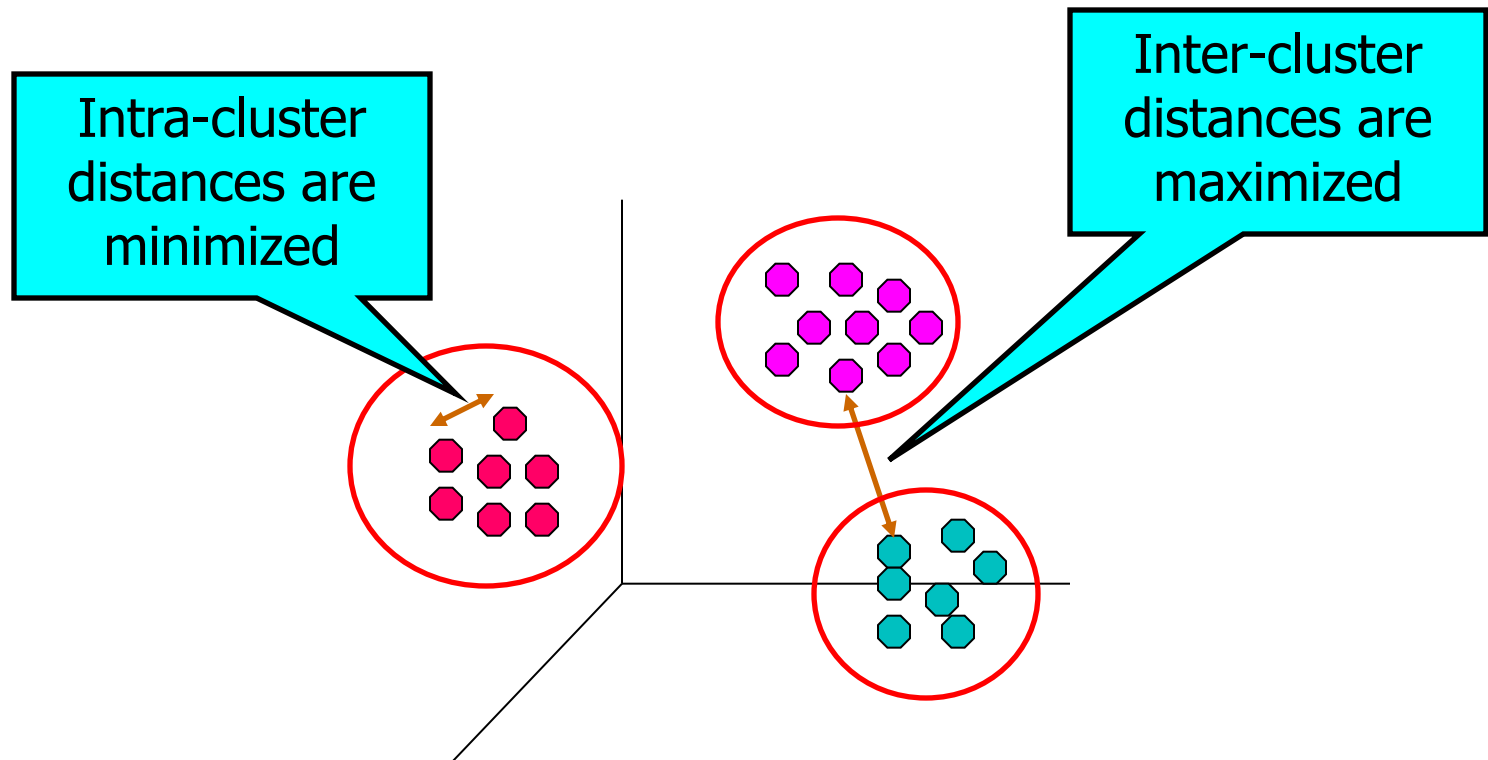


# What is Cluster Analysis?

- 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



# Applications of Cluster Analysis

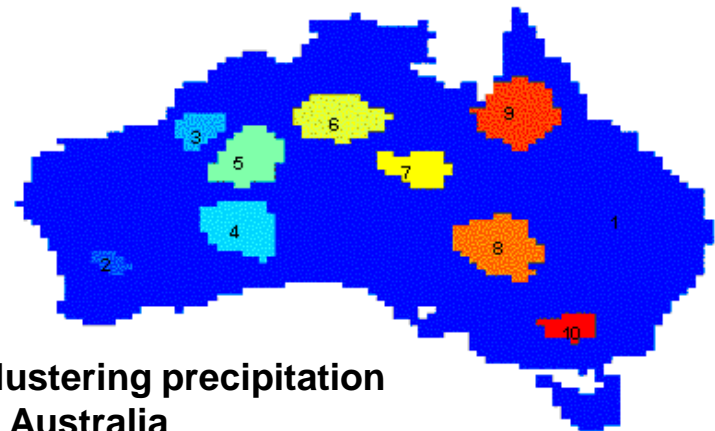
## ● Understanding

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

	<i>Discovered Clusters</i>	<i>Industry Group</i>
<b>1</b>	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
<b>2</b>	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
<b>3</b>	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
<b>4</b>	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

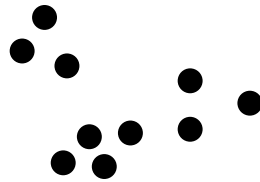
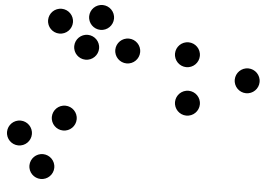
## ● Summarization

- Reduce the size of large data sets

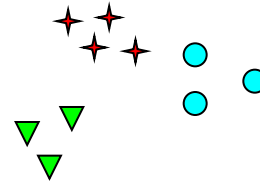


Clustering precipitation  
in Australia

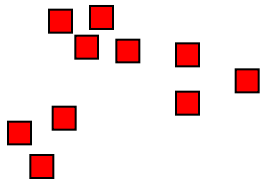
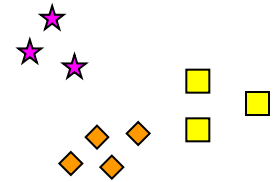
# Notion of a Cluster can be Ambiguous



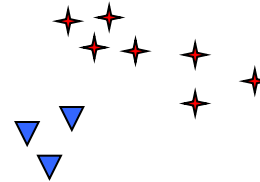
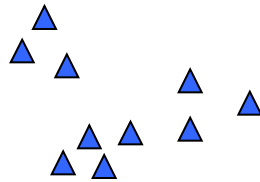
How many clusters?



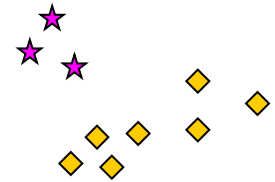
Six Clusters



Two Clusters



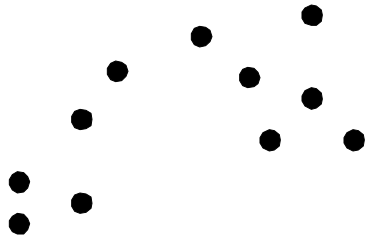
Four Clusters



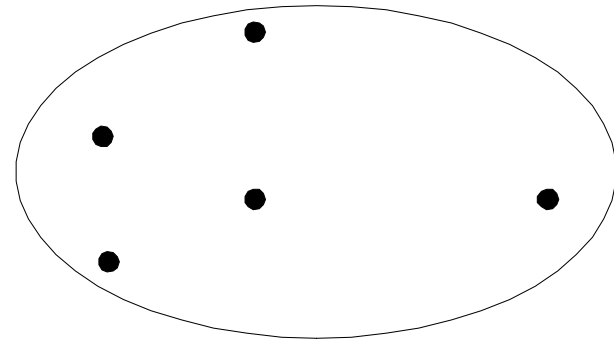
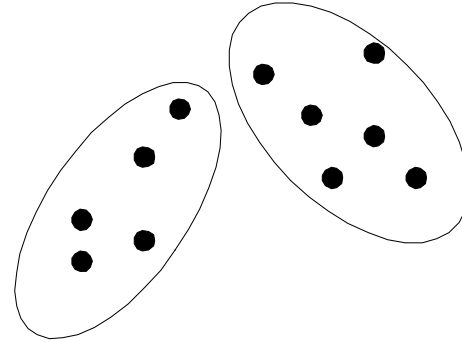
# Types of Clusterings

- A **clustering** gives set of clusters
- Partitional Clustering
  - A division 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

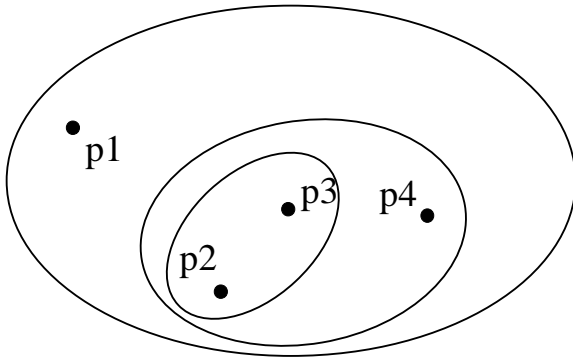


Original Points

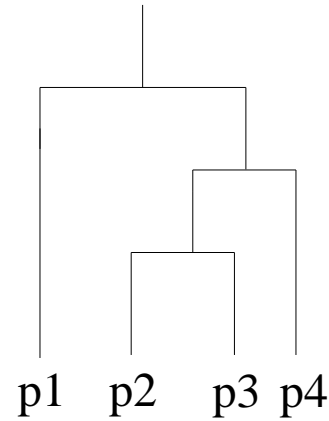


A Partitional Clustering

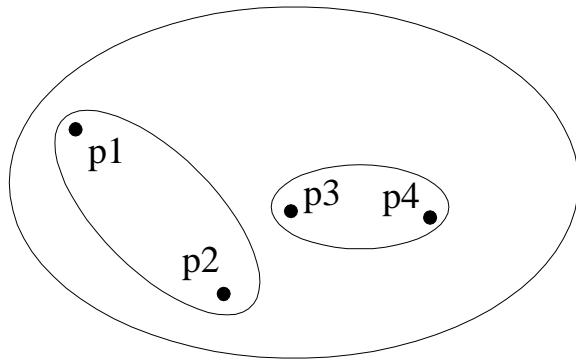
# Hierarchical Clustering



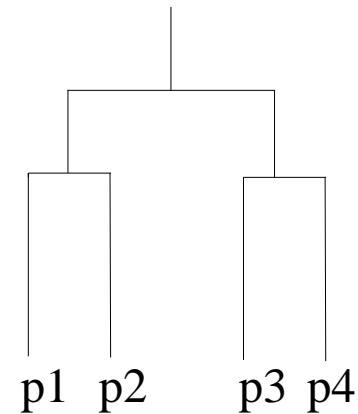
**Traditional Hierarchical Clustering**



**Traditional Dendrogram**



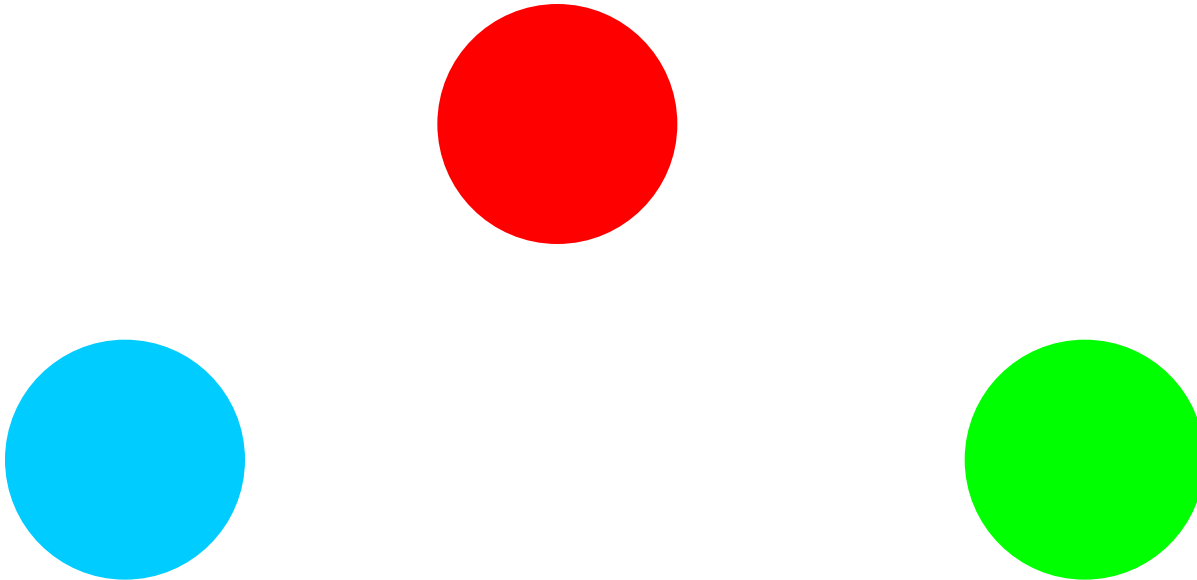
**Non-traditional Hierarchical Clustering**



**Non-traditional Dendrogram**

# Types of Clusters: Well-Separated

- Well-Separated Clusters:
  - A cluster is a set of points such that **any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.**

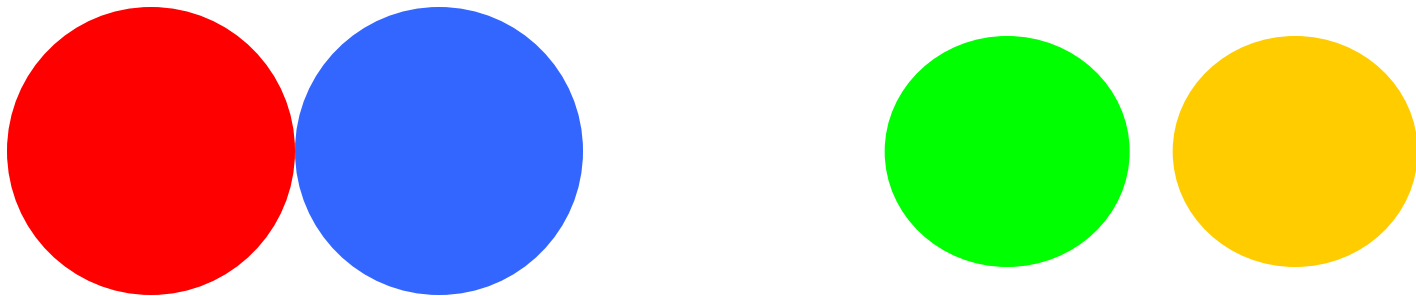


**3 well-separated clusters**

# 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

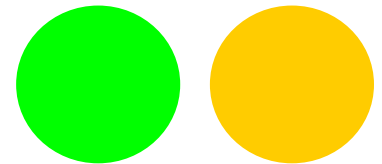
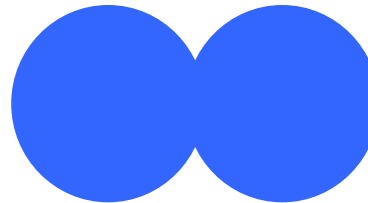
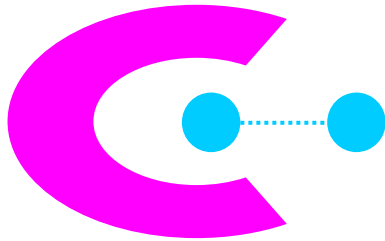
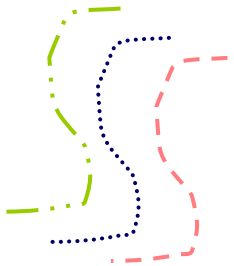


**4 center-based clusters**



# Types of Clusters: Contiguity-Based

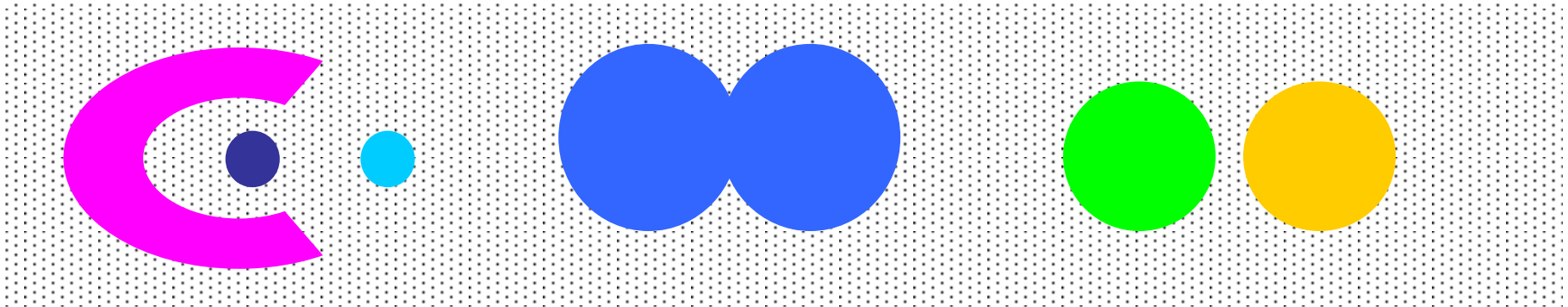
- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



**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.

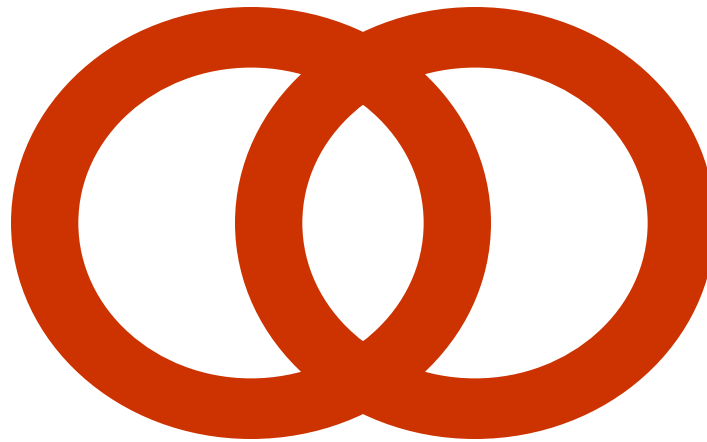


**6 density-based clusters**

# Types of Clusters: Conceptual Clusters

- Shared Property or Conceptual Clusters
  - Finds clusters that share some common property or represent a particular concept.

.



**2 Overlapping Circles**

# Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- Density-based clustering

# K-means Clustering

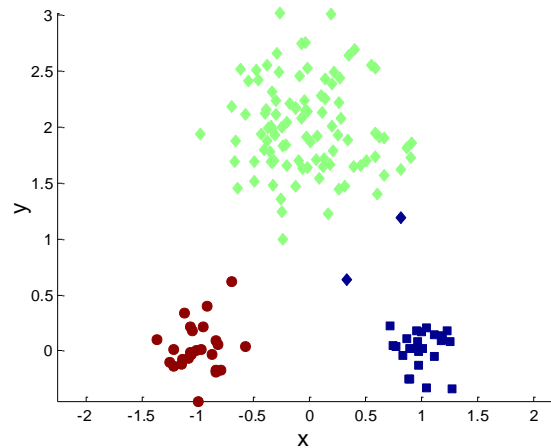
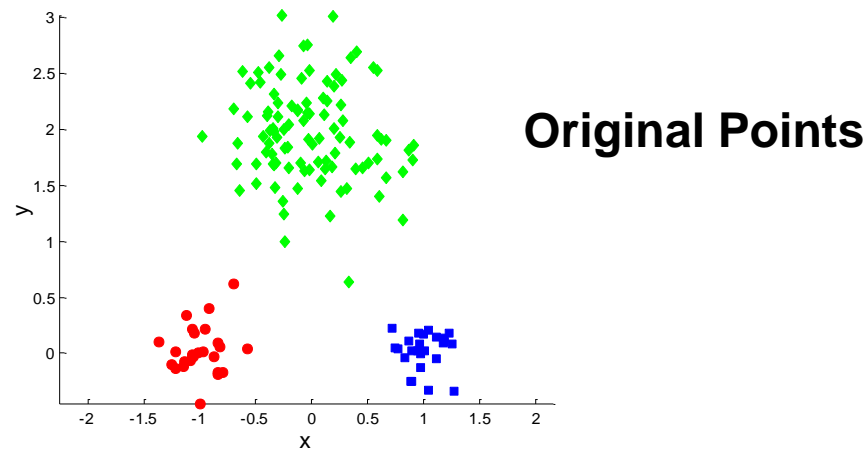
- Partitional clustering approach
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters,  $K$ , must be specified
- The basic algorithm is very simple

- 
- 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
-

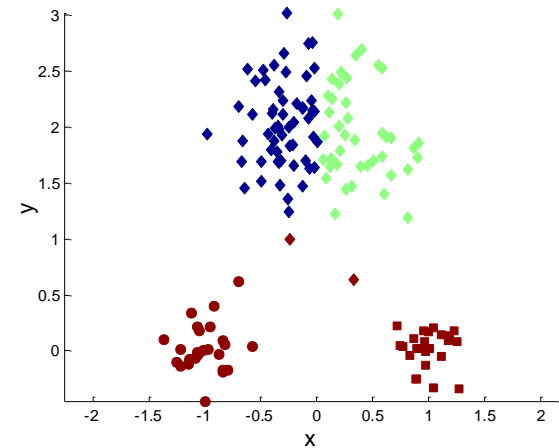
# K-means Clustering – Details

- 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



**Optimal Clustering**



**Sub-optimal Clustering**

# Evaluating K-means Clusters

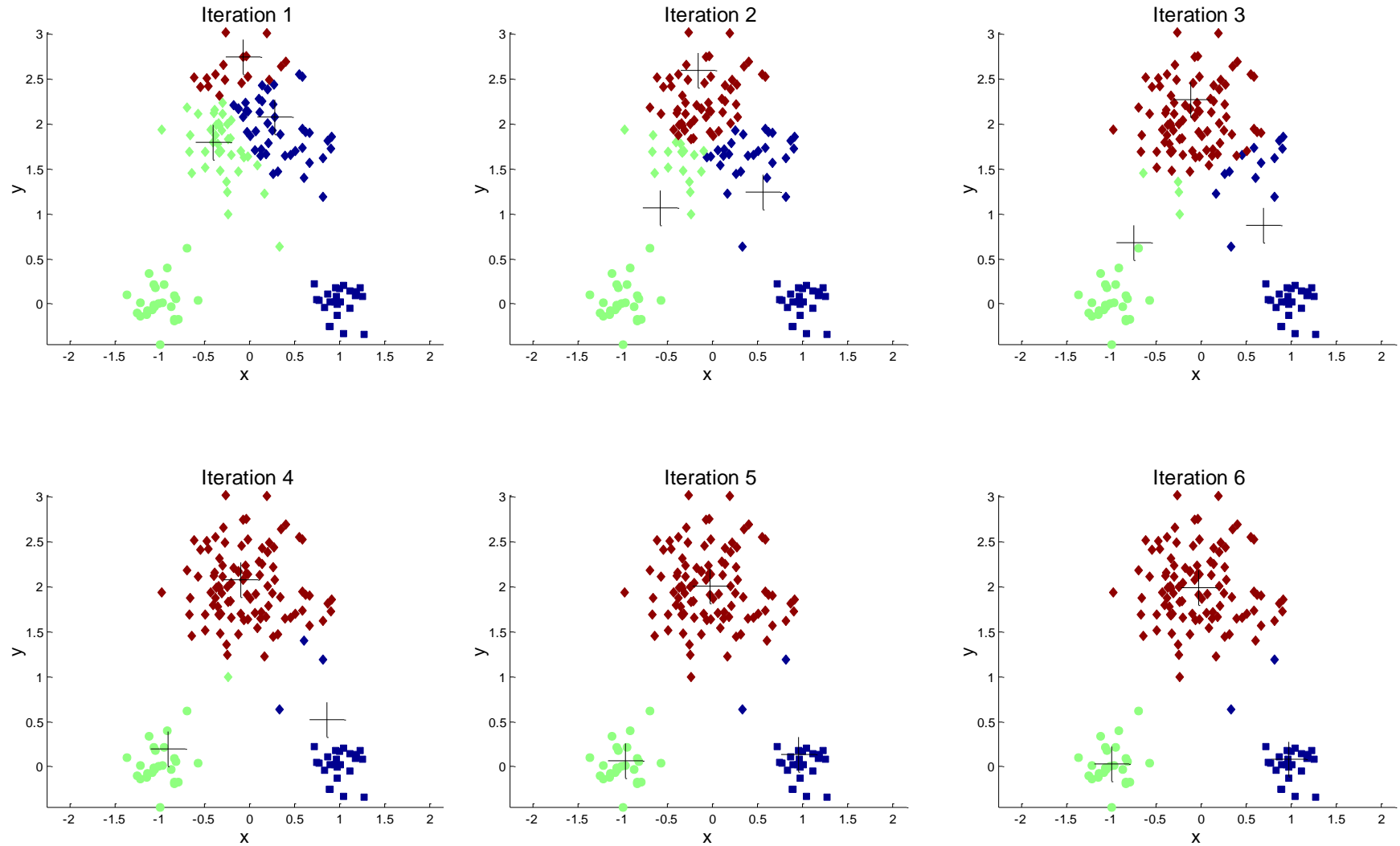
- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

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

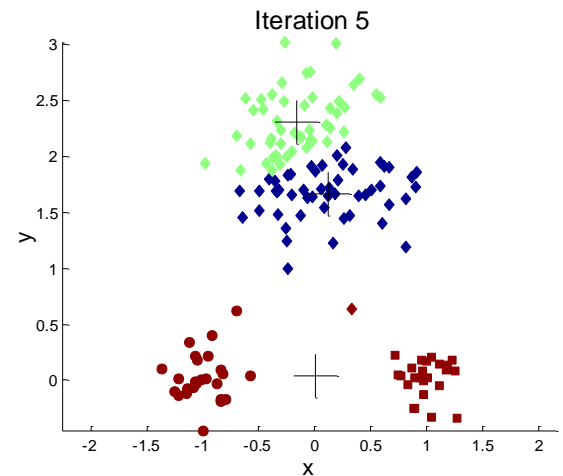
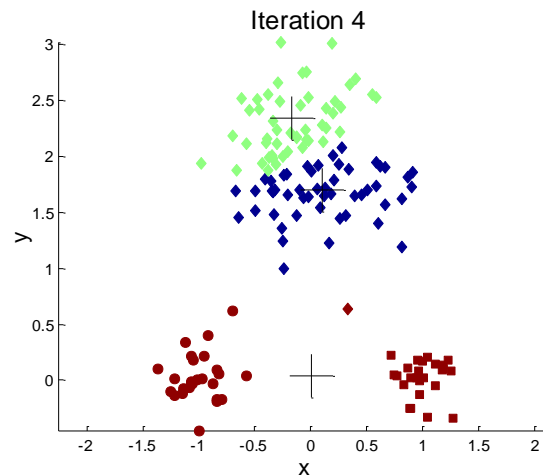
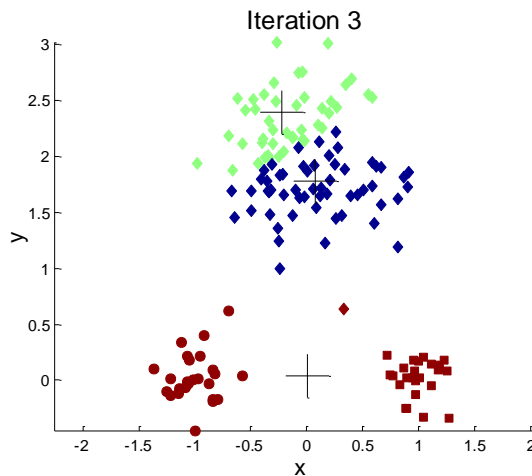
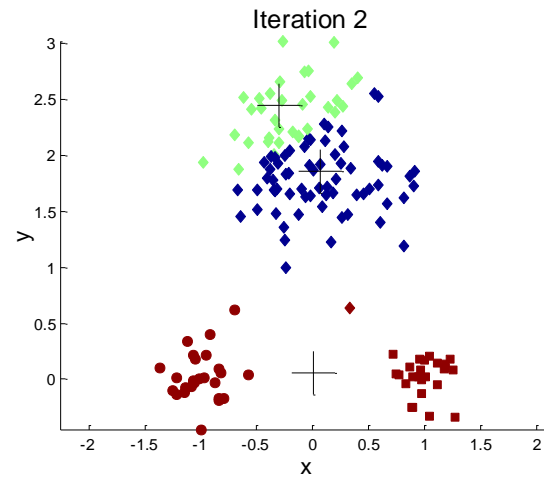
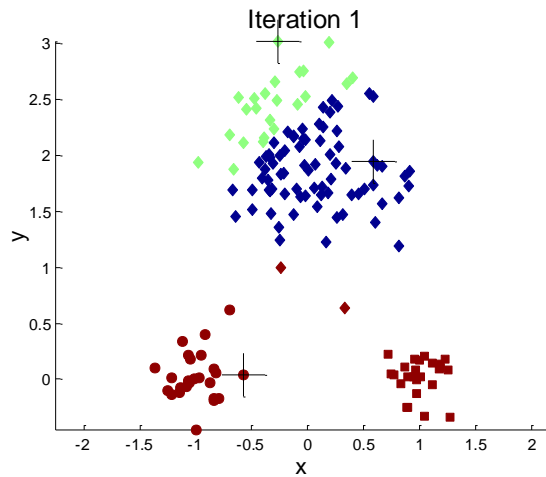
- $x$  is a data point in cluster  $C_i$  and  $m_i$  is the representative point for cluster  $C_i$ 
  - ◆ can show that  $m_i$  corresponds to the center (mean) of the cluster
- Given two clusters, we can choose the one with the smallest error
- One easy way to reduce SSE is to increase  $K$ , the number of clusters
  - ◆ A good clustering with smaller  $K$  can have a lower SSE than a poor clustering with higher  $K$



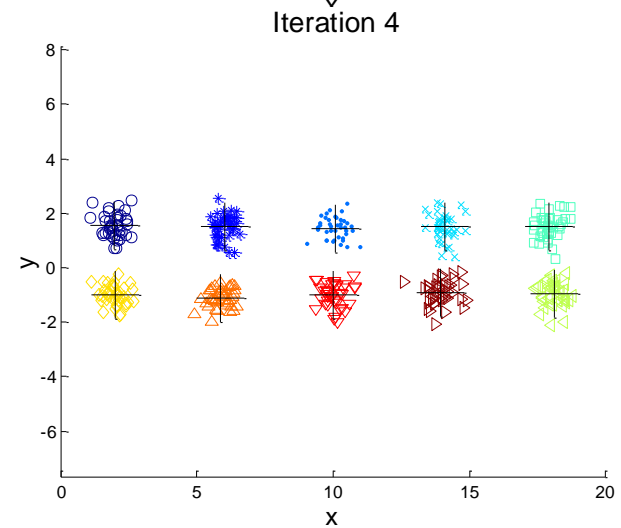
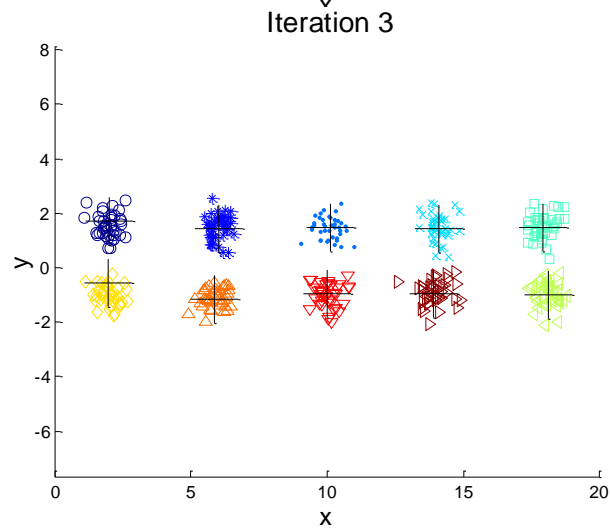
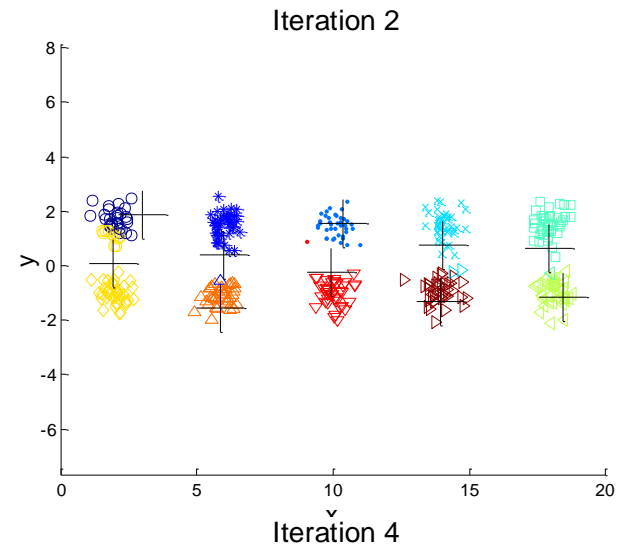
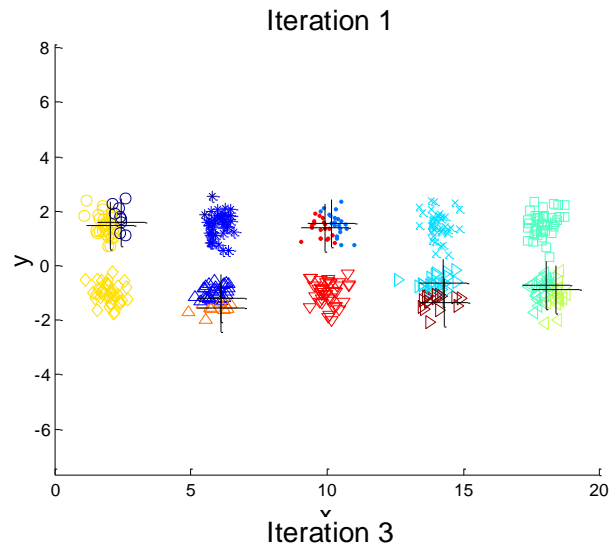
# Importance of Choosing Initial Centroids



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

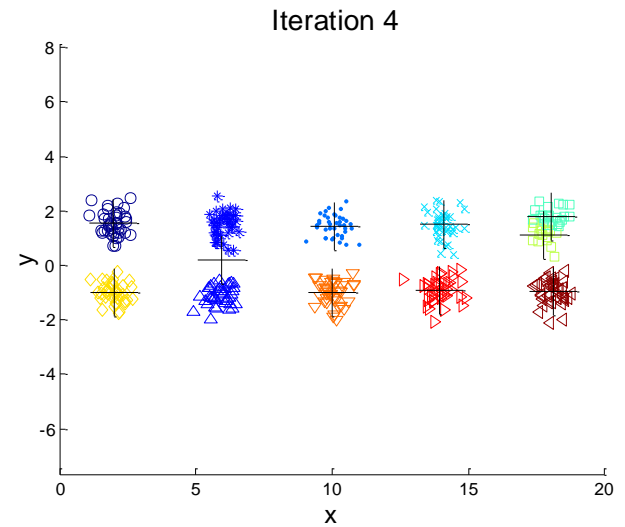
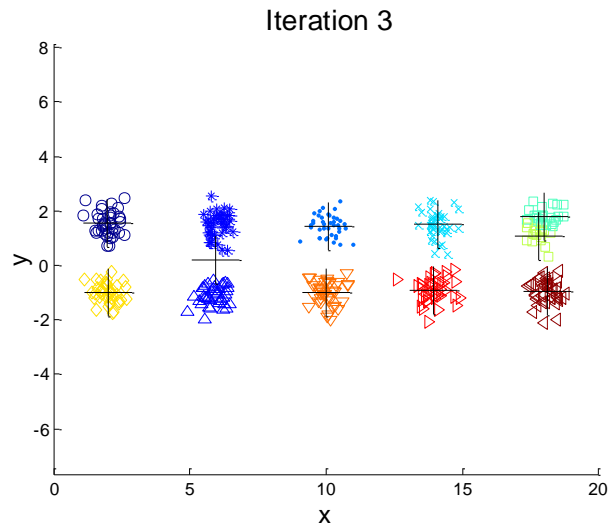
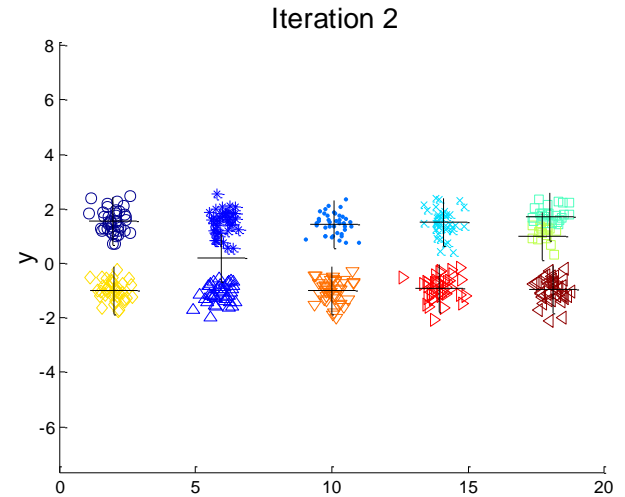
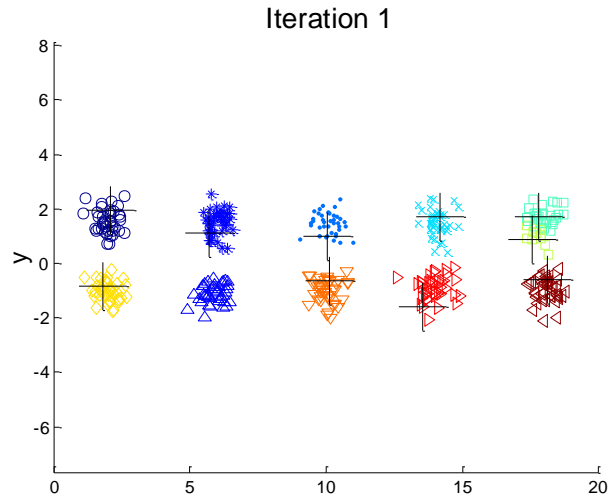


# 10 Clusters Example



**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
- Sample and use hierarchical clustering to determine initial centroids
- Select more than  $k$  initial centroids and then select among these initial centroids
  - Select most widely separated
- Postprocessing
- Bisecting K-means
  - Not as susceptible to initialization issues

# Updating Centers Incrementally

- In the basic K-means algorithm, centroids are updated after all points are assigned to a centroid
- An alternative is to update the centroids after each assignment (incremental approach)
  - Each assignment updates zero or two centroids
  - More expensive
  - Introduces an order dependency
  - Never get an empty cluster
  - Can use “weights” to change the impact

# Pre-processing and Post-processing

- Pre-processing
  - Normalize the data
  - Eliminate outliers
- Post-processing
  - Eliminate small clusters that may represent outliers
  - Split 'loose' clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are 'close' and that have relatively low SSE
  - Can use these steps during the clustering process

# Bisecting K-means

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

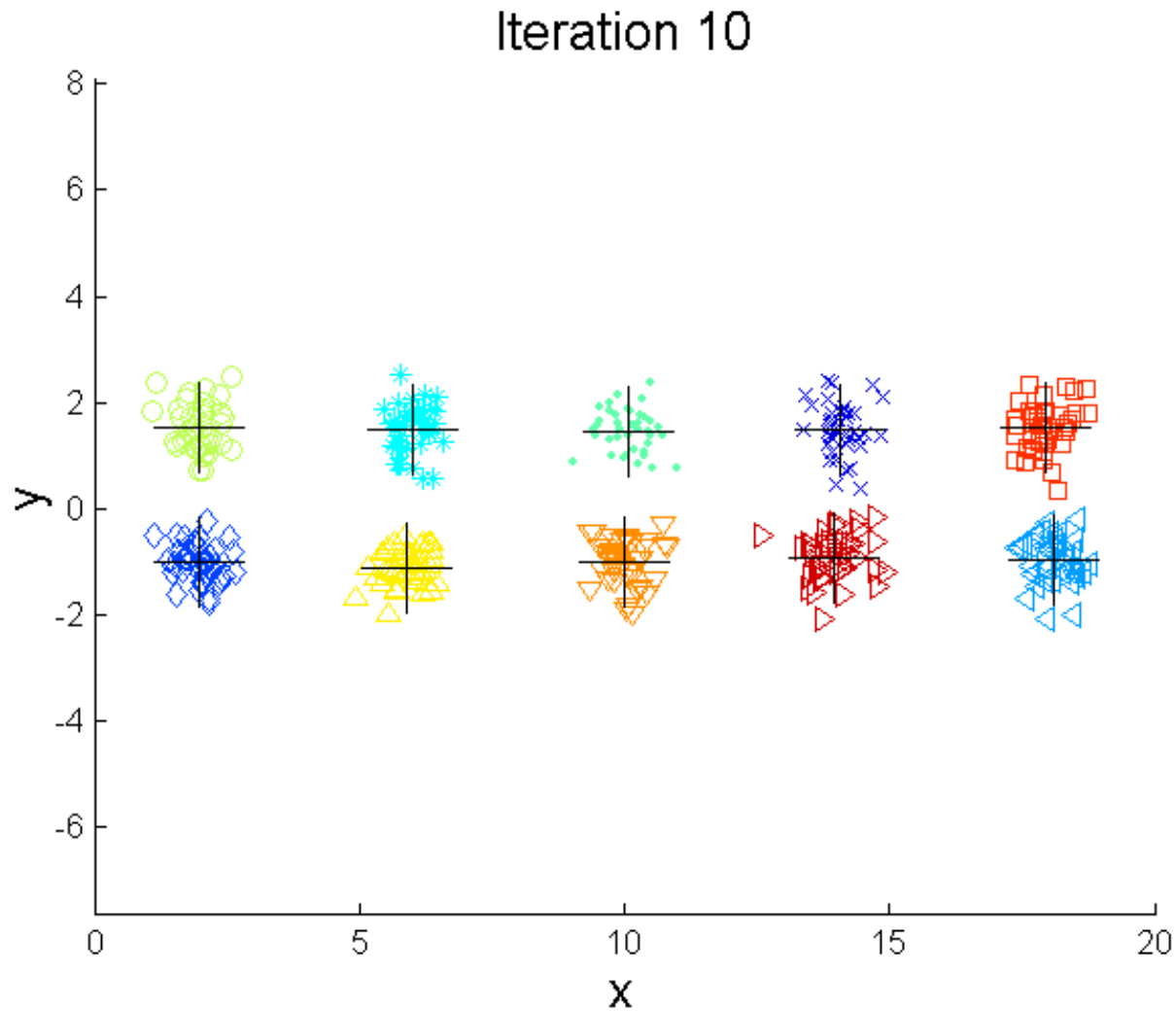
---

```
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
```

---



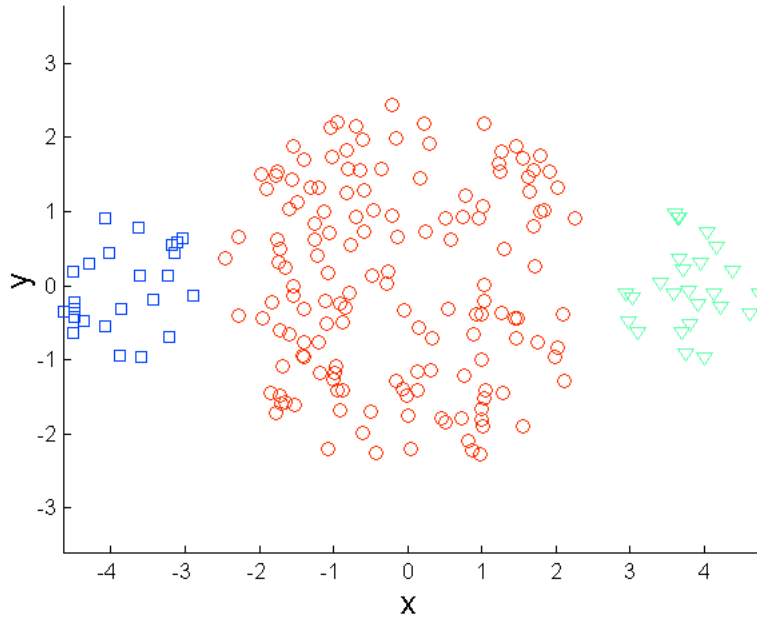
# Bisecting K-means Example



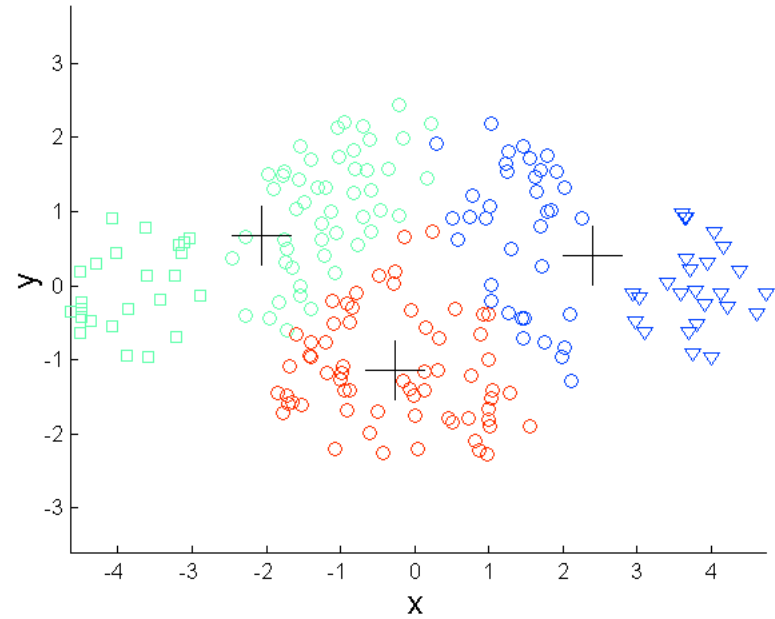
# Limitations of K-means

- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes
- K-means has problems when the data contains outliers.

# Limitations of K-means: Differing Sizes

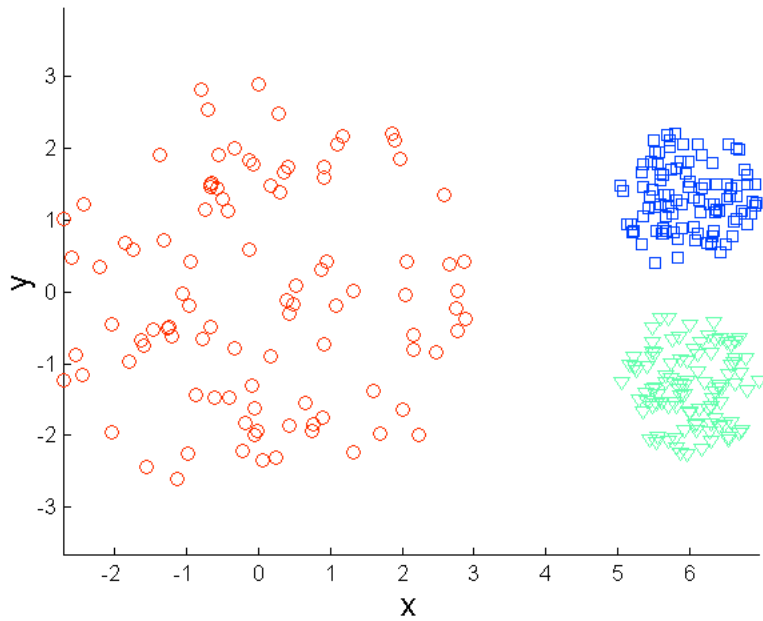


**Original Points**

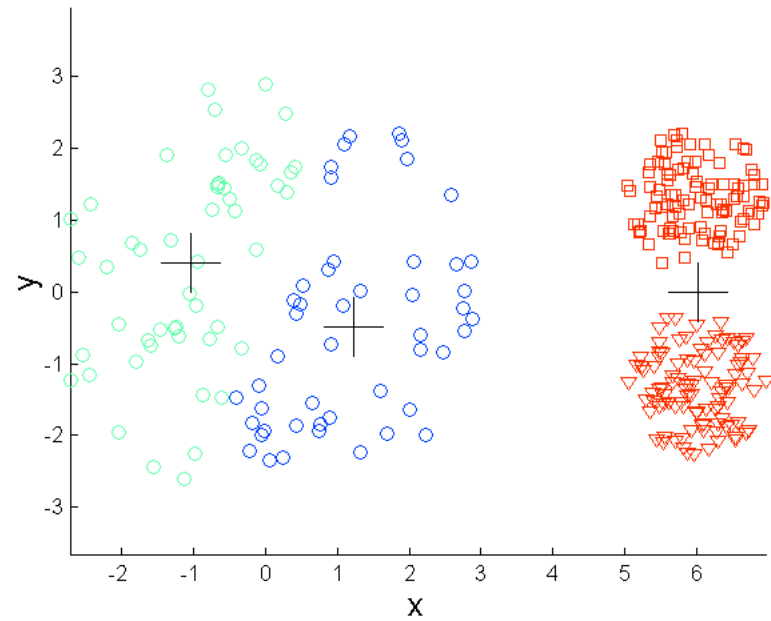


**K-means (3 Clusters)**

# Limitations of K-means: Differing Density

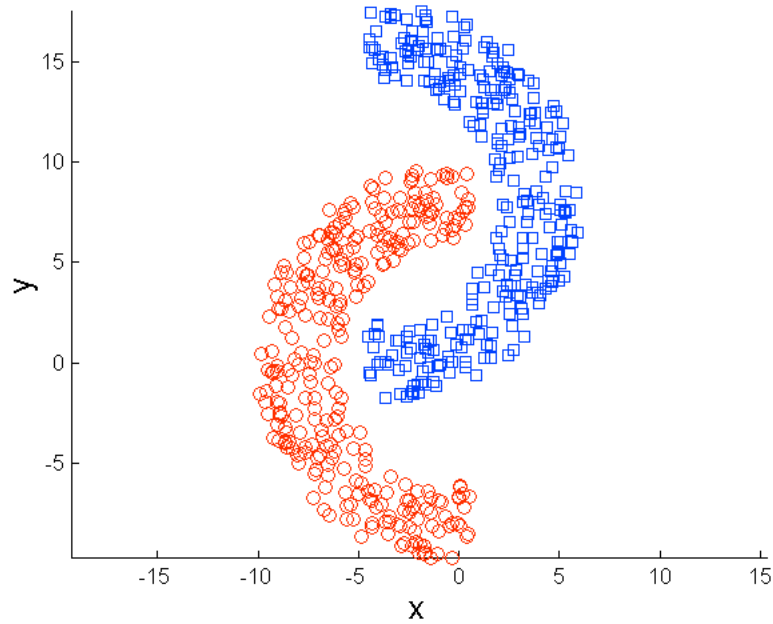


**Original Points**

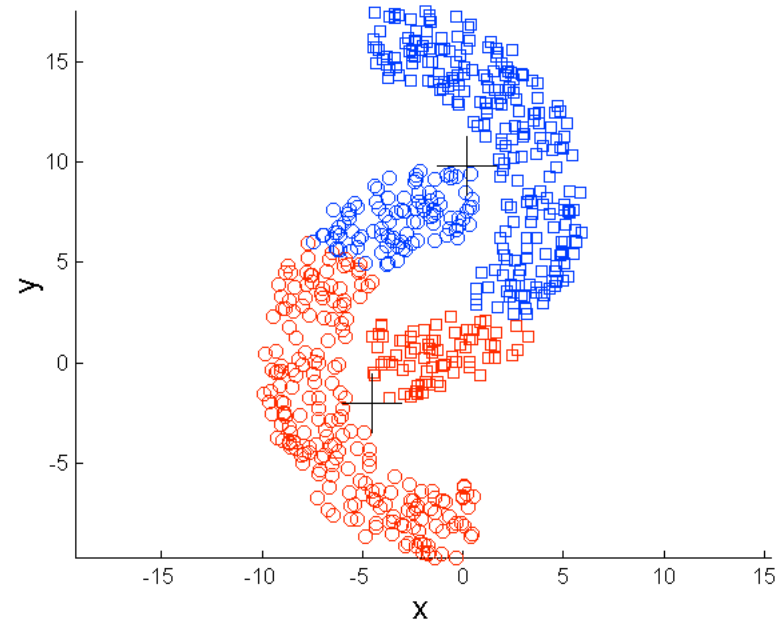


**K-means (3 Clusters)**

# Limitations of K-means: Non-globular Shapes

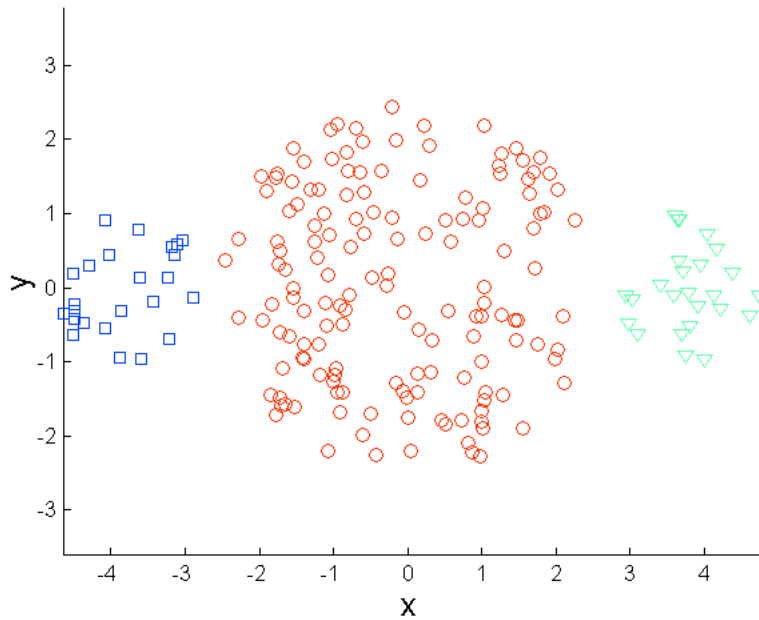


**Original Points**

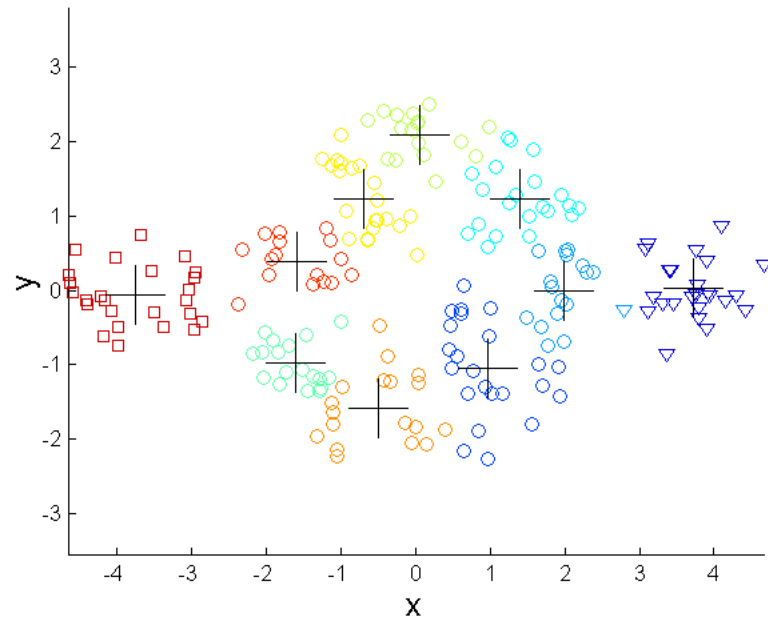


**K-means (2 Clusters)**

# Overcoming K-means Limitations



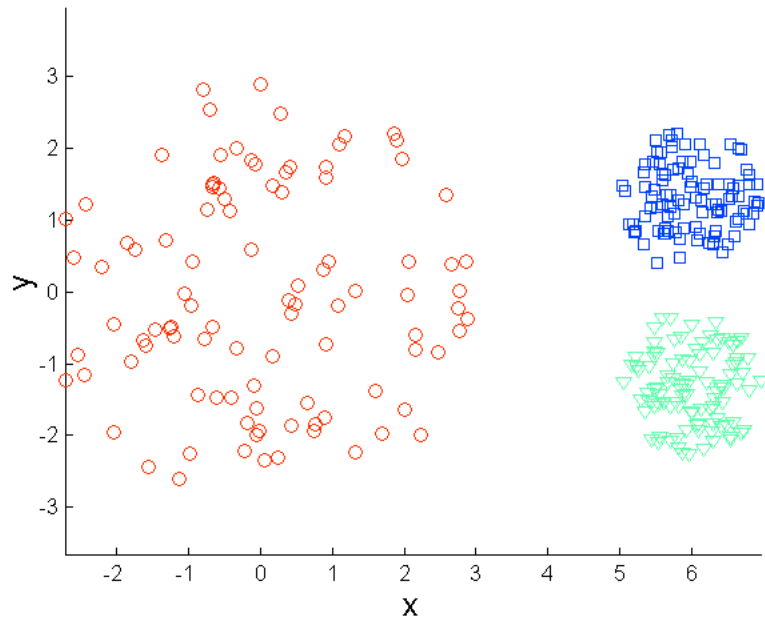
**Original Points**



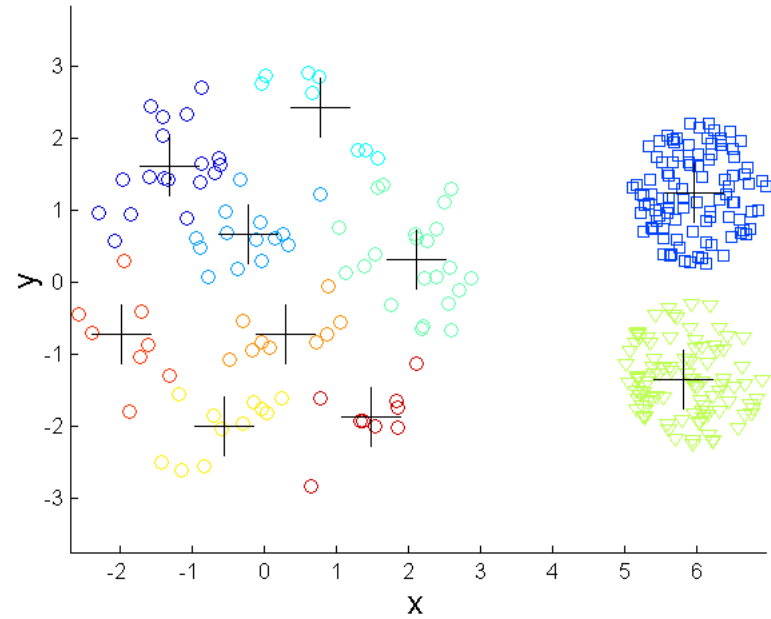
**K-means Clusters**

One solution is to use many clusters.  
Find parts of clusters, but need to put together.

# Overcoming K-means Limitations

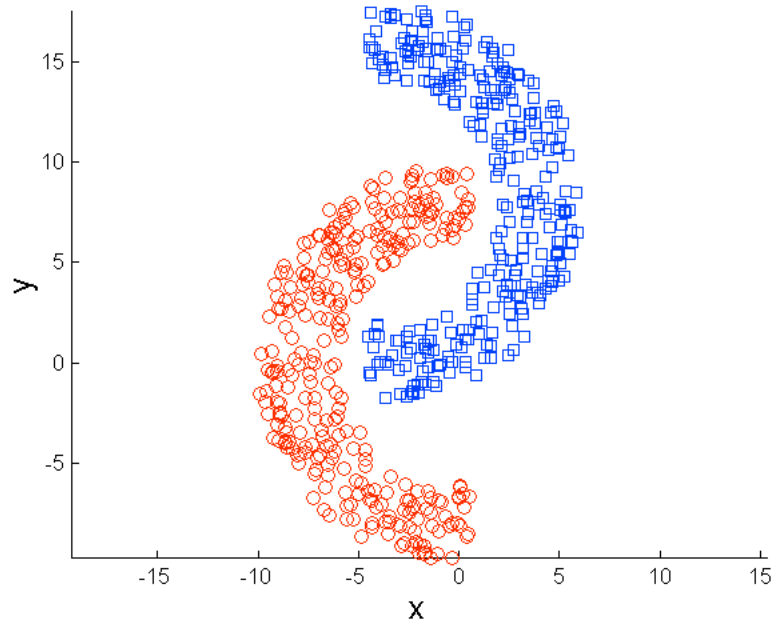


**Original Points**

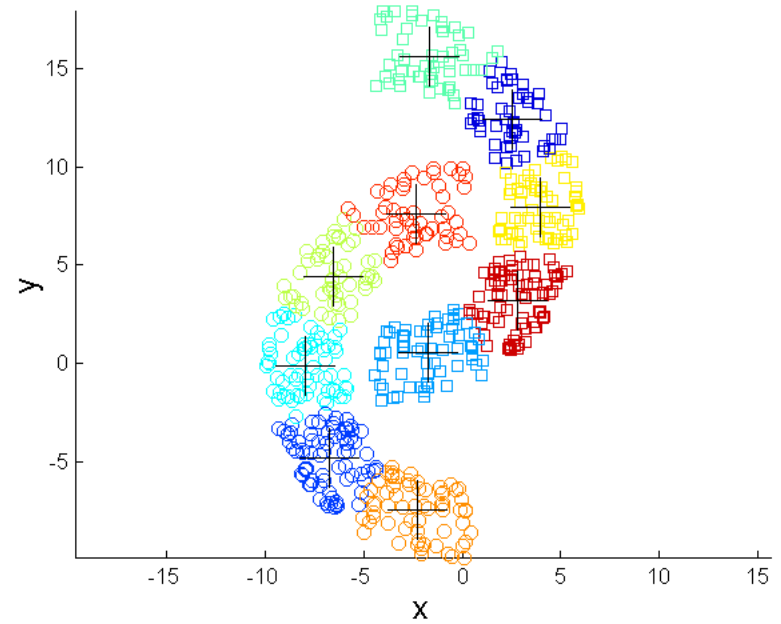


**K-means Clusters**

# Overcoming K-means Limitations



**Original Points**



**K-means Clusters**



# Hierarchical Clustering

Clustering comes in a form of a tree – *dendrogram* visualizing how data contribute to individual clusters

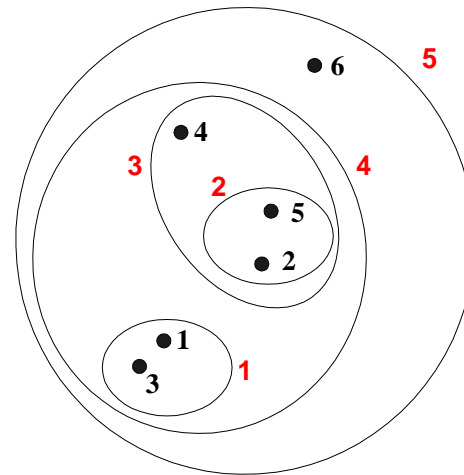
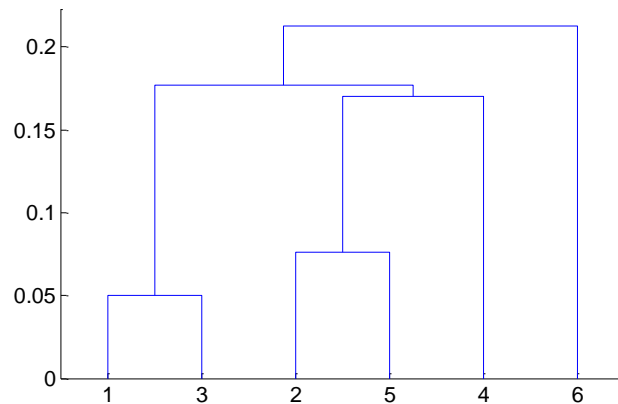
Dendrogram is a binary tree

Clustering is realized in a successive manner through:

- successive splits, or *Top-down* (divisive approach)
- successive aggregations or *Bottom – up* (agglomerative approach)

# 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



# Strengths of Hierarchical Clustering

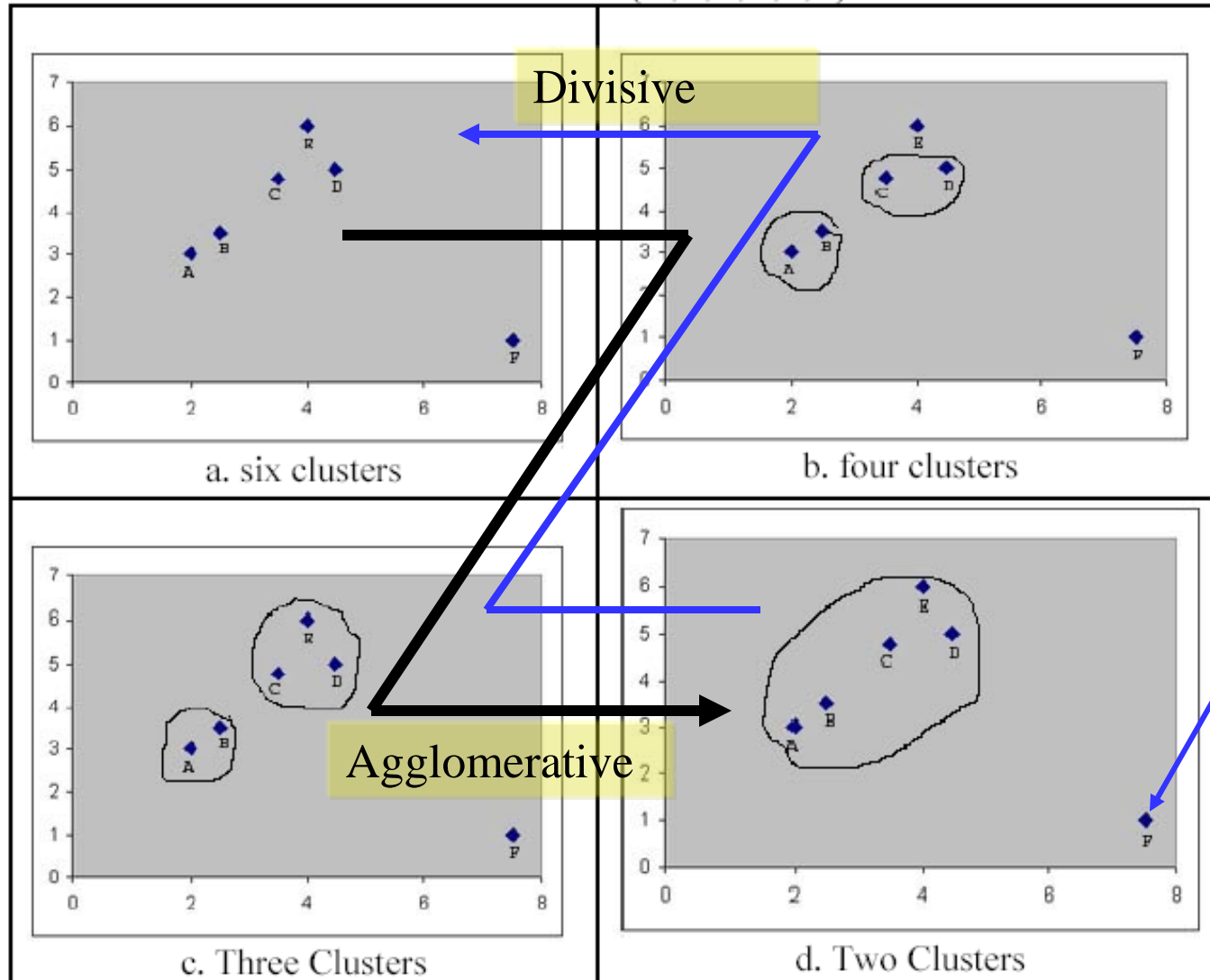
- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by ‘cutting’ the dendrogram at the proper level
- They may correspond to **meaningful taxonomies**
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

# 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 a point (or there are  $k$  clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

## Example of Hierarchical Clustering

Consider we need to cluster six elements {A,B,C,D,E,F}.

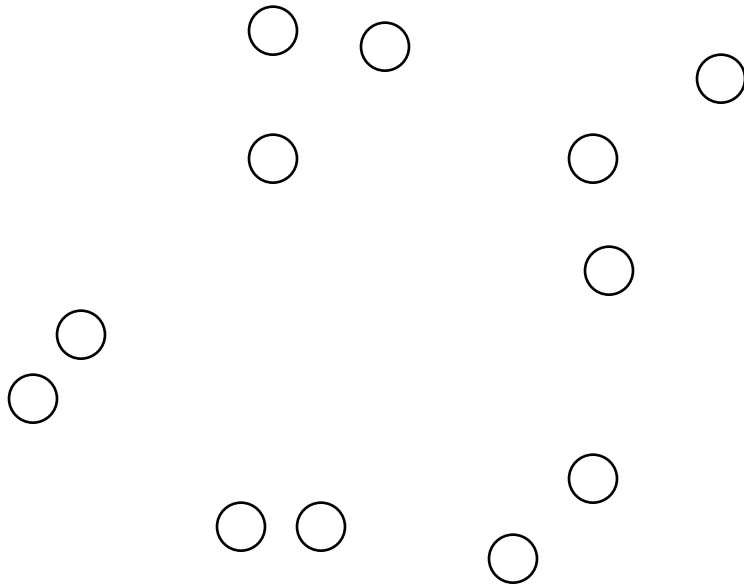


# Agglomerative Clustering Algorithm

- More 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



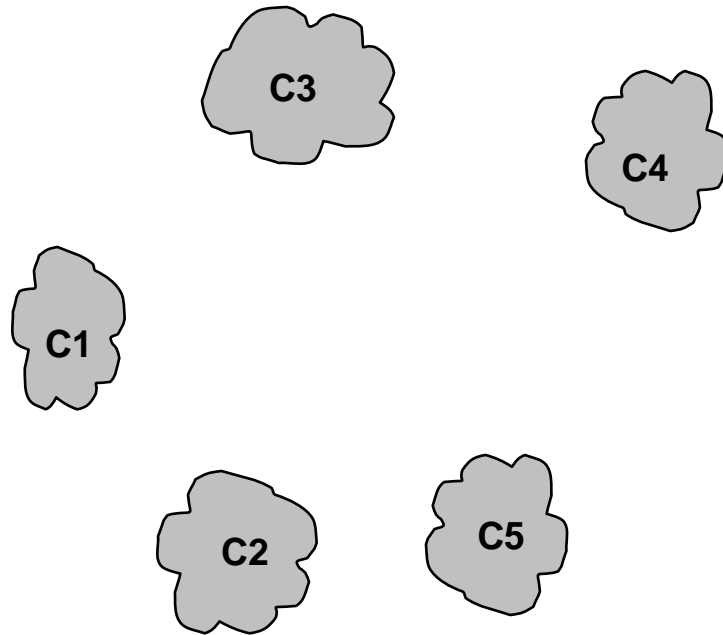
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

**Proximity Matrix**



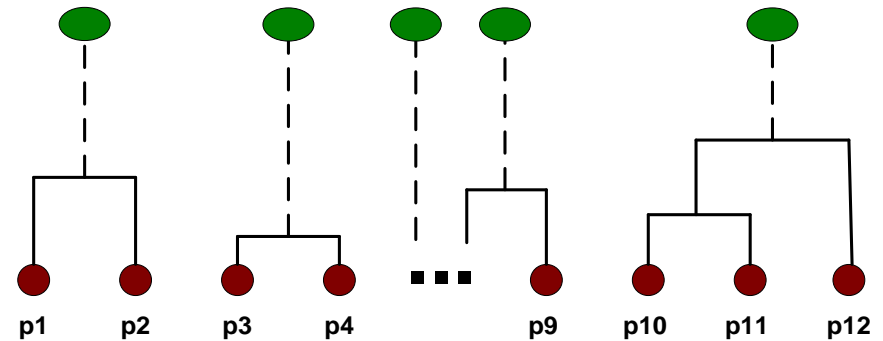
# Intermediate Situation

- After some merging steps, we have some clusters



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

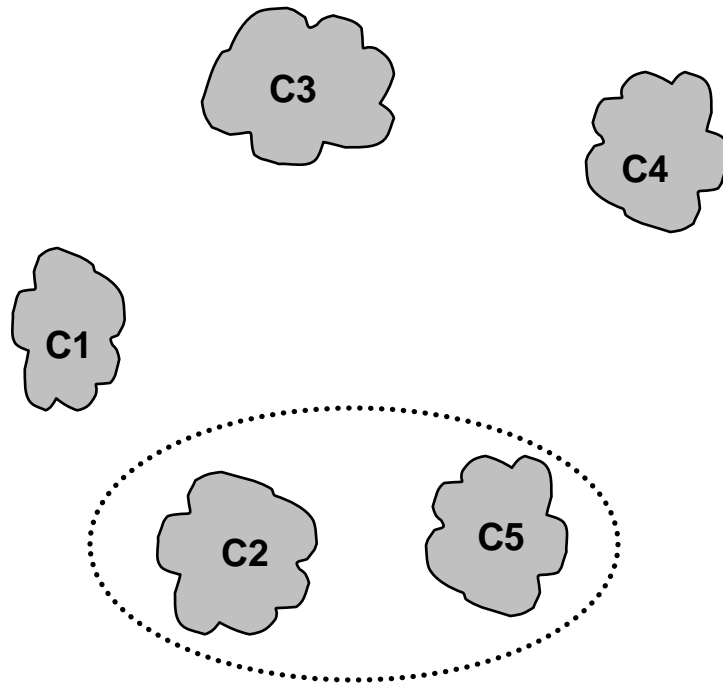
**Proximity Matrix**





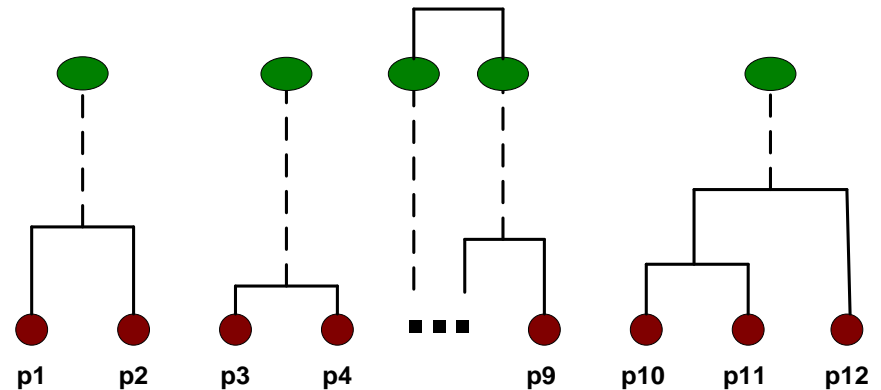
# Intermediate Situation

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



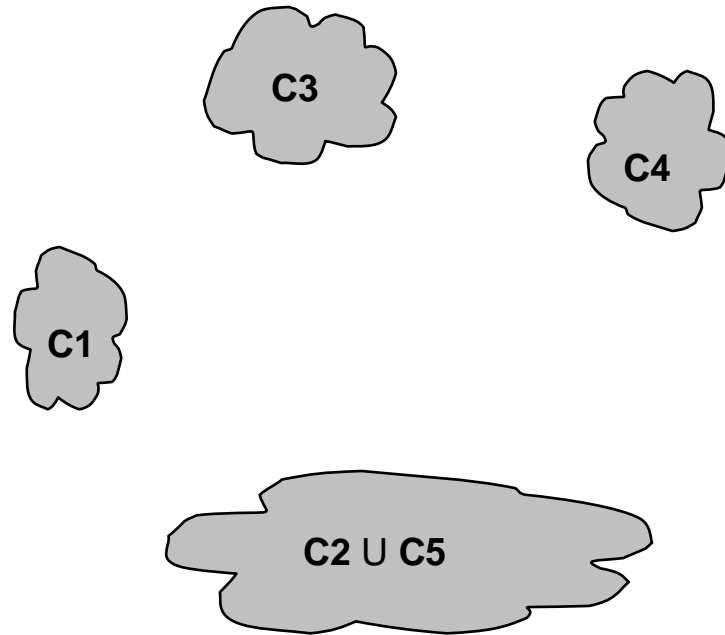
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



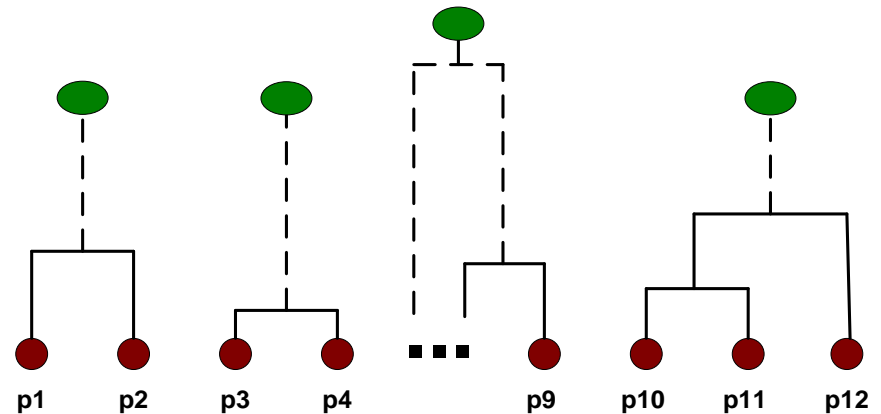
# After Merging

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

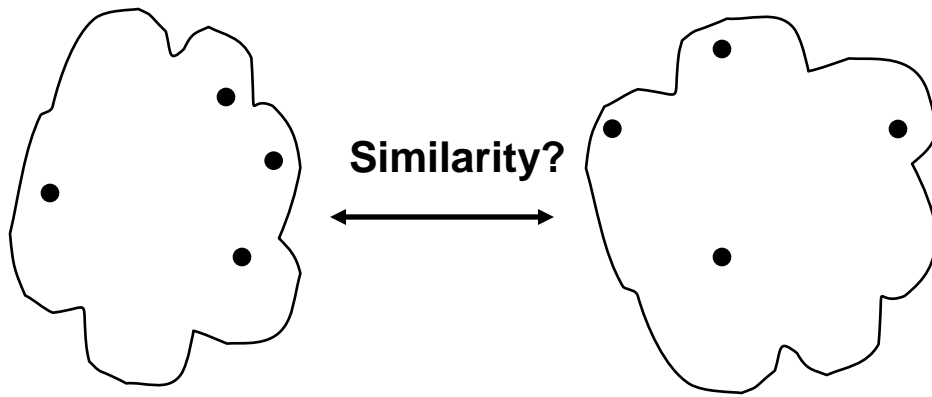


		$C2 \cup C5$			
		C1	C5	C3	C4
$C2 \cup C5$	C1		?		
	C5	?	?	?	?
	C3		?		
	C4		?		

Proximity Matrix



# How to Define Inter-Cluster Similarity

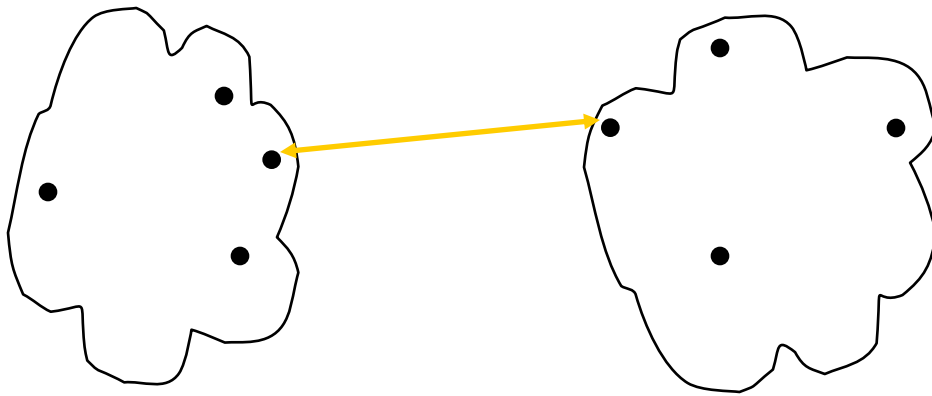


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

**Proximity Matrix**

# How to Define Inter-Cluster Similarity

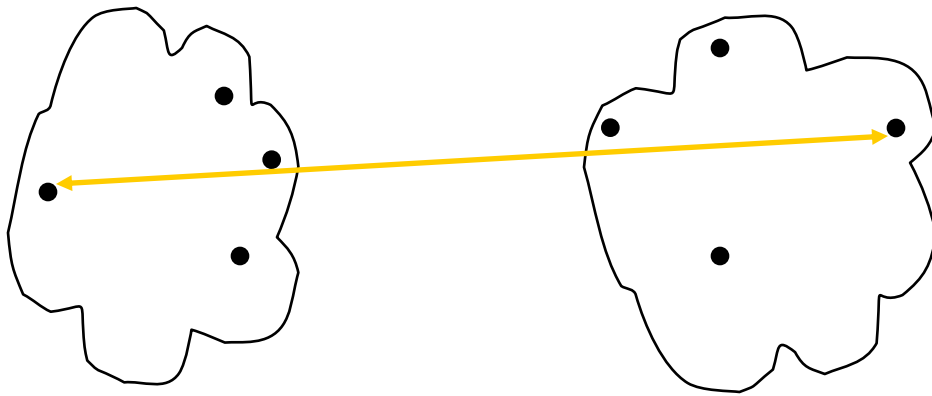


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

· Proximity Matrix

# How to Define Inter-Cluster Similarity

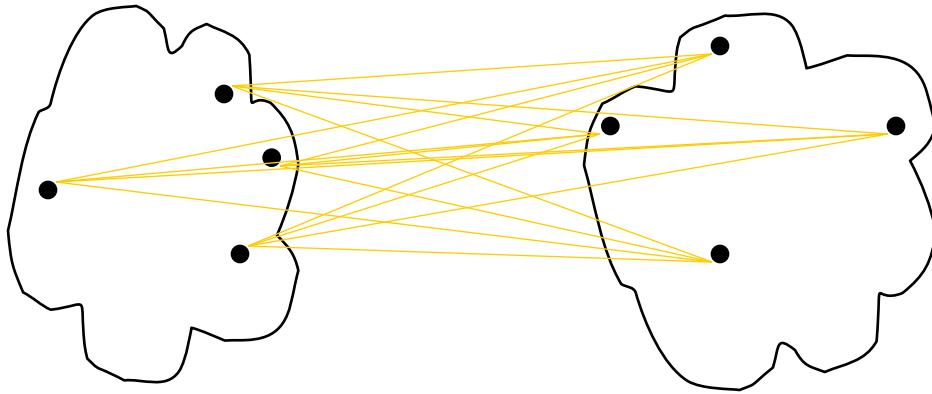


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

**Proximity Matrix**

# How to Define Inter-Cluster Similarity

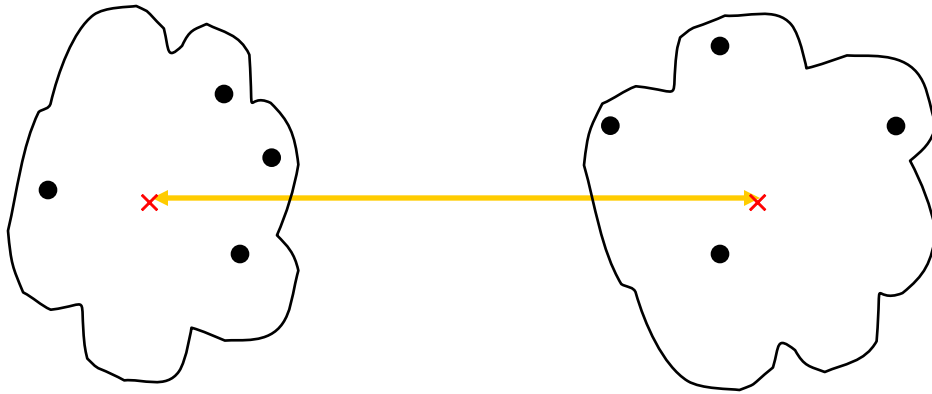


- MIN
- MAX
- **Group Average**
- Distance Between Centroids
- Other methods driven by an objective function
  - Ward's Method uses squared error

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

· **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

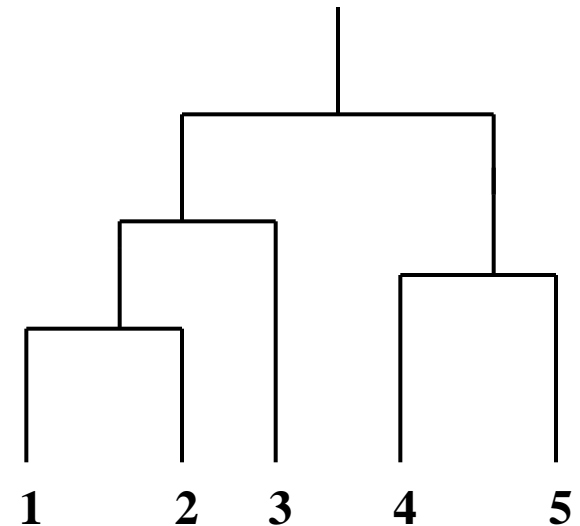
	p1	p2	p3	p4	p5	...
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

**Proximity Matrix**

# Cluster Similarity: MIN or Single Link

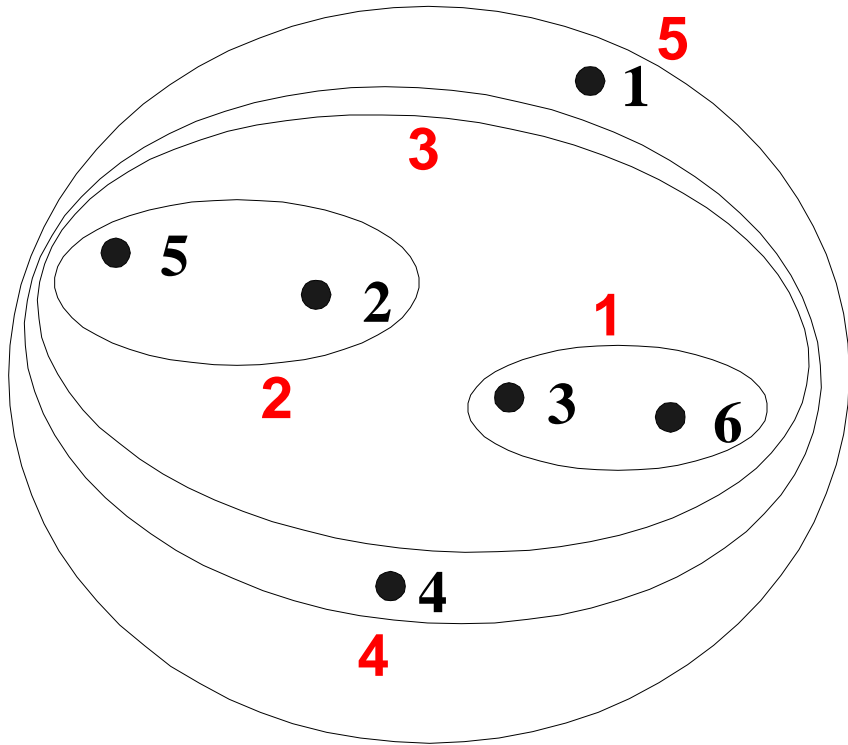
- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph.

	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00

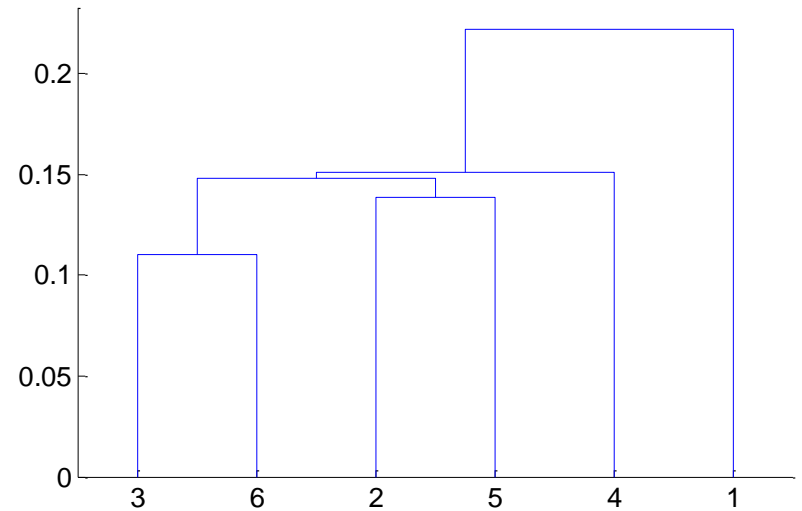




# Hierarchical Clustering: MIN



**Nested Clusters**

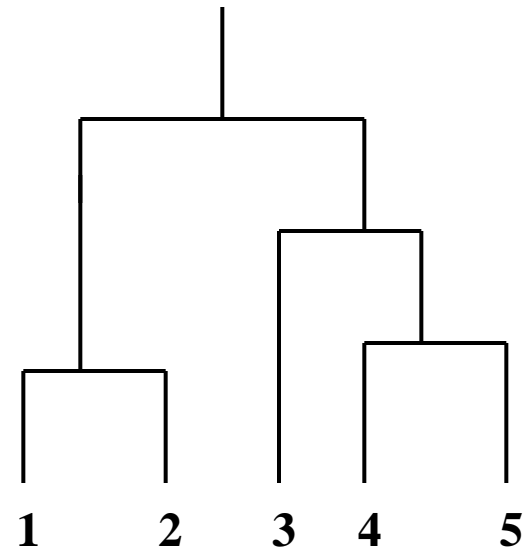


**Dendrogram**

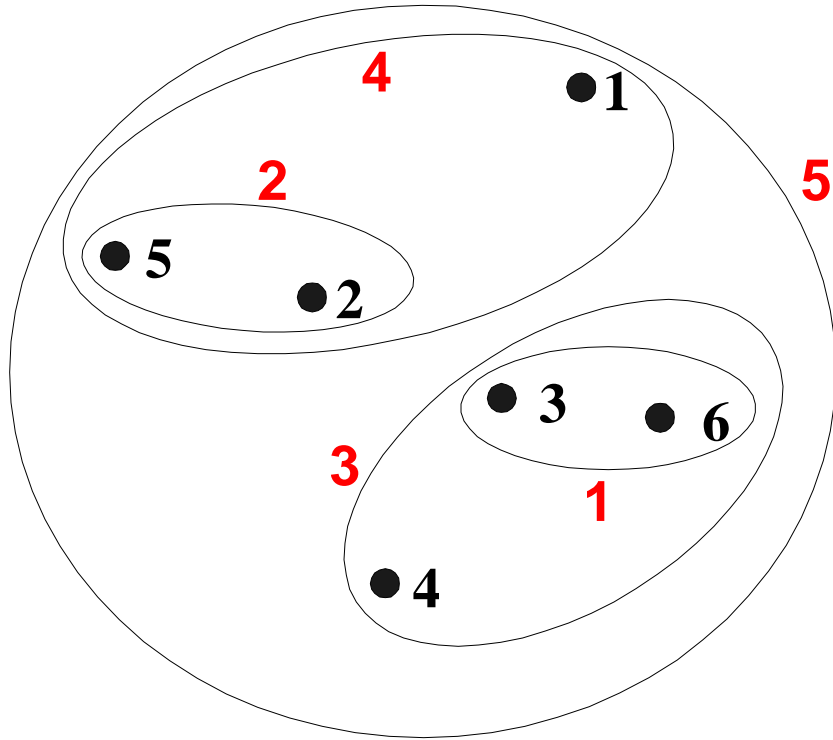
# Cluster Similarity: MAX or Complete Linkage

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

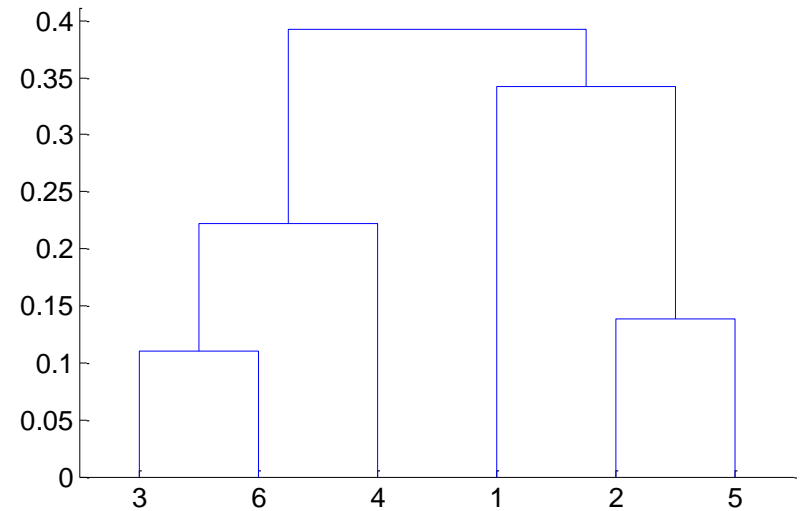
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



# Hierarchical Clustering: MAX



**Nested Clusters**



**Dendrogram**

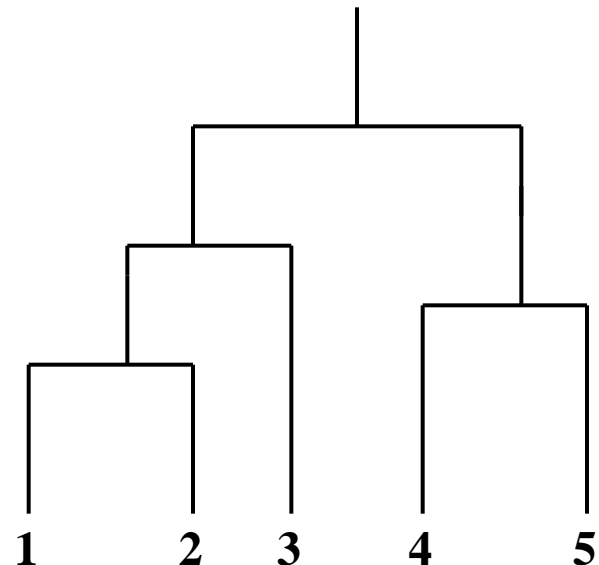
# Cluster Similarity: Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

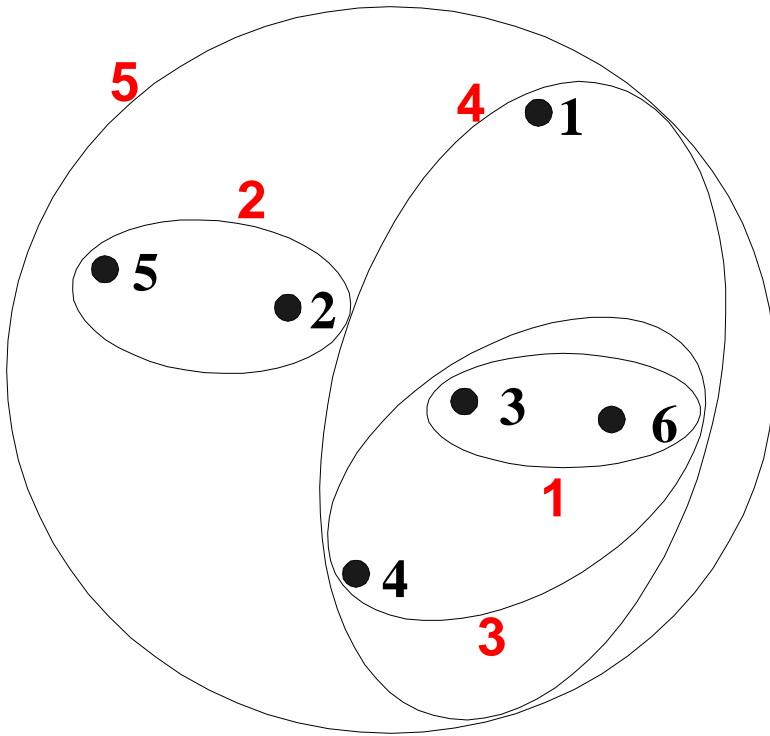
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

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

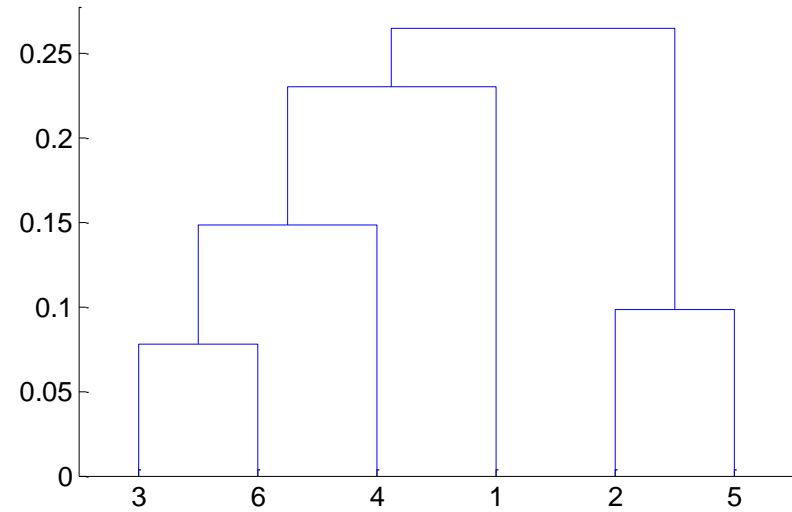
	I1	I2	I3	I4	I5
I1	1.00	0.90	0.10	0.65	0.20
I2	0.90	1.00	0.70	0.60	0.50
I3	0.10	0.70	1.00	0.40	0.30
I4	0.65	0.60	0.40	1.00	0.80
I5	0.20	0.50	0.30	0.80	1.00



# 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: Time and Space requirements

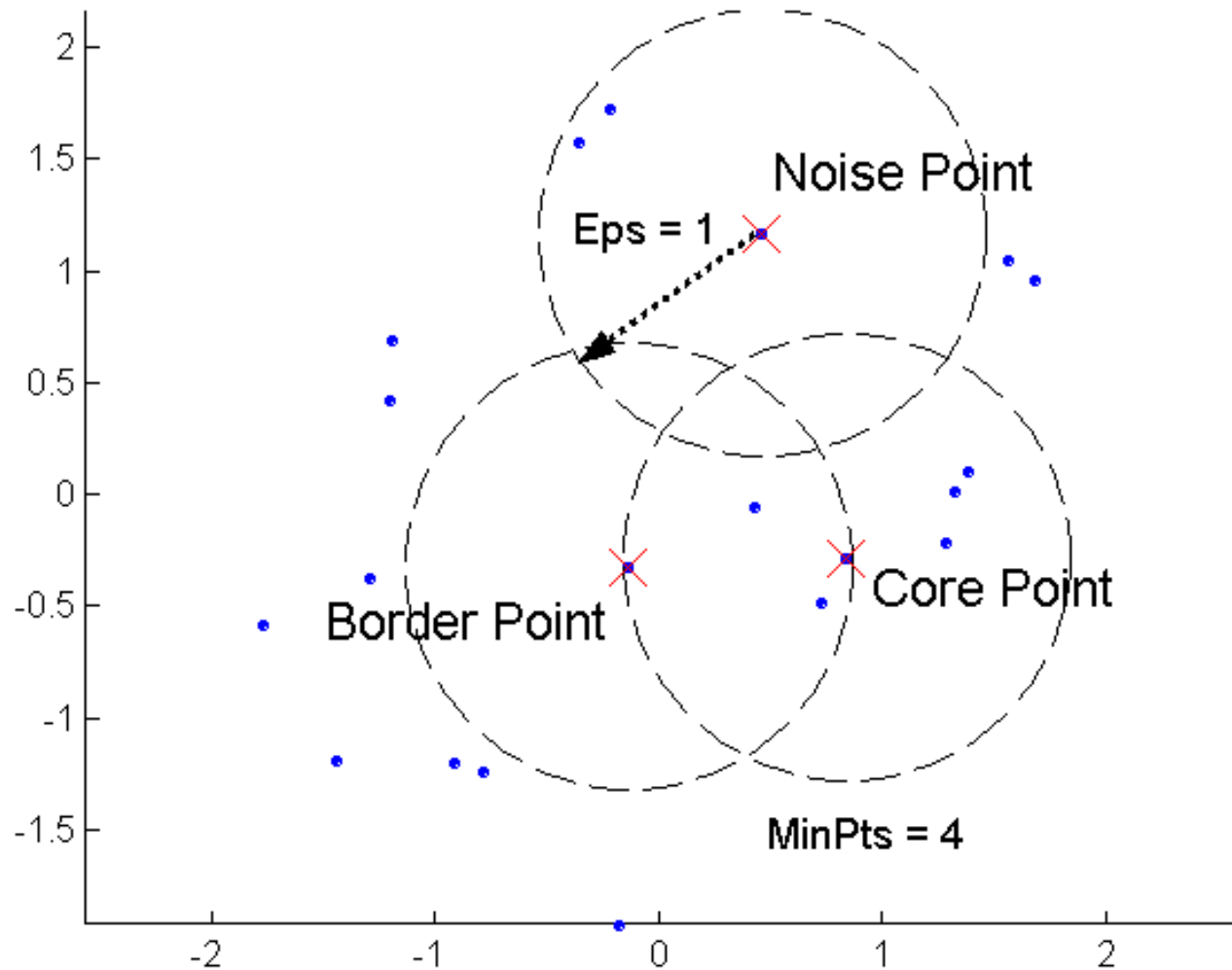
- $O(N^2)$  space since it uses the proximity matrix.
  - $N$  is the number of points.
- $O(N^3)$  time in many cases
  - There are  $N$  steps and at each step the size,  $N^2$ , proximity matrix must be updated and searched
  - Complexity can be reduced to  $O(N^2 \log(N))$  time for some approaches



# DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Epsilon)
  - A point is a **core point** if it has more than a specified number of points (MinPts) within Eps
    - ◆ These are points that are at the interior of a cluster
  - A **border point** has less than MinPts within Eps, 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



# DBSCAN Algorithm

- Eliminate noise points
- Perform clustering on the remaining 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**

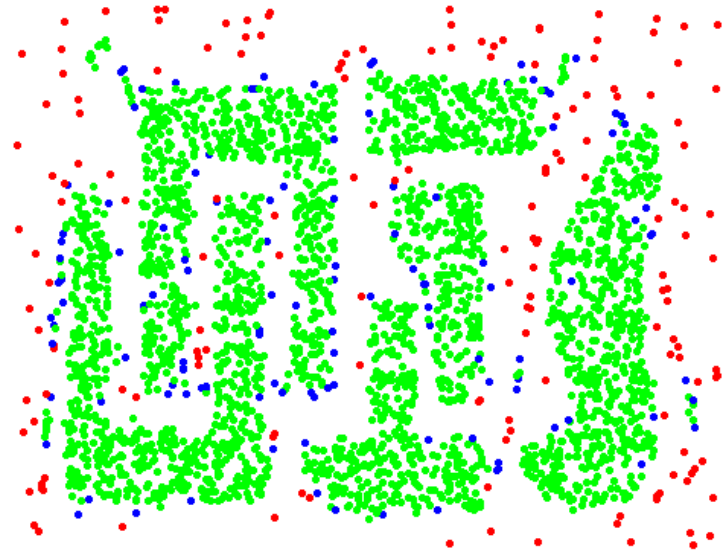
**end for**

**end for**

# DBSCAN: Core, Border and Noise Points



Original Points



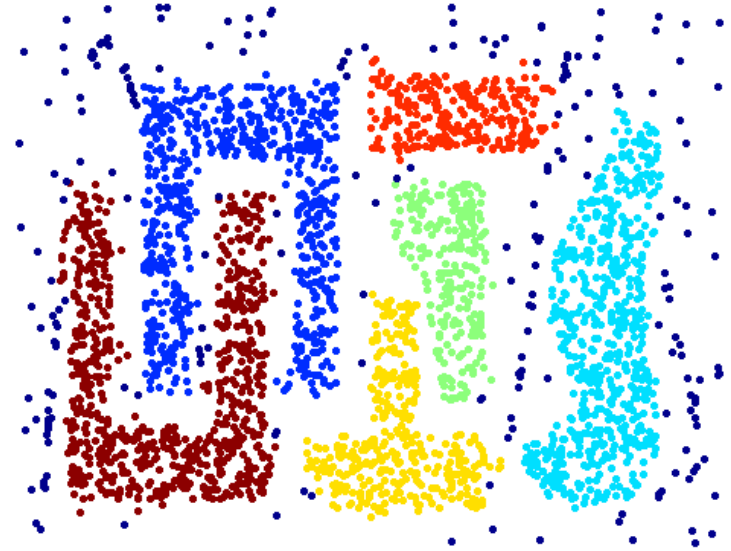
Point types: **core**,  
**border** and **noise**

Eps = 10, MinPts = 4

# When DBSCAN Works Well



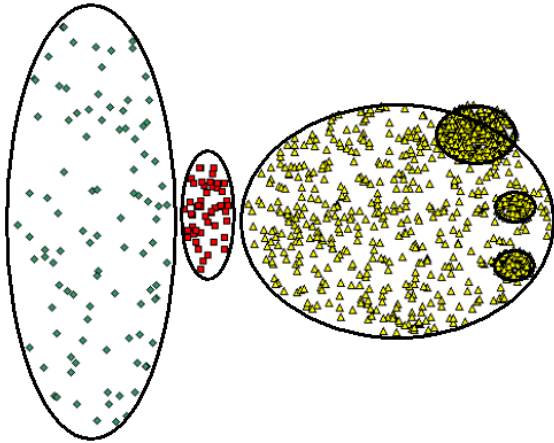
Original Points



Clusters

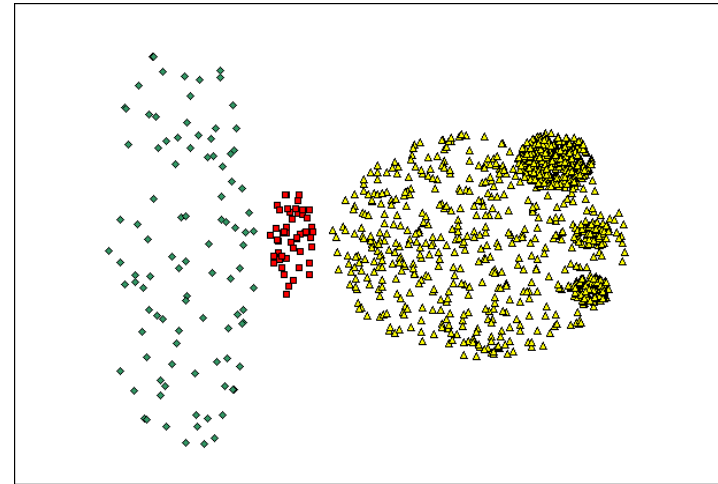
- Resistant to Noise
- Can handle clusters of different shapes and sizes

# When DBSCAN Does NOT Work Well

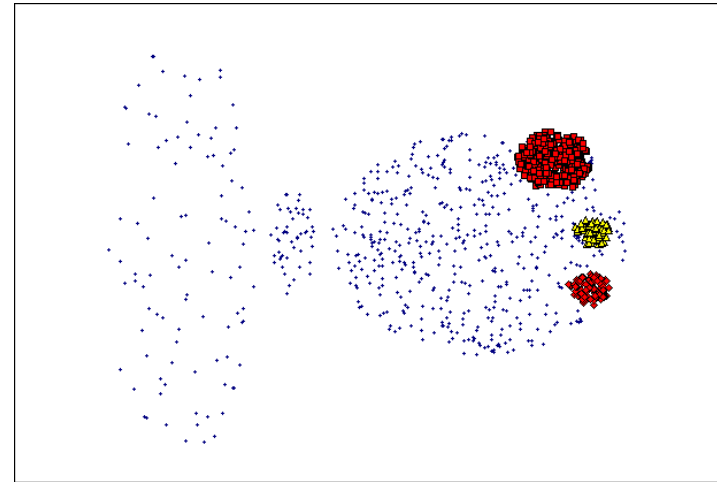


**Original Points**

- Varying densities
- High-dimensional data



(MinPts=4, Eps=9.75).



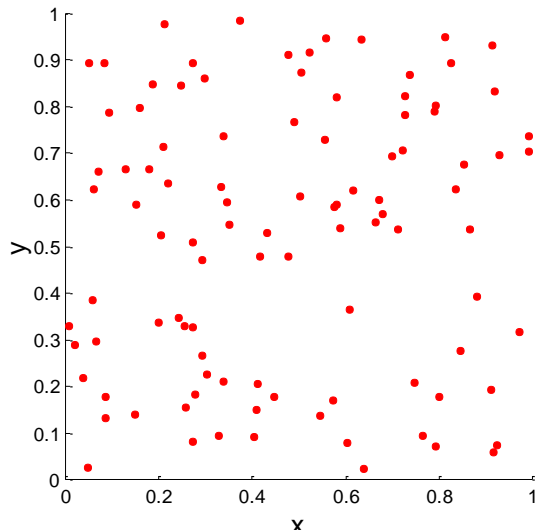
(MinPts=4, Eps=9.92)

# Cluster Validity

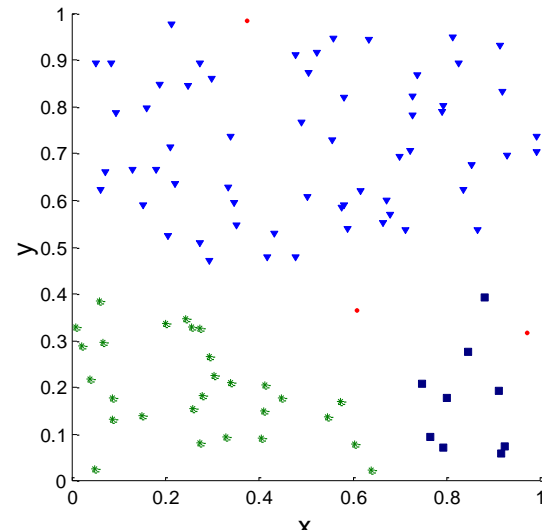
- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the “goodness” of the resulting clusters?
- Why to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms

# Clusters found in Random Data

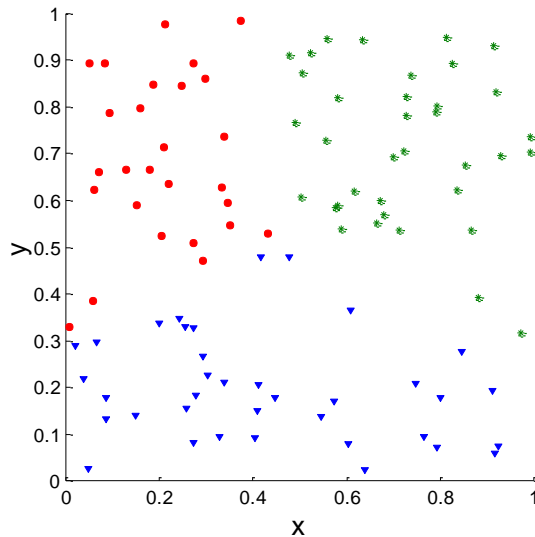
Random  
Points



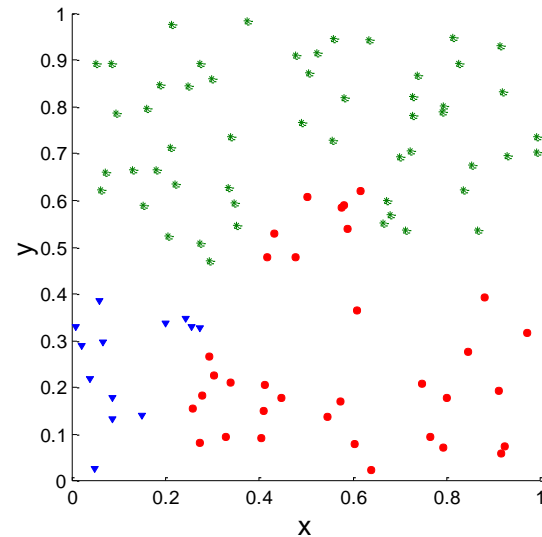
DBSCAN



K-means



Complete  
Link





# Different Aspects of Cluster Validation

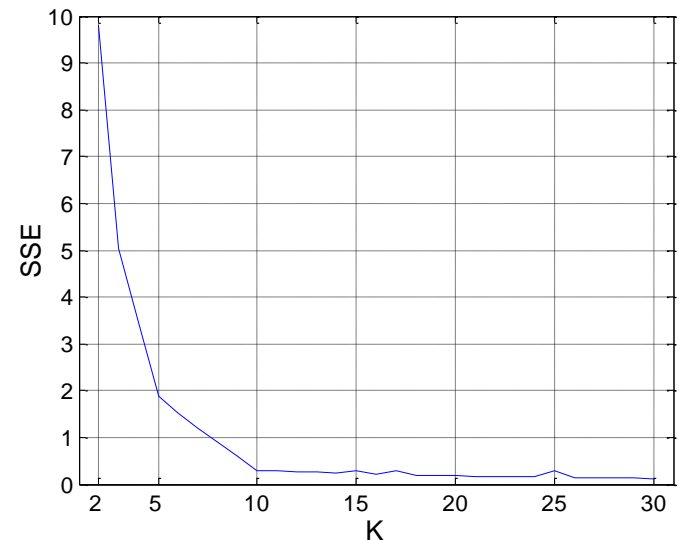
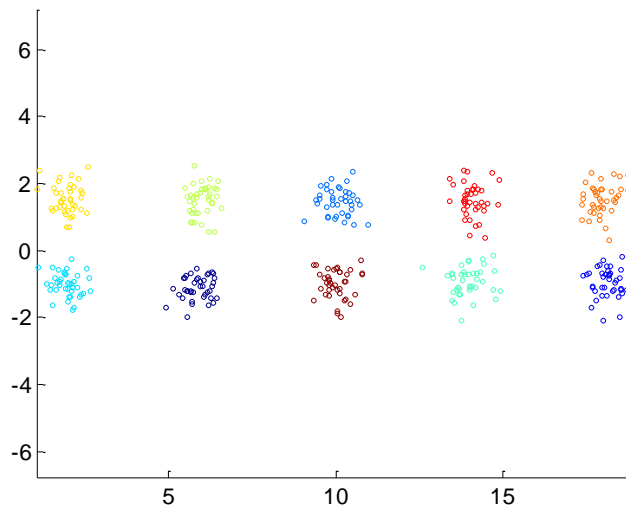
1. Determining the **clustering tendency** of a set of data, i.e., distinguishing whether **non-random structure actually exists** in the data.
2. Comparing the results of a cluster analysis to **externally known results**, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data ***without* reference to external information**.
  - Use only the data
4. Comparing the results of **two different sets of cluster analyses** to determine which is **better**.
5. Determining the '**correct**' number of clusters.

# 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:** Used to measure the extent to which cluster labels match externally supplied class labels.
    - ◆ Entropy
  - **Internal Index:** 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

# Internal Measures: SSE

- Clusters in more complicated figures aren't well separated
- Internal Index: Used to measure the goodness of a clustering structure without respect to external information
  - SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters



# Internal 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)

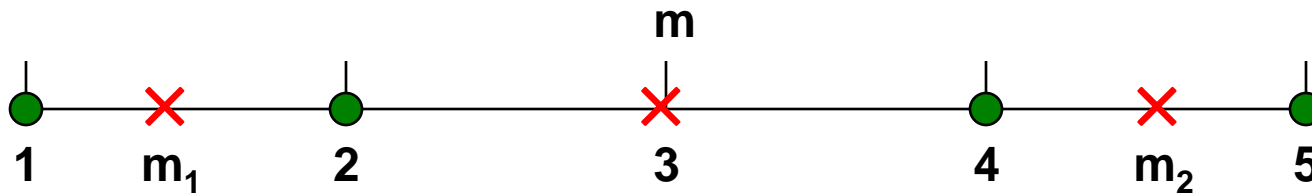
$$WSS = \sum_i \sum_{x \in C_i} (x - m_i)^2$$

- Separation is measured by the **between cluster sum of squares**
$$BSS = \sum_i |C_i| (m - m_i)^2$$

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

# Internal Measures: Cohesion and Separation

- Example: SSE
  - $BSS + WSS = \text{constant}$



**K=1 cluster:**

$$WSS = (1 - 3)^2 + (2 - 3)^2 + (4 - 3)^2 + (5 - 3)^2 = 10$$

$$BSS = 4 \times (3 - 3)^2 = 0$$

$$Total = 10 + 0 = 10$$

**K=2 clusters:**

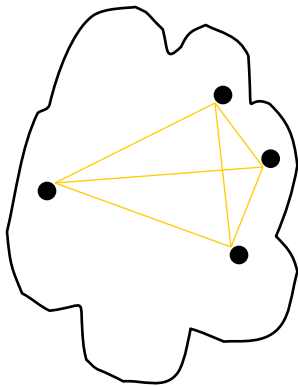
$$WSS = (1 - 1.5)^2 + (2 - 1.5)^2 + (4 - 4.5)^2 + (5 - 4.5)^2 = 1$$

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

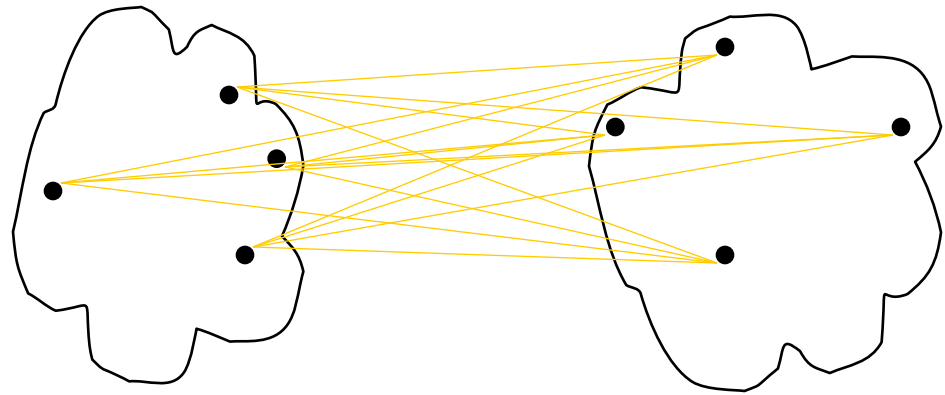
$$Total = 1 + 9 = 10$$

# Internal 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.



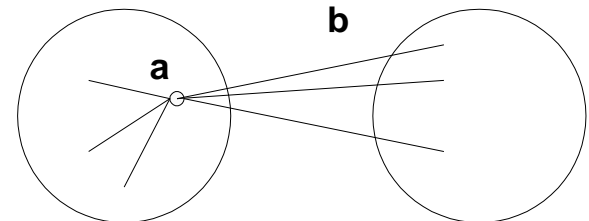
cohesion



separation

# Internal Measures: Silhouette Coefficient

- Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points
- 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
$$s = 1 - a/b \quad \text{if } a < b, \quad (\text{or } s = b/a - 1 \quad \text{if } a \geq b, \text{ not the usual case})$$
  - Typically between 0 and 1.
  - The closer to 1 the better.



# External 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.,  $e = \sum_{j=1}^K \frac{m_j}{m} e_j$ , 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  $purity_j = \max_i p_{ij}$  and the overall purity of a clustering by  $purity = \sum_{j=1}^K \frac{m_j}{m} purity_j$ .



# 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