

The background of the slide is an abstract composition. It features a series of thick, dark, wavy lines that sweep across the frame from the top left towards the bottom right. These lines are layered over a light gray background that contains a grid of faint, semi-transparent numbers (0-9) scattered across it. The overall aesthetic is modern and technical, suggesting data analysis or computer science.

# 7. Clustering

# Clustering

Clustering is the process of finding meaningful groups in data.

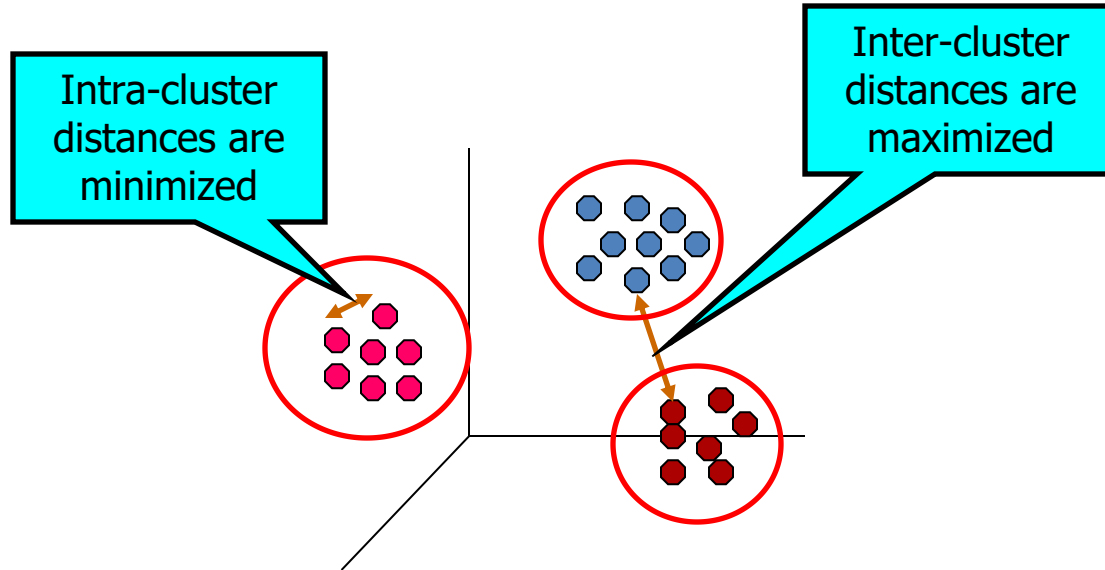
*For example, customers of a company can be grouped based on the purchase behavior. In recent years, clustering has even found its use in political elections*

Clustering to describe the data

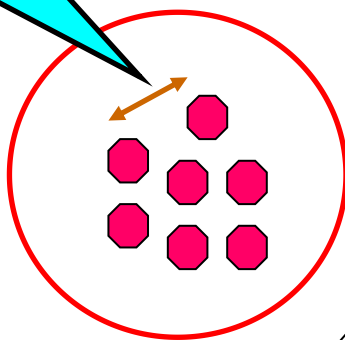
Clustering for pre-processing

# What is Cluster Analysis?

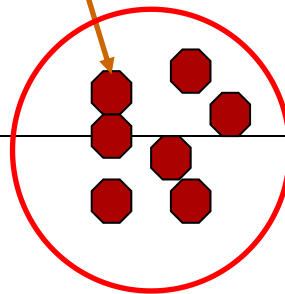
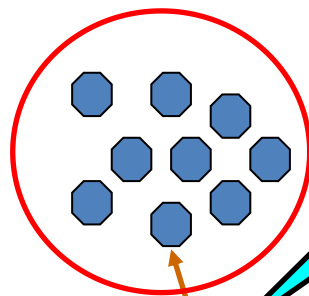
- Given a set of objects, place them in groups such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups




Intra-cluster distances are  
minimized



Inter-cluster distances are  
maximized



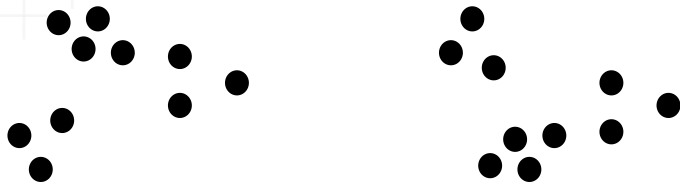
# Types of clustering

- 
1. Exclusive or strict partitioning clusters
  2. Overlapping clusters
  3. Hierarchical clusters

## **Based Algorithmic approach**

1. Distance based clustering
2. Density clustering
3. Hierarchical clustering

# How many clusters?



How many clusters?



Six Clusters



Two Clusters

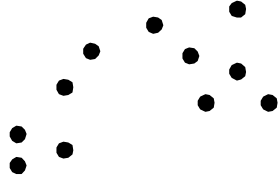


Four Clusters

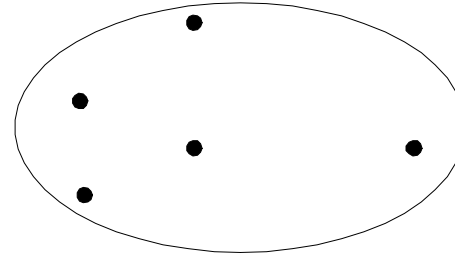
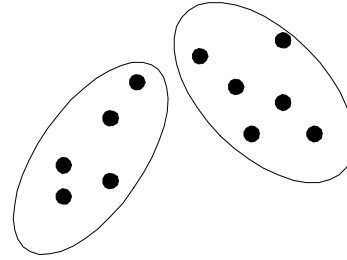
# Types of Clusterings

- 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)
  - **Hierarchical clustering**
    - A set of nested clusters organized as a hierarchical tree

# Partitional Clustering



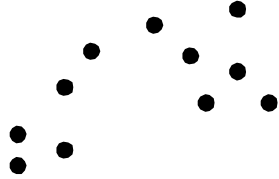
**Original Points**



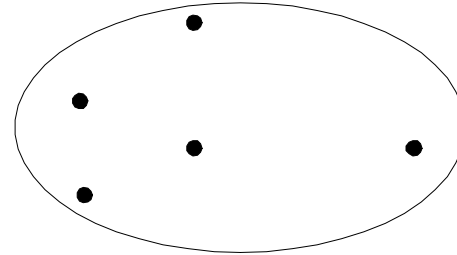
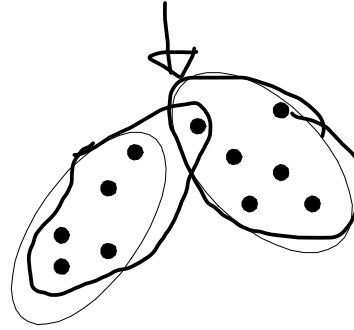
**A Partitional  
Clustering**



# Overlapping Clustering

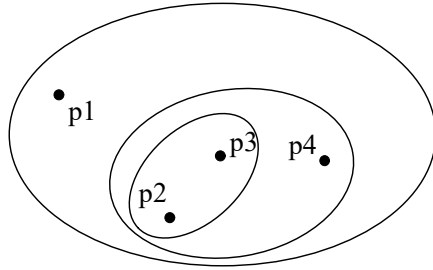


**Original Points**

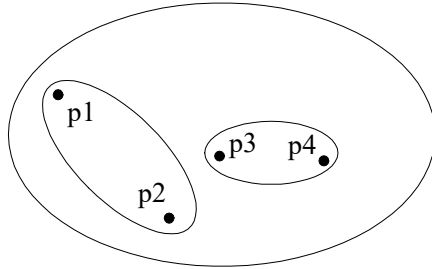


**A Partitional  
Clustering**

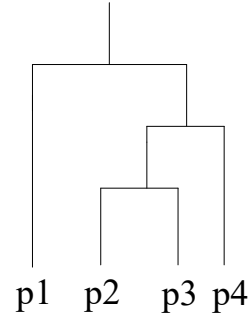
# Hierarchical Clustering



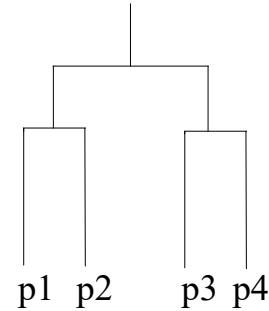
**Traditional Hierarchical Clustering**



**Non-traditional Hierarchical Clustering**



**Traditional Dendrogram**



**Non-traditional Dendrogram**

## Characteristics of the Input Data Are Important

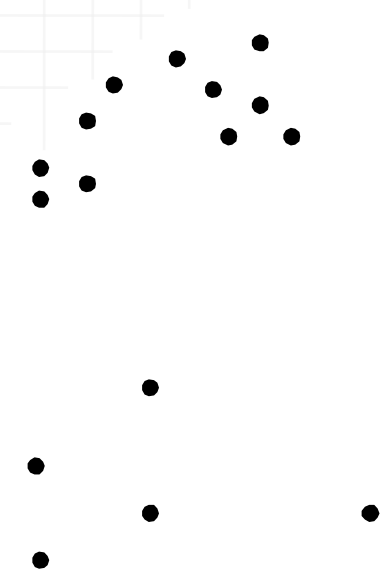
- Type of proximity (similarity) or density measure
  - Central to clustering
  - Depends on data and application
- Data characteristics that affect proximity and/or density are
  - Dimensionality (# of columns)
    - Sparseness
  - Attribute type
  - Special relationships in the data
    - For example, autocorrelation
  - Distribution of the data
- Noise and Outliers
  - Often interfere with the operation of the clustering algorithm

# K-means Clustering

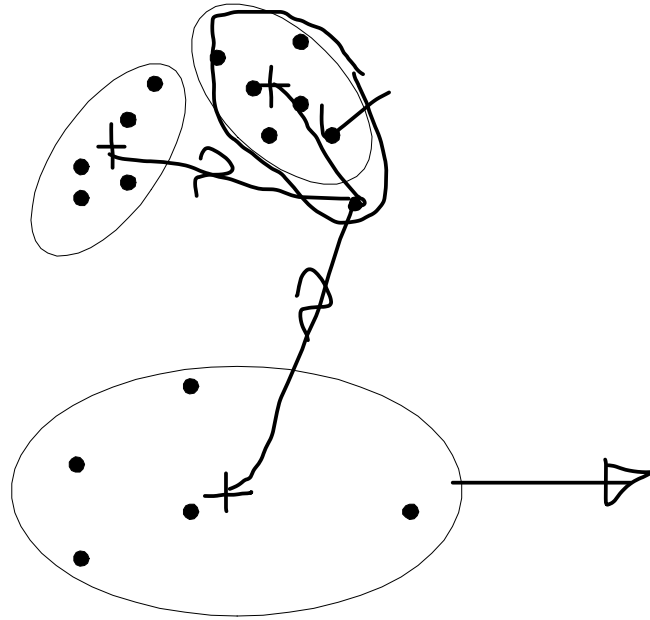
- Partitional clustering approach
- Number of clusters,  $K$ , must be specified
- Each cluster is associated with a **centroid** (center point) (mean)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

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

# Cluster Centroid



Original Points



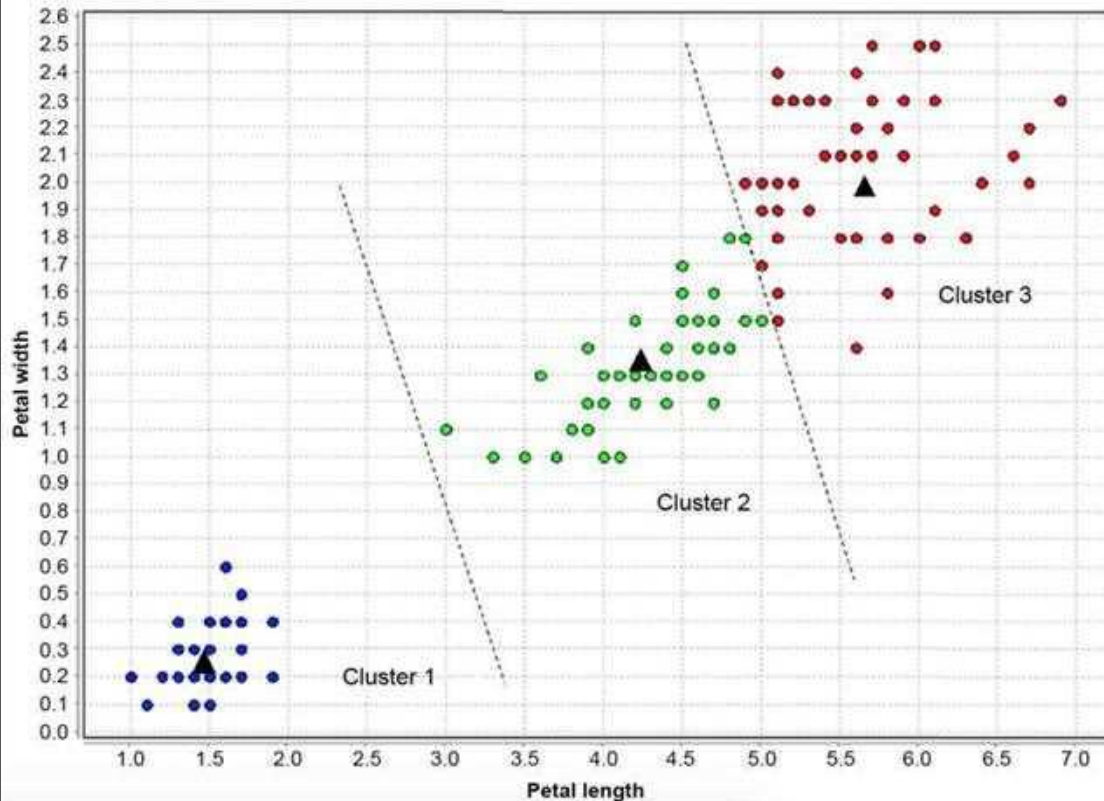
A Partitional  
Clustering

	$x_1$	$x_2$	$x_3$
1			
2			
...			
5			
	$\frac{\sum x_1}{5}$	$\frac{\sum x_2}{5}$	...

# K-means Clustering – Details

- Simple iterative algorithm.
  - Choose initial centroids;
  - repeat {assign each point to a nearest centroid; re-compute cluster centroids}
  - until centroids stop changing.
- Initial centroids are often chosen randomly.
  - Clusters produced can vary from one run to another
- The centroid is (typically) the mean of the points in the cluster, but other definitions are possible
- K-means will converge for common proximity measures with appropriately defined centroid
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to 'Until relatively few points change clusters'

# k Partitions



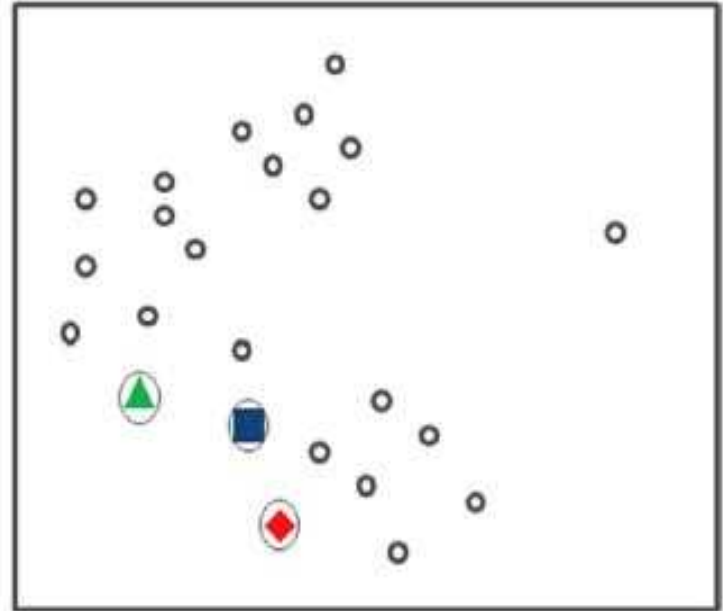
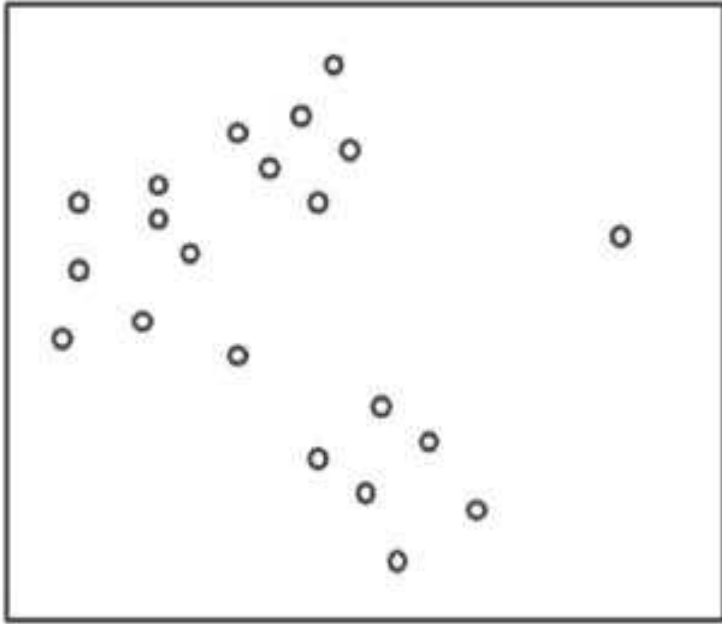
k-means algorithm divides the data space into  $k$  partitions or boundaries, where the centroid in each partition is the prototype of the clusters

# K Means Algorithm Steps (recap)

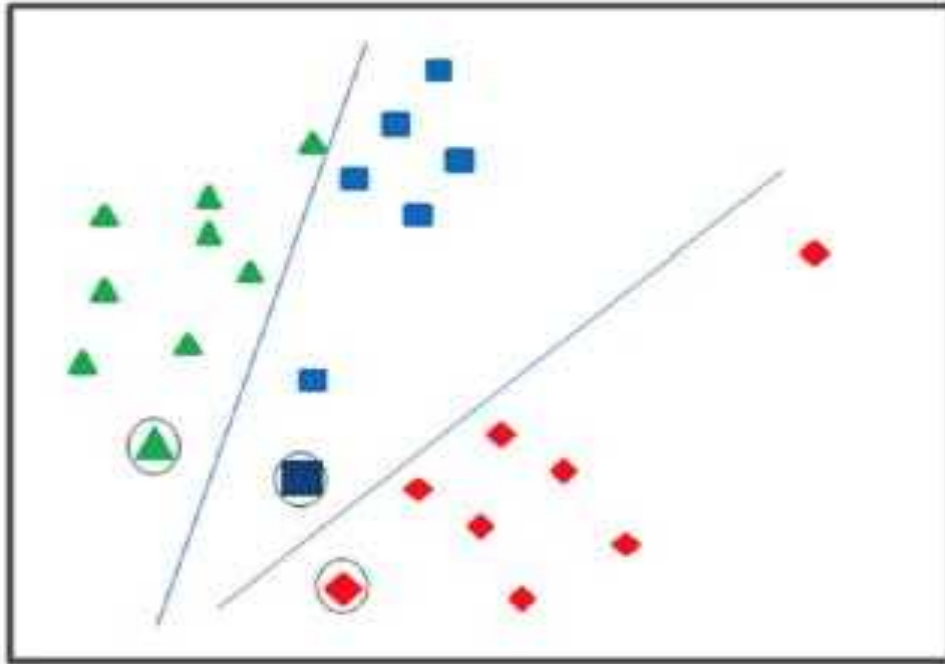
1. Initialize means (centroids)
2. Assign data points to cluster means
3. Calculate new means (based on the assigned points for each cluster)
4. Repeat the process until convergence (no further change in cluster assignments)



# Step 1: Initiate Centroids

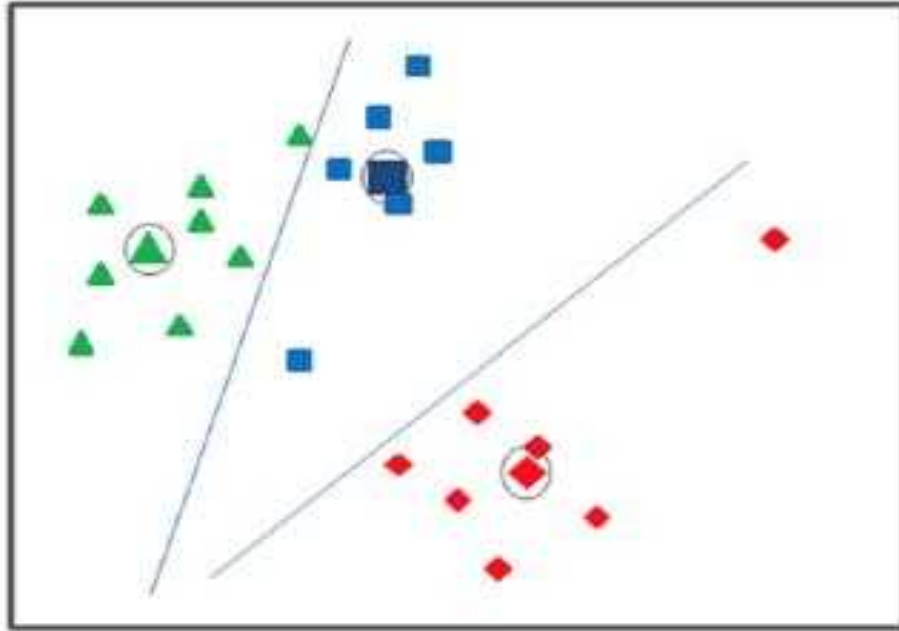


## Step 2: Assign Data Points



$$\text{Distance } d = \sqrt{(x_1 - c_1)^2 + (x_2 - c_2)^2 + \dots + (x_n - c_n)^2}$$

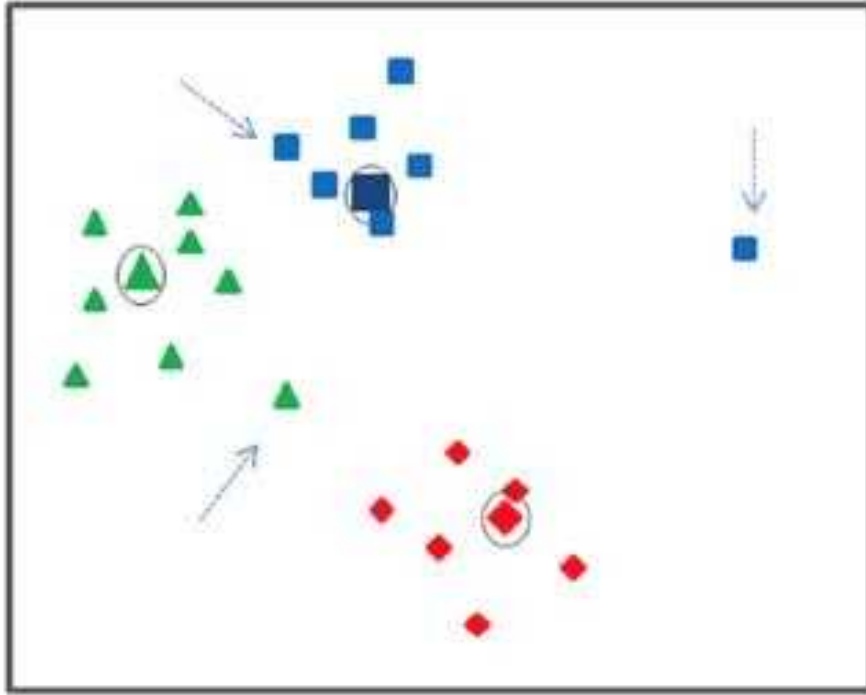
# Step 3: Calculate New Centroids



Minimizing the sum of squared errors (SSE)

$$SSE = \sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|^2$$

## Step 4: Repeat Assignment and Calculate New Centroids



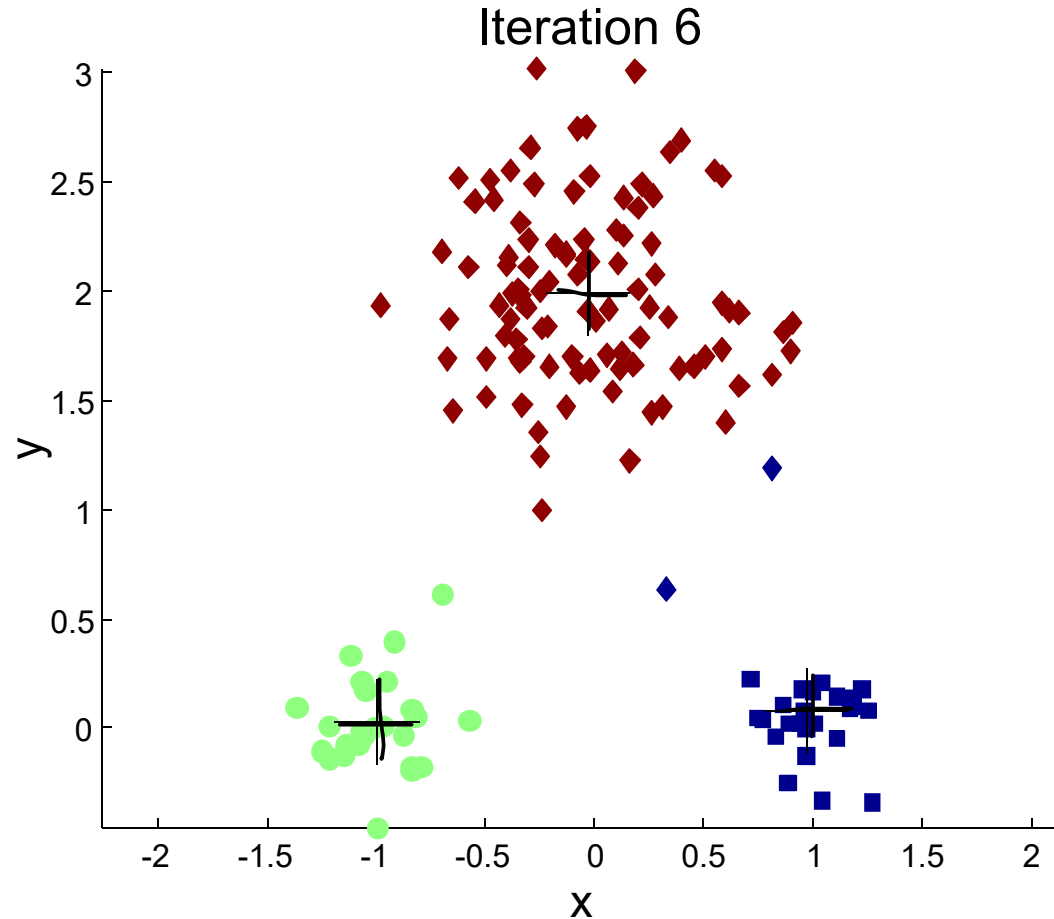
# Step 5: Termination

No further change in assignment of data points happens or, in other words, no significant change in centroids are noted.

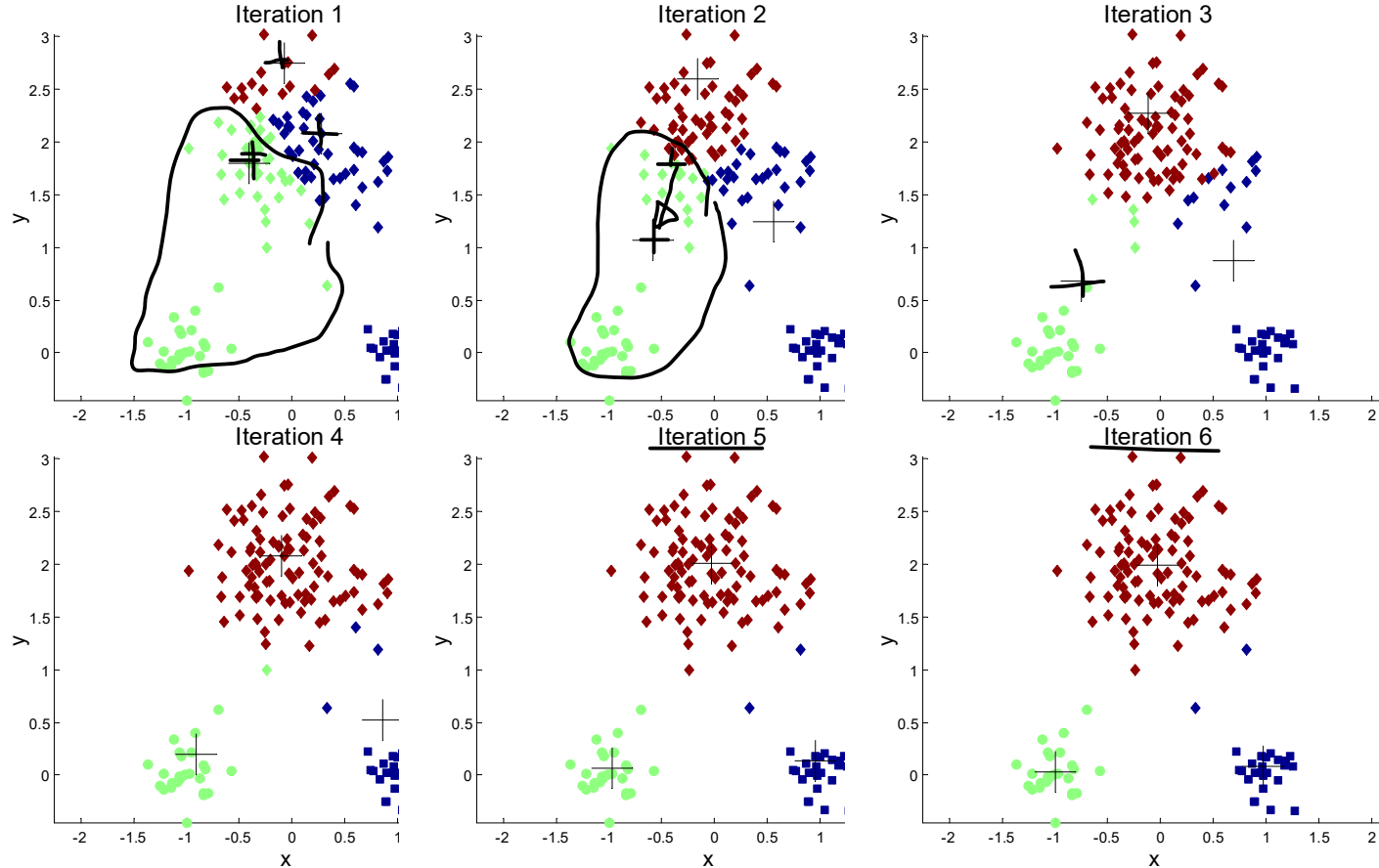
## Evaluation of Clusters

- Minimize total SSE

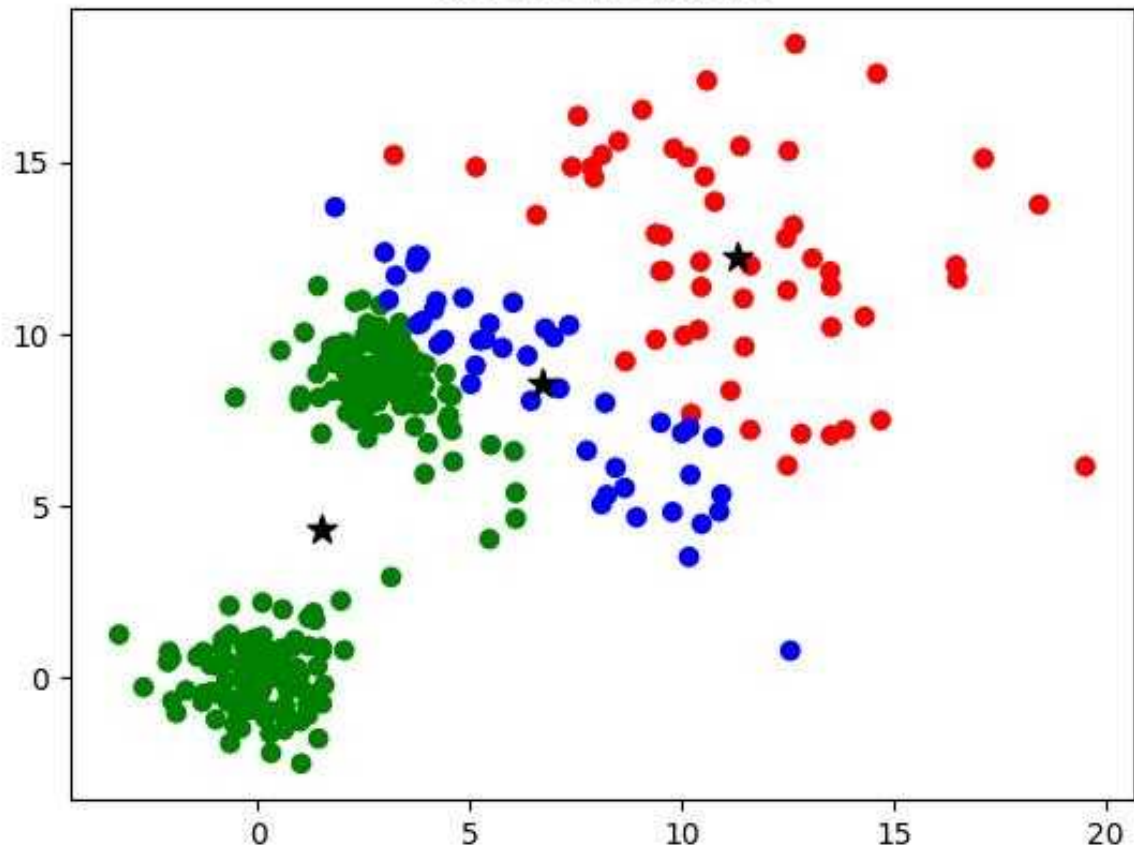
# Example of K-means Clustering



# Example of K-means Clustering

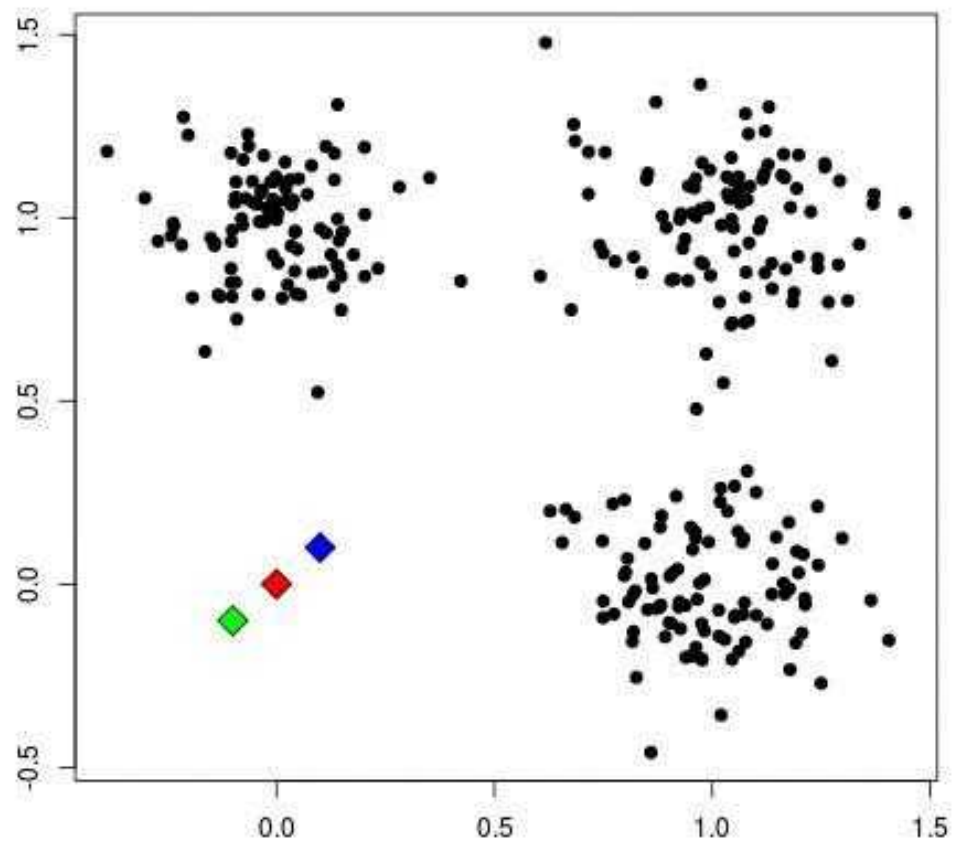


Kmeans Iteration 1





Start!



# K-means Objective Function

- A common objective function (used with Euclidean distance measure) is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster center
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} dist^2(m_i, x)$$

- $x$  is a data point in cluster  $C_i$  and  $m_i$  is the centroid (mean) for cluster  $C_i$
- SSE improves in each iteration of K-means until it reaches a local or global minima.

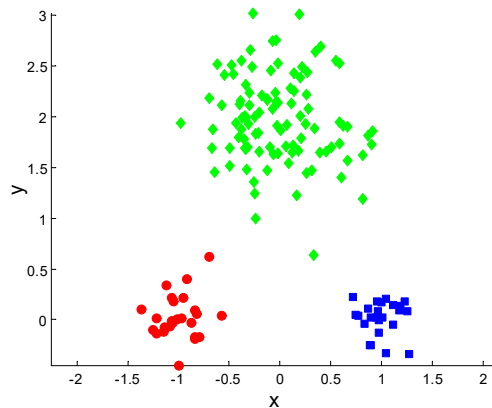
# K-means Objective Function

The diagram illustrates the K-means objective function  $J$  with the following components and annotations:

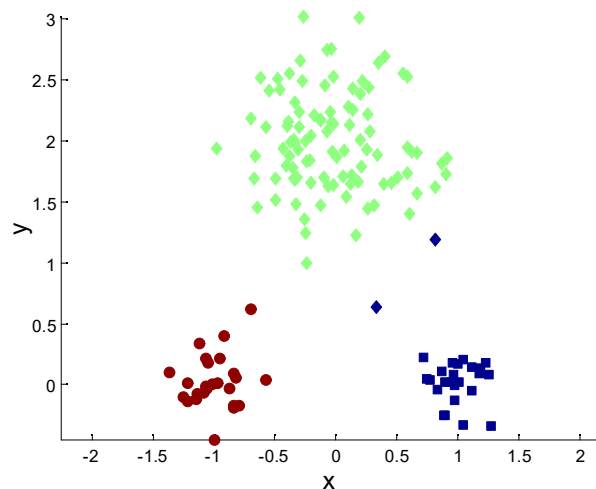
- number of clusters**: Points to the variable  $k$  in the outer summation.
- number of cases**: Points to the variable  $n$  in the inner summation.
- case  $i$** : Points to the variable  $i$  in the inner summation.
- centroid for cluster  $j$** : Points to the variable  $c_j$ .
- Distance function**: A bracket under the term  $\|x_i^{(j)} - c_j\|^2$  identifies it as the squared distance between a case and its cluster centroid.
- objective function**: Points to the variable  $J$ .

$$J = \sum_{j=1}^k \sum_{i=1}^n \|x_i^{(j)} - c_j\|^2$$

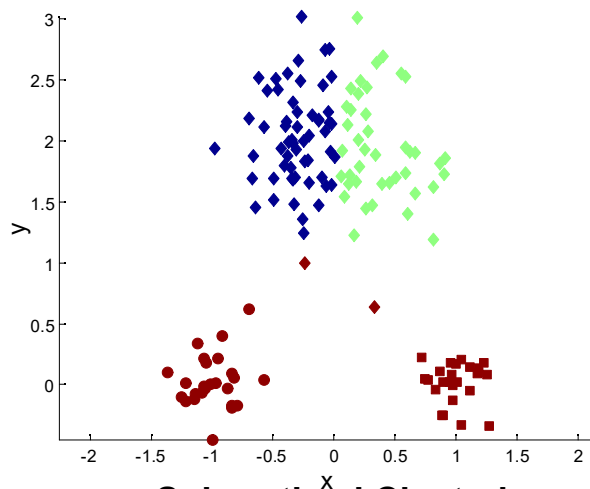
# Two different K-means Clustering



Original Points



Optimal Clustering



Sub-optimal Clustering

# K Mean Example

- Suppose we want to group the visitors to a website using just their age (one-dimensional space) as follows:
  - $n = 19$
  - 15,15,16,19,19,20,20,21,22,28,35,40,41,42,43,44,60,61,65
- Initial clusters (random centroid or average):
  - $k = 2$
  - $c1 = 16$
  - $c2 = 22$

# Iteration 1

$x_i$	$c_1$	$c_2$	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	16	22	1	7	1	15.33
15	16	22	1	7	1	
16	16	22	0	6	1	
19	16	22	9	3	2	36.25
19	16	22	9	3	2	
20	16	22	16	2	2	
20	16	22	16	2	2	
21	16	22	25	1	2	
22	16	22	36	0	2	
28	16	22	12	6	2	
35	16	22	19	13	2	
40	16	22	24	18	2	
41	16	22	25	19	2	
42	16	22	26	20	2	
43	16	22	27	21	2	
44	16	22	28	22	2	
60	16	22	44	38	2	
61	16	22	45	39	2	
65	16	22	49	43	2	

$c_1 = 15.33$  $c_2 = 36.25$

$Distance\ 1 = |x_i - c_1|$  $Distance\ 2 = |x_i - c_2|$

# Iteration 2

$x_i$	$c_1$	$c_2$	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	15.33	36.25	0.33	21.25	1	18.56
15	15.33	36.25	0.33	21.25	1	
16	15.33	36.25	0.67	20.25	1	
19	15.33	36.25	3.67	17.25	1	
19	15.33	36.25	3.67	17.25	1	
20	15.33	36.25	4.67	16.25	1	
20	15.33	36.25	4.67	16.25	1	
21	15.33	36.25	5.67	15.25	1	
22	15.33	36.25	6.67	14.25	1	
28	15.33	36.25	12.67	8.25	2	45.9
35	15.33	36.25	19.67	1.25	2	
40	15.33	36.25	24.67	3.75	2	
41	15.33	36.25	25.67	4.75	2	
42	15.33	36.25	26.67	5.75	2	
43	15.33	36.25	27.67	6.75	2	
44	15.33	36.25	28.67	7.75	2	
60	15.33	36.25	44.67	23.75	2	
61	15.33	36.25	45.67	24.75	2	
65	15.33	36.25	49.67	28.75	2	

c1 = 18.56  
c2 = 45.9

# Iteration 3

$x_i$	$c_1$	$c_2$	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	18.56	45.9	3.56	30.9	1	19.50
15	18.56	45.9	3.56	30.9	1	
16	18.56	45.9	2.56	29.9	1	
19	18.56	45.9	0.44	26.9	1	
19	18.56	45.9	0.44	26.9	1	
20	18.56	45.9	1.44	25.9	1	
20	18.56	45.9	1.44	25.9	1	
21	18.56	45.9	2.44	24.9	1	
22	18.56	45.9	3.44	23.9	1	
28	18.56	45.9	9.44	17.9	1	
35	18.56	45.9	16.44	10.9	2	47.89
40	18.56	45.9	21.44	5.9	2	
41	18.56	45.9	22.44	4.9	2	
42	18.56	45.9	23.44	3.9	2	
43	18.56	45.9	24.44	2.9	2	
44	18.56	45.9	25.44	1.9	2	
60	18.56	45.9	41.44	14.1	2	
61	18.56	45.9	42.44	15.1	2	
65	18.56	45.9	46.44	19.1	2	

c1 = 19.5  
c2 = 47.89



# Iteration 4

$x_i$	$c_1$	$c_2$	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	19.5	47.89	4.50	32.89	1	19.50
15	19.5	47.89	4.50	32.89	1	
16	19.5	47.89	3.50	31.89	1	
19	19.5	47.89	0.50	28.89	1	
19	19.5	47.89	0.50	28.89	1	
20	19.5	47.89	0.50	27.89	1	
20	19.5	47.89	0.50	27.89	1	
21	19.5	47.89	1.50	26.89	1	
22	19.5	47.89	2.50	25.89	1	
28	19.5	47.89	8.50	19.89	1	
35	19.5	47.89	15.50	12.89	2	47.89
40	19.5	47.89	20.50	7.89	2	
41	19.5	47.89	21.50	6.89	2	
42	19.5	47.89	22.50	5.89	2	
43	19.5	47.89	23.50	4.89	2	
44	19.5	47.89	24.50	3.89	2	
60	19.5	47.89	40.50	12.11	2	
61	19.5	47.89	41.50	13.11	2	
65	19.5	47.89	45.50	17.11	2	

c1 = 19.5  
c2 = 47.89

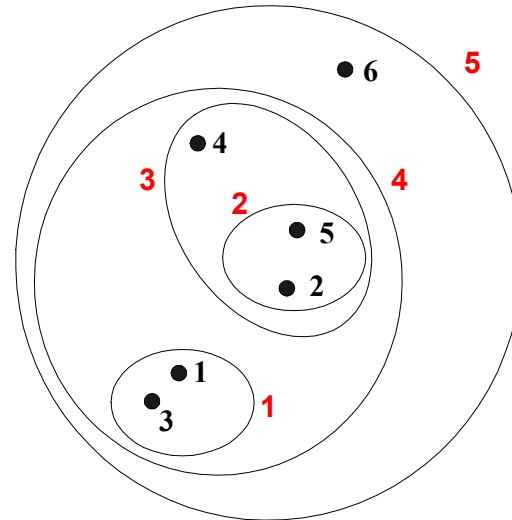
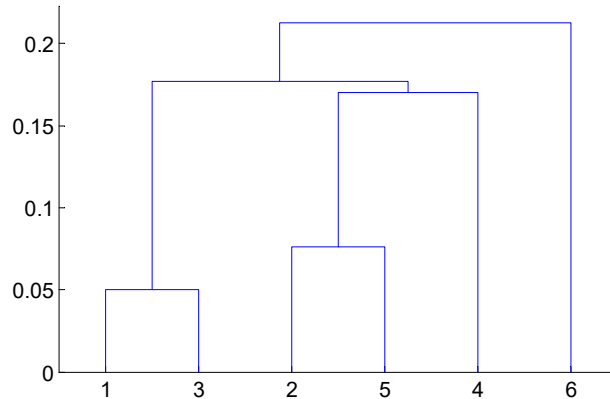
## K mean result notes

- No change between iterations 3 and 4 has been noted.
- By using clustering, 2 groups have been identified 15-28 and 35-65.
- The initial choice of centroids can affect the output clusters, so the algorithm is often run multiple times with different starting conditions in order to get a fair view of what the clusters should be.

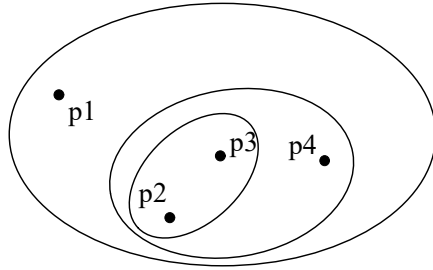
Iteration	Centroid 1 (C1)	Centroid 2 (C2)
Initial	16	22
1	15.33	36.25
2	18.56	45.90
3	19.50	47.89
4	19.50	47.89

# Hierarchical Clustering

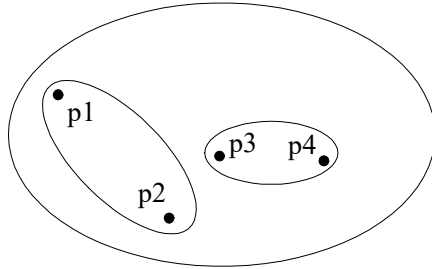
- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits



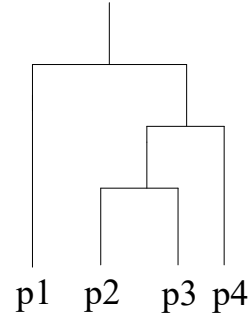
# Hierarchical Clustering



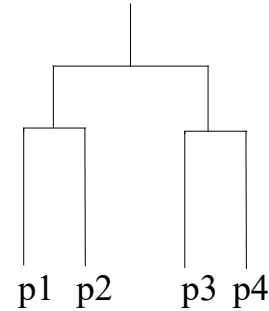
**Traditional Hierarchical Clustering**



**Non-traditional Hierarchical Clustering**

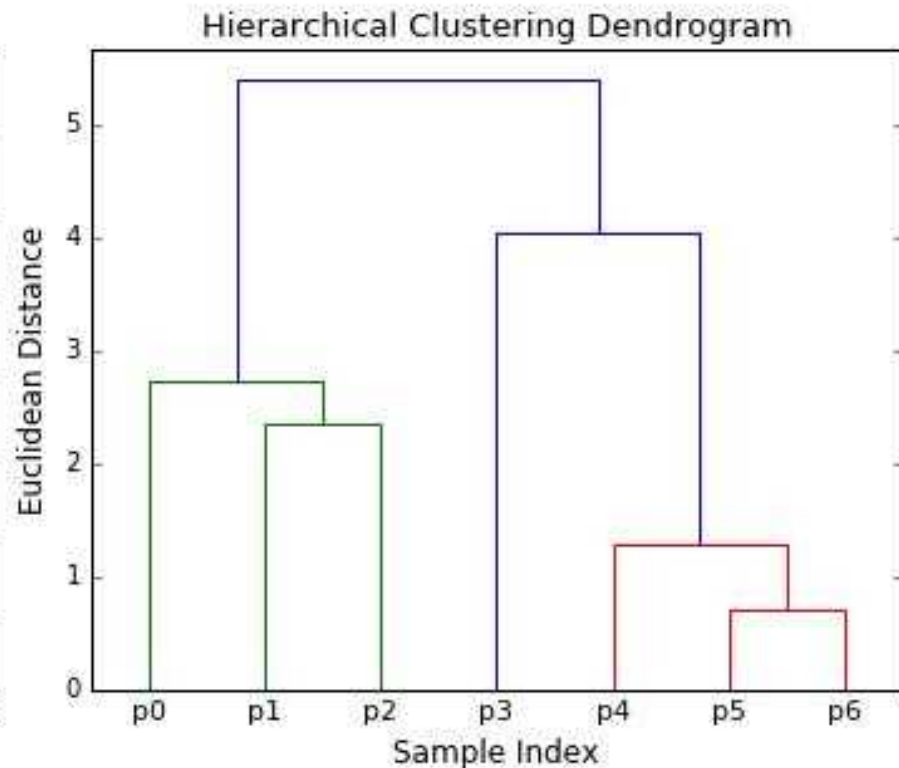
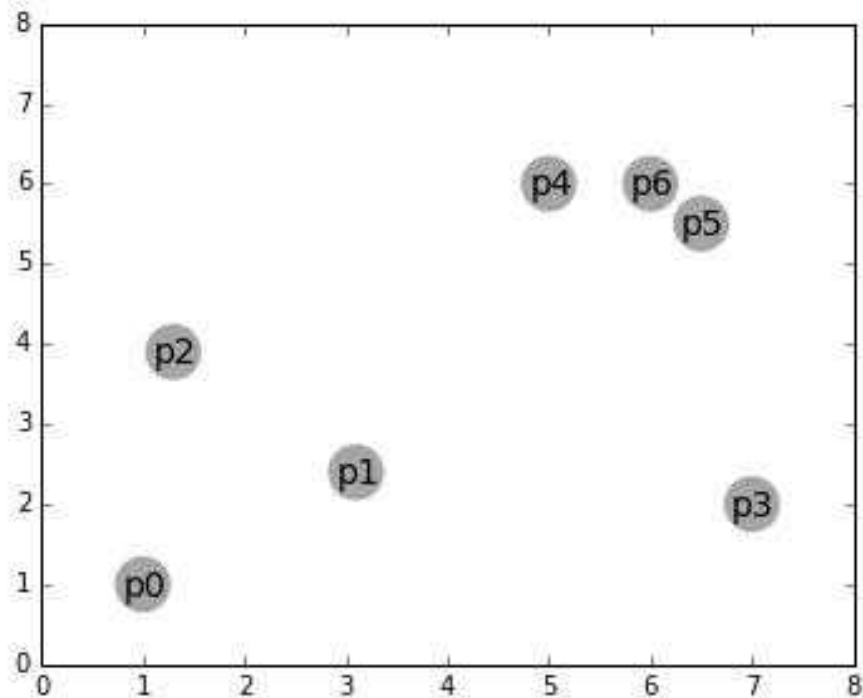


**Traditional Dendrogram**



**Non-traditional Dendrogram**

# Hierarchical Clustering Animation



# Strengths of Hierarchical Clustering

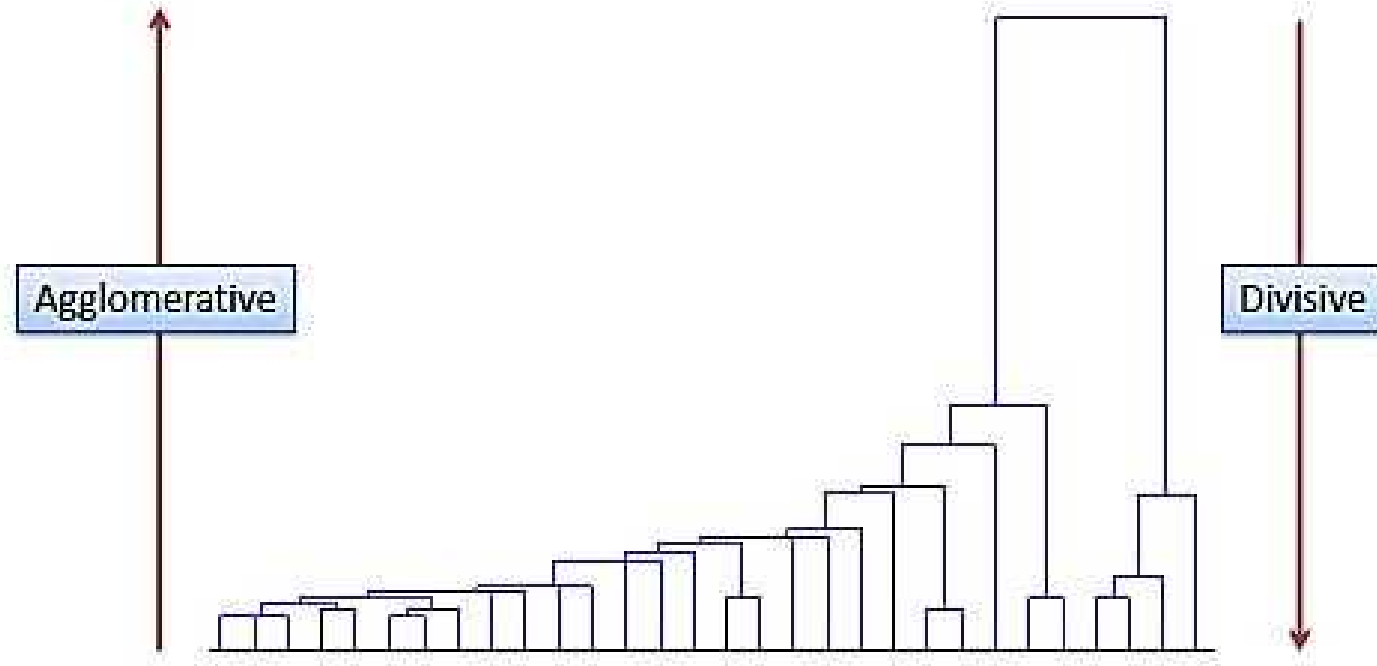
- Do not have to assume any number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level
- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

# Hierarchical Clustering

- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or  $k$  clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains an individual point (or there are  $k$  clusters)

# Agglomerative vs divisive hierarchical clustering

## Hierarchical Clustering



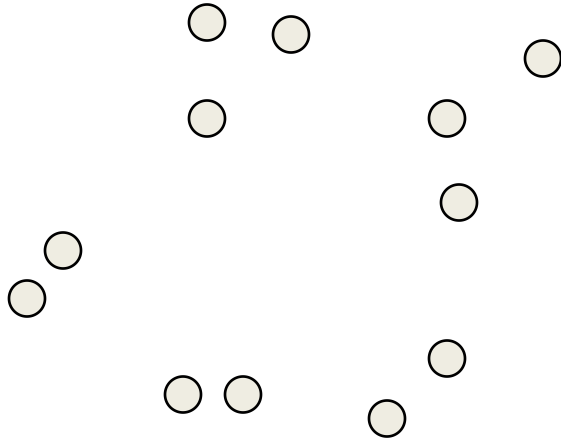


# Agglomerative Clustering Algorithm

- **Key Idea: Successively merge closest clusters**
- Basic algorithm
  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

# Steps 1 and 2

- Start with clusters of individual points and a proximity matrix



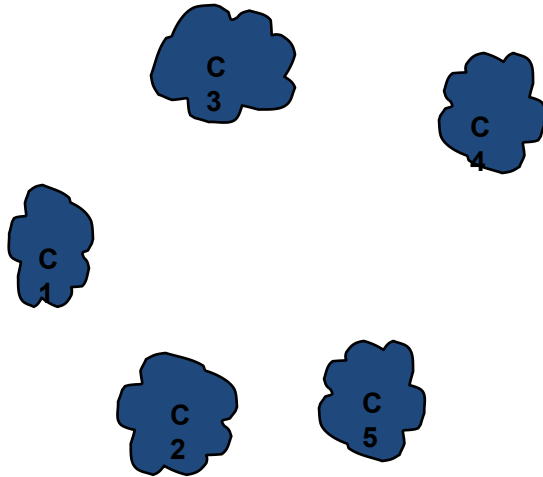
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

**Proximity Matrix**



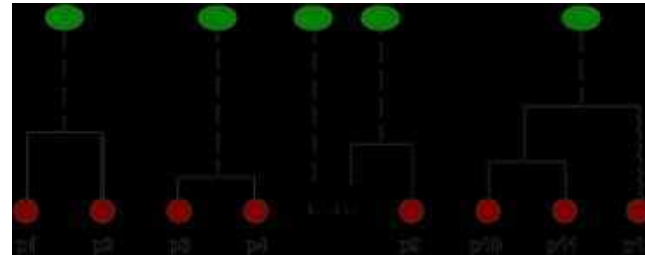
# Intermediate Situation

- After some merging steps, we have some clusters



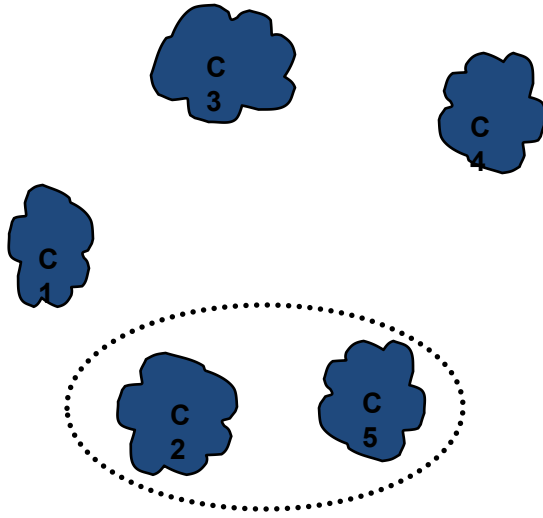
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



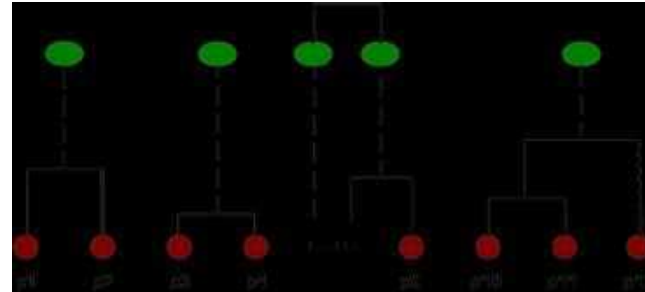
## Step 4

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



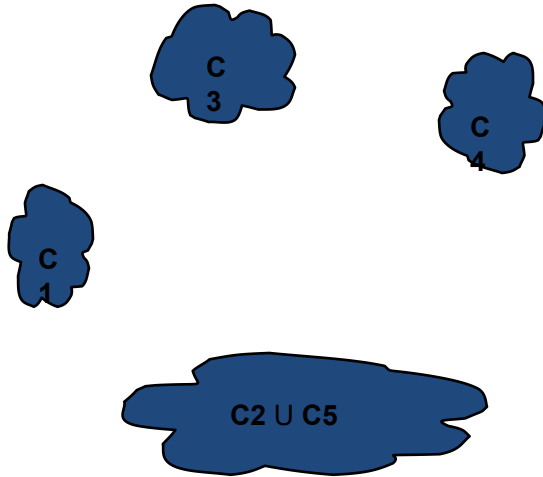
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



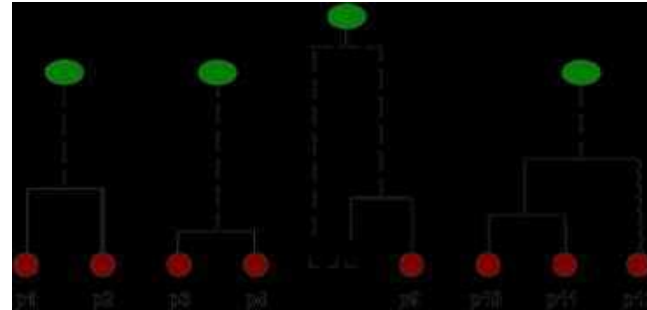
# Step 5

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

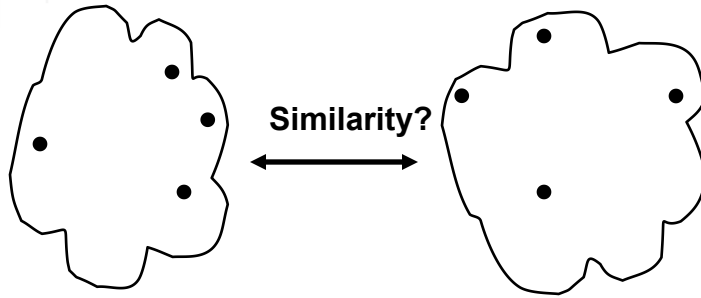


	C1	$C2 \cup C5$	C3	C4
C1		?		
$C2 \cup C5$	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



# How to Define Inter-Cluster Distance

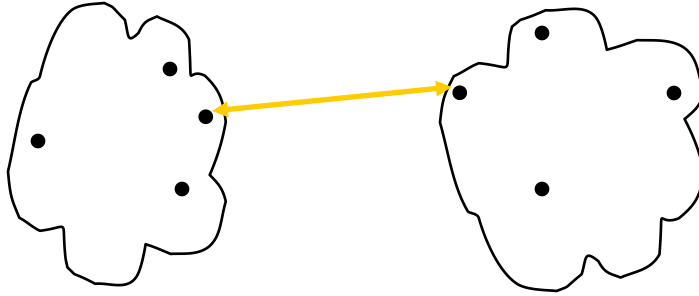


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

**Proximity Matrix**

# How to Define Inter-Cluster Similarity

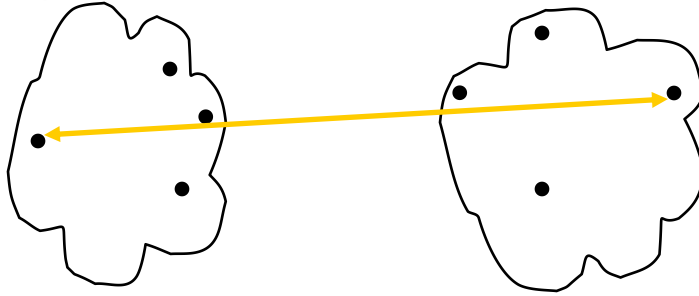


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

**Proximity Matrix**

# How to Define Inter-Cluster Similarity



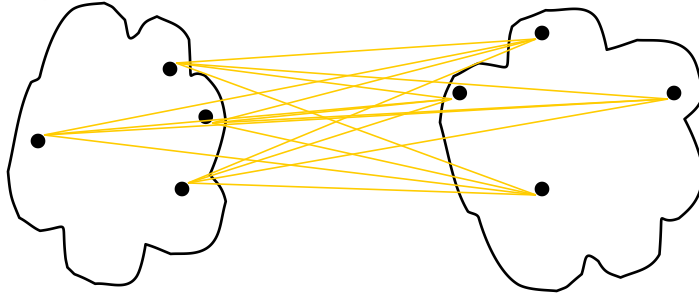
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

**Proximity Matrix**



# How to Define Inter-Cluster Similarity

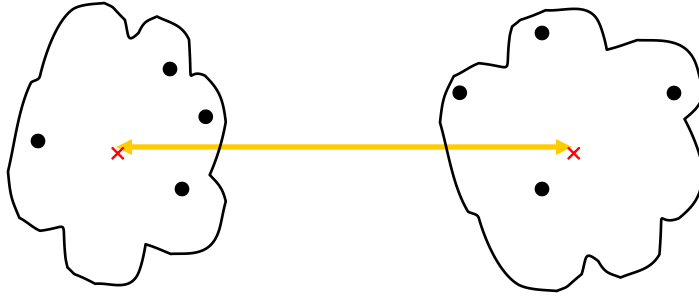


- MIN
- MAX
- **Group Average**
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

**Proximity Matrix**

# How to Define Inter-Cluster Similarity

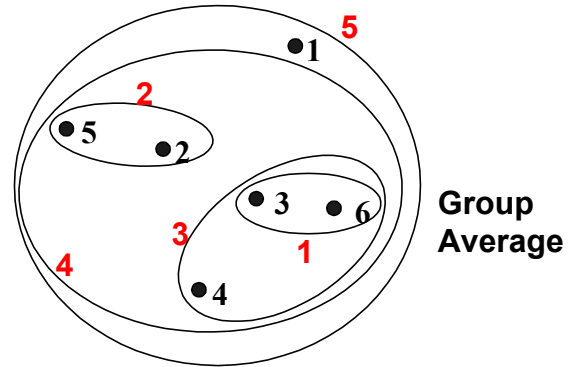
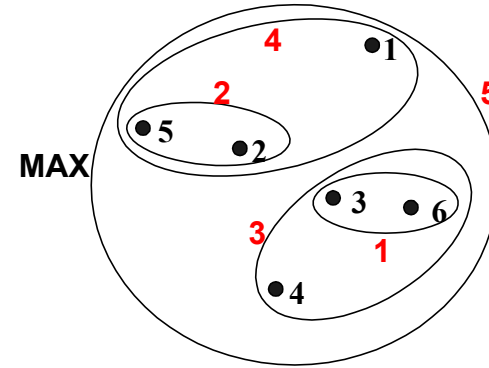
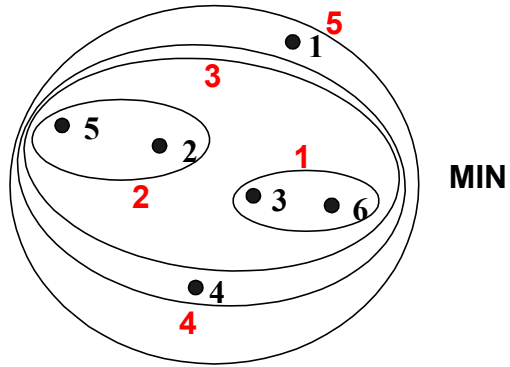


- MIN
- MAX
- Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function
  - Ward's Method uses squared error

	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

**Proximity Matrix**

# Hierarchical Clustering: Comparison

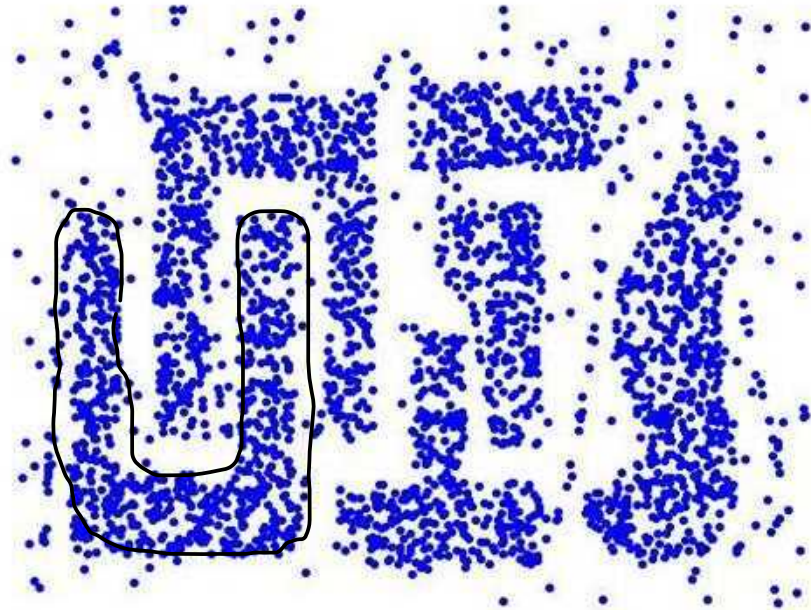




# Density based clustering

# Density Based Clustering

- Clusters are regions of high density that are separated from one another by regions of low density.

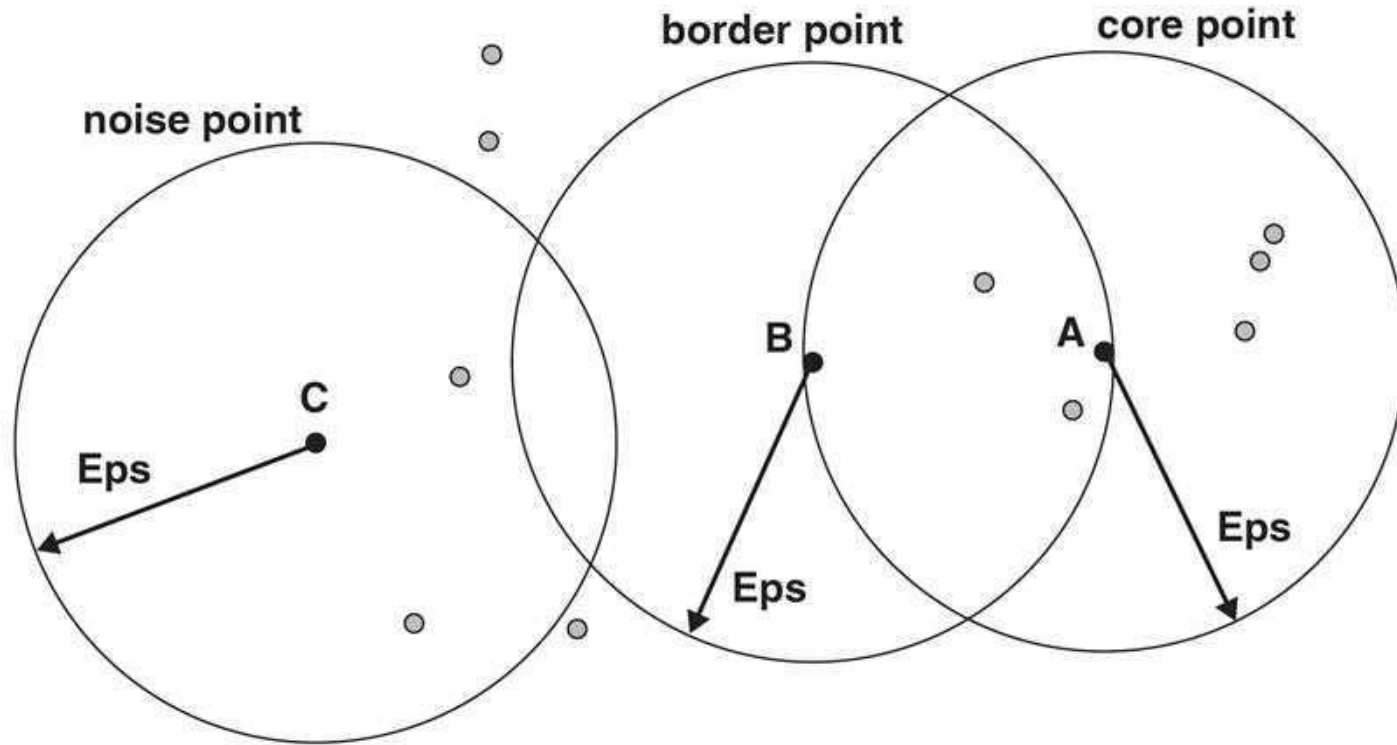


# DBSCAN

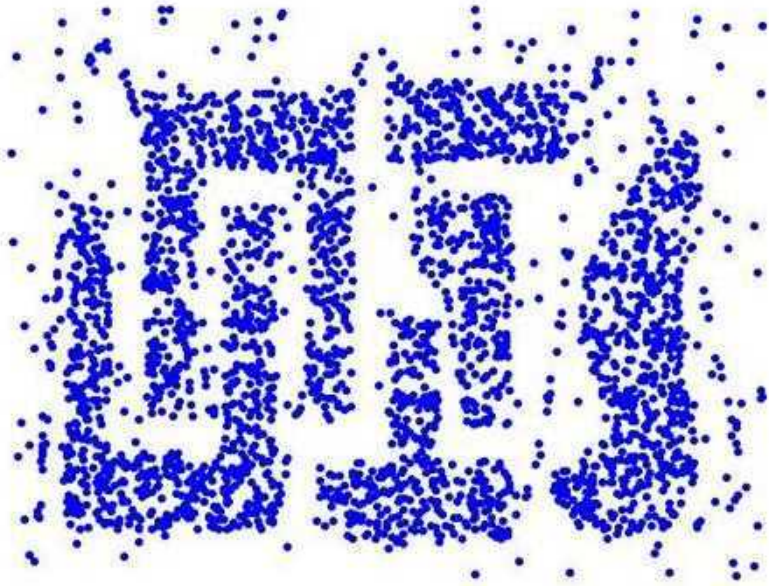
- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has at least a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
    - Counts the point itself
  - A border 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: Core, Border, and Noise Points

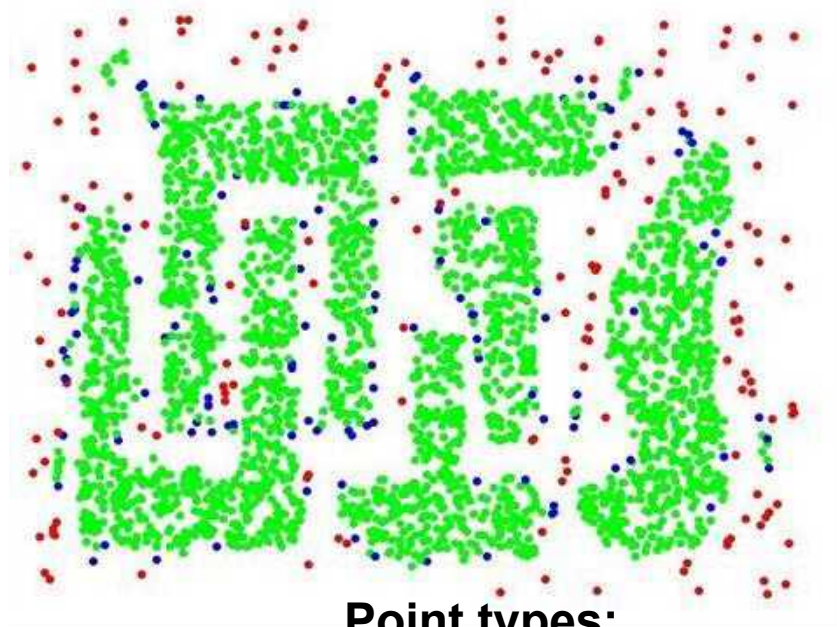
MinPts = 7



## DBSCAN: Core, Border and Noise Points



Original Points



Point types:

Core

Border

Noise

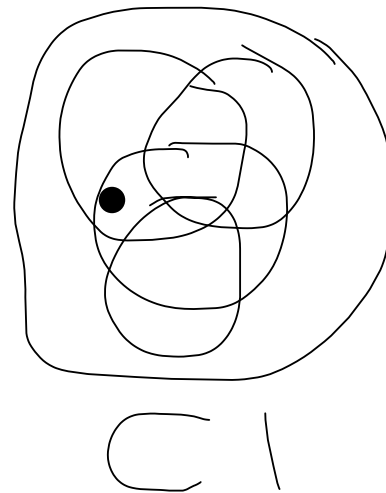
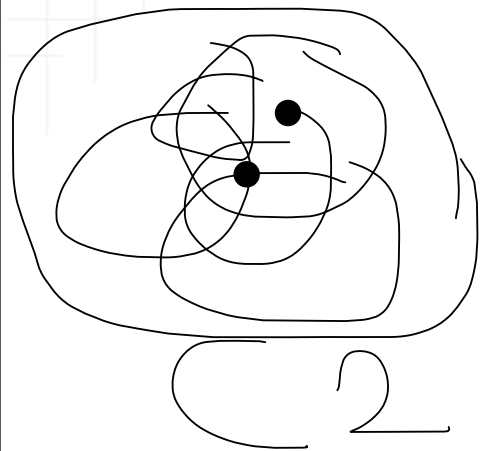
Cluster points

Eps = 10, MinPts = 4

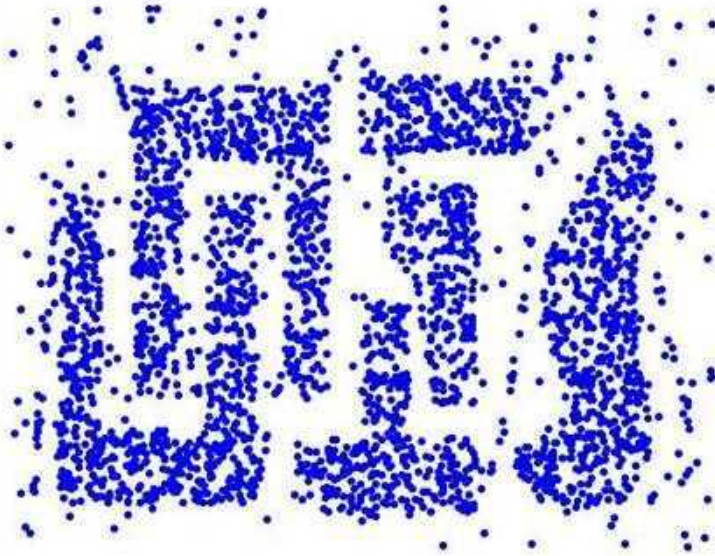


# DBSCAN Algorithm

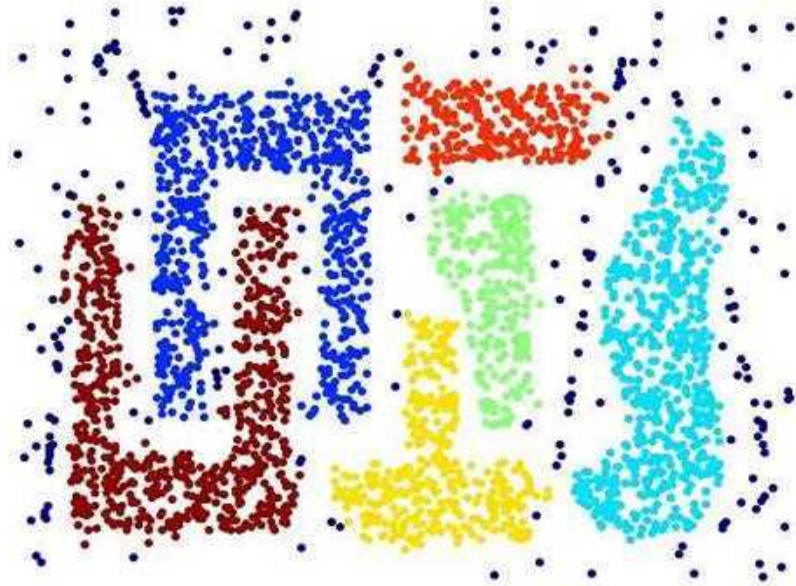
- Form clusters using core points, and assign border points to one of its neighboring clusters
- 1: Label all points as core, border, or noise points.
  - 2: Eliminate noise points.
  - 3: Put an edge between all core points within a distance  $Eps$  of each other.
  - 4: Make each group of connected core points into a separate cluster.
  - 5: Assign each border point to one of the clusters of its associated core points



## When DBSCAN Works Well



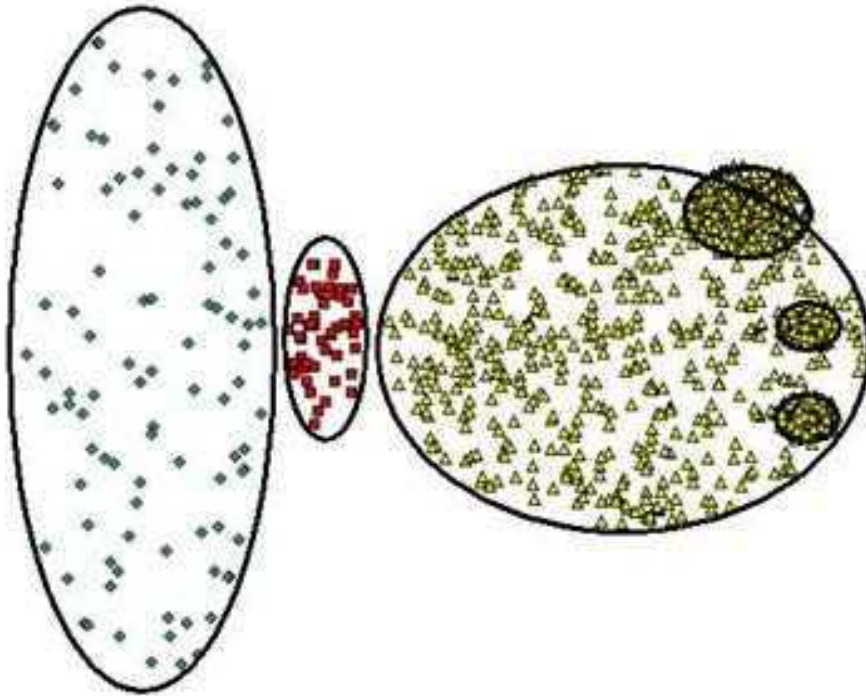
Original Points



Clusters (dark blue points indicate noise)

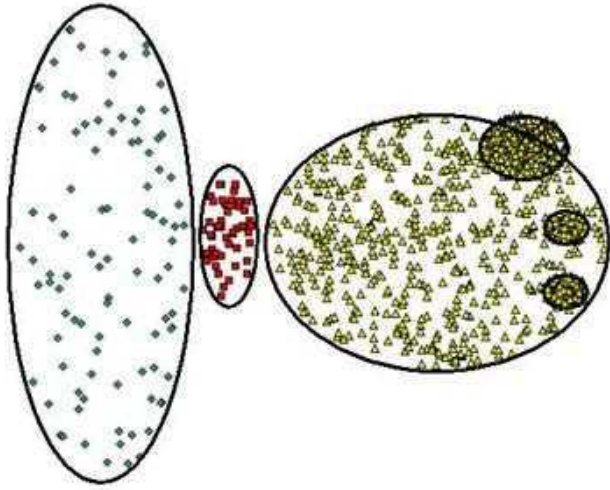
- Can handle clusters of different shapes and sizes
- Resistant to noise

## When DBSCAN Does NOT Work Well



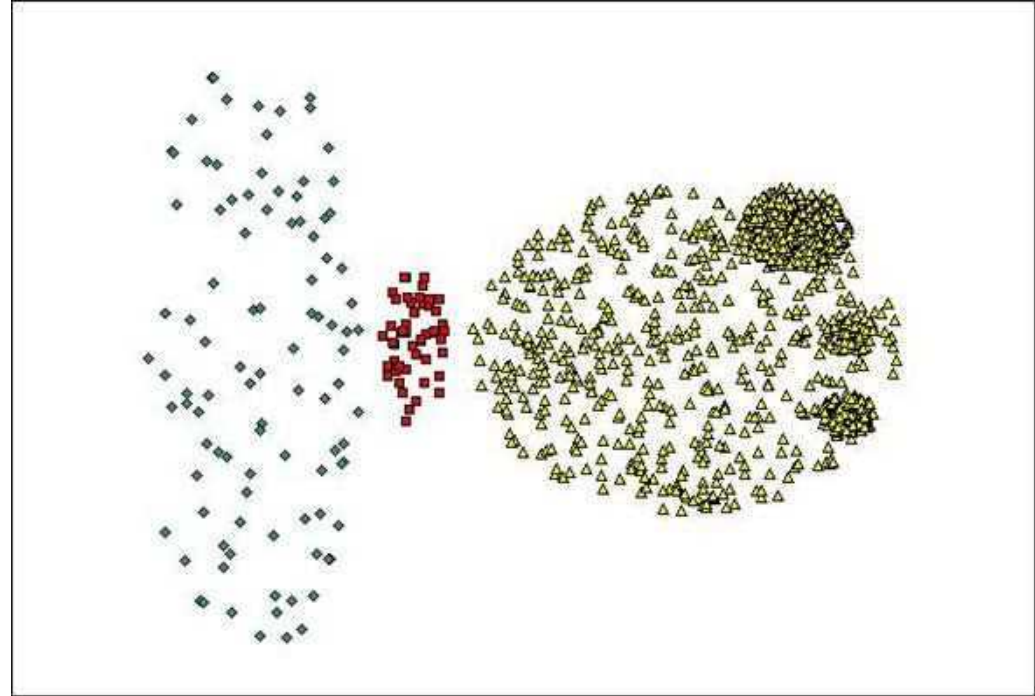
Original Points

## When DBSCAN Does NOT Work Well



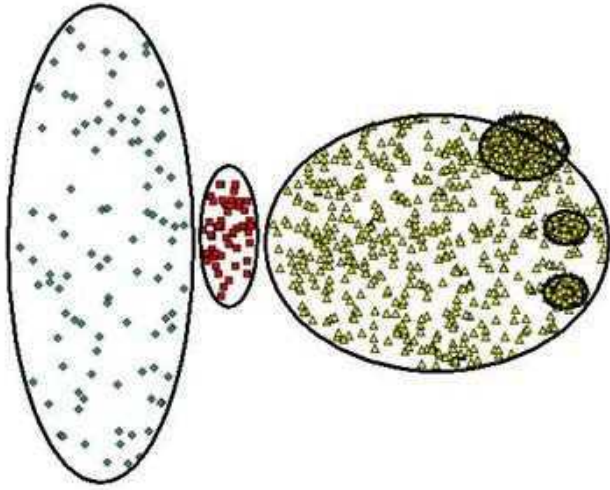
Original Points

- Varying densities
- High-dimensional data



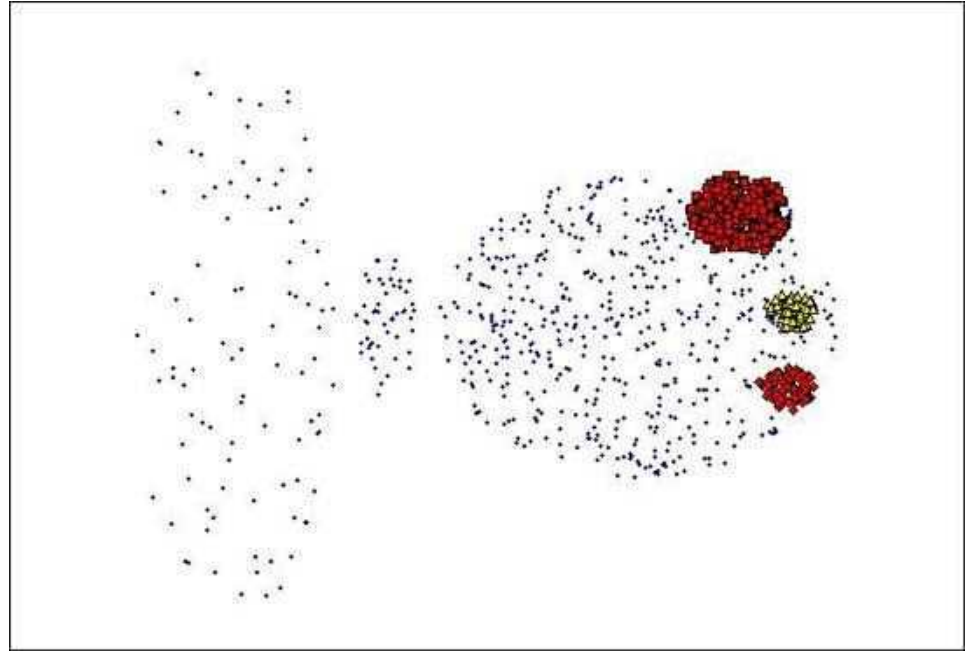
(MinPts=4, Eps=9.92).

## When DBSCAN Does NOT Work Well



**Original Points**

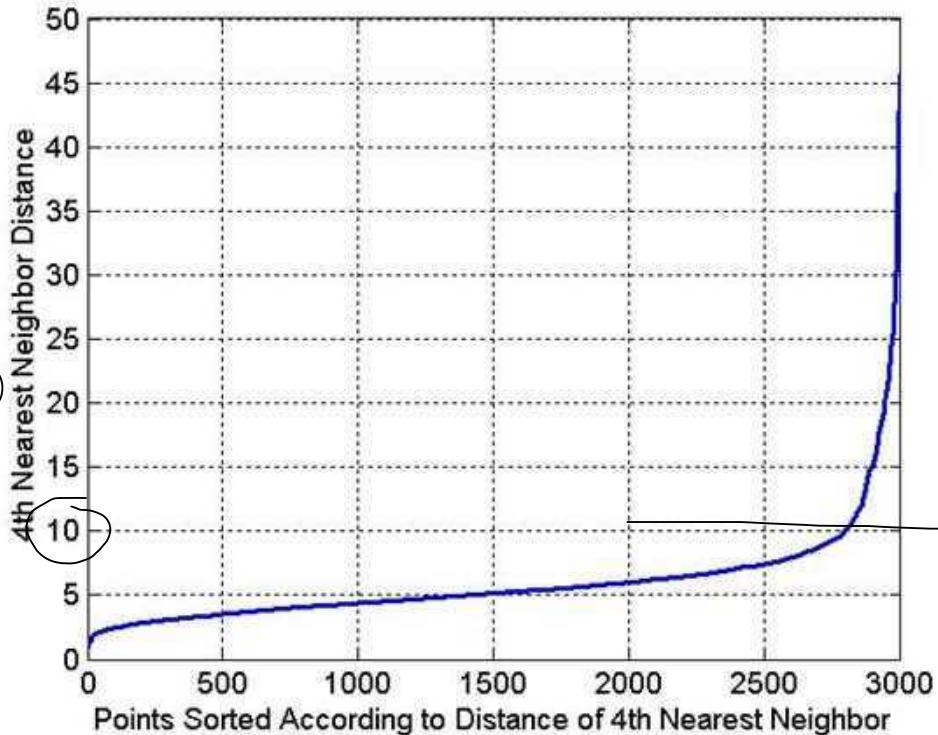
- Varying densities
- High-dimensional data



(MinPts=4, Eps=9.75)

## DBSCAN: Determining EPS and MinPts

- Idea is that for points in a cluster, their  $k^{\text{th}}$  nearest neighbors are at close distance
- Noise points have the  $k^{\text{th}}$  nearest neighbor at farther distance
- So, plot sorted distance of every point to its  $k^{\text{th}}$  nearest neighbor



MinPts = 4

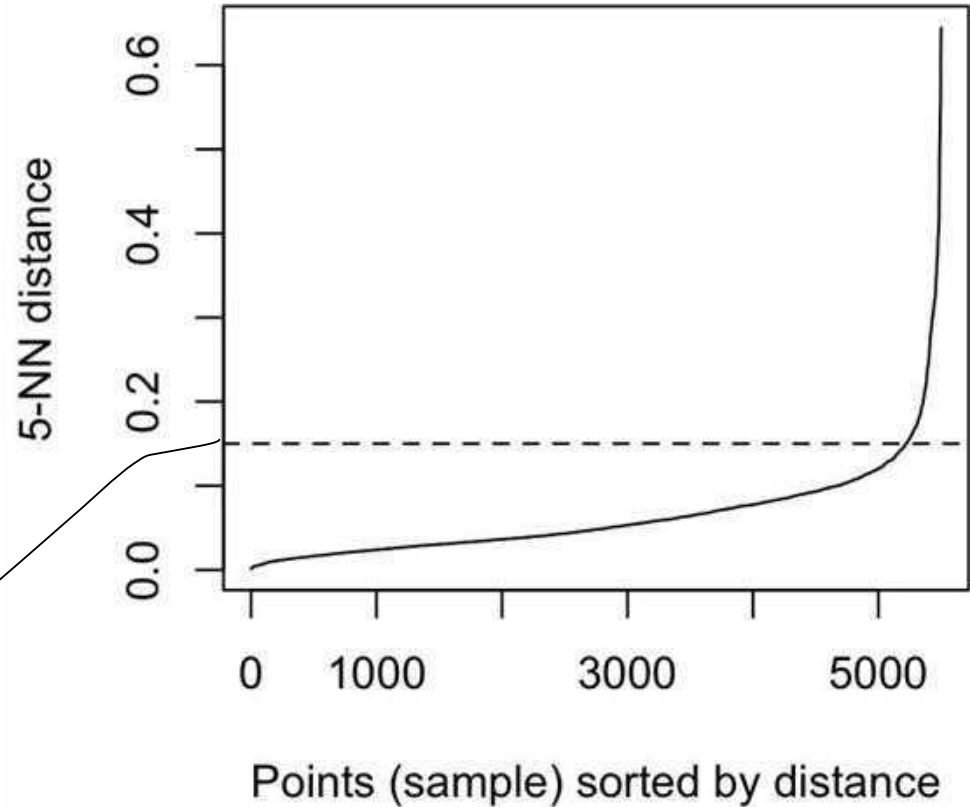
Eps = 10



# DBSCAN Optimal parameters

- The aim is to determine the “knee”, which corresponds to the optimal eps parameter.
- A knee corresponds to a threshold where a sharp change occurs along the k-distance curve.

$$\text{Min } P+ = 5$$
$$\text{Eps} = r = 0.17$$







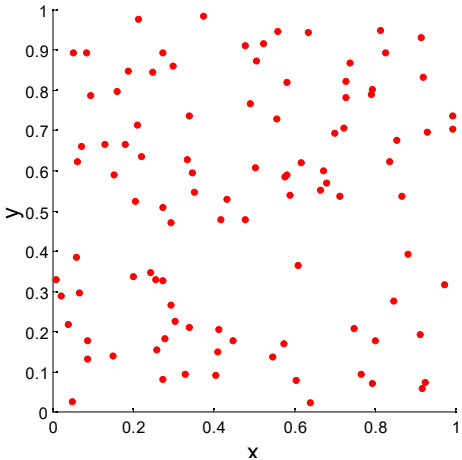
# Cluster Validity

# Cluster Validity

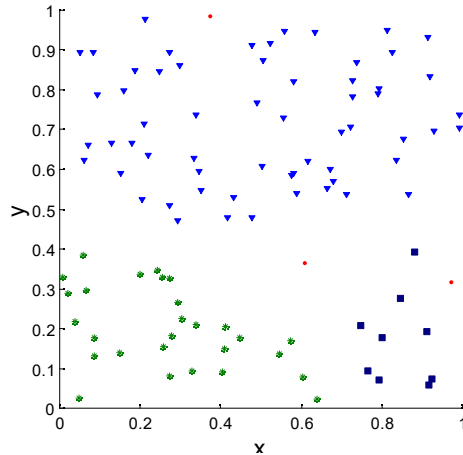
- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?
- But “clusters are in the eye of the beholder”!
  - In practice the clusters we find are defined by the clustering algorithm
- 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

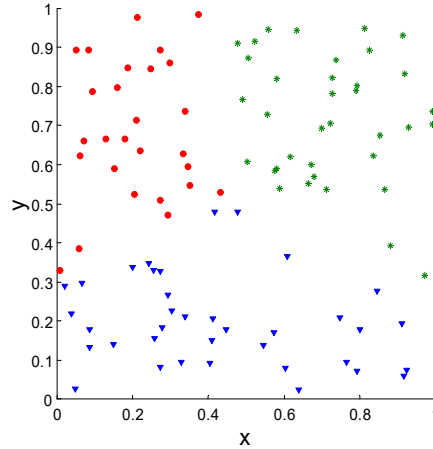
Random Points



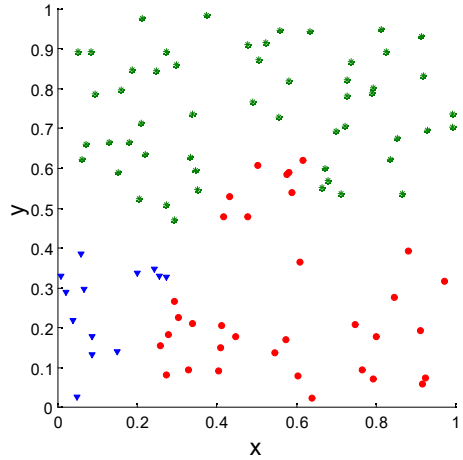
DBSCAN



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 two types.
  - **Supervised:** Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
    - Often called *external indices* because they use information external to the data
  - **Unsupervised:** Used to measure the goodness of a clustering structure *without* respect to external information.
    - Sum of Squared Error (SSE)
    - Often called *internal indices* because they only use information in the data
- You can use supervised or unsupervised measures to compare clusters or clustering

## Supervised Measures of Cluster Validity: Entropy and Purity

**Table 5.9.** K-means Clustering Results for LA Document Data Set

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports	Entropy	Purity
1	3	5	40	506	96	27	1.2270	0.7474
2	4	7	280	29	39	2	1.1472	0.7756
3	1	1	1	7	4	671	0.1813	0.9796
4	10	162	3	119	73	2	1.7487	0.4390
5	331	22	5	70	13	23	1.3976	0.7134
6	5	358	12	212	48	13	1.5523	0.5525
Total	354	555	341	943	273	738	1.1450	0.7203

**entropy** For each cluster, the class distribution of the data is calculated first, i.e., for cluster  $j$  we compute  $p_{ij}$ , the ‘probability’ that a member of cluster  $j$  belongs to class  $i$  as follows:  $p_{ij} = m_{ij}/m_j$ , where  $m_j$  is the number of values in cluster  $j$  and  $m_{ij}$  is the number of values of class  $i$  in cluster  $j$ . Then using this class distribution, the entropy of each cluster  $j$  is calculated using the standard formula  $e_j = \sum_{i=1}^L p_{ij} \log_2 p_{ij}$ , where the  $L$  is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster, i.e.,  $e = \sum_{i=1}^K \frac{m_i}{m} e_j$ , where  $m_j$  is the size of cluster  $j$ ,  $K$  is the number of clusters, and  $m$  is the total number of data points.

**purity** Using the terminology derived for entropy, the purity of cluster  $j$ , is given by  $\text{purity}_j = \max p_{ij}$  and the overall purity of a clustering by  $\text{purity} = \sum_{i=1}^K \frac{m_i}{m} \text{purity}_j$ .

## Unsupervised Measures: Cohesion and Separation

- **Cluster Cohesion:** Measures how closely related are objects in a cluster
  - Example: SSE
- **Cluster Separation:** Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)
  - Separation is measured by the between cluster sum of squares

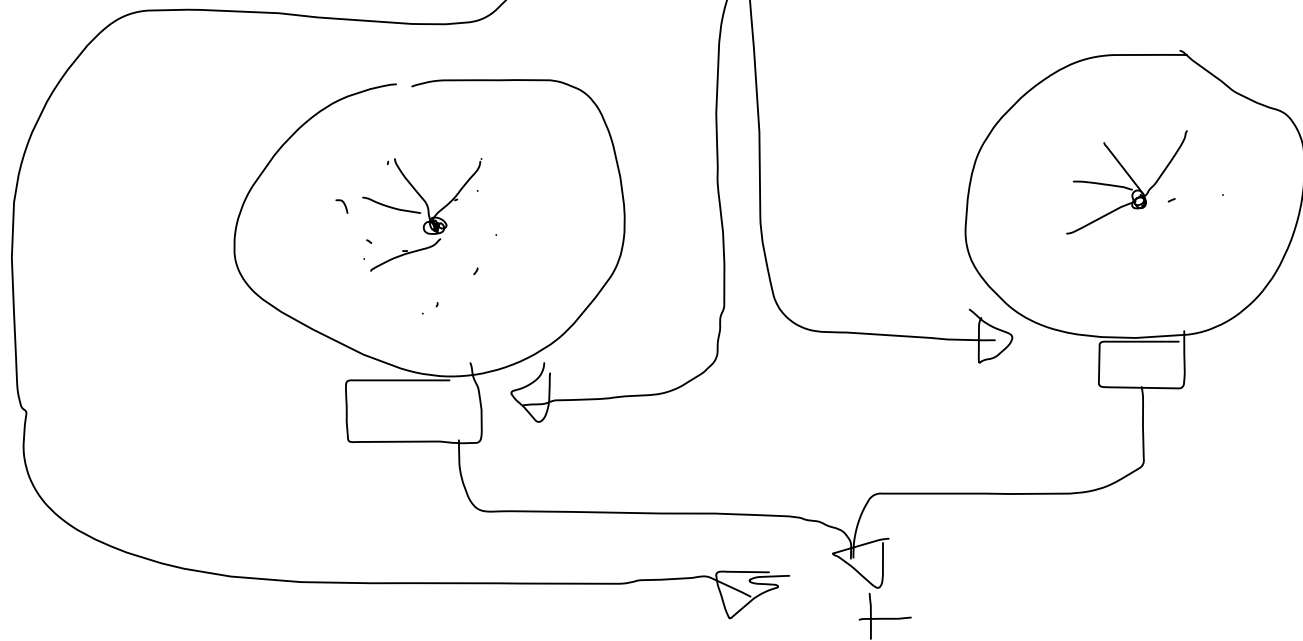
$$SSE = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

Where  $|C_i|$  is the size of cluster  $i$

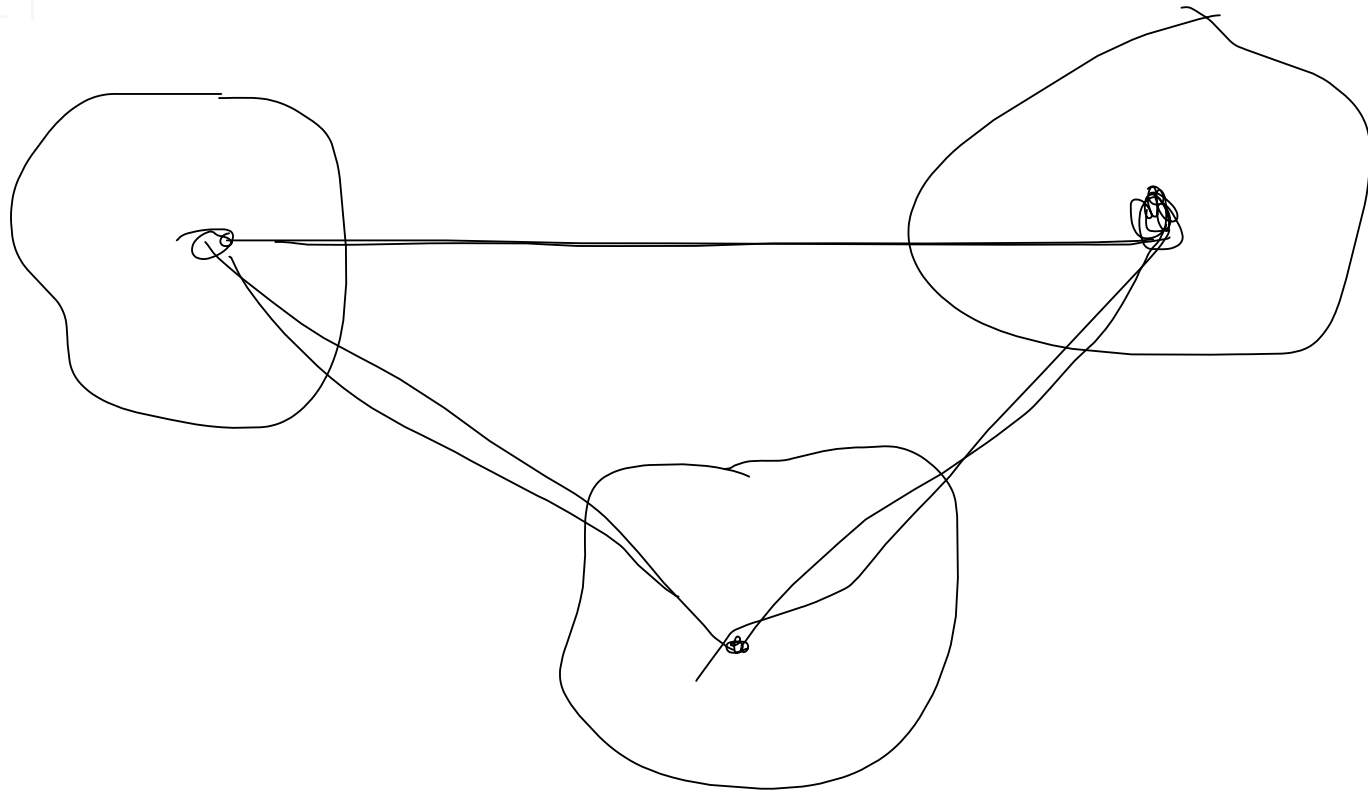
$$SSB = \sum_i |C_i| (m - m_i)^2$$

SS E

$$\sum \sum (X - M \dot{e})$$

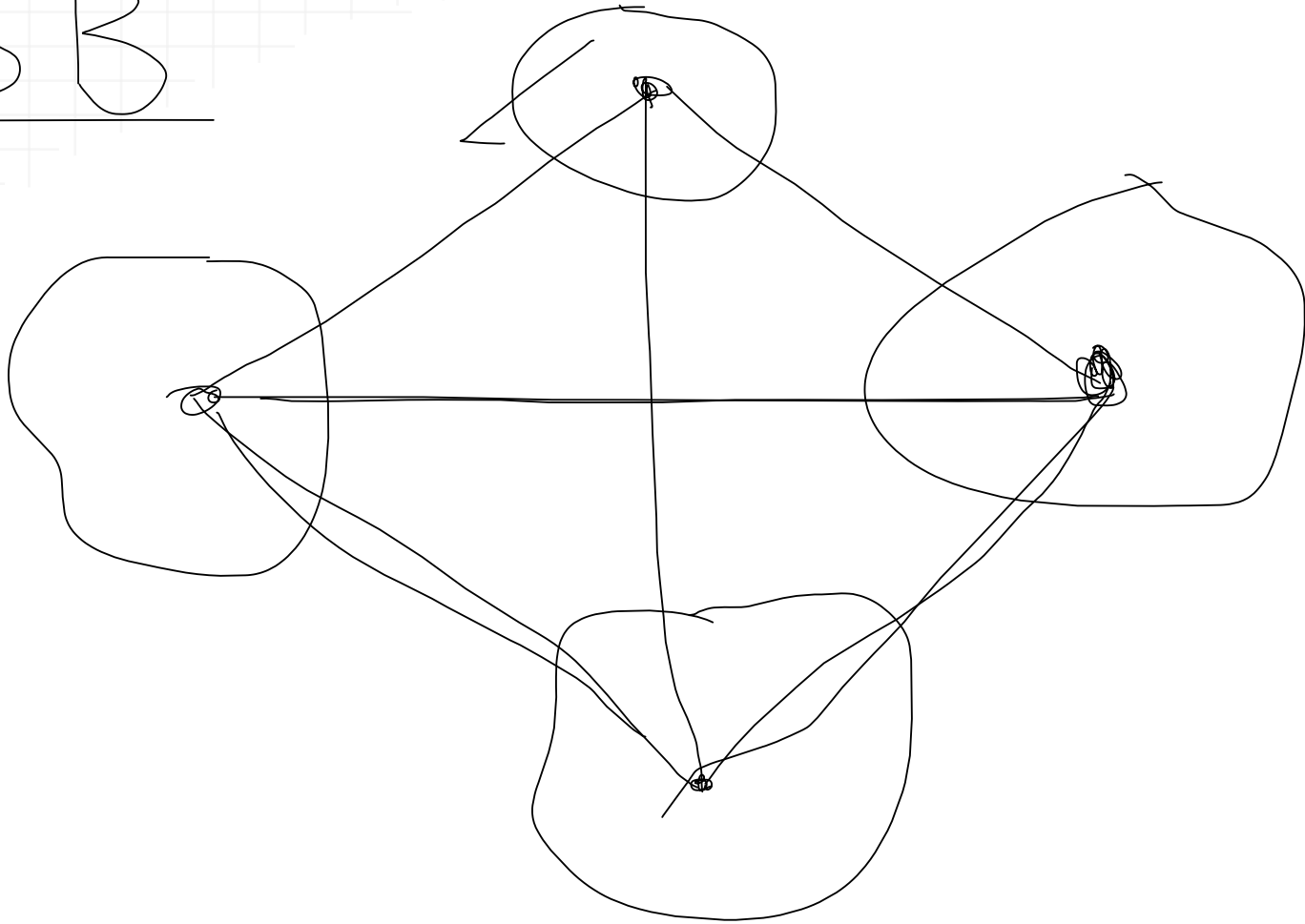


SSB



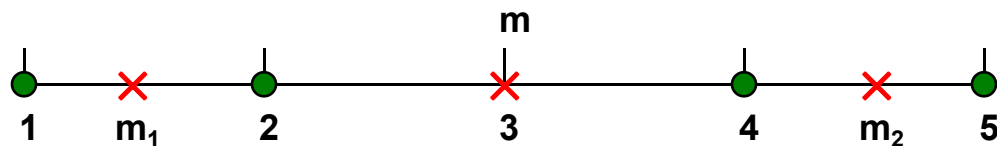


SSB



# Unsupervised Measures: Cohesion and Separation

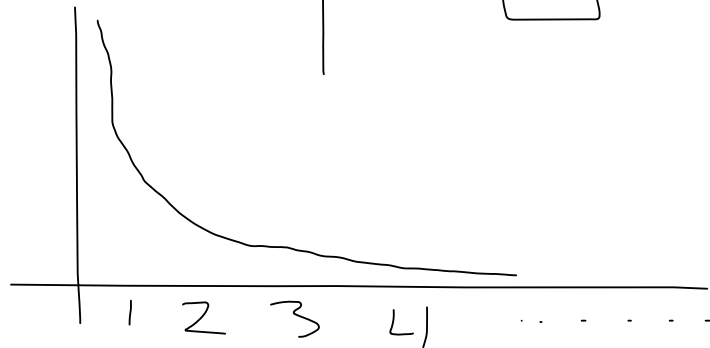
- Example: SSE
  - SSB + SSE = constant



**K=1 cluster:**  $SSE = (1 - 3)^2 + (2 - 3)^2 + (4 - 3)^2 + (5 - 3)^2 = 10$   
 $SSB = 4 \times (3 - 3)^2 = 0$   
 $Total = 10 + 0 = 10$

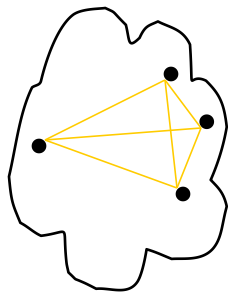
**K=2 clusters:**  $SSE = (1 - 1.5)^2 + (2 - 1.5)^2 + (4 - 4.5)^2 + (5 - 4.5)^2 = 1$   
 $SSB = 2 \times (3 - 1.5)^2 + 2 \times (4.5 - 3)^2 = 9$   
 $Total = 1 + 9 = 10$

1 <	SSE	SSB	<sup>0.8</sup> <sup>0.2</sup> $SSE + SSB$
1 2 3 4 ...			

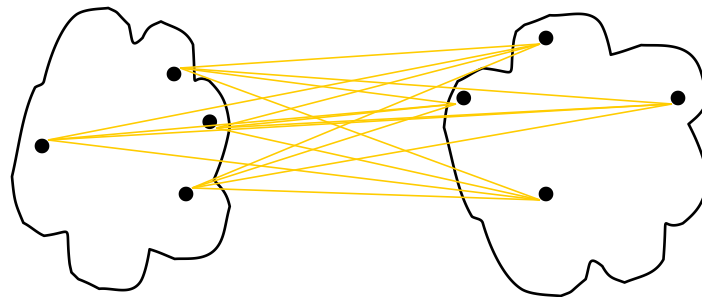


## Unsupervised Measures: Cohesion and Separation

- A proximity graph-based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



cohesion



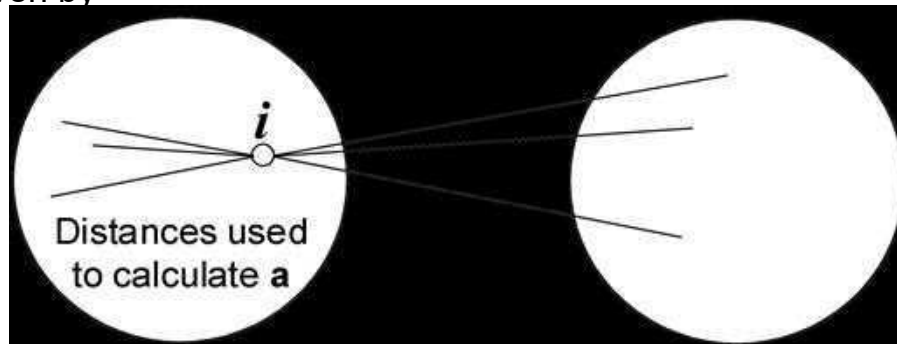
separation

## Unsupervised Measures: Silhouette Coefficient

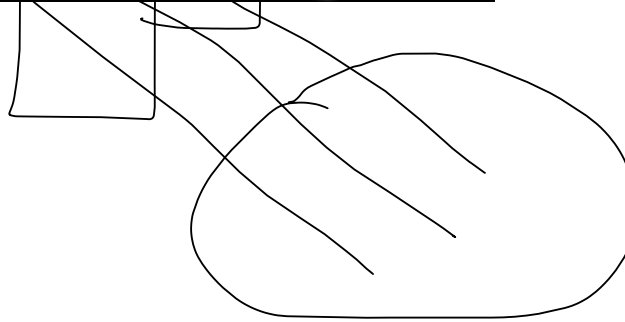
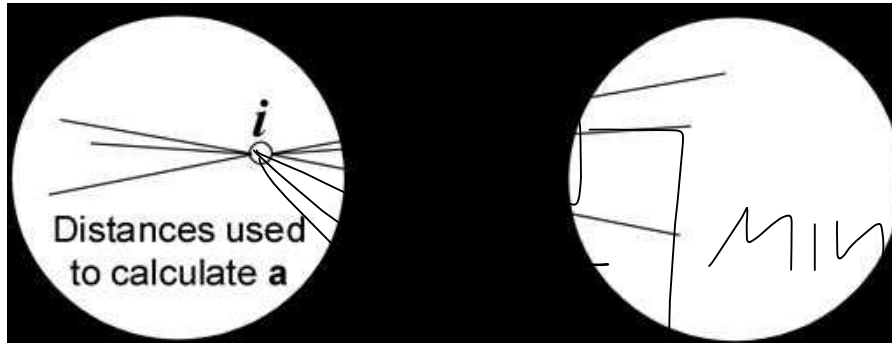
- Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point,  $i$ 
  - Calculate  $a$  = average distance of  $i$  to the points in its cluster
  - Calculate  $b$  = min (average distance of  $i$  to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

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

- Value can vary between -1 and 1
  - Typically ranges between 0 and 1.
  - The closer to 1 the better.
- Can calculate the average silhouette coefficient for a cluster or a clustering



## Unsupervised Measures: Silhouette Coefficient



# Measuring Cluster Validity Via Correlation

- Two matrices
  - Proximity Matrix
  - Ideal Similarity Matrix
    - One row and one column for each data point
    - An entry is 1 if the associated pair of points belong to the same cluster
    - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
  - Since the matrices are symmetric, only the correlation between  $n(n-1) / 2$  entries needs to be calculated.
- High magnitude of correlation indicates that points that belong to the same cluster are close to each other.
  - Correlation may be positive or negative depending on whether the similarity matrix is a similarity or dissimilarity matrix
- Not a good measure for some density or contiguity based clusters.

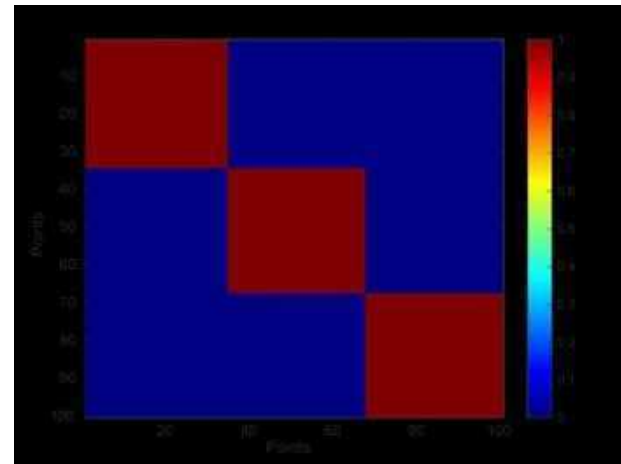
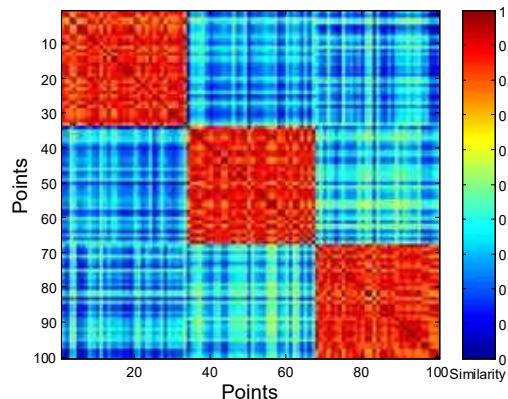
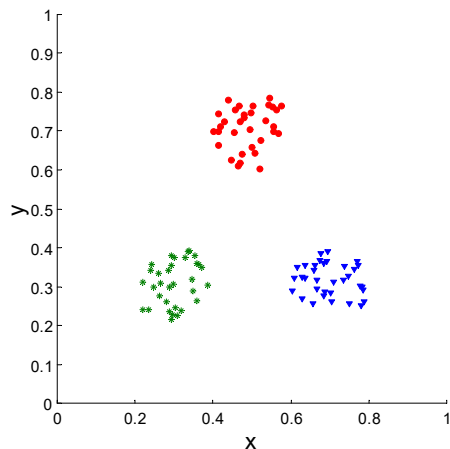
	Pro X					
	$P_1$	$P_2$	$P_3$	-	.	.
$P_1$	0	0.3				
$P_2$	0.3	0				
$P_3$	0	0.2	0			
.						
.						
.						

Sim					
	$P_1$	$P_2$	$P_3$	-	...
$P_1$	1	0	1	0	1
$P_2$					
$P_3$					



# Measuring Cluster Validity Via Correlation

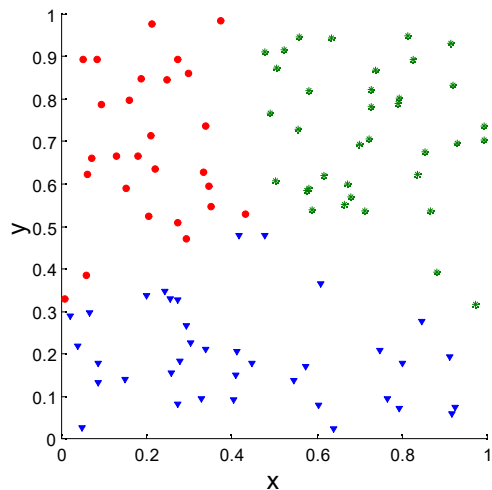
- Correlation of ideal similarity and proximity matrices for the K-means clusterings of the following well-clustered data set.



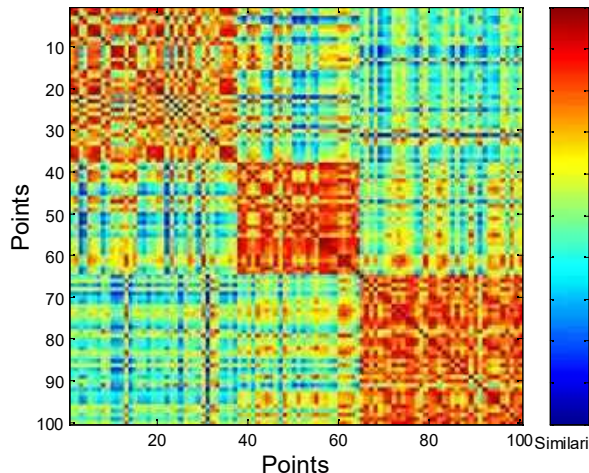
**Corr = 0.9235**

# Measuring Cluster Validity Via Correlation

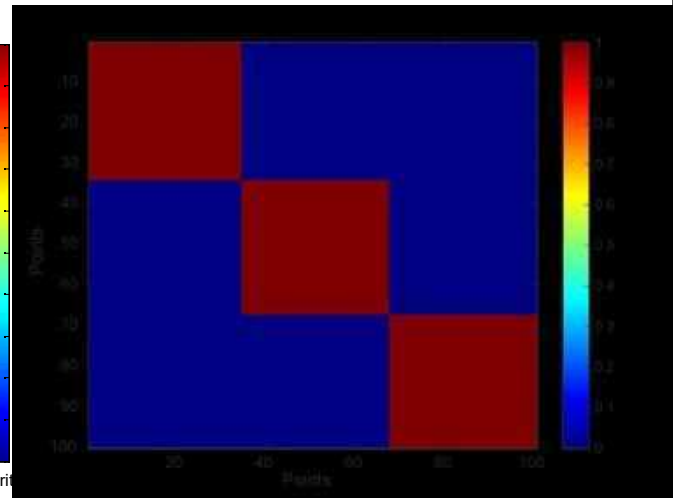
- Correlation of ideal similarity and proximity matrices for the K-means clusterings of the following random data set.



K-means

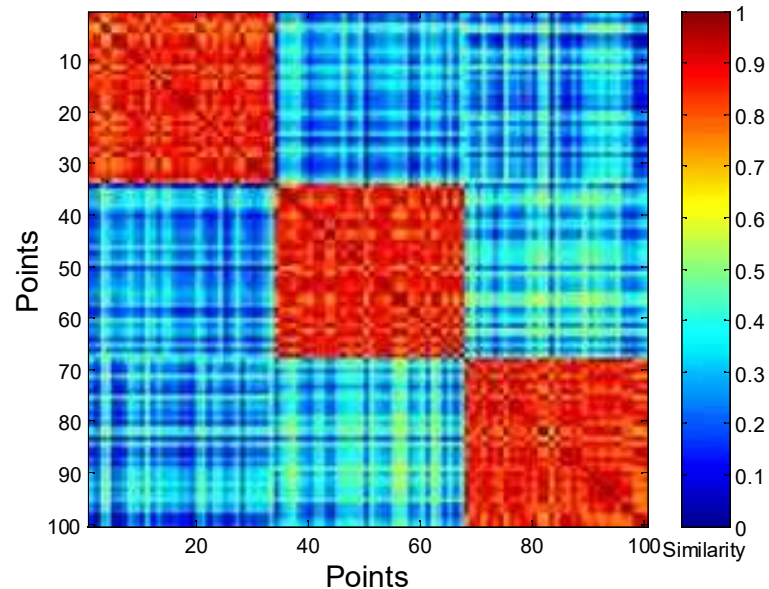
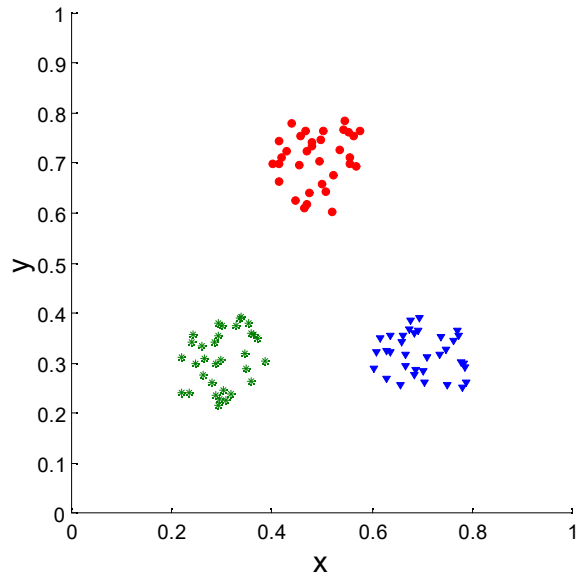


Corr = 0.5810



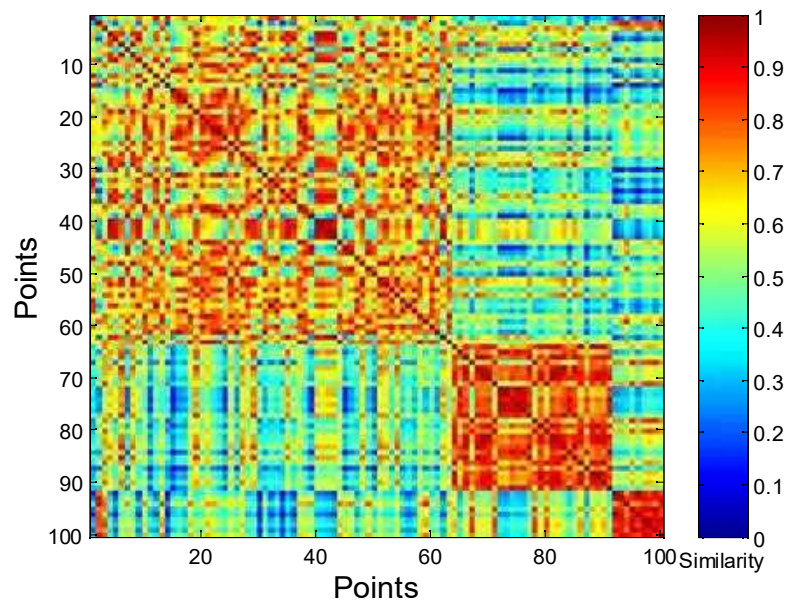
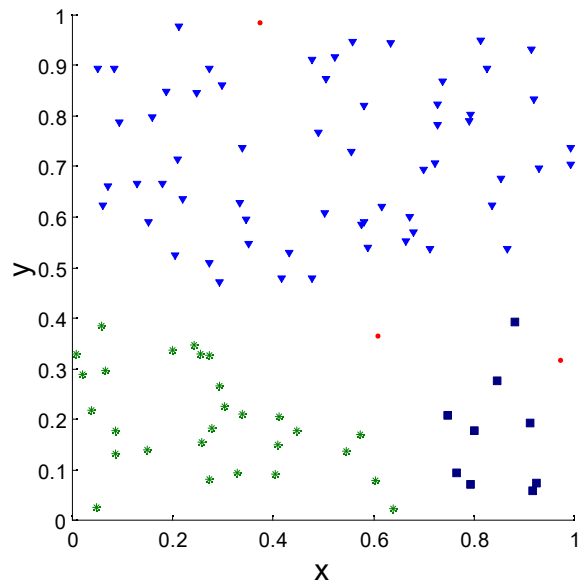
## Judging a Clustering Visually by its Similarity Matrix

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



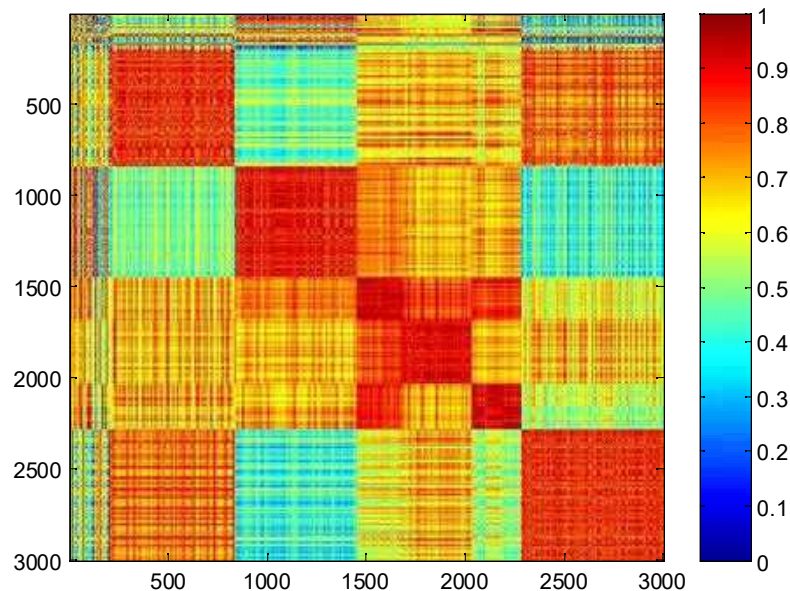
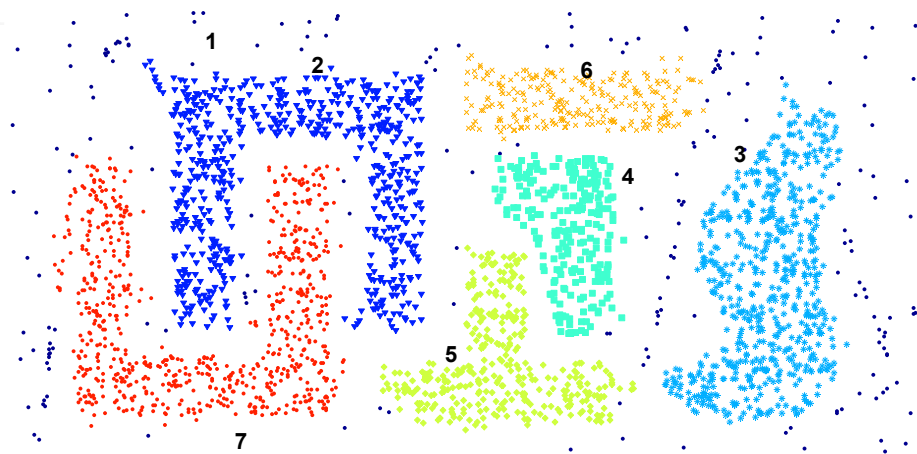
# Judging a Clustering Visually by its Similarity Matrix

- Clusters in random data are not so crisp



**DBSCAN**

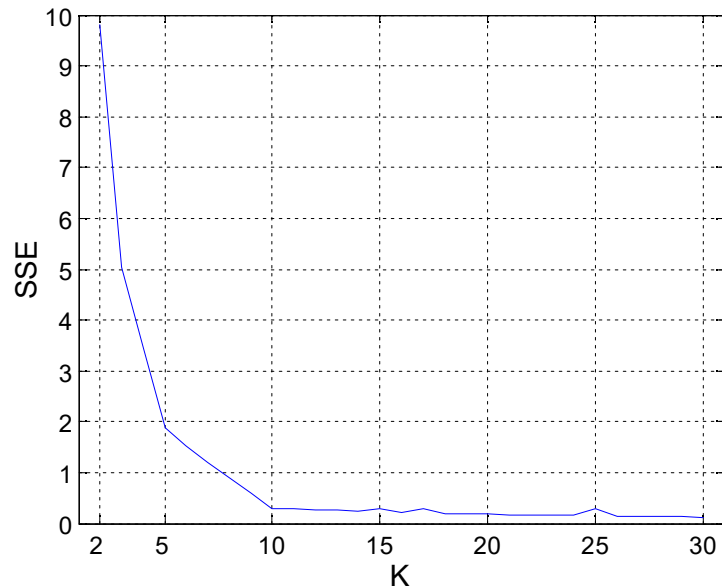
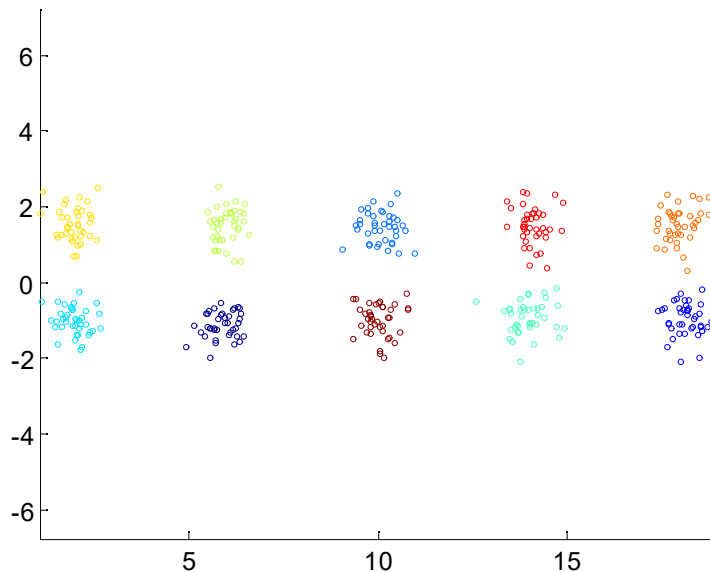
# Judging a Clustering Visually by its Similarity Matrix



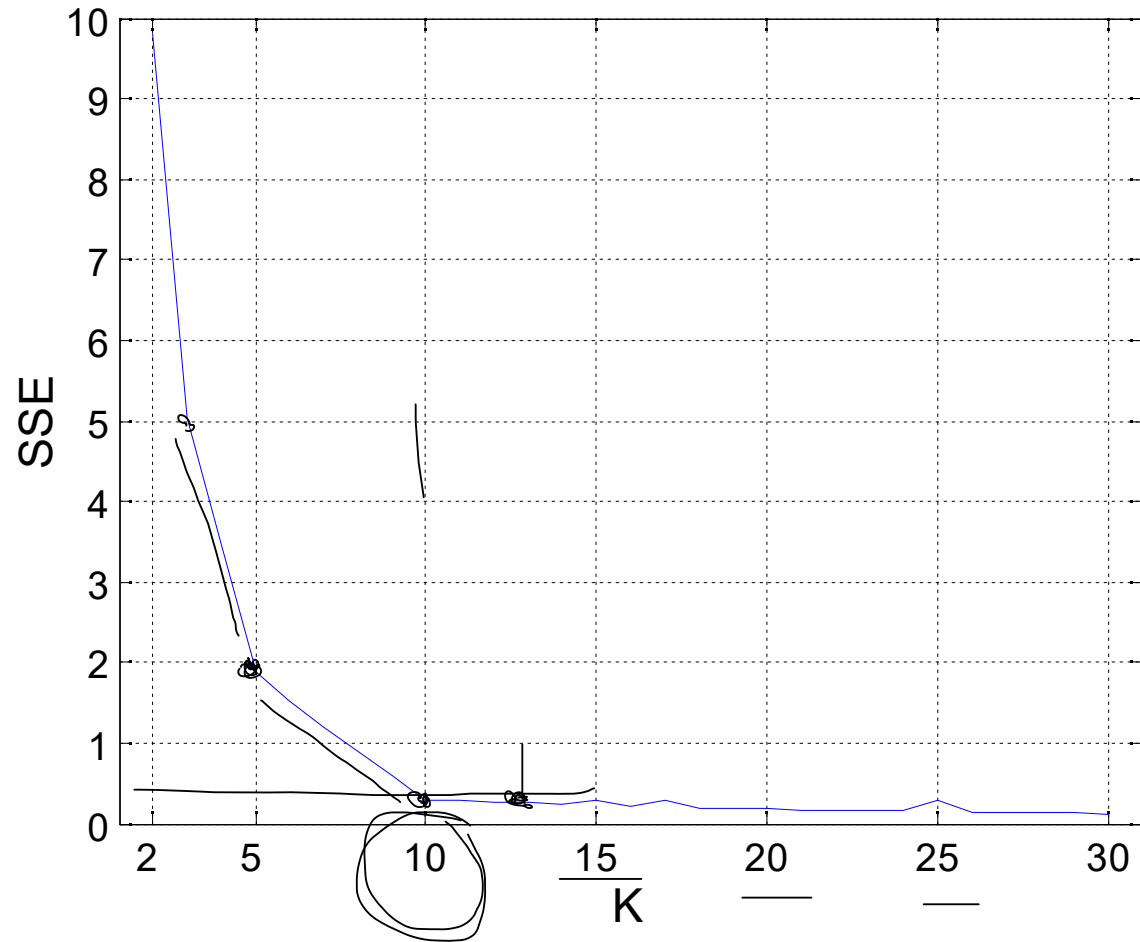
**DBSCAN**

## Determining the Correct Number of Clusters

- SSE is good for comparing two clusterings or two clusters
- SSE can also be used to estimate the number of clusters

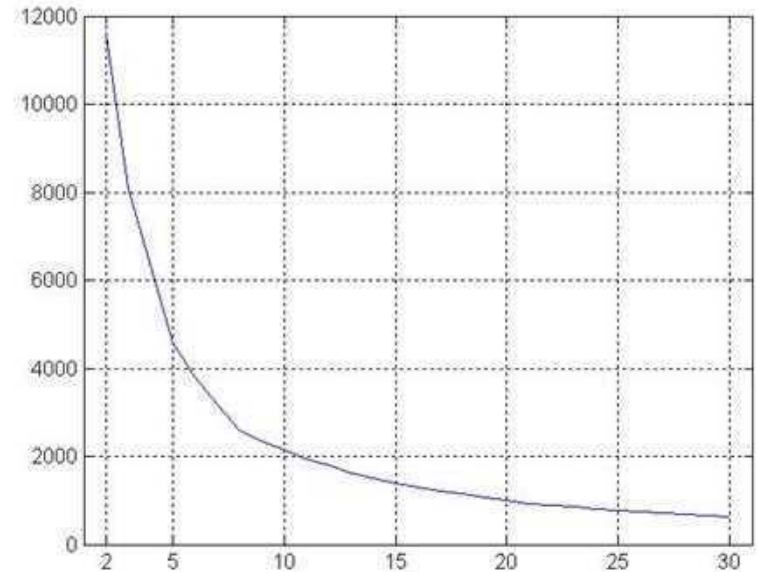
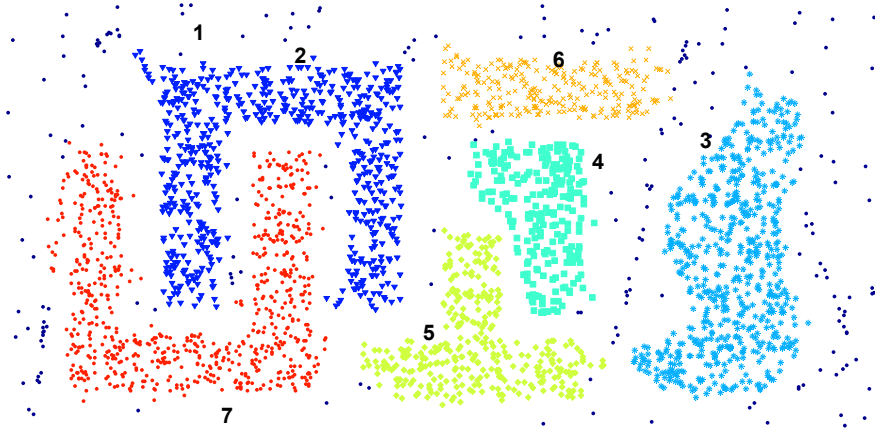


## Determining the Correct Number of Clusters



# Determining the Correct Number of Clusters

- SSE curve for a more complicated data set

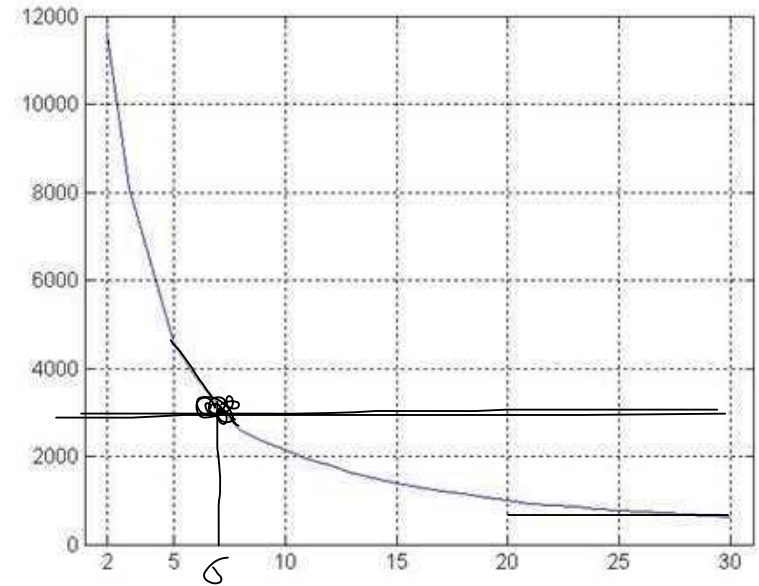
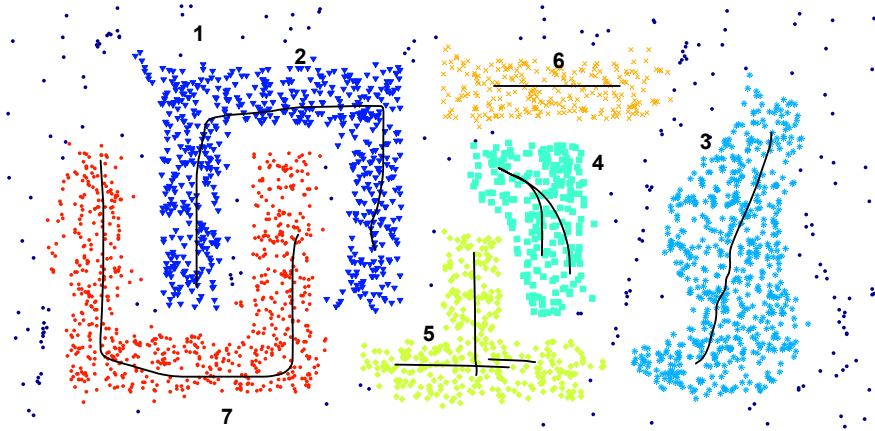
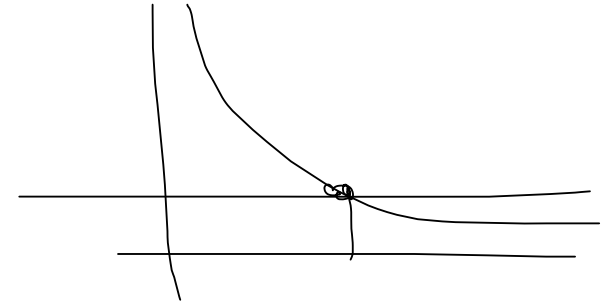


**SSE of clusters found using K-means**



# Determining the Correct Number of Clusters

- SSE curve for a more complicated data set

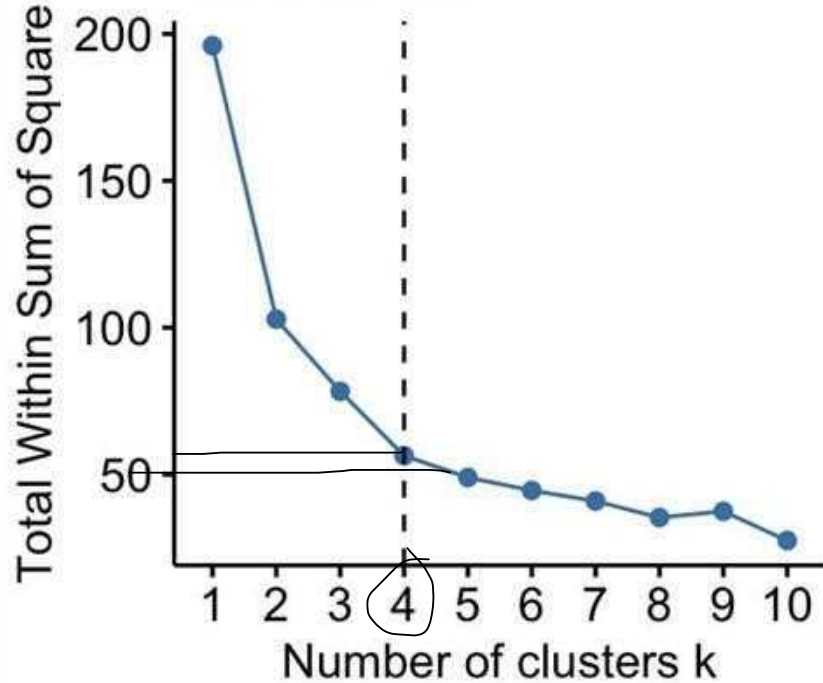


SSE of clusters found using K-means

# Optimal Number of clusters

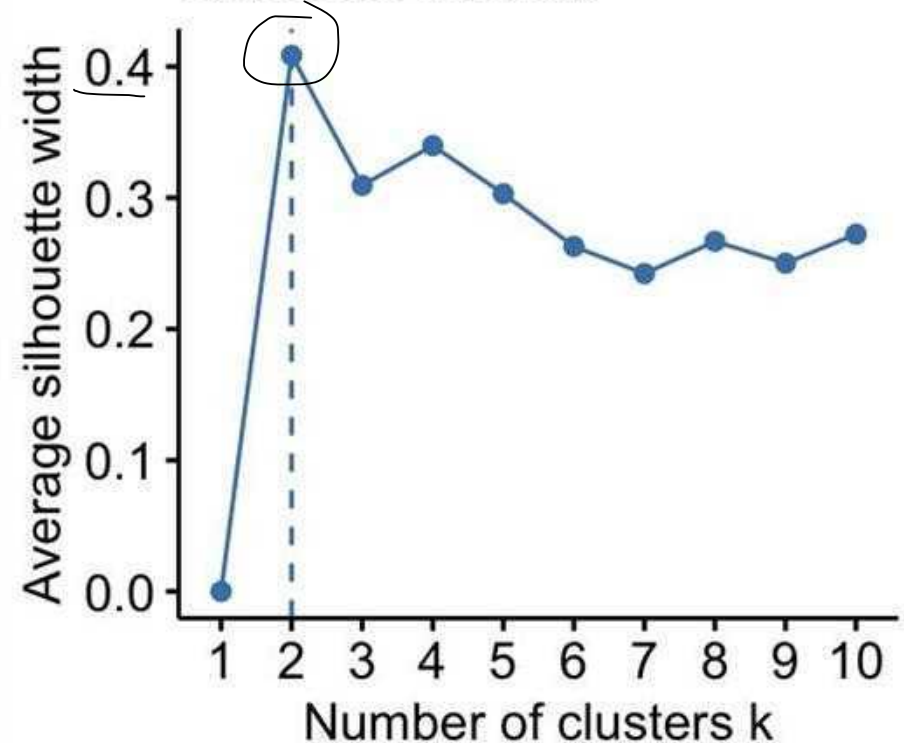
Optimal number of clusters

Elbow method



Optimal number of clusters

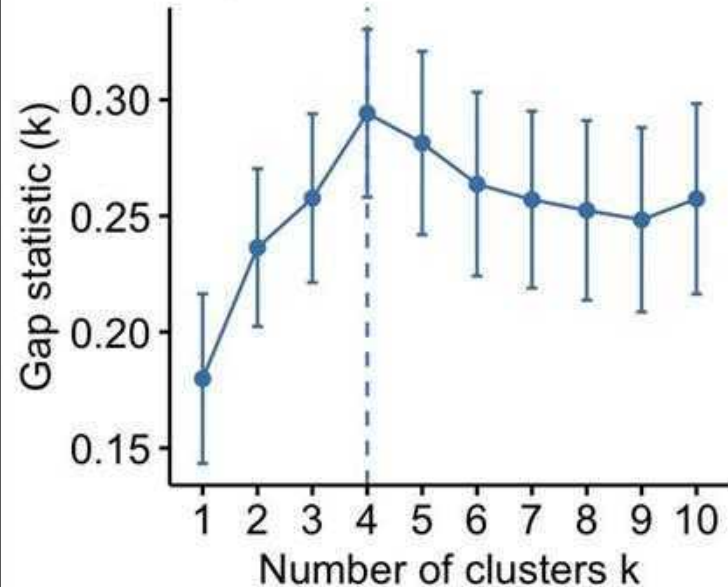
Silhouette method



# Optimal Number of clusters

Optimal number of clusters:

Gap statistic method



- Elbow method: 4 clusters solution suggested
- Silhouette method: 2 clusters solution suggested
- Gap statistic method: 4 clusters solution suggested

# Elbow method

1. Compute clustering algorithm (e.g., k-means clustering) for different values of  $k$ . For instance, by varying  $k$  from 1 to 10 clusters.
2. For each  $k$ , calculate the total within-cluster sum of square (wss) (SSE).
3. Plot the curve of wss according to the number of clusters  $k$ .
4. The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters.

## Final Comment on Cluster Validity

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

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.”

### ***Algorithms for Clustering Data, Jain and Dubes***

- H. Xiong and Z. Li. *Clustering Validation Measures*. In C. C. Aggarwal and C. K. Reddy, editors, *Data Clustering: Algorithms and Applications*, pages 571–605. Chapman & Hall/CRC, 2013.