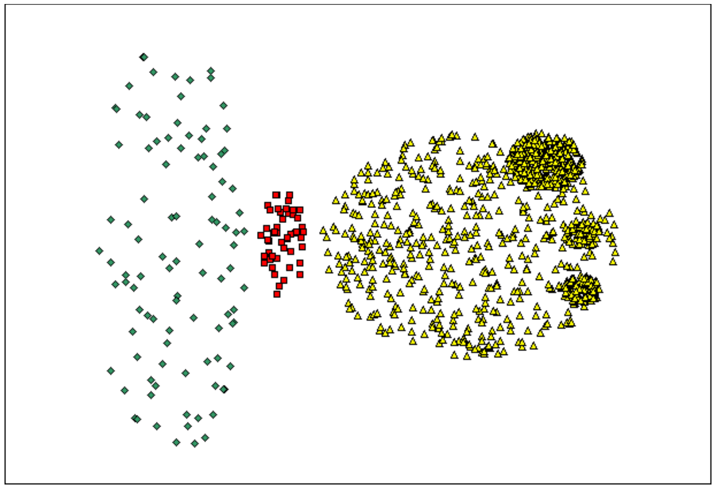
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

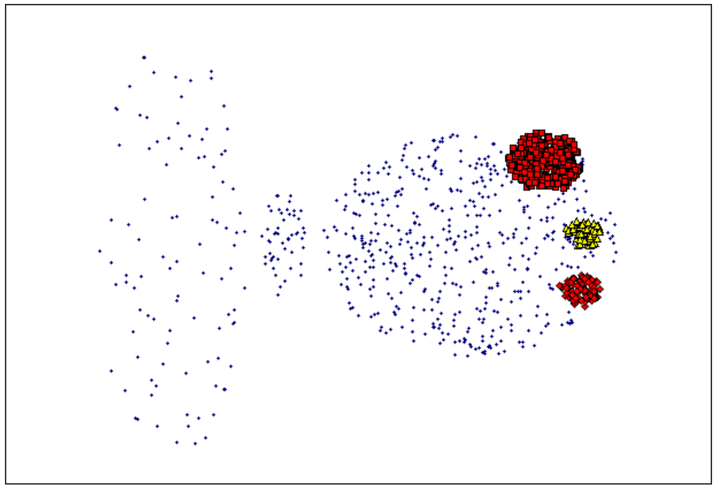
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.75).

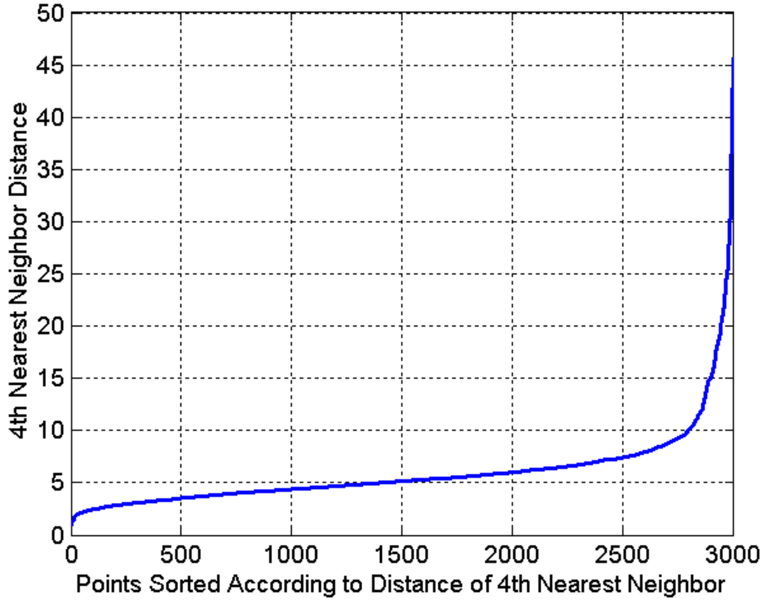


(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

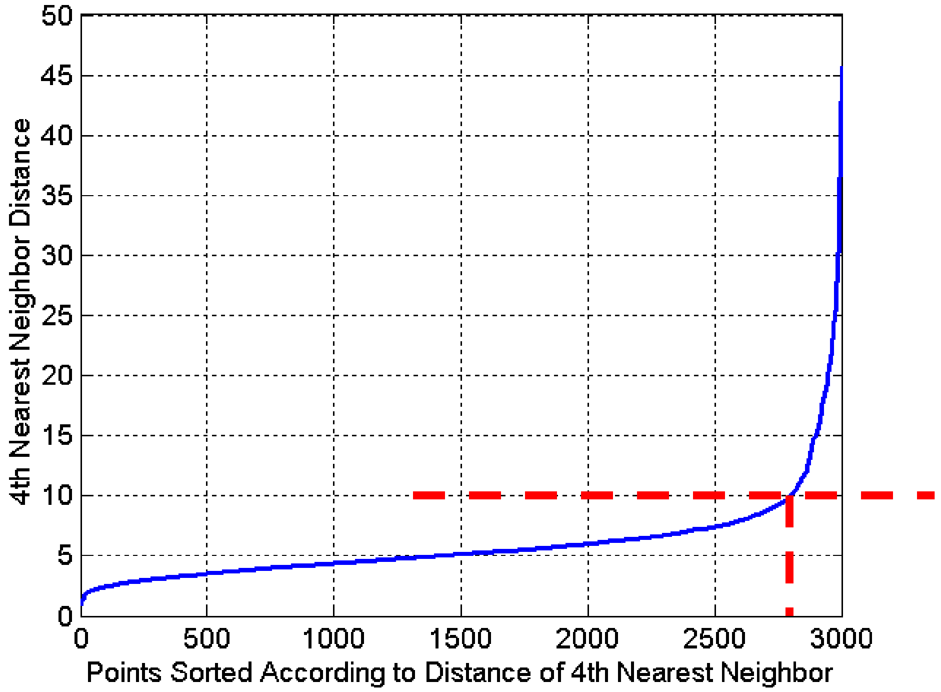
DBSCAN: Determining EPS and MinPts

- Compute the *k*-dist for all the data points for some *k (k=MinPts)*
- Sort them in increasing order
- Plot the sorted values
- It is expected to see a sharp change at the value of k-dist that corresponds to a suitable value of Eps



DBSCAN: Determining EPS and MinPts

- Points for which *k*-dist is less than *Eps* will be labeled as core points, while other points will be labeled as noise or border points.



Time and Space Complexity

- In the worst case, time complexity is $O(m^2)$, where m is the number of points.
- However, in low-dimensional spaces, using some data structures, the time complexity can be as low as $O(m \log m)$ in the average case.
- The space requirement is $O(m)$ because it is necessary to keep only a small amount of data for each point, i.e., the cluster label and the identification of each point as a core, border, or noise point.

Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall

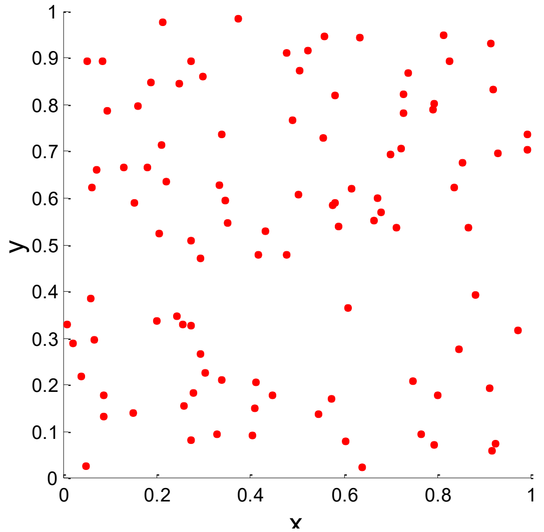
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?

- But “clusters are in the eye of the beholder”!

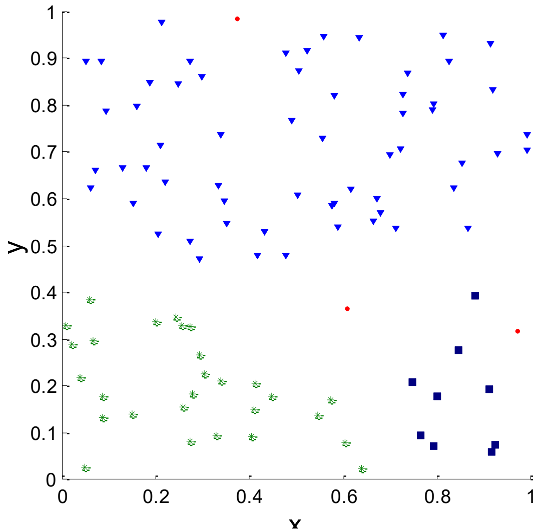
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Clusters found in Random Data

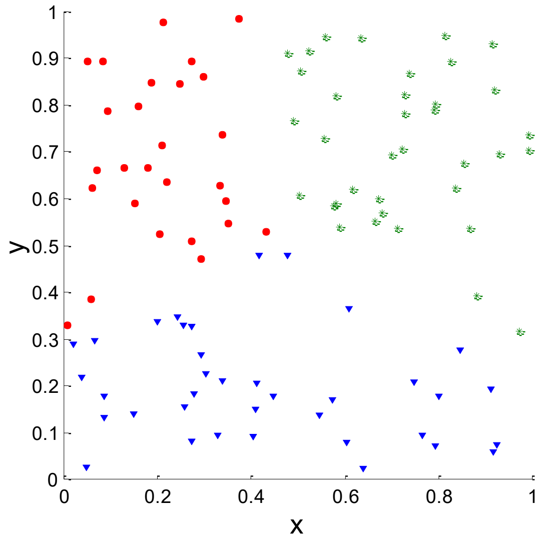
Random Points



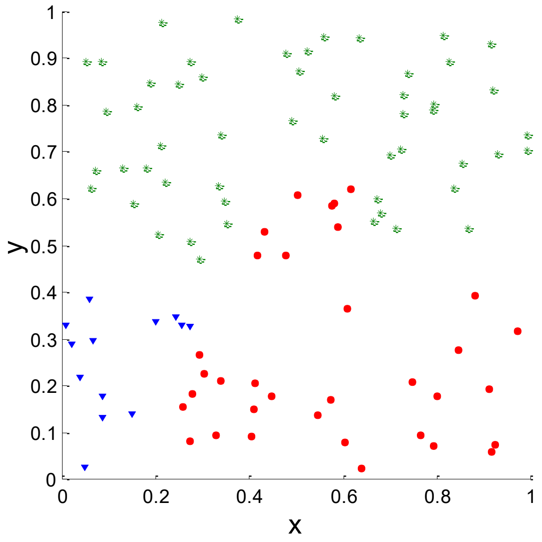
DBSCAN



K-means



Complete Link



Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
  - External Index (supervised): Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
  - Internal Index (unsupervised): Used to measure the goodness of a clustering structure *without* respect to external information.
    - Sum of Squared Error (SSE)
  - Relative Index: Used to compare two different clusterings or clusters.
    - Often an external or internal index is used for this function, e.g., SSE or entropy
- Sometimes these are referred to as criteria instead of indices
  - However, sometimes criterion is the general strategy and index is the numerical measure that implements the criterion.

Unsupervised Evaluation Using Cohesion and Separation

- Many internal measures of cluster validity for partitional clustering schemes are based on the notions of **cohesion** or **separation**
- In general, we can consider expressing overall cluster validity for a set of *K* clusters as a weighted sum of the validity of individual clusters

$$overall\ validity = \sum_{i=1}^{K} w_i\ validity(C_i).$$

- The *validity* function can be cohesion, separation, or some combination of these quantities
- The weights will vary depending on the cluster validity measure

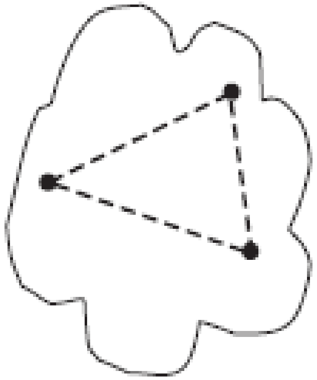
Unsupervised Evaluation Using Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster

- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters

Unsupervised Evaluation Using Cohesion and Separation

- **Graph-Based View of Cohesion and Separation**
  - The cohesion of a cluster can be defined as the sum of the weights of the links in the proximity graph that connect points within the cluster

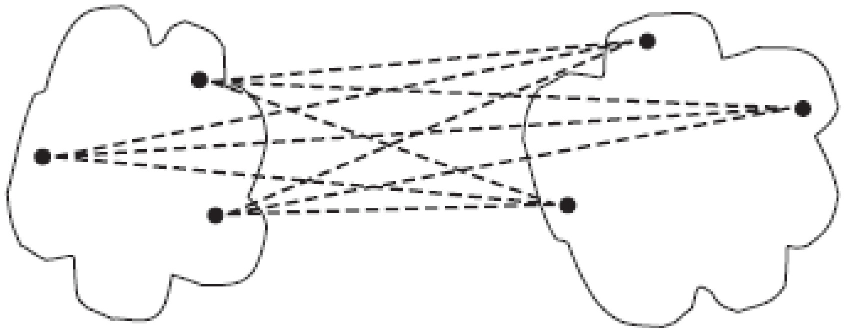


$$cohesion(C_i) = \sum_{\substack{\mathbf{x} \in C_i \\ \mathbf{y} \in C_i}} proximity(\mathbf{x}, \mathbf{y})$$

Proximity function can be a similarity or a dissimilarity
For similarity, higher values are better for cohesion while lower values are better for separation. For dissimilarity, the opposite is true.

Unsupervised Evaluation Using Cohesion and Separation

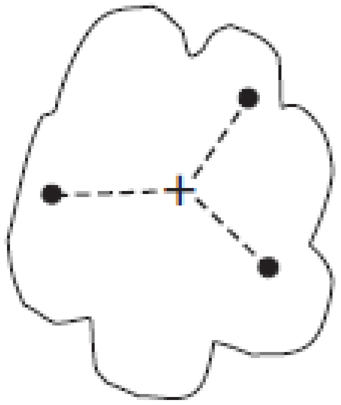
- **Graph-Based View of Cohesion and Separation**
  - The separation between two clusters can be measured by the sum of the weights of the links from points in one cluster to points in the other cluster



$$separation(C_i, C_j) = \sum_{\substack{\mathbf{x}\in C_i \\ \mathbf{y}\in C_j}} proximity(\mathbf{x}, \mathbf{y})$$

Unsupervised Evaluation Using Cohesion and Separation

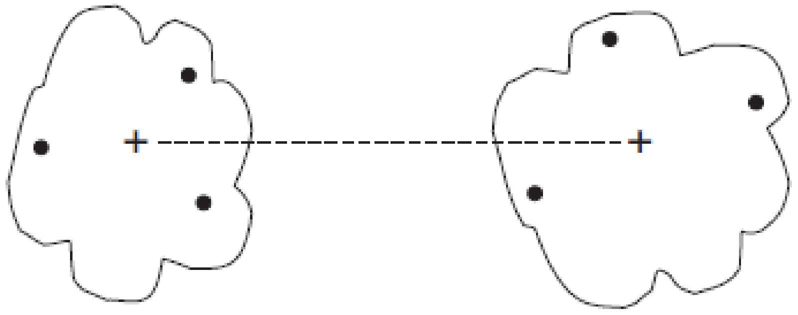
- **Prototype-Based View of Cohesion and Separation**
  - The cohesion of a cluster can be defined as the sum of the proximities with respect to the prototype (centroid or medoid) of the cluster



$$cohesion(C_i) = \sum_{\mathbf{x} \in C_i} proximity(\mathbf{x}, \mathbf{c}_i)$$

Unsupervised Evaluation Using Cohesion and Separation

- **Prototype-Based View of Cohesion and Separation**
  - The separation between two clusters can be measured by the proximity of the two cluster prototypes

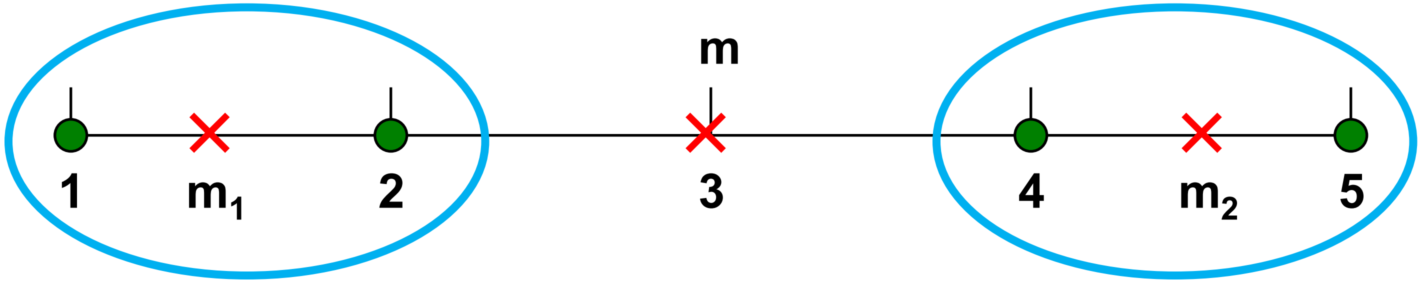


$$\begin{aligned} separation(C_i, C_j) &= proximity(\mathbf{c}_i, \mathbf{c}_j) \\ separation(C_i) &= proximity(\mathbf{c}_i, \mathbf{c}) \end{aligned}$$

Unsupervised Evaluation Using Cohesion and Separation

Prototype-Based View of Cohesion and Separation

- Example:
  - WSS + BSS (if proximity be the squared Euclidean distance)



Coesão = WSS (Within Sum of Squares)

$$SSE = WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

Separação = BSS (Between Sum of Squares)

$$BSS = \sum_i |C_i| (m - m_i)^2$$

K=2 clusters:

$$SSE = WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

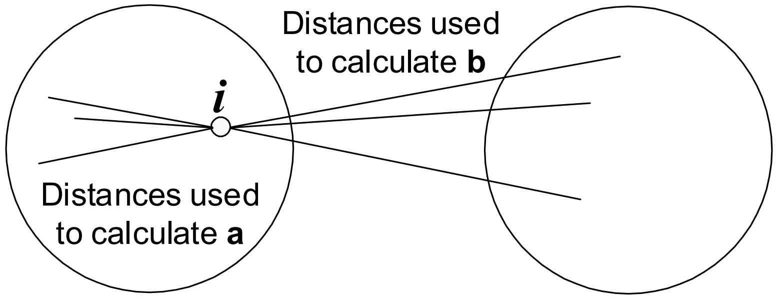
$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

Evaluating Objects, Clusters, Clustering

- Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, i
  - Calculate $\boldsymbol{a}$ = average distance of i to the points in its cluster
  - Calculate $\boldsymbol{b}$ = min (average distance of i to points in another cluster)
  - The silhouette coefficient for a point is then given by

    $s_i = (b - a) / \max(a, b)$


Distances used to calculate **b**
i
Distances used to calculate **a**

  - The value of the silhouette coefficient can vary between −1 and 1

Evaluating Objects, Clusters, Clustering

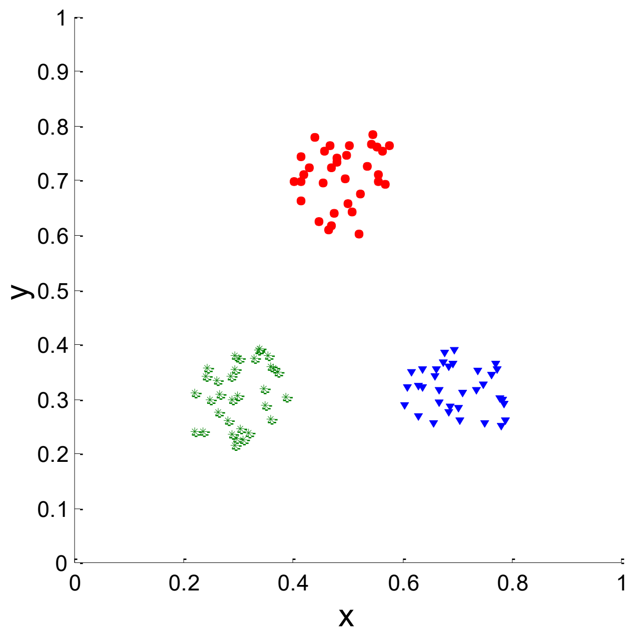
- **The Silhouette Coefficient**
  - We can compute the average silhouette coefficient of a cluster by simply taking the average of the silhouette coefficients of points belonging to the cluster
  - An overall measure of the goodness of a clustering can be obtained by computing the average silhouette coefficient of all points ($\bar{s}(K)$)
    - Um modo de escolher o melhor valor de K é selecionar aquele que resulta no maior valor de $\bar{s}(K)$

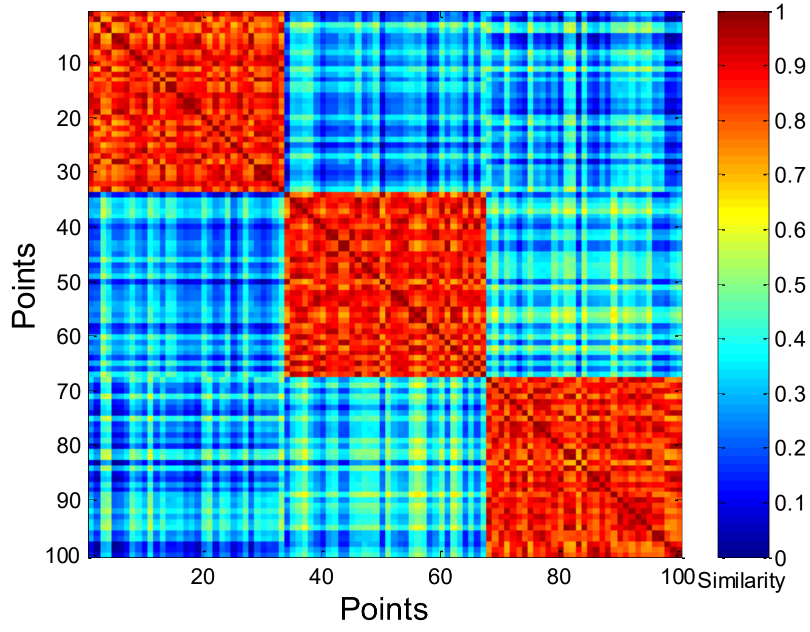
Evaluating Objects, Clusters, Clustering

- **The Silhouette Coefficient**
  - O coeficiente silhueta, SC, é o máximo $\bar{s}(K)$ para K = 2, 3, ..., (n − 1)
  - SC é uma medida da quantidade de estrutura descoberta por um algoritmo de agrupamento
    - SC $\leq$ 0.25 significa que não foi encontrada uma estrutura substancial
    - 0.26 $\leq$ SC $\leq$ 0.5 indica que a estrutura encontrada é fraca e pode ser artificial
    - 0.5 $\leq$ SC $\leq$ 0.7 significa que uma estrutura razoável foi encontrada
    - 0.71 $\leq$ SC $\leq$ 1 indica que foi encontrada uma estrutura forte

Using Similarity Matrix for Cluster Validation

- Order the similarity matrix with respect to cluster labels and inspect visually.

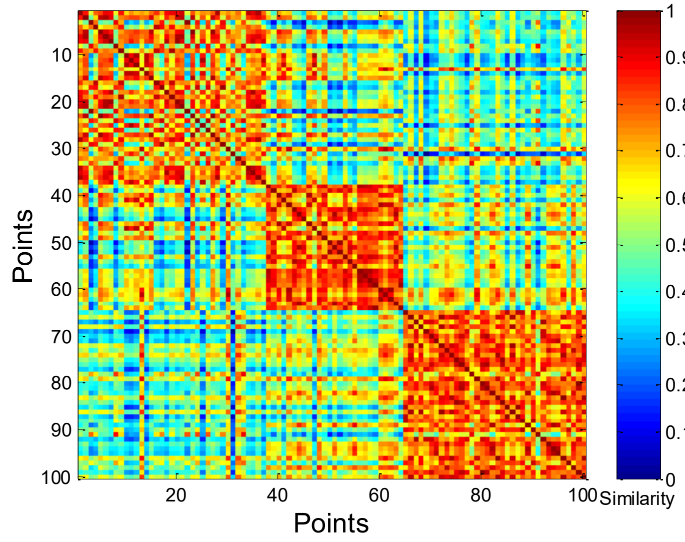


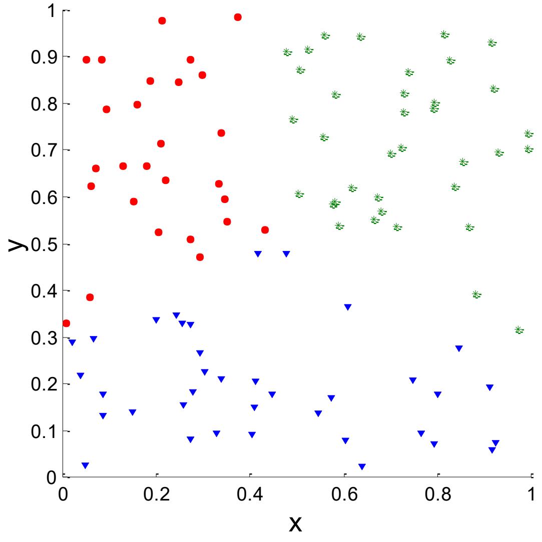


K-means

Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

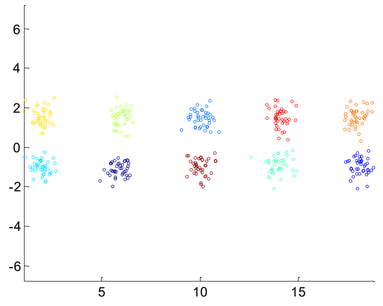


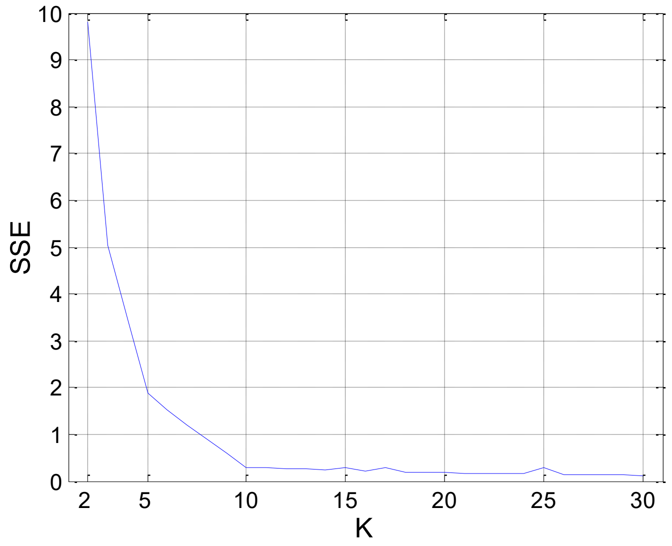


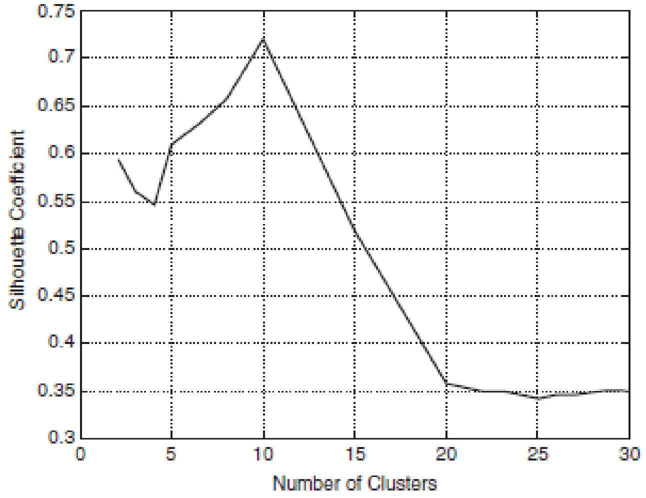
K-means

Determining the Correct Number of Clusters

- Various unsupervised cluster evaluation measures can be used to approximately determine the correct or natural number of clusters







Supervised Measures of Cluster Validity

- When we have external information about data, it is typically in the form of externally derived class labels for the data objects.
- In such cases, the usual procedure is to measure the degree of correspondence between the cluster labels and the class labels.
- But why is this of interest? After all, if we have the class labels, then what is the point in performing a cluster analysis?

Supervised Measures of Cluster Validity

- Motivations for such an analysis include, for example, the comparison of clustering techniques with the “ground truth” or the evaluation of the extent to which a manual classification process can be automatically produced by cluster analysis, e.g., the clustering of news articles.

Supervised Measures of Cluster Validity

- We consider two different kinds of approaches
  - **Classification-oriented**
    - Use measures from classification, such as entropy, purity, and the *F*-measure. These measures evaluate the extent to which a cluster contains objects of a single class
  - **Similarity-oriented**
    - Related to the similarity measures for binary data, such as the Jaccard measure. These approaches measure the extent to which two objects that are in the same class are in the same cluster and vice versa

Supervised Measures of Cluster Validity

- **Similarity-Oriented Measures of Cluster Validity**
  - The measures are based on the premise that any two objects that are in the same cluster should be in the same class and vice versa
  - We can view this approach to cluster validity as involving the comparison of two matrices:
    - (1) the **ideal cluster similarity matrix**, which has a 1 in the *ij*th entry if two objects, *i* and *j*, are in the same cluster and 0, otherwise
    - (2) a **class similarity matrix** defined with respect to class labels, which has a 1 in the *ij*th entry if two objects, *i* and *j*, belong to the same class, and a 0 otherwise

Supervised Measures of Cluster Validity

- **Similarity-Oriented Measures of Cluster Validity**
  - Take the correlation of these two matrices as the measure of cluster validity (**known as Hubert's Γ statistic**)
  - Consider five data points, *p*1*, p*2*, p*3*, p*4*, and p*5, two clusters, *C*1 = *{p*1*, p2, p3}* and *C2* = *{p*4*, p5}*, and two classes, *L*1 = *{p*1*, p2}* and *L2* = *{p*3*, p*4*, p5}*. The ideal cluster and class similarity matrices are given in Tables 7.10 and 7.11. The correlation between the entries of these two matrices is 0.359.

Table 7.10. Ideal cluster similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	1	0	0
p2	1	1	1	0	0
p3	1	1	1	0	0
p4	0	0	0	1	1
p5	0	0	0	1	1

Table 7.11. Class similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	0	0	0
p2	1	1	0	0	0
p3	0	0	1	1	1
p4	0	0	1	1	1
p5	0	0	1	1	1

Supervised Measures of Cluster Validity

- **Similarity-Oriented Measures of Cluster Validity**
  - We can convert these two matrices into binary vectors

f_{00} = number of pairs of objects having a different class and a different cluster
f_{01} = number of pairs of objects having a different class and the same cluster
f_{10} = number of pairs of objects having the same class and a different cluster
f_{11} = number of pairs of objects having the same class and the same cluster

$$\text{Rand statistic} = \frac{f_{00} + f_{11}}{f_{00} + f_{01} + f_{10} + f_{11}}$$

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

Supervised Measures of Cluster Validity

- Similarity-Oriented Measures of Cluster Validity

Table 7.10. Ideal cluster similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	1	0	0
p2	1	1	1	0	0
p3	1	1	1	0	0
p4	0	0	0	1	1
p5	0	0	0	1	1

Table 7.11. Class similarity matrix.

Point	p1	p2	p3	p4	p5
p1	1	1	0	0	0
p2	1	1	0	0	0
p3	0	0	1	1	1
p4	0	0	1	1	1
p5	0	0	1	1	1

	Same Cluster	Different Cluster
Same Class	f_{11}	f_{10}
Different Class	f_{01}	f_{00}

p1, p2	**p2, p3**	**p3, p4**
p1, p3	**p2, p4**	**p3, p5**
p1, p4	**p2, p5**	**p4, p5**
p1, p5		

$f_{00} = 4$, $f_{01} = 2$, $f_{10} = 2$, $f_{11} = 2$

Rand statistic = (2 + 4)/10 = 0.6

Jaccard coefficient = 2/(2+2+2) = 0.33

Final Comment on Cluster Validity

"The validation of clustering structures is the most difficult and frustrating part of cluster analysis."

Algorithms for Clustering Data, Jain and Dubes

- In conclusion, it is important to realize that clustering is often used as an exploratory data technique whose goal is often not to provide a crisp answer, but rather to provide some insight into the underlying structure of the data.
  - In this situation, cluster validity indices are useful to the extent they are useful to that end goal.

Demais Algoritmos

- k-medoids
- Scalable Algorithms
  - CLARA/CLARANS
    - Scalable implementation of the k-medoids algorithm
  - BIRCH
    - Top-down hierarchical generalization of the k-means algorithm
  - CURE
    - Bottom-up agglomerative approach to clustering

k-medoids

- The main distinguishing feature of the k-medoids algorithm is that the representatives are always selected from the database D, and this difference necessitates changes to the basic structure of the k-representatives algorithm

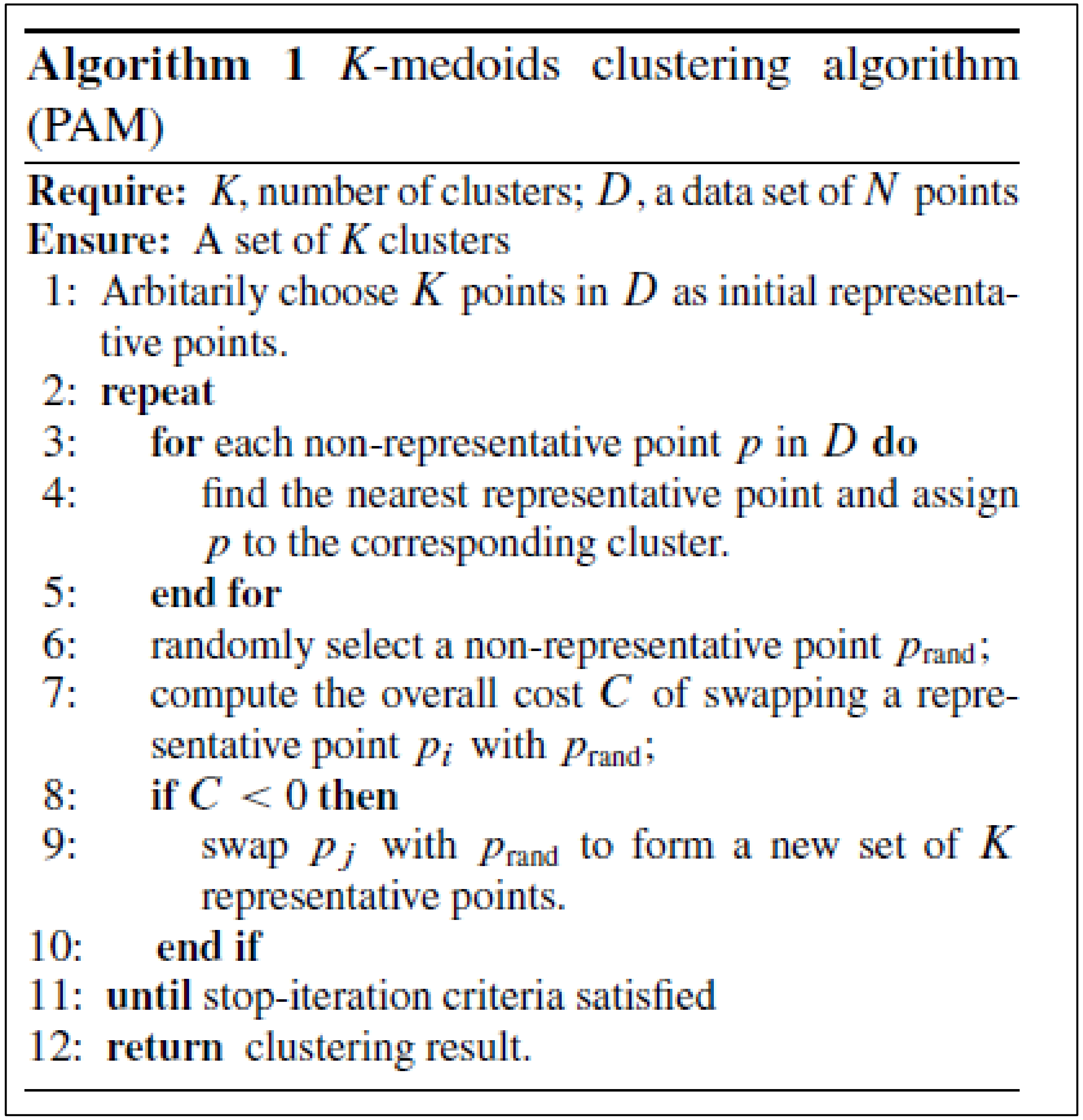
- The goal is to minimize the average dissimilarity of objects to their closest selected object = minimize the sum of the dissimilarities between object and their closest selected object

k-medoids

- Why it is sometimes desirable to select the representatives from D?
  - The representative of a k-means cluster may be distorted by outliers in that cluster (vide figura)
  - It is sometimes difficult to compute the optimal central representative of a set of data points of a complex data type
- Therefore, a key property of the k-medoids algorithm is that it can be defined virtually on any data type, as long as an appropriate similarity or distance function can be defined on the data type

k-medoids

https://link.springer.com/referenceworkentry/10.1007/978-1-4899-7687-1_432



Algorithm 1 *K*-medoids clustering algorithm (PAM)

```
Require: $K$, number of clusters; $D$, a data set of $N$ points
Ensure: A set of $K$ clusters
 1: Arbitarily choose $K$ points in $D$ as initial representative points.
 2: repeat
 3:     for each non-representative point $p$ in $D$ do
 4:         find the nearest representative point and assign $p$ to the corresponding cluster.
 5:     end for
 6:     randomly select a non-representative point $p_{\mathrm{rand}}$;
 7:     compute the overall cost $C$ of swapping a representative point $p_i$ with $p_{\mathrm{rand}}$;
 8:     if $C < 0$ then
 9:         swap $p_j$ with $p_{\mathrm{rand}}$ to form a new set of $K$ representative points.
10:     end if
11: until stop-iteration criteria satisfied
12: return clustering result.
```

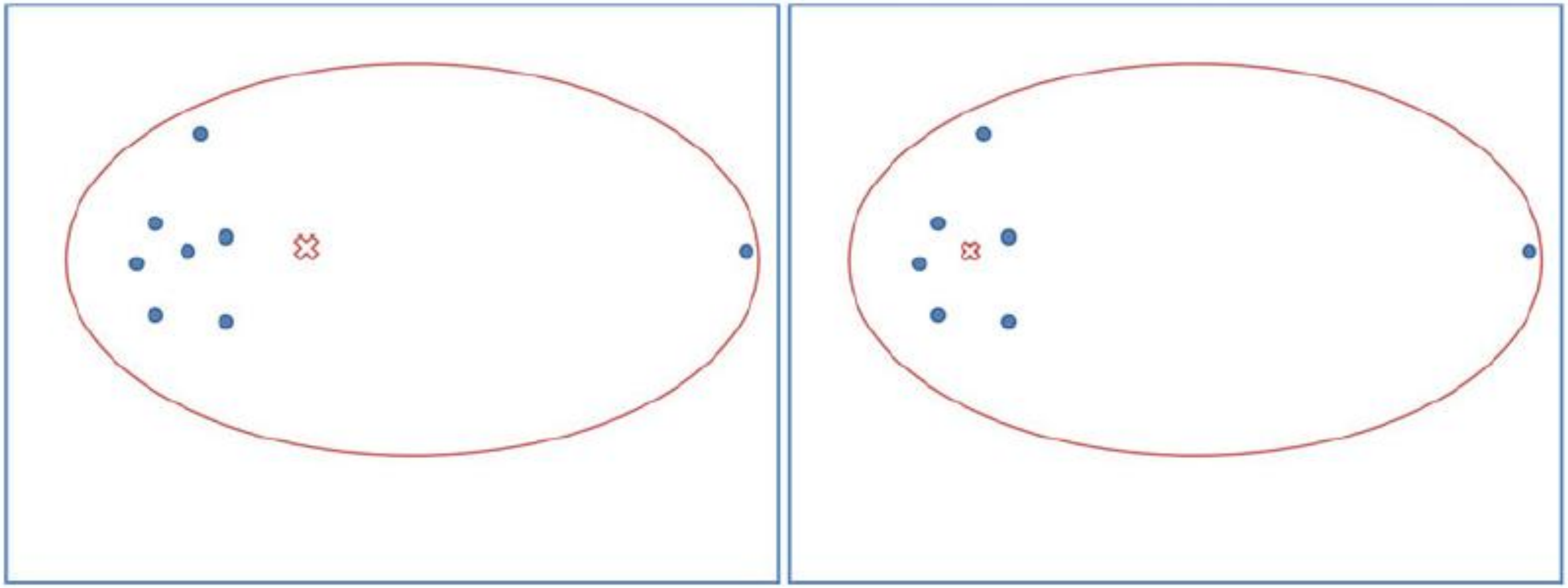
$$O(K(N-K)^2 I).$$

K-Medoids Clustering is a clustering method more robust to outliers than K-Means

Representative algorithms include Partitioning Around Medoids (PAM), CLARA, CLARANS, etc.

k-medoids

https://link.springer.com/referenceworkentry/10.1007/978-1-4899-7687-1_432



K-Medoids Clustering, Fig. 1 Mean vs. medoid in 2D space. In both figures (**a**) and (**b**), the group of points in the right forms a cluster and the rightmost point is an outlier. *The red point* represents the center found by mean or medoid

Demais Algoritmos

- Categorical Data
  - k-modes
  - ROCK
- High-Dimensional Data
  - CLIQUE
- Graph-Based Algorithms
- etc.

Demais Tópicos

- Agrupamento semissupervisionado
  - https://github.com/datamole-ai/active-semi-supervised-clustering
- Agrupamento Ativo
  - https://github.com/datamole-ai/active-semi-supervised-clustering
- Comitê de Agrupamentos
  - https://github.com/NaegleLab/OpenEnsembles
- Interpretabilidade
  - https://docs.interpretable.ai/stable/examples/clustering/

Demais Referências

- Relative Clustering Validity Criteria: A Comparative Overview, 2010
- Discovering Knowledge in Data: An Introduction to Data Mining; Larose, D. T.; 2005

Recursos

- PyCaret
  - https://pycaret.org/
- PyClustering
  - https://pyclustering.github.io/docs/0.8.2/html/index.html
- scikit-learn
  - https://scikit-learn.org/stable/modules/clustering.html