# Data Mining: Clustering

## Knowledge Discovery in Databases

**Dino Pedreschi**

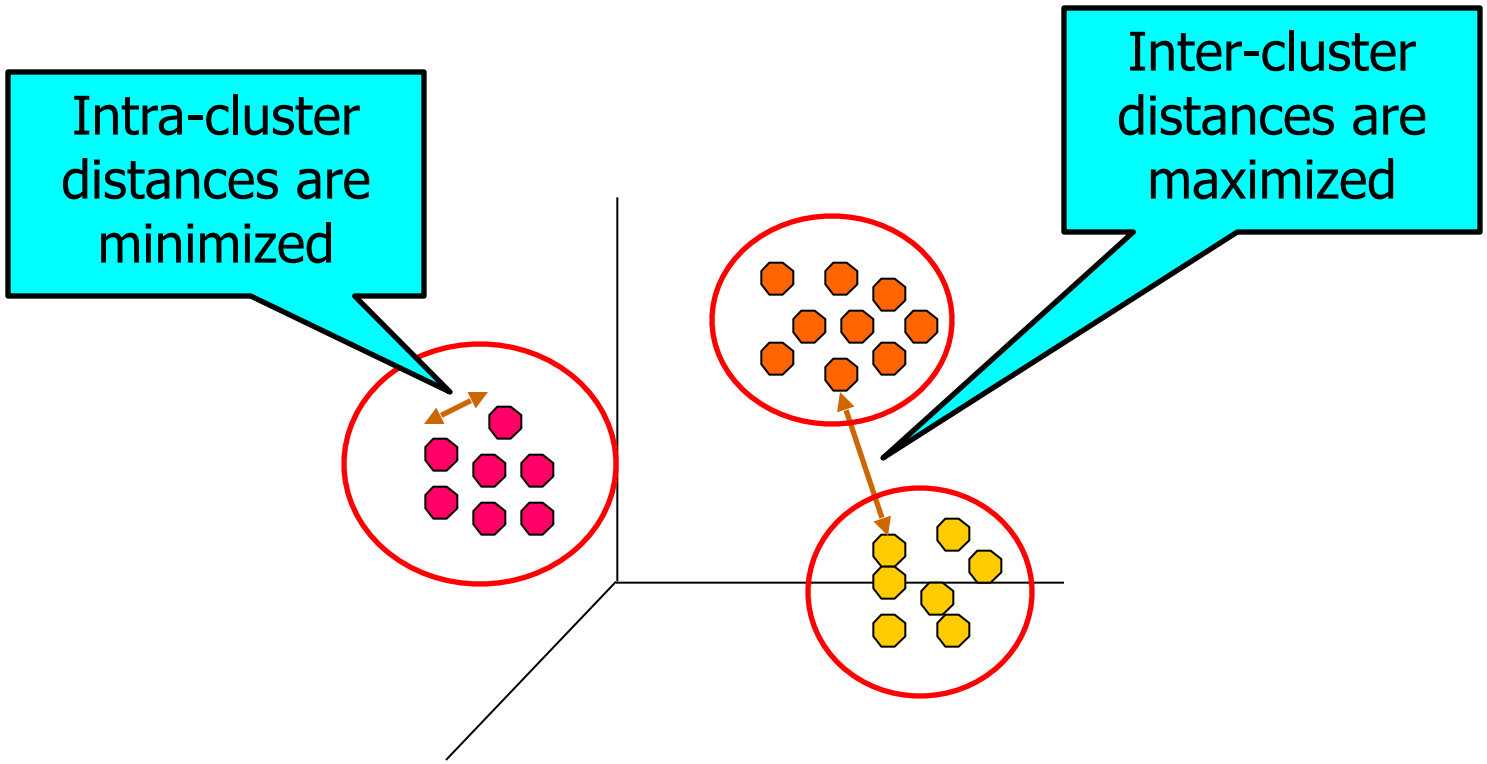
**Pisa KDD Lab, ISTI-CNR & Univ. Pisa**

**http://kdd.isti.cnr.it/**

Master MAINS

# 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**



# Similarity and Dissimilarity

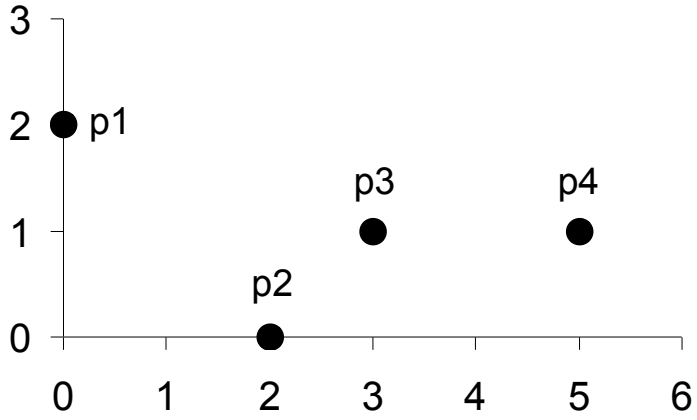
- **Similarity**
  - Numerical measure of how alike two data objects are.
  - Is higher when objects are more alike.
  - Often falls in the range [0,1]
- **Dissimilarity**
  - Numerical measure of how different are two data objects
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies
- **Proximity refers to a similarity or dissimilarity**

# Euclidean Distance

- Euclidean Distance

$$dist = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$$

# Euclidean Distance




| point | x | y |
|---|---|---|
| p1 | 0 | 2 |
| p2 | 2 | 0 |
| p3 | 3 | 1 |
| p4 | 5 | 1 |

| | p1 | p2 | p3 | p4 |
|---|---|---|---|---|
| p1 | 0 | 2.828 | 3.162 | 5.099 |
| p2 | 2.828 | 0 | 1.414 | 3.162 |
| p3 | 3.162 | 1.414 | 0 | 2 |
| p4 | 5.099 | 3.162 | 2 | 0 |

Distance Matrix

# Minkowski Distance

- Minkowski Distance is a generalization of Euclidean Distance

$$dist = \left( \sum_{k=1}^{n} | p_k - q_k |^r \right)^{\frac{1}{r}}$$

# Minkowski Distance: Examples

- *r* = 1. City block (Manhattan, taxicab, $L_1$ norm) distance.
  - A common example of this is the Hamming distance, which is just the number of bits that are different between two binary vectors
- *r* = 2. Euclidean distance
- $r \rightarrow \infty$. "supremum" ($L_{max}$ norm, $L_\infty$ norm) distance.
  - This is the maximum difference between any component of the vectors
- Do not confuse *r* with *n*, i.e., all these distances are defined for all numbers of dimensions.

# Minkowski Distance

| point | x | y |
|---|---|---|
| p1 | 0 | 2 |
| p2 | 2 | 0 |
| p3 | 3 | 1 |
| p4 | 5 | 1 |

| L1 | p1 | p2 | p3 | p4 |
|---|---|---|---|---|
| p1 | 0 | 4 | 4 | 6 |
| p2 | 4 | 0 | 2 | 4 |
| p3 | 4 | 2 | 0 | 2 |
| p4 | 6 | 4 | 2 | 0 |

| L2 | p1 | p2 | p3 | p4 |
|---|---|---|---|---|
| p1 | 0 | 2.828 | 3.162 | 5.099 |
| p2 | 2.828 | 0 | 1.414 | 3.162 |
| p3 | 3.162 | 1.414 | 0 | 2 |
| p4 | 5.099 | 3.162 | 2 | 0 |

| $L_\infty$ | p1 | p2 | p3 | p4 |
|---|---|---|---|---|
| p1 | 0 | 2 | 3 | 5 |
| p2 | 2 | 0 | 1 | 3 |
| p3 | 3 | 1 | 0 | 2 |
| p4 | 5 | 3 | 2 | 0 |

Distance Matrix

# Common Properties of a Distance

- **Distances, such as the Euclidean distance, have some well known properties.**

  1. $d(p, q) \geq 0$ for all $p$ and $q$ and $d(p, q) = 0$ only if $p = q$. (Positive definiteness)
  2. $d(p, q) = d(q, p)$ for all $p$ and $q$. (Symmetry)
  3. $d(p, r) \leq d(p, q) + d(q, r)$ for all points $p$, $q$, and $r$. (Triangle Inequality)

  where $d(p, q)$ is the distance (dissimilarity) between points (data objects), $p$ and $q$.

- **A distance that satisfies these properties is a metric**

# Common Properties of a Similarity

- **Similarities, also have some well known properties.**

  1. ***s(p, q) = 1*** **(or maximum similarity) only if** ***p = q*****.**

  2. ***s(p, q) = s(q, p)*** **for all** ***p*** **and** ***q*****. (Symmetry)**

  **where** ***s(p, q)*** **is the similarity between points (data objects),** ***p*** **and** ***q*****.**

# Similarity Between Binary Vectors

- Common situation is that objects, *p* and *q*, have only binary attributes
- Compute similarities using the following quantities

  $M_{01}$ = the number of attributes where p was 0 and q was 1

  $M_{10}$ = the number of attributes where p was 1 and q was 0

  $M_{00}$ = the number of attributes where p was 0 and q was 0

  $M_{11}$ = the number of attributes where p was 1 and q was 1
- Simple Matching and Jaccard Coefficients

  SMC = number of matches / number of attributes

  = (M11 + M00) / (M01 + M10 + M11 + M00)

  J = number of 11 matches / number of not-both-zero attributes values

  = (M11) / (M01 + M10 + M11)

# SMC versus Jaccard: Example

*p* = 1 0 0 0 0 0 0 0 0 0

*q* = 0 0 0 0 0 0 1 0 0 1

$M_{01}$ = 2 (the number of attributes where p was 0 and q was 1)

$M_{10}$ = 1 (the number of attributes where p was 1 and q was 0)

$M_{00}$ = 7 (the number of attributes where p was 0 and q was 0)

$M_{11}$ = 0 (the number of attributes where p was 1 and q was 1)

$SMC = (M_{11} + M_{00})/(M_{01} + M_{10} + M_{11} + M_{00}) = (0+7) / (2+1+0+7) = 0.7$

$J = (M_{11}) / (M_{01} + M_{10} + M_{11}) = 0 / (2 + 1 + 0) = 0$

# Cosine Similarity

- If $d_1$ and $d_2$ are two document vectors, then

$$\cos(d_1, d_2) = (d_1 \bullet d_2) / ||d_1|| \, ||d_2|| ,$$

where • indicates vector dot product and $||d||$ is the length of vector $d$.

- Example:

$d_1$ = 3 2 0 5 0 0 0 2 0 0

$d_2$ = 1 0 0 0 0 0 0 1 0 2

$d_1 \bullet d_2$ = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5

$||d_1|| = (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$

$||d_2|| = (1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5} = (6)^{0.5} = 2.245$

$\cos(d_1, d_2) = .3150$

# General Approach for Combining Similarities

- **Sometimes attributes are of many different types, but an overall similarity is needed.**

1. For the $k^{th}$ attribute, compute a similarity, $s_k$, in the range $[0, 1]$.
2. Define an indicator variable, $\delta_k$, for the $k_{th}$ attribute as follows:

$$\delta_k = \begin{cases} 0 & \text{if the } k^{th} \text{ attribute is a binary asymmetric attribute and both objects have a value of 0, or if one of the objects has a missing values for the } k^{th} \text{ attribute} \\ 1 & \text{otherwise} \end{cases}$$

3. Compute the overall similarity between the two objects using the following formula:

$$similarity(p, q) = \frac{\sum_{k=1}^{n} \delta_k s_k}{\sum_{k=1}^{n} \delta_k}$$

# Using Weights to Combine Similarities

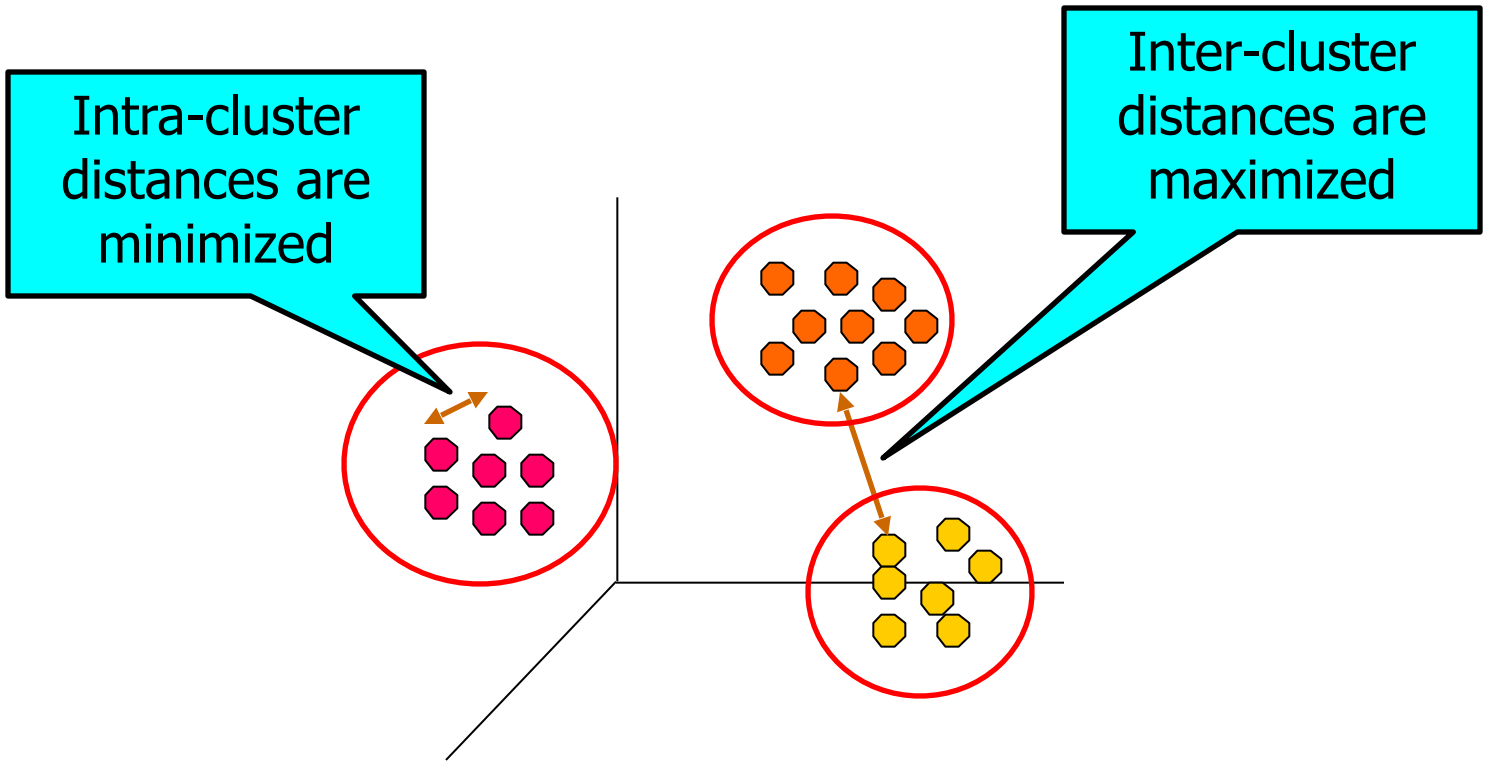
- **May not want to treat all attributes the same.**
  - **Use weights $w_k$ which are between 0 and 1 and sum to 1.**

$$similarity(p,q) = \frac{\sum_{k=1}^{n} w_k \delta_k s_k}{\sum_{k=1}^{n} \delta_k}$$

$$distance(p,q) = \left( \sum_{k=1}^{n} w_k |p_k - q_k|^r \right)^{1/r}$$

# 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

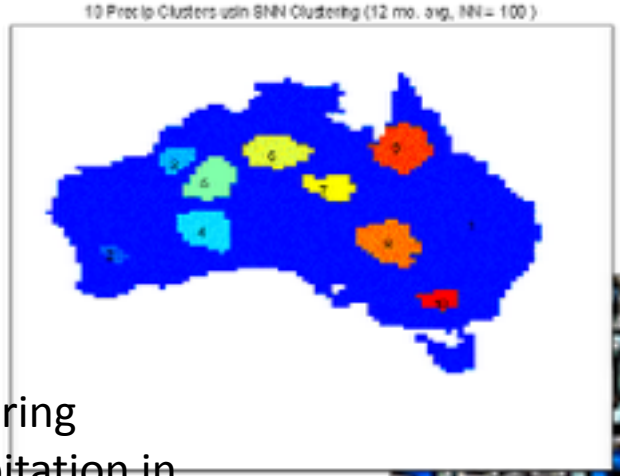
## Understanding

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

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

## Summarization

- Reduce the size of large data sets




Clustering precipitation in Australia

# What is not Cluster Analysis?

- **Supervised classification**
  - Have class label information
- **Simple segmentation**
  - Dividing students into different registration groups alphabetically, by last name
- **Results of a query**
  - Groupings are a result of an external specification
- **Graph partitioning**
  - Some mutual relevance and synergy, but areas are not identical

# Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

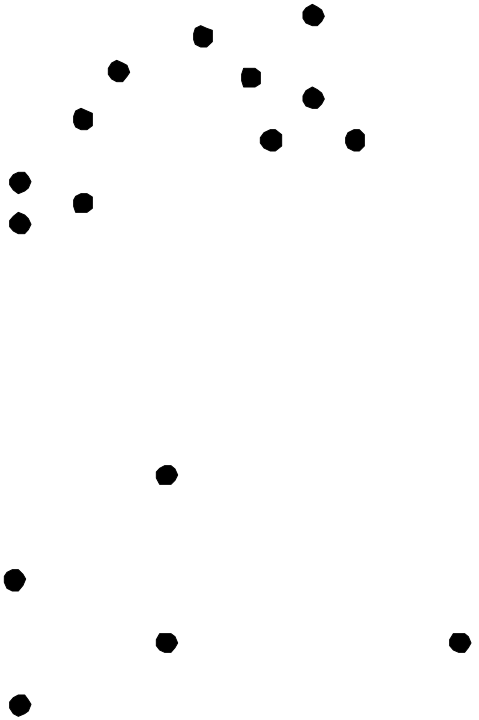
Two Clusters

Four Clusters

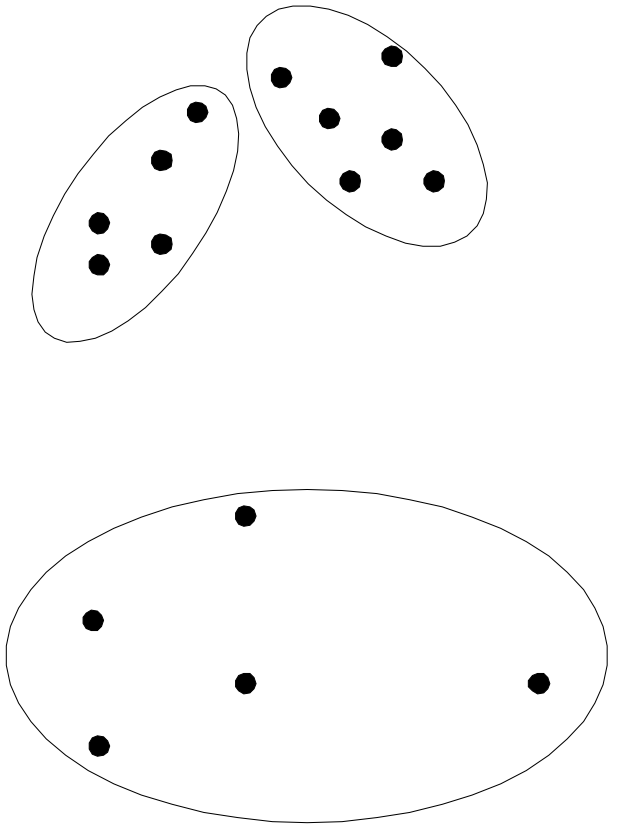
# Types of Clusterings

- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
- Partitional Clustering
  - A division 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

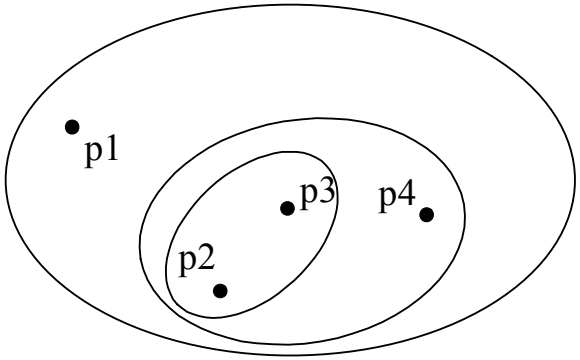


Original Points

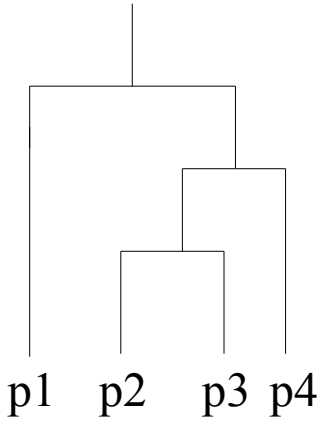


A Partitional Clustering

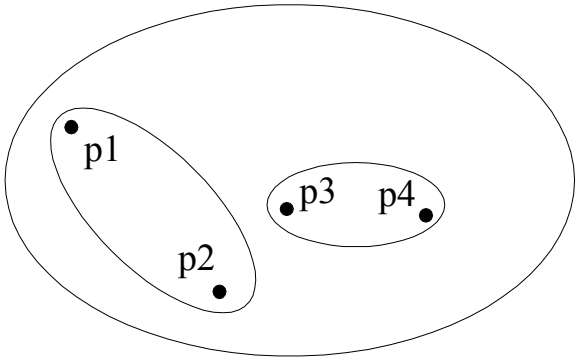
# Hierarchical Clustering



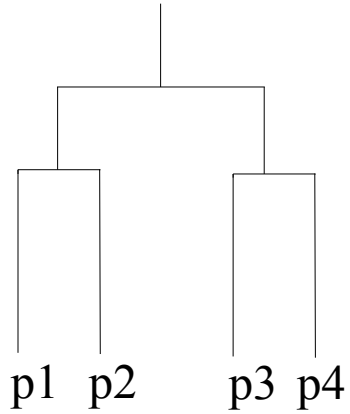

Traditional Hierarchical Clustering




Traditional Dendrogram




Non-traditional Hierarchical Clustering




Non-traditional Dendrogram

# Other Distinctions Between Sets of Clusters

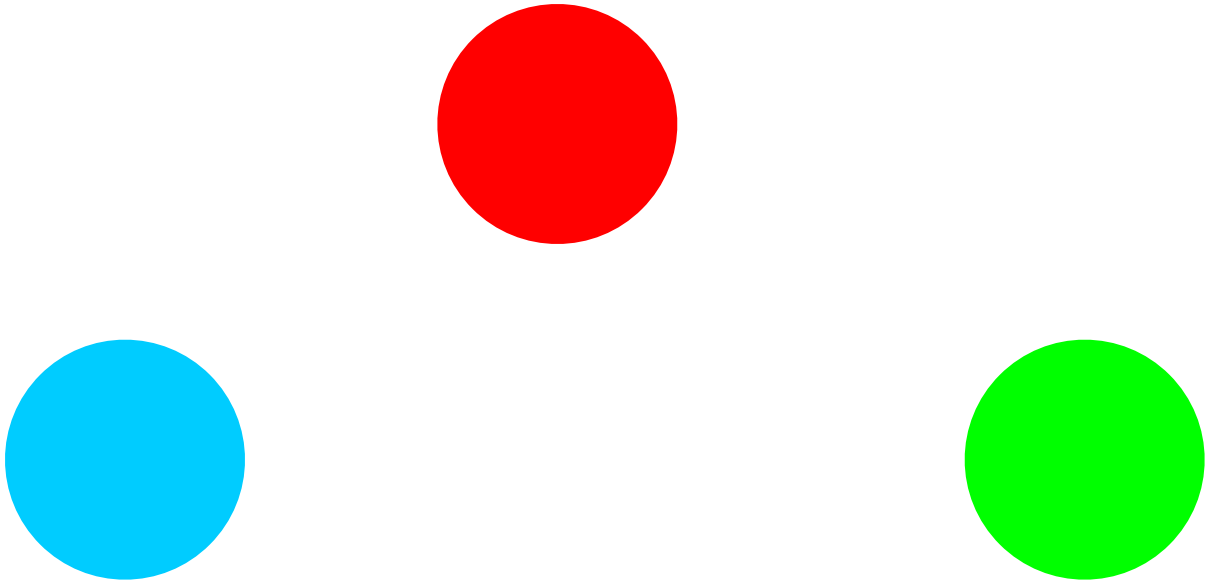
- **Exclusive versus non-exclusive**
  - In non-exclusive clusterings, points may belong to multiple clusters.
  - Can represent multiple classes or ‘border’ points
- **Fuzzy versus non-fuzzy**
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must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

# Types of Clusters

- Well-separated clusters
- Center-based clusters
- Contiguous clusters
- Density-based clusters
- Property or Conceptual
- Described by an Objective Function

# Types of Clusters: Well-Separated

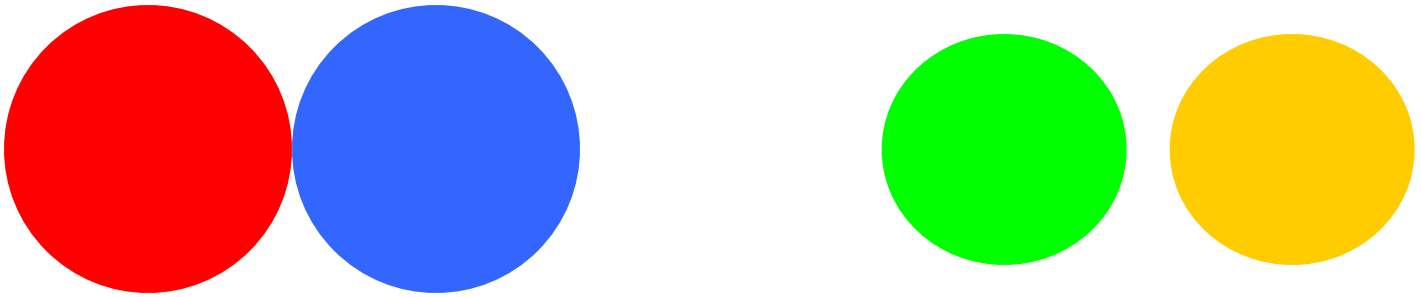
- **Well-Separated Clusters:**
  - **A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.**



3 well-separated clusters

# Types of Clusters: Center-Based

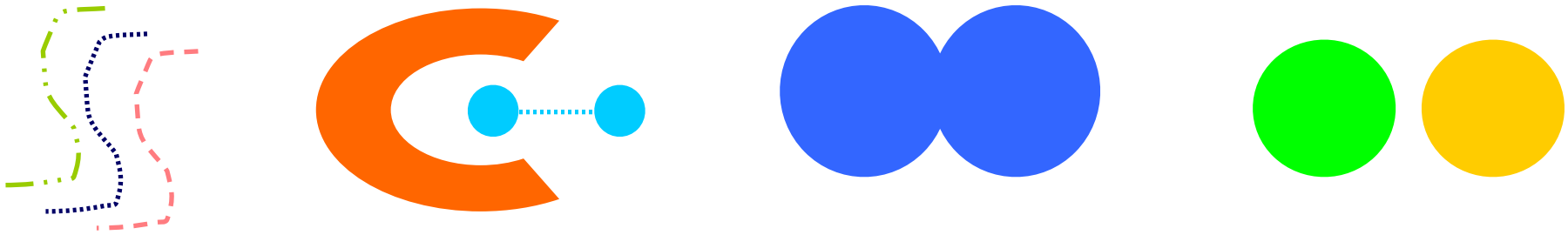
- **Center-based**
  - **A cluster is a set of objects such that an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster**
  - **The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most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



4 center-based clusters

# Types of Clusters: Contiguity-Based

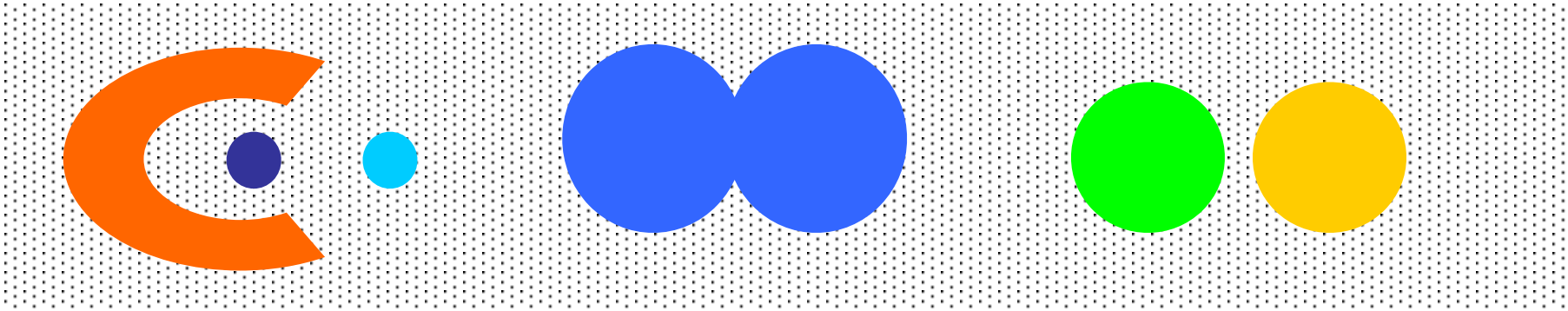
- **Contiguous Cluster (Nearest neighbor or Transitive)**
  - **A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.**



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

# Characteristics of the Input Data Are Important

- Type of proximity or density measure
  - This is a derived measure, but central to clustering
- Sparseness
  - Dictates type of similarity
  - Adds to efficiency
- Attribute type
  - Dictates type of similarity
- Type of Data
  - Dictates type of similarity
  - Other characteristics, e.g., autocorrelation
- Dimensionality
- Noise and Outliers
- Type of Distribution

# Clustering Algorithms

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

# K-means Clustering

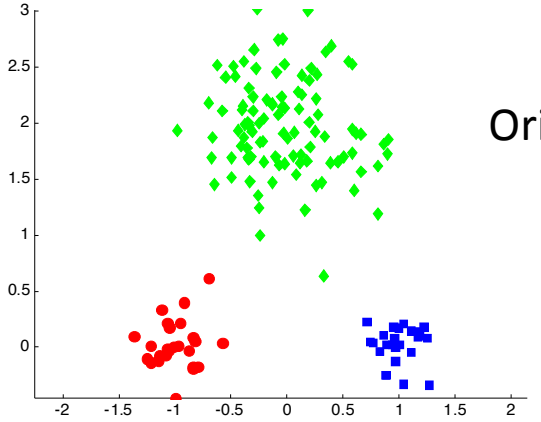
- Partitional clustering approach
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified
- The basic algorithm is very simple

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

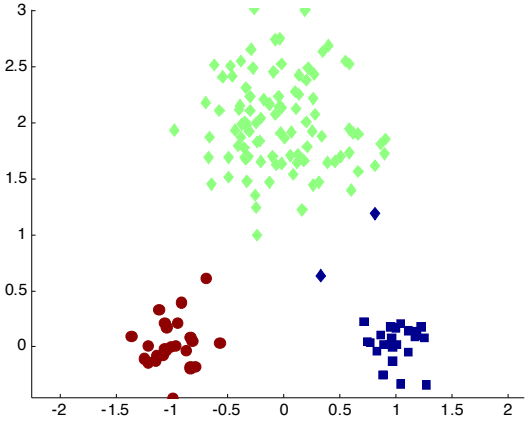
# K-means Clustering – Details

- 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.

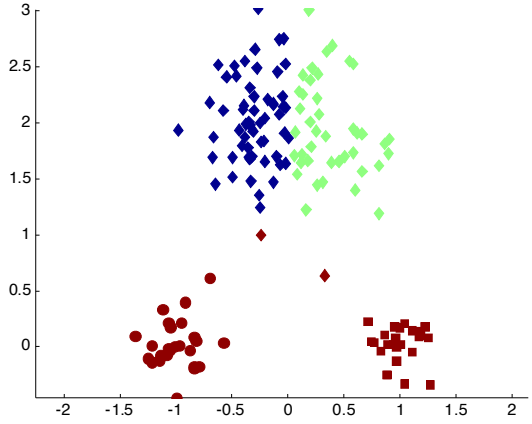
# Two different K-means Clusterings




Original Points

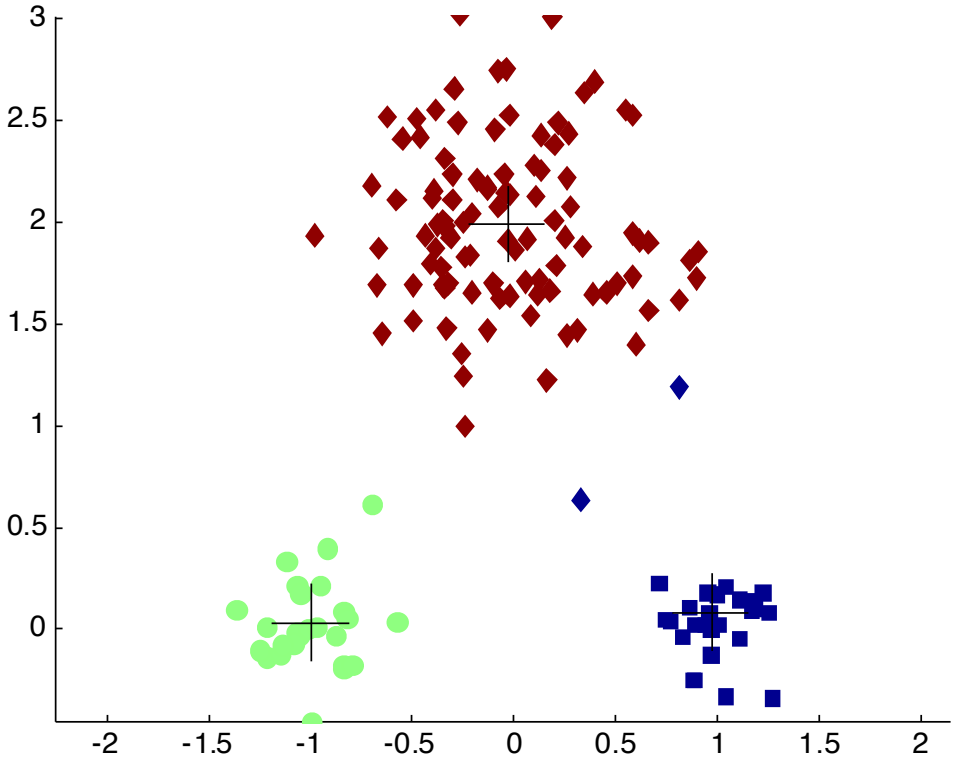



Optimal Clustering

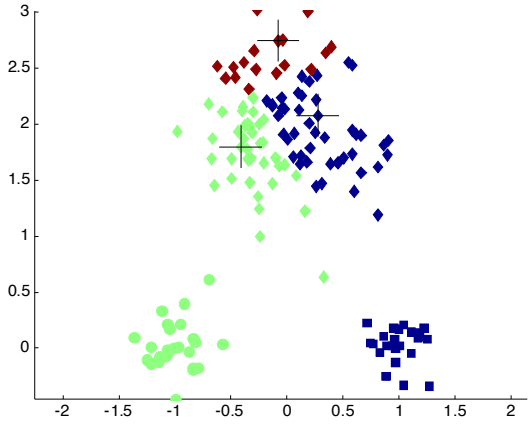
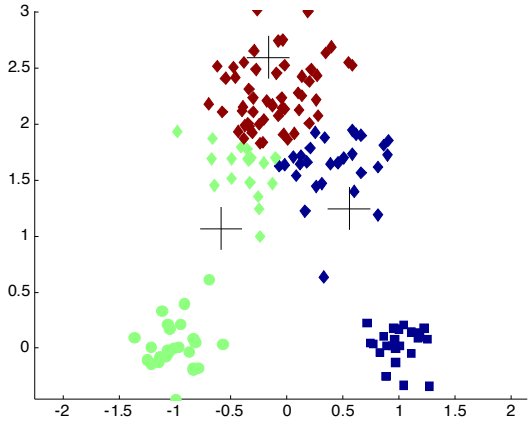
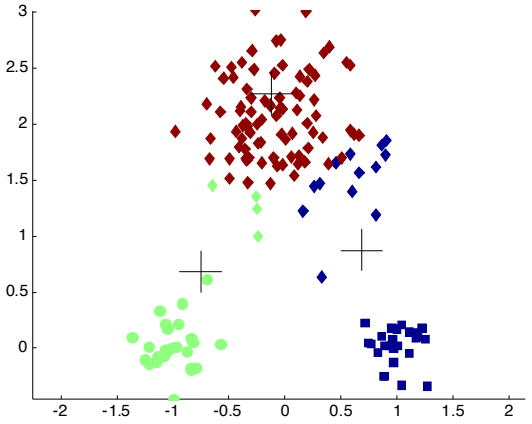
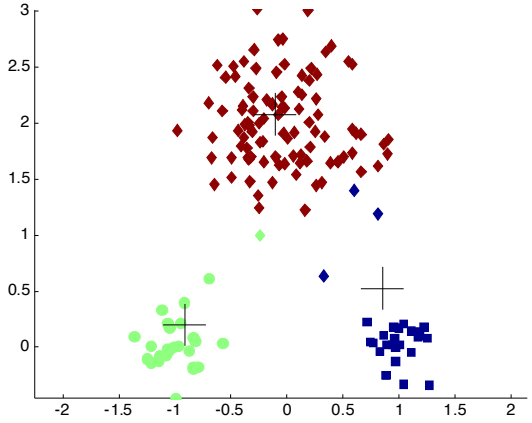
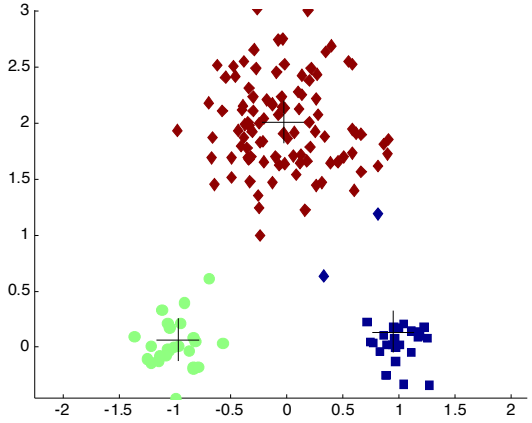
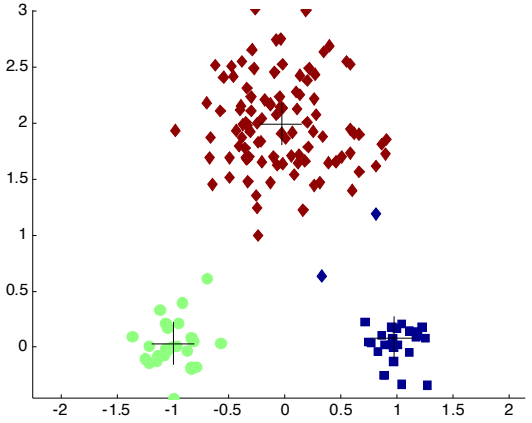



Sub-optimal Clustering

# Importance of Choosing Initial Centroids



# Importance of Choosing Initial Centroids

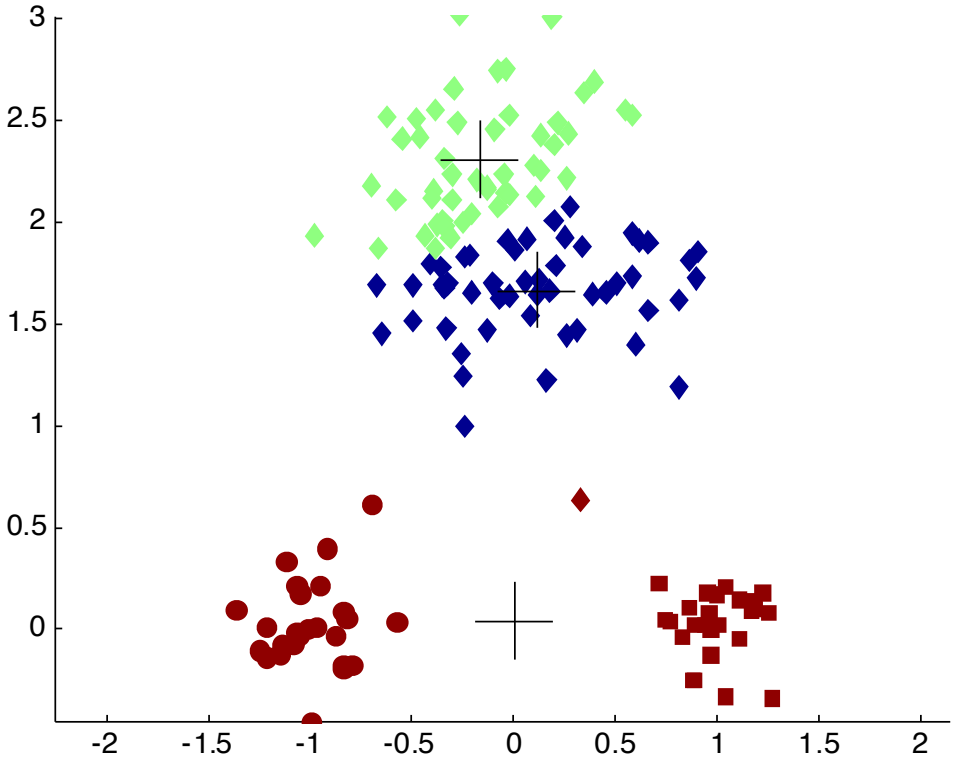
# Evaluating K-means Clusters

- **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.**

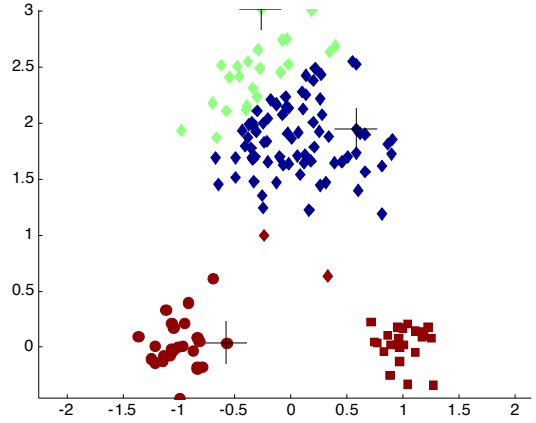
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

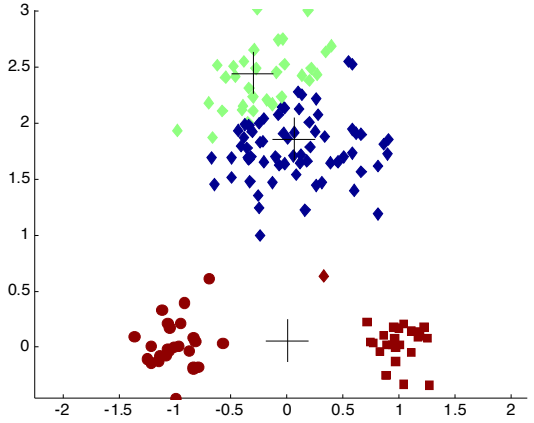
  - ***x* is a data point in cluster $C_i$ and $m_i$ is the representative point for cluster $C_i$**
    - can show that $m_i$ corresponds to the center (mean) of the cluster
  - **Given two clusters, we can choose the one with the smallest error**
  - **One easy way to reduce SSE is to increase K, the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

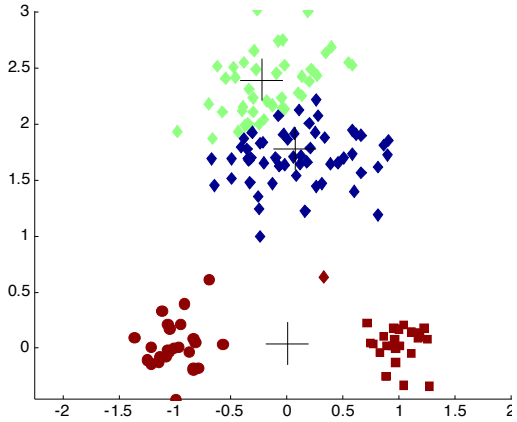
# Importance of Choosing Initial Centroids …

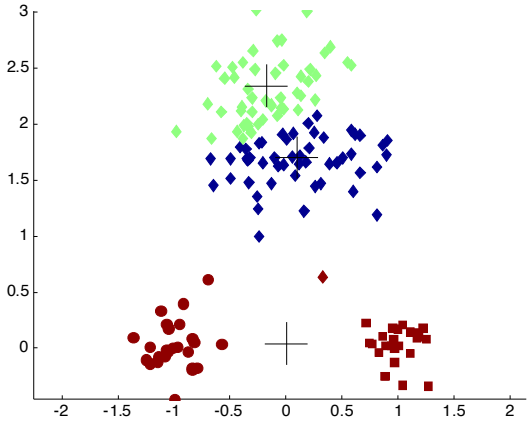


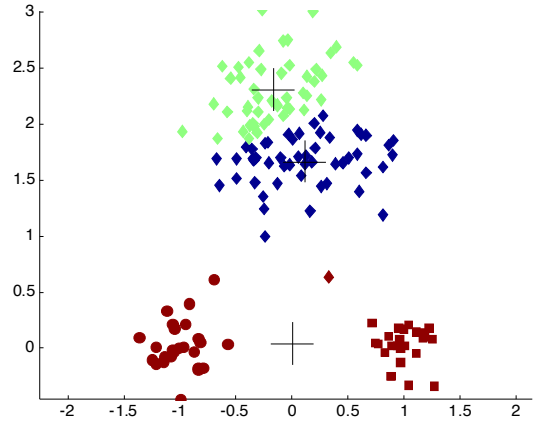
# Importance of Choosing Initial Centroids …



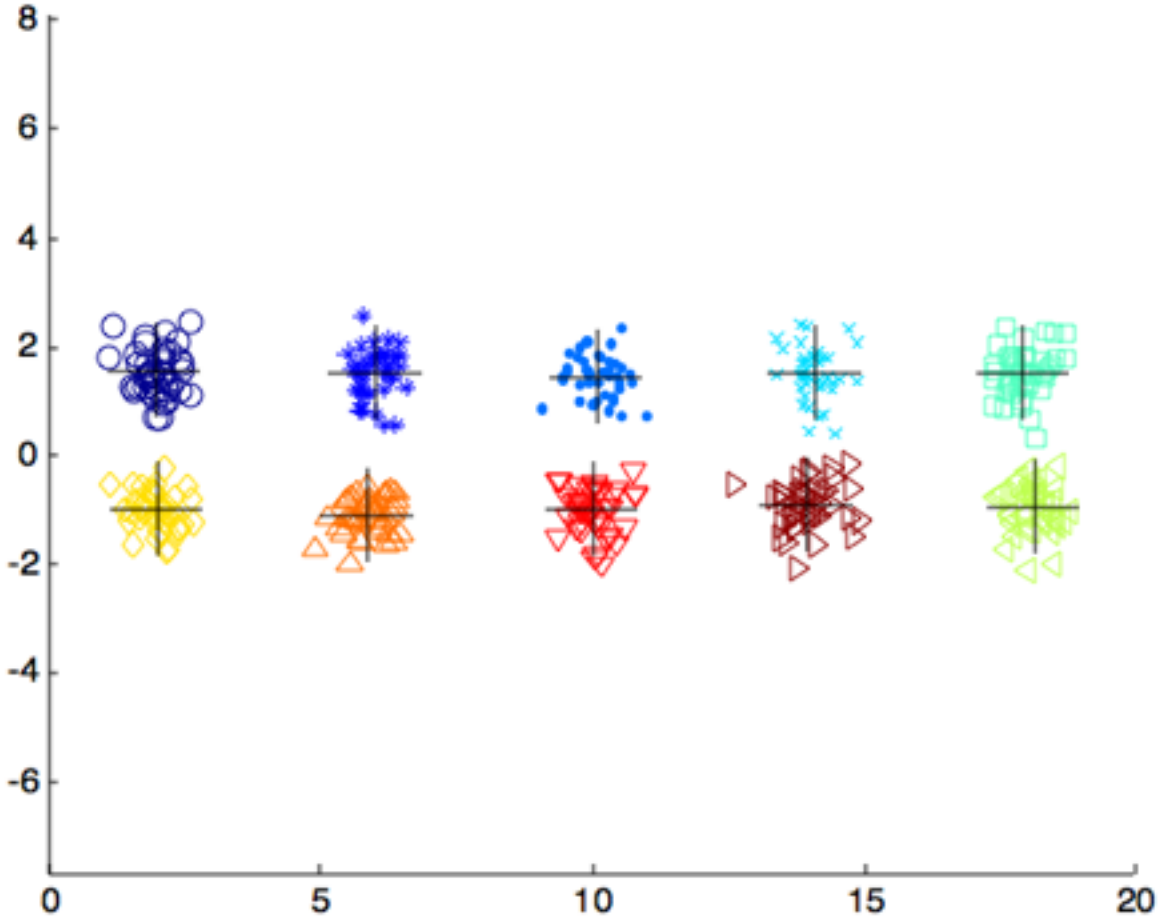



# Problems with Selecting Initial Points

- If there are K 'real' clusters then the chance of selecting one centroid from each cluster is small.
  - Chance is relatively small when K is large
  - If clusters are the same size, n, then

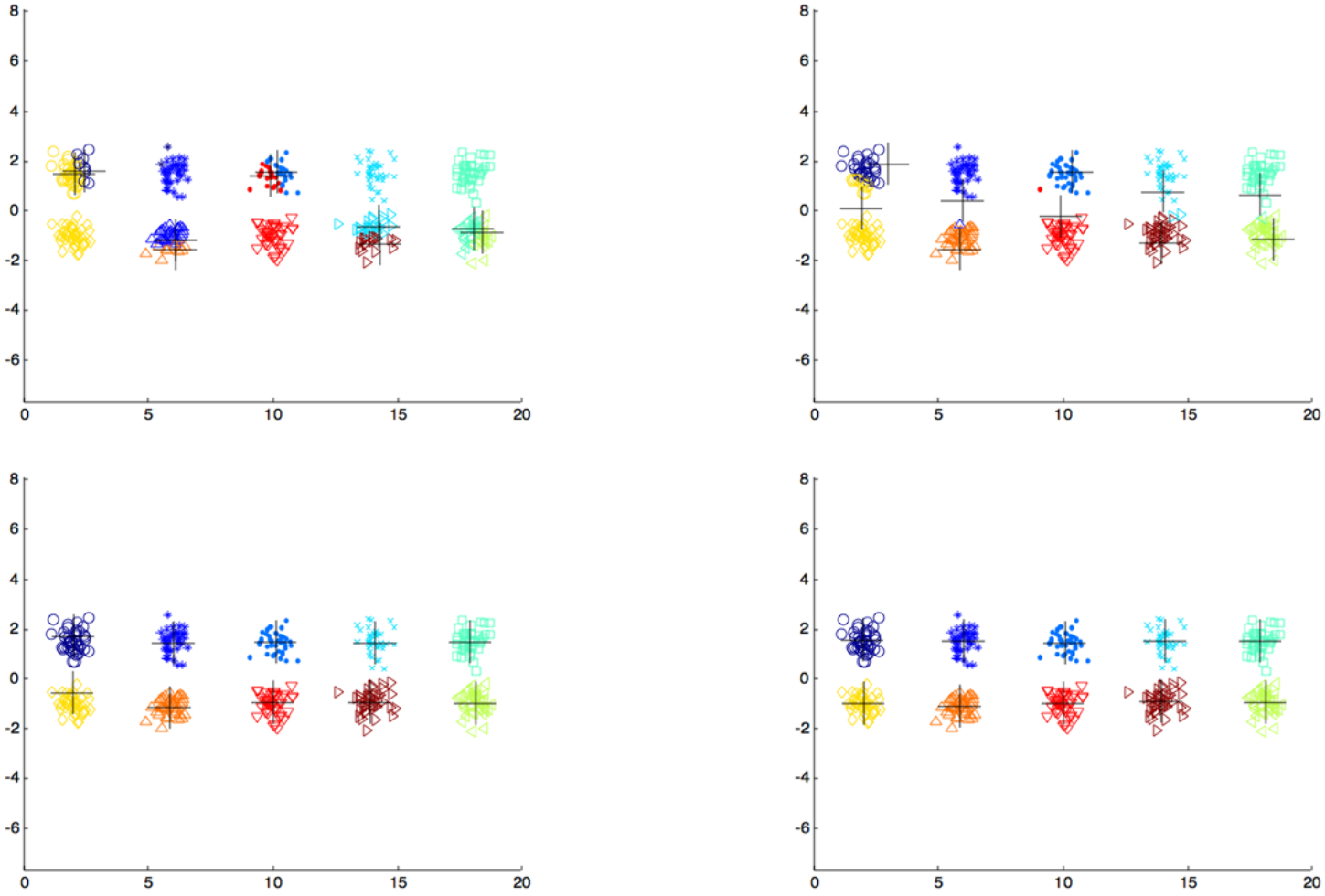
$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

# 10 Clusters Example



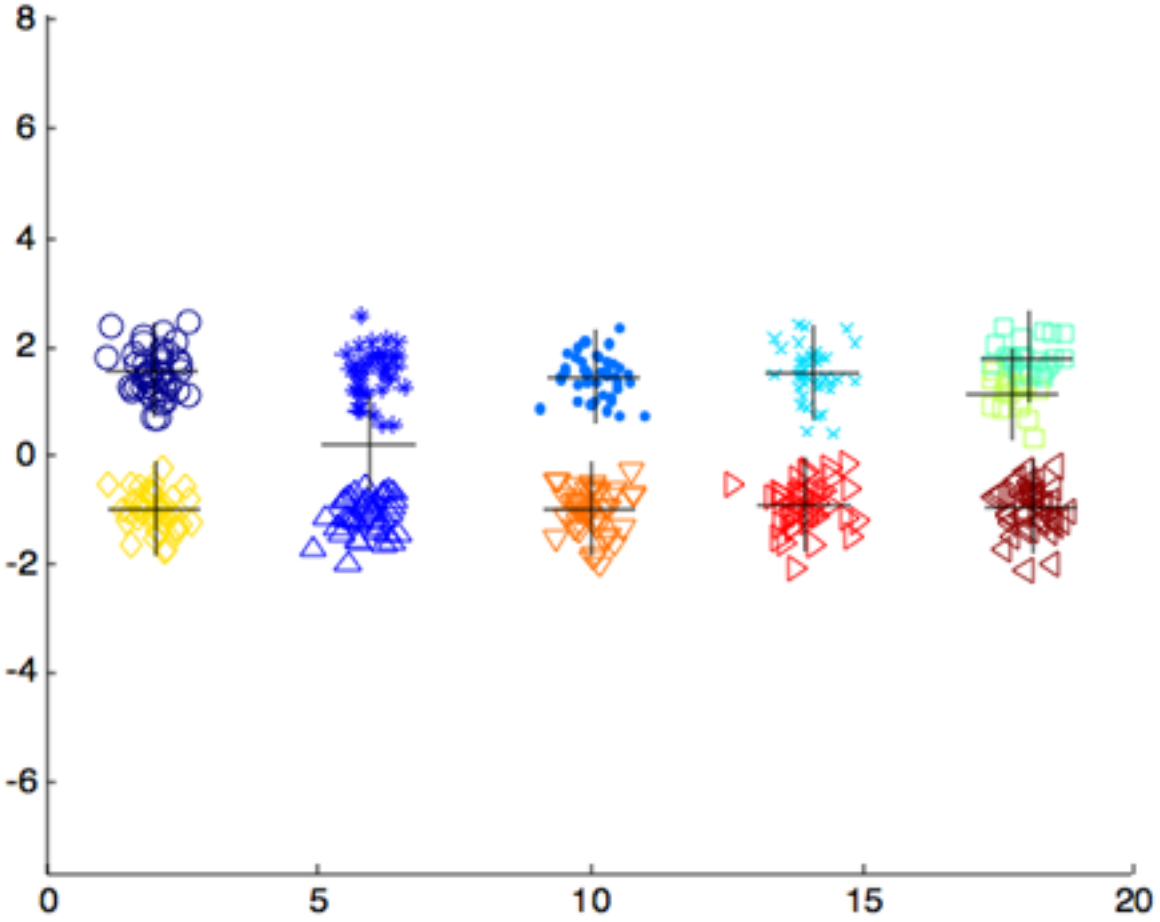

Starting with two initial centroids in one cluster of each pair of clusters

# 10 Clusters Example



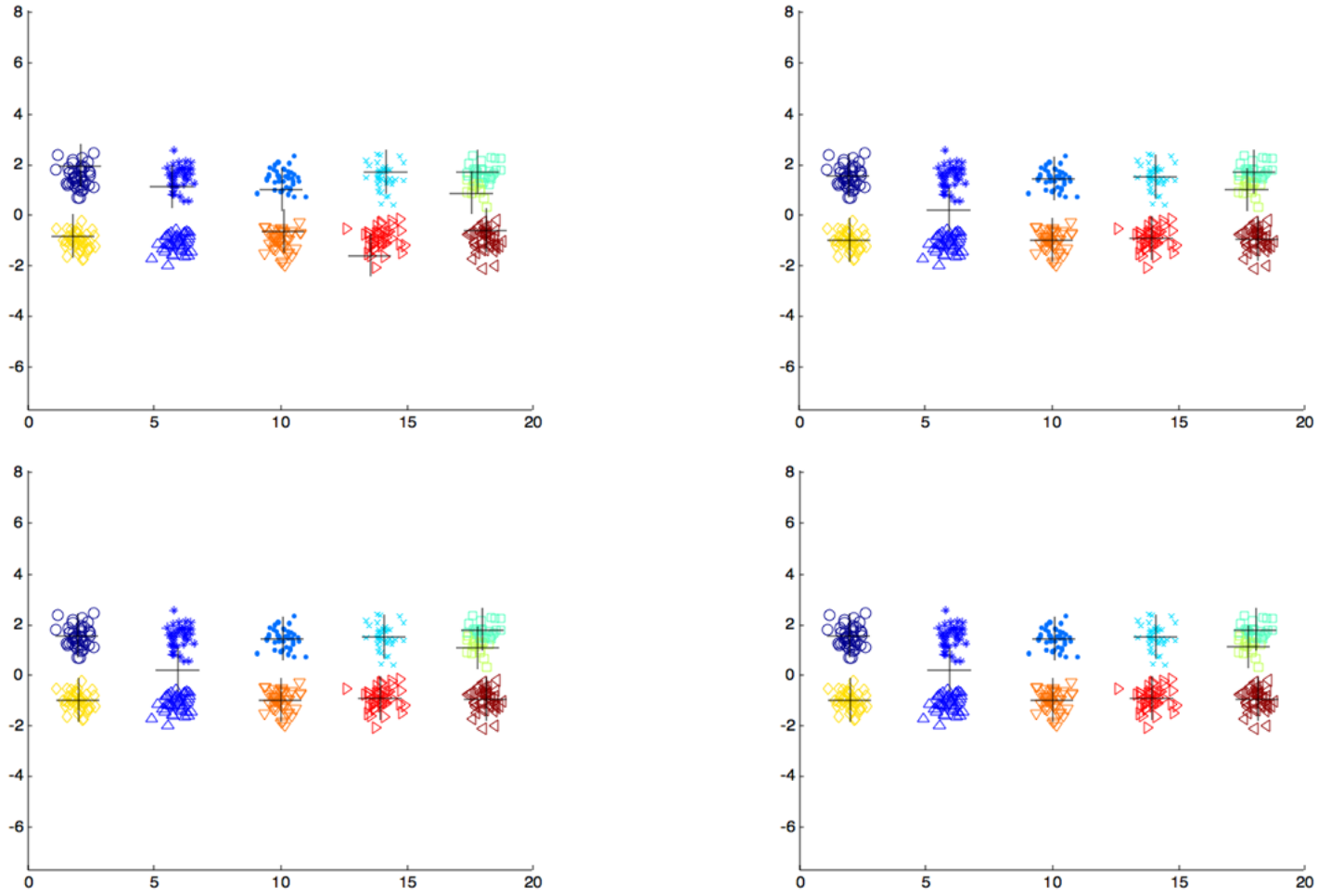

Starting with two initial centroids in one cluster of each pair of clusters

# 10 Clusters Example




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

# 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
  - Helps, but probability is not on your side
- 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

# Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters

- Several strategies
  - Choose the point that contributes most to SSE
  - Choose a point from the cluster with the highest SSE
  - If there are several empty clusters, the above can be repeated several times.

# 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**
    - ISODATA

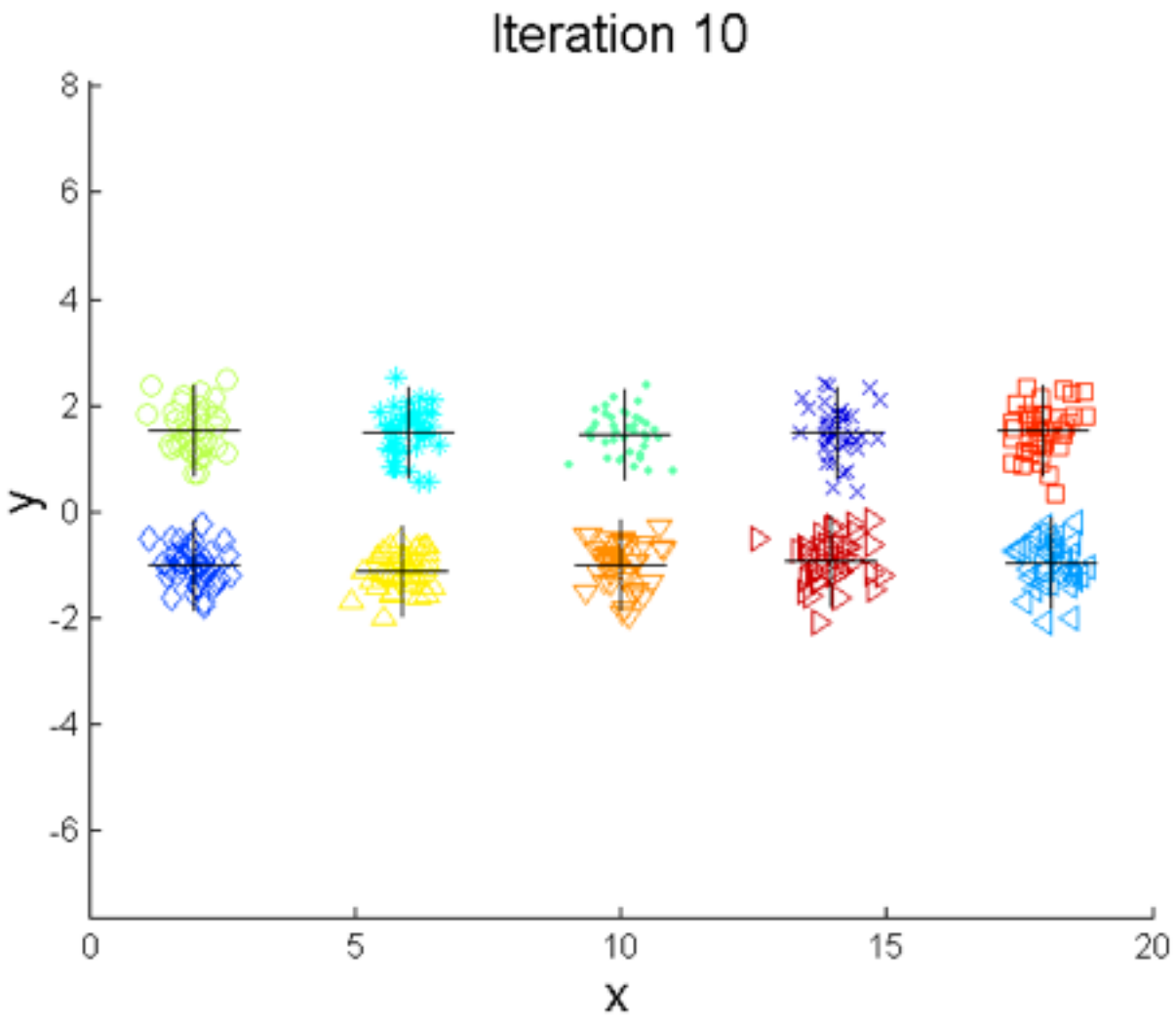
# Bisecting K-means

## 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

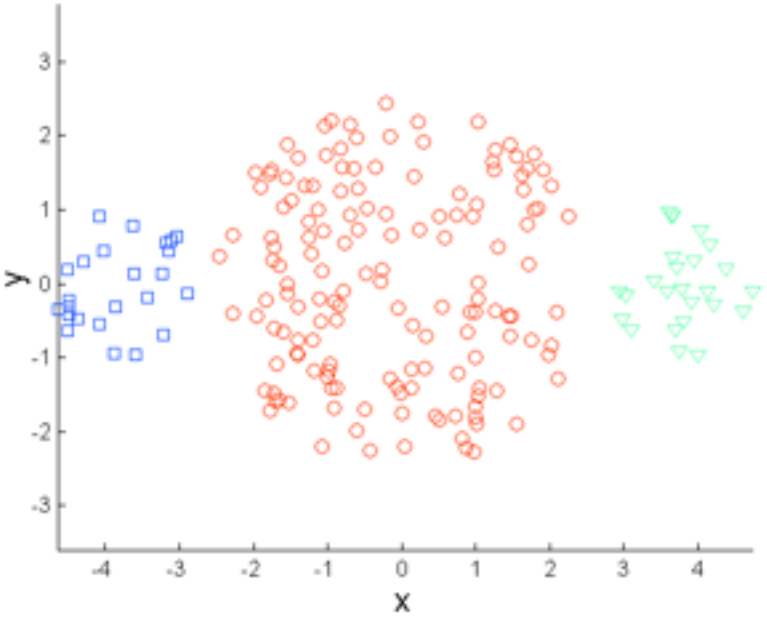


# Limitations of K-means

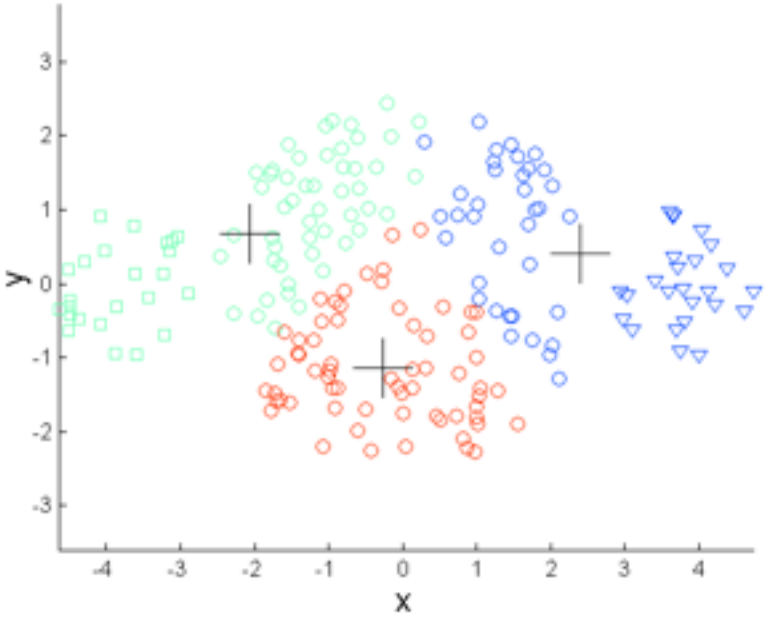
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

- K-means has problems when the data contains outliers.

# Limitations of K-means: Differing Sizes

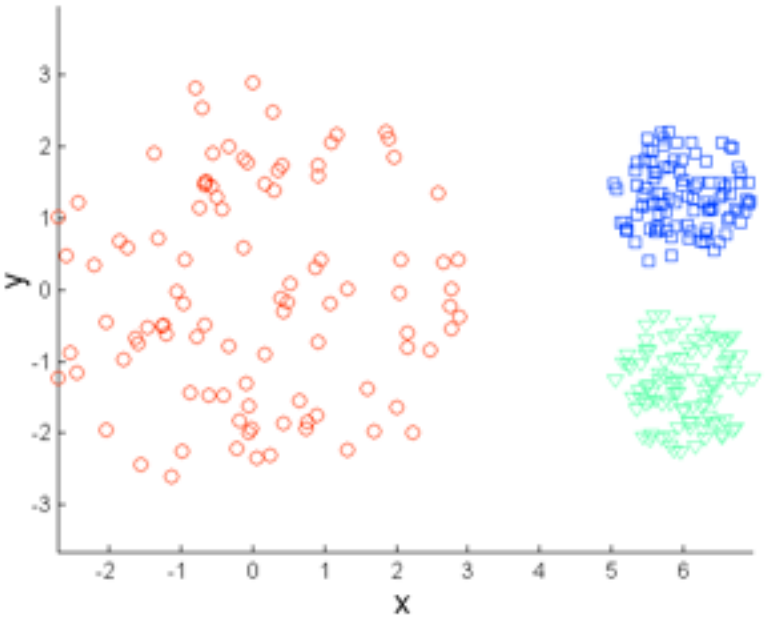



Original Points

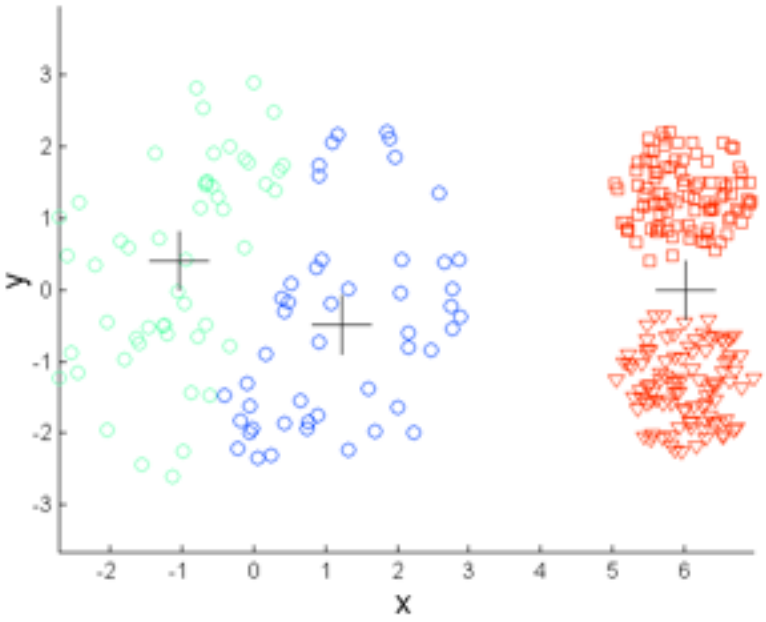



K-means (3 Clusters)

# Limitations of K-means: Differing Density

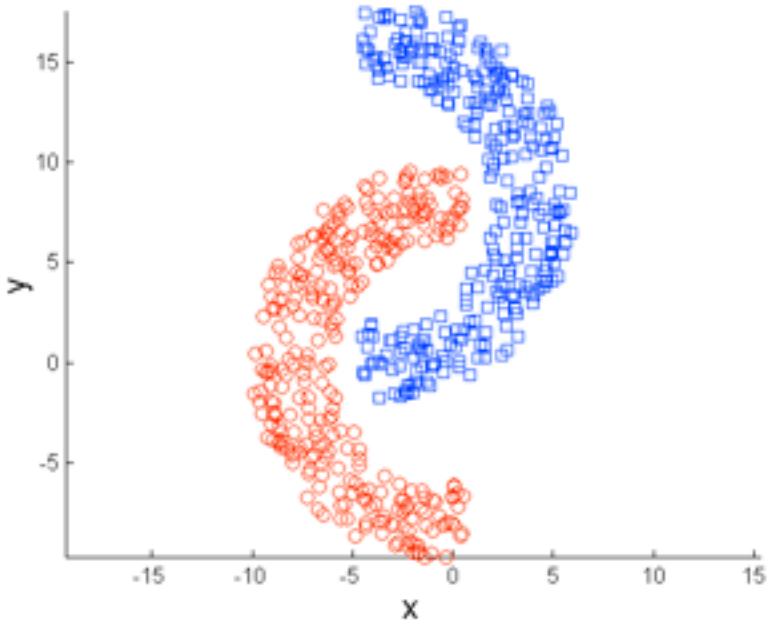



Original Points

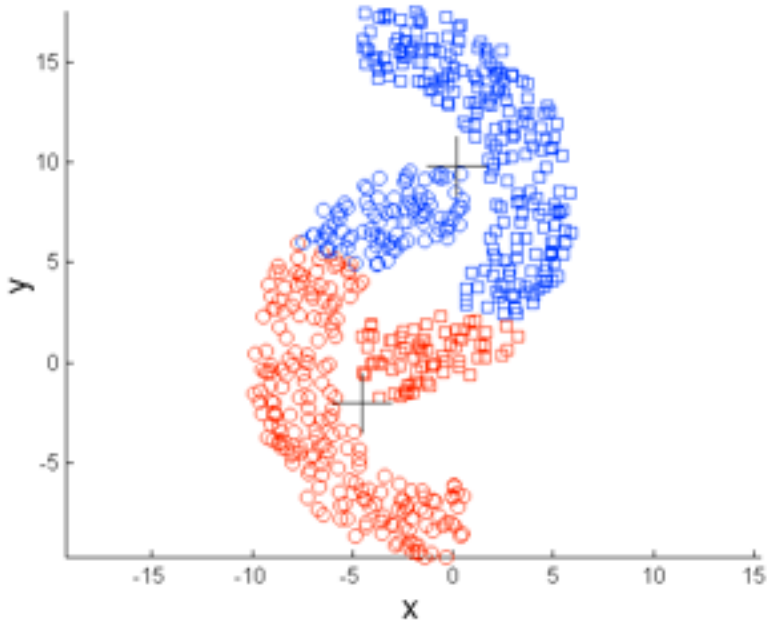



K-means (3 Clusters)

# Limitations of K-means: Non-globular Shapes

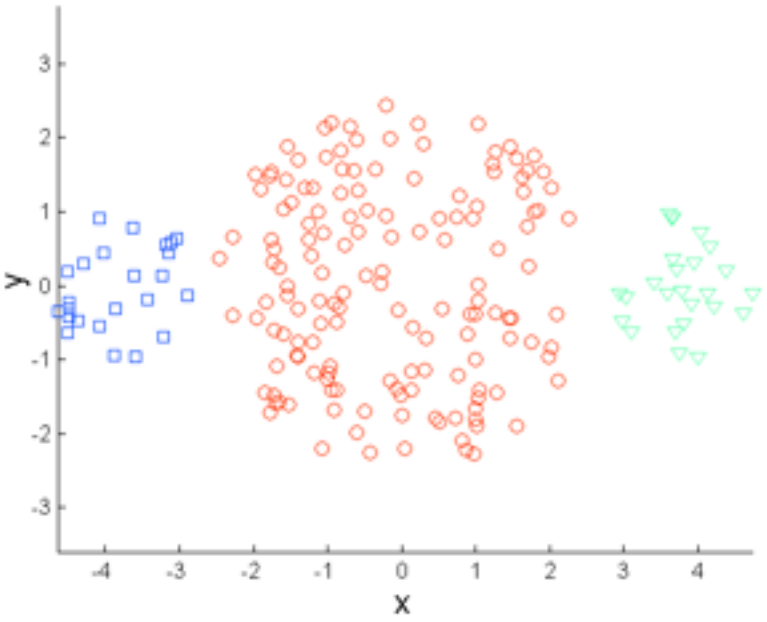



Original Points

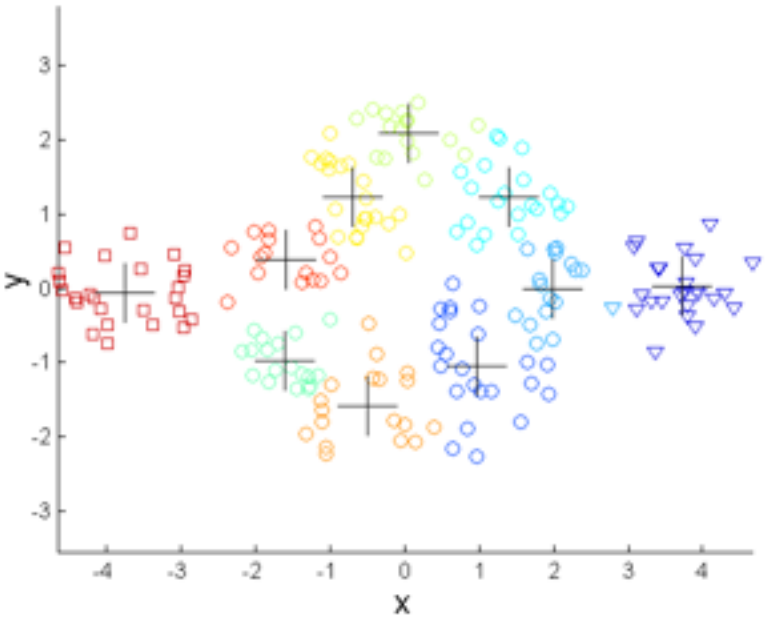



K-means (2 Clusters)

# Overcoming K-means Limitations




Original Points

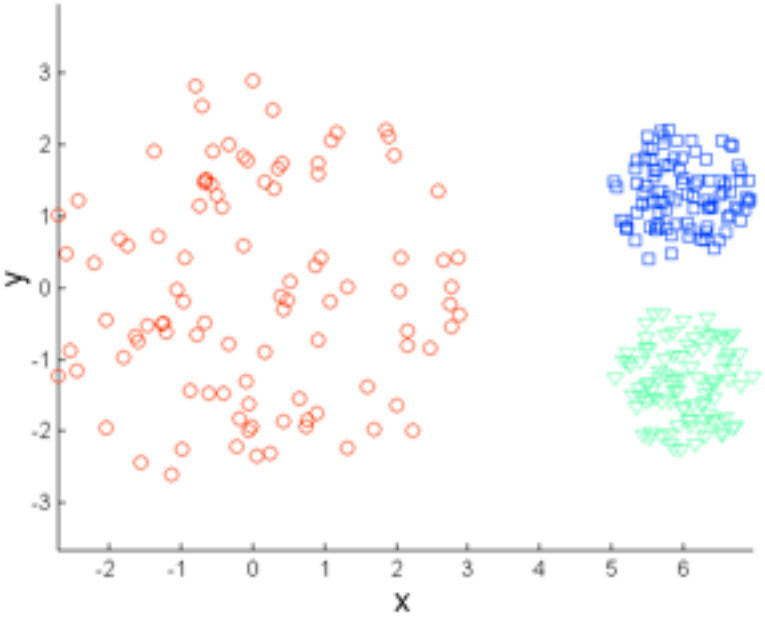



K-means Clusters

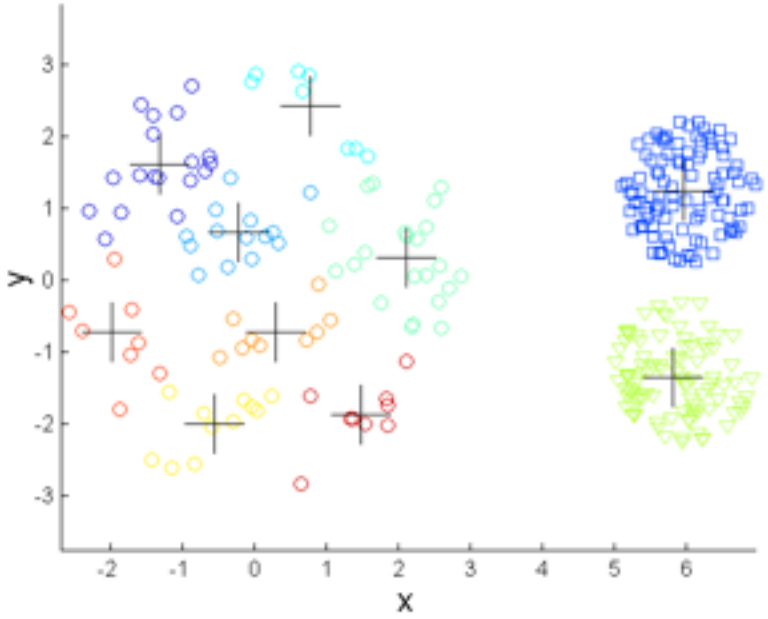
One solution is to use many clusters.

Find parts of clusters, but need to put together.

# Overcoming K-means Limitations

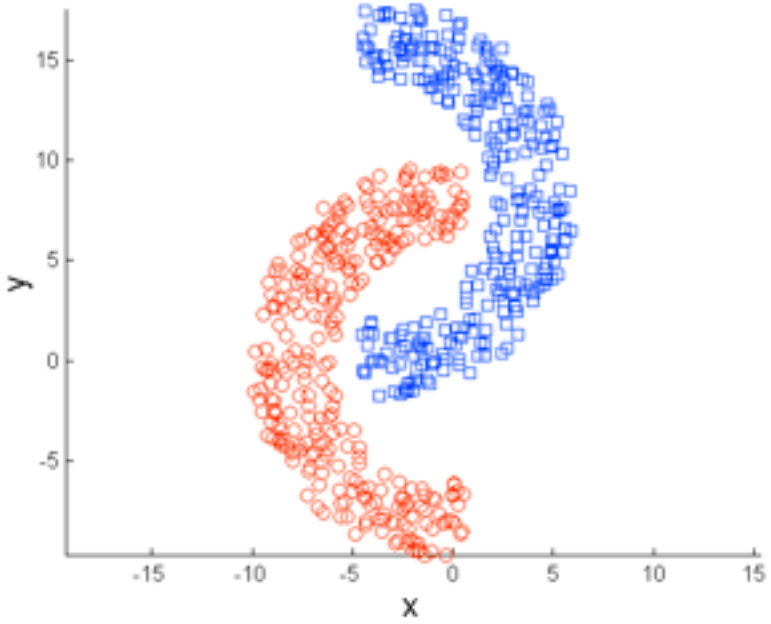



Original Points

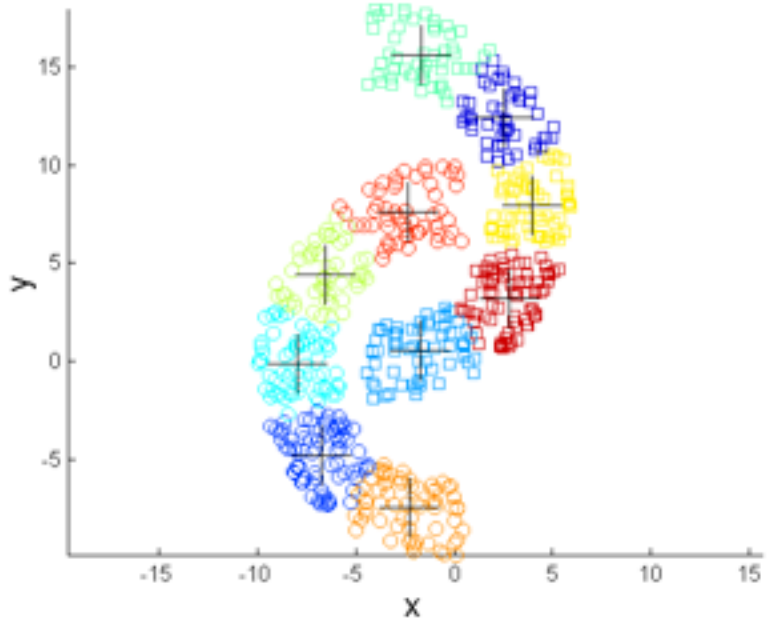



K-means Clusters

# Overcoming K-means Limitations



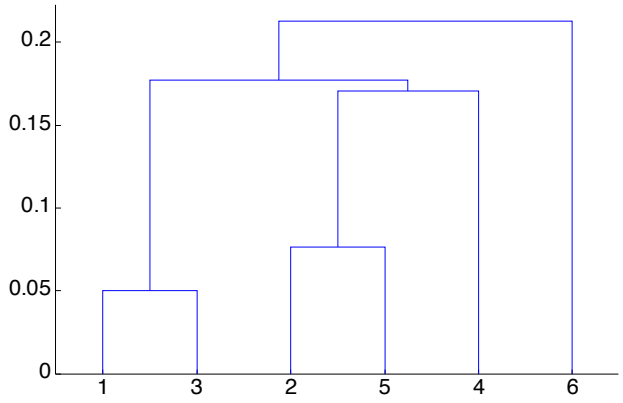

Original Points

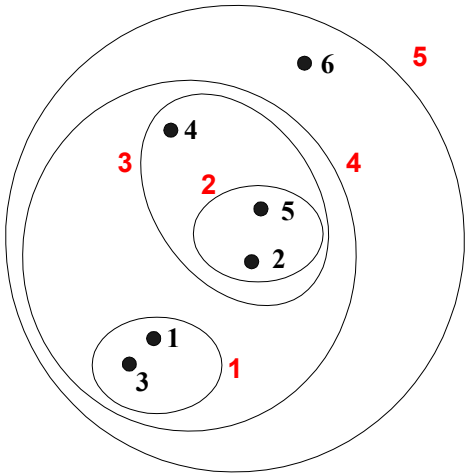



K-means Clusters

# Hierarchical Clustering

- 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 dendogram at the proper level

- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

# Hierarchical Clustering

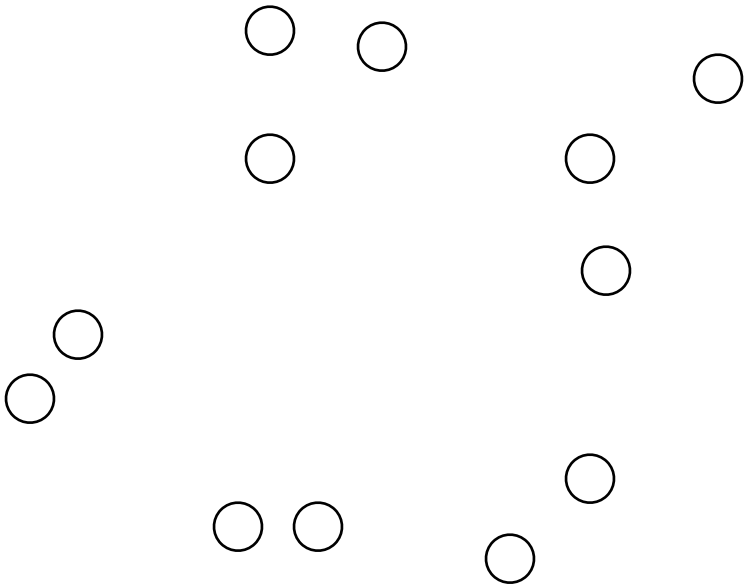
- **Two main types of hierarchical clustering**
  - **Agglomerative:**
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - **Divisive:**
    - Start with one, all-inclusive cluster
    - At each step, 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**

# 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**

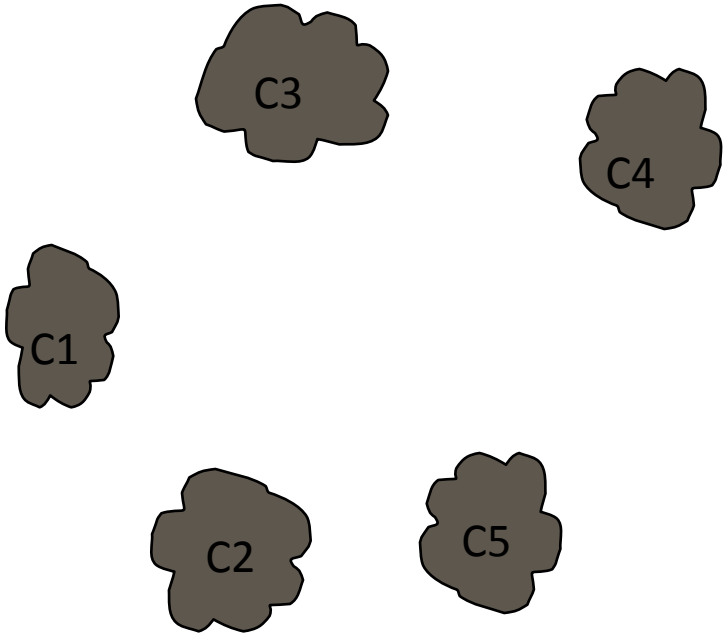


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

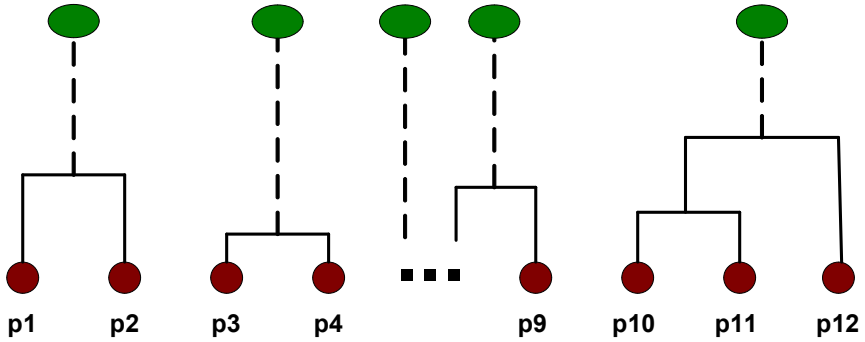
# Intermediate Situation

- **After some merging steps, we have some clusters**



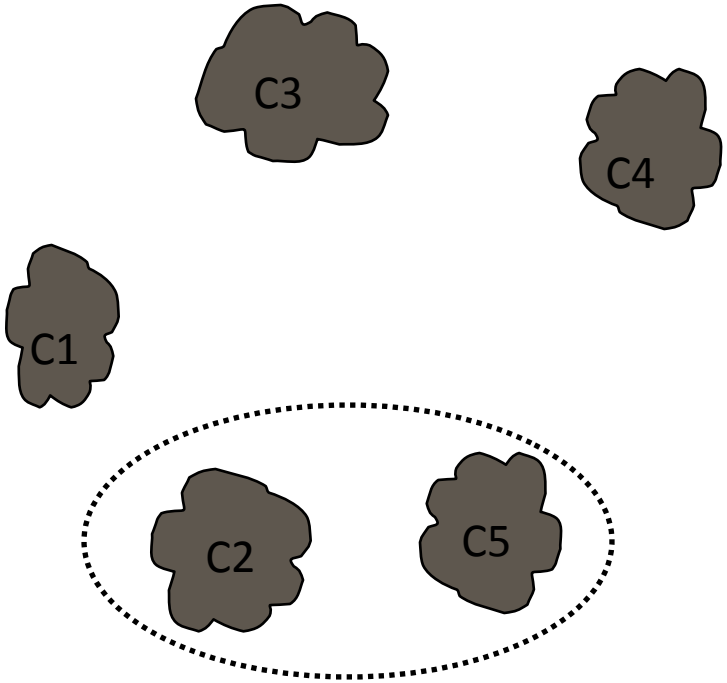

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

Proximity Matrix



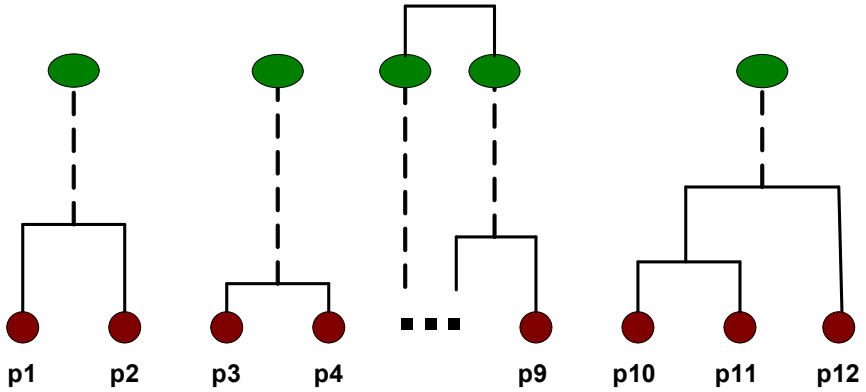
# Intermediate Situation

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



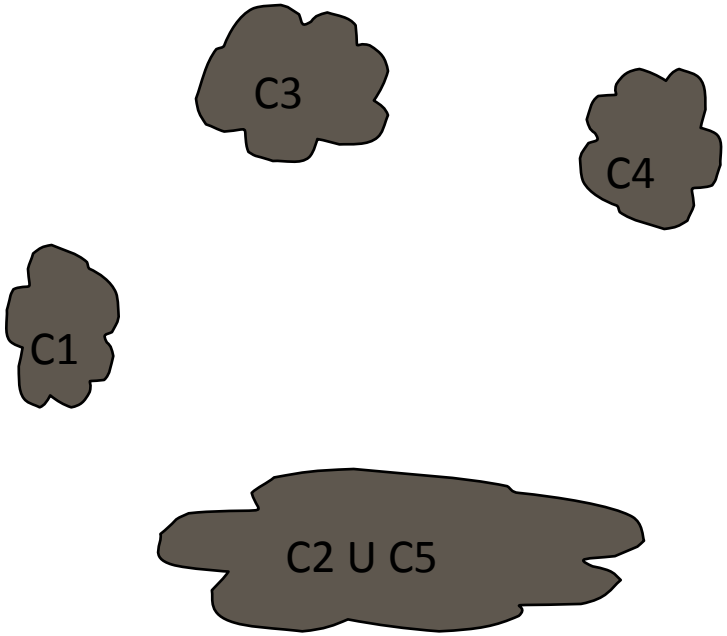

| | C1 | C2 | C3 | C4 | C5 |
|---|---|---|---|---|---|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

Proximity Matrix



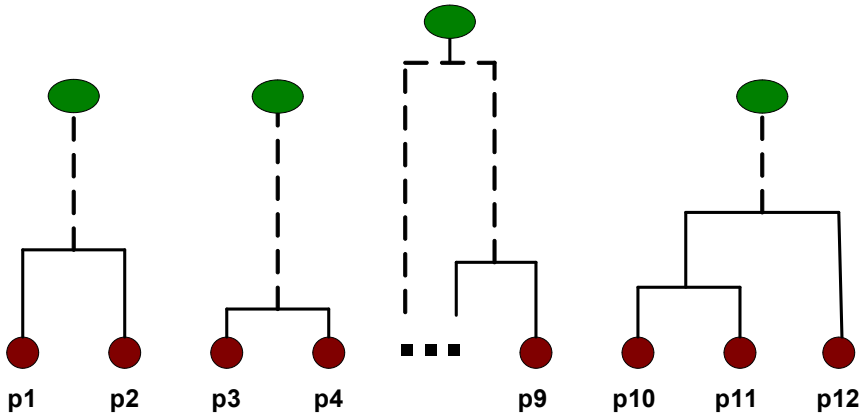
# After Merging

- **The question is "How do we update the proximity matrix?"**

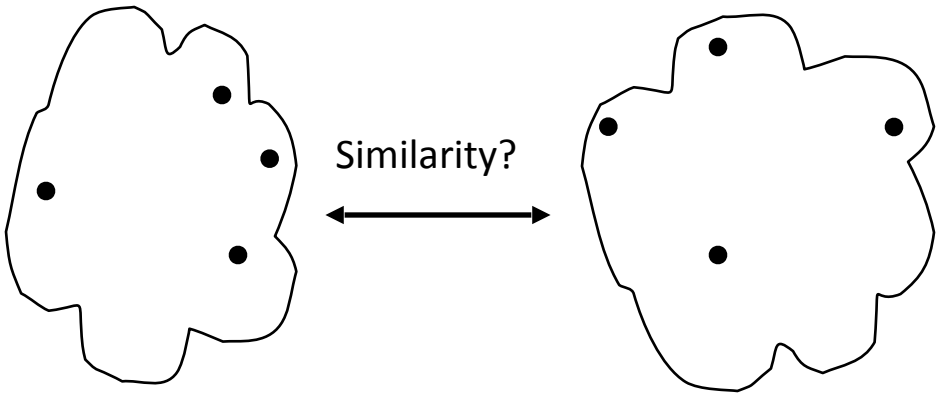



| | C1 | C2 U C5 | C3 | C4 |
|---|---|---|---|---|
| C1 | | ? | | |
| C2 U C5 | ? | ? | ? | ? |
| C3 | | ? | | |
| C4 | | ? | | |

Proximity Matrix



# How to Define Inter-Cluster Similarity

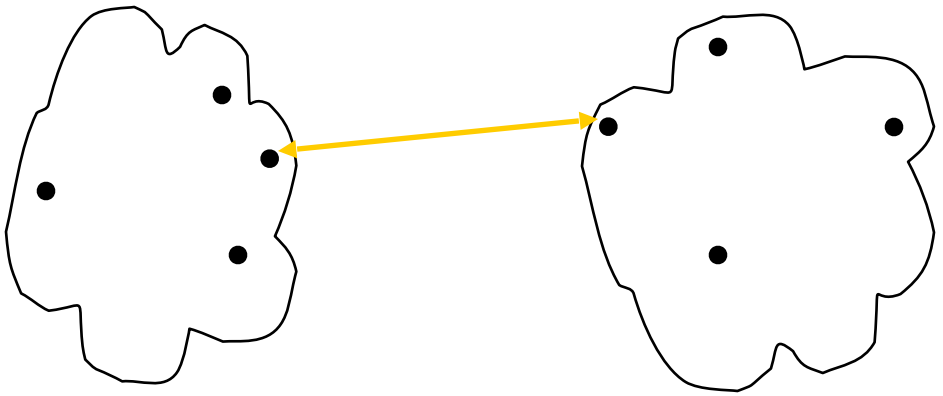



| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

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

# How to Define Inter-Cluster Similarity

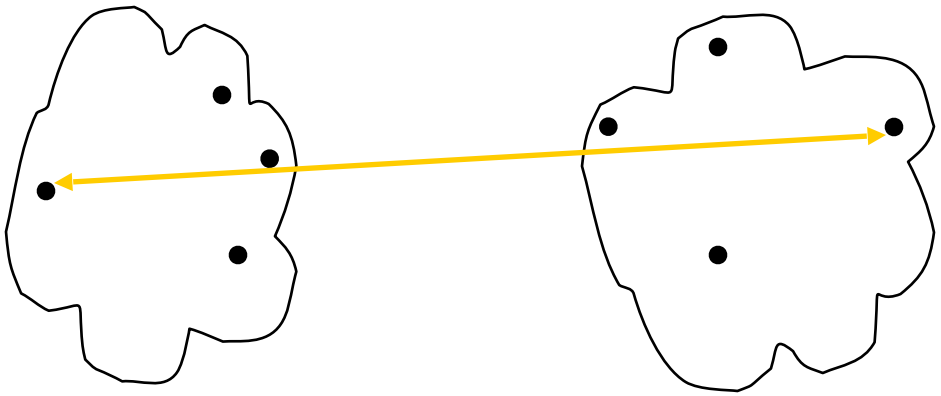


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

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

# How to Define Inter-Cluster Similarity

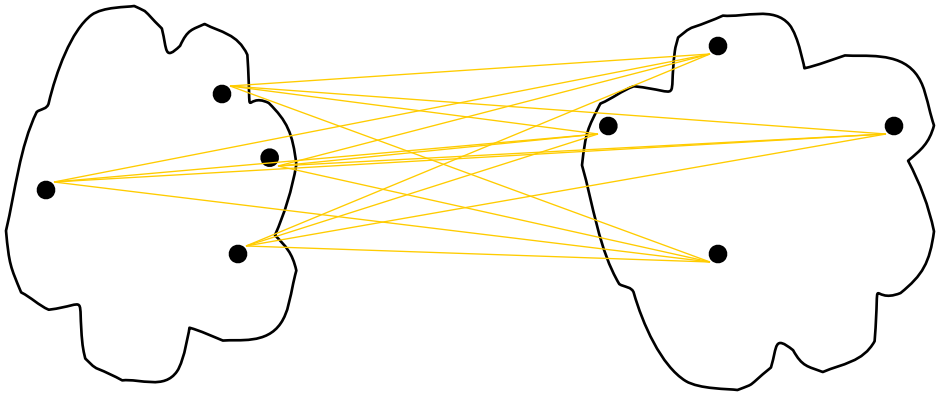


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

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

# How to Define Inter-Cluster Similarity

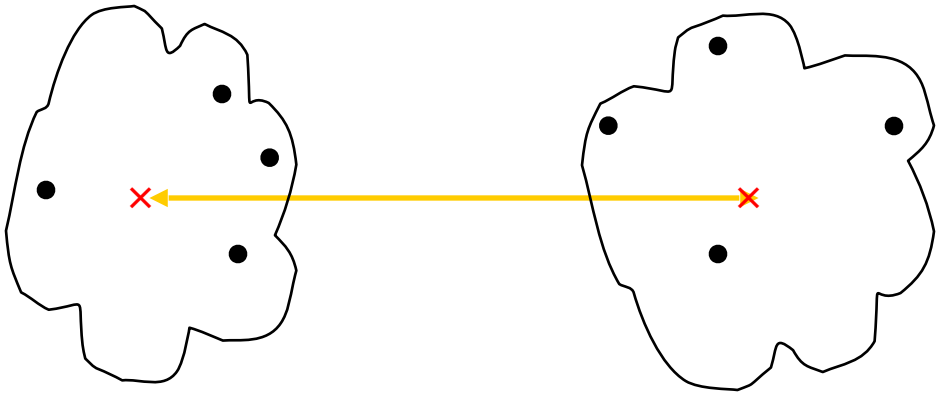


| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

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

# How to Define Inter-Cluster Similarity



| | p1 | p2 | p3 | p4 | p5 | . . . |
|---|---|---|---|---|---|---|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

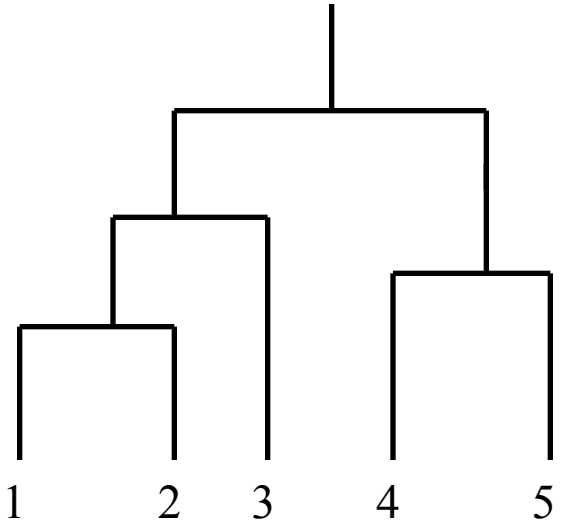
Proximity Matrix

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

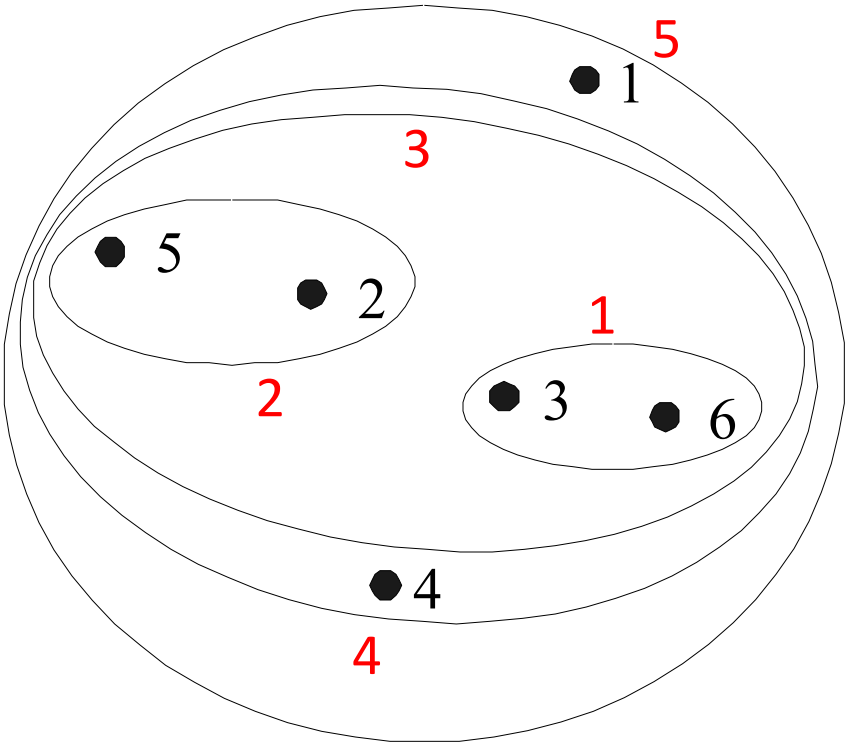
# Cluster Similarity: MIN or Single Link

- **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.**

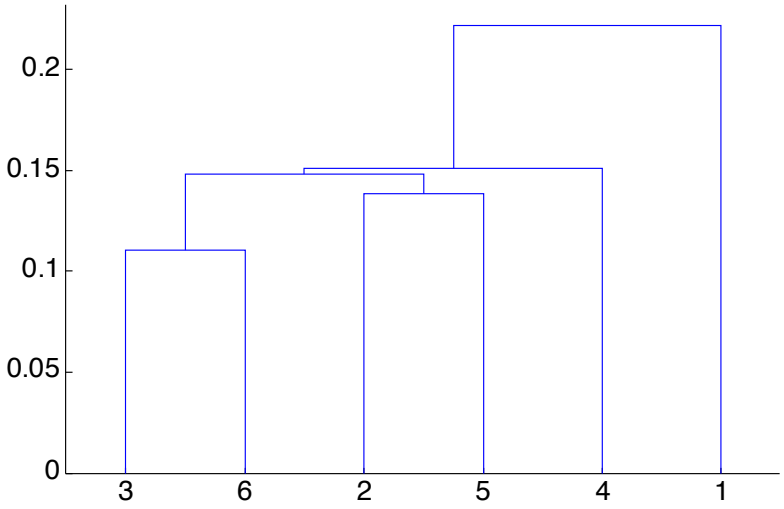
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Hierarchical Clustering: MIN

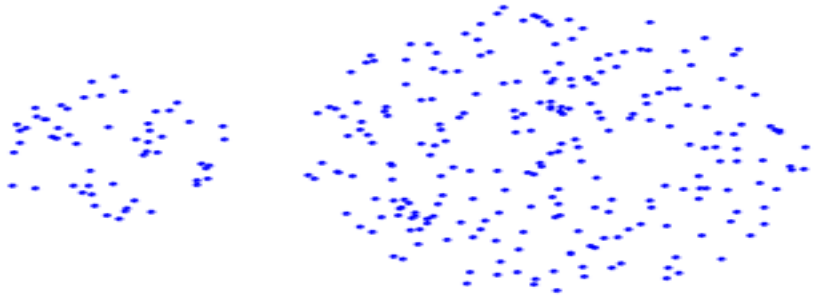



Nested Clusters




Dendrogram

# Strength of MIN

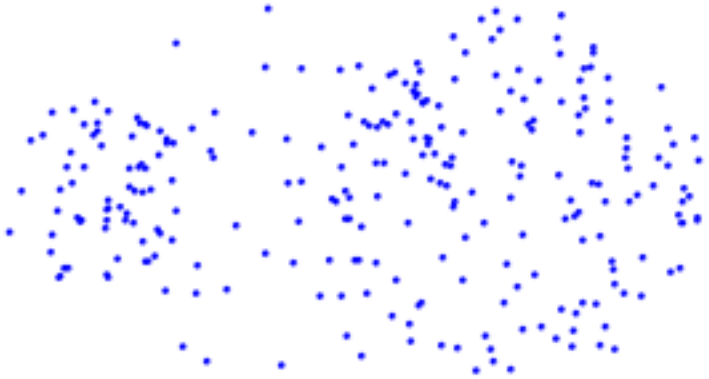


Original Points

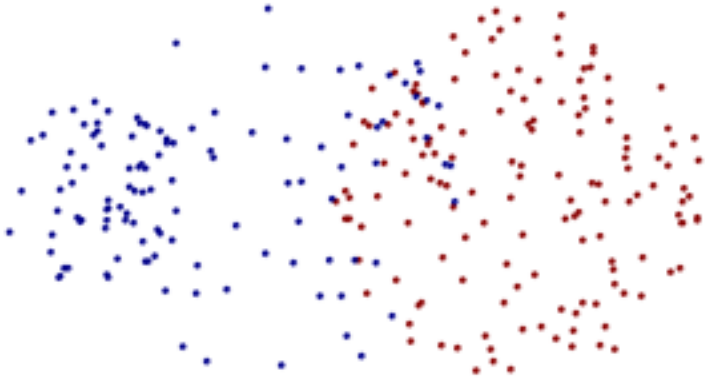
Two Clusters

- Can handle non-elliptical shapes

# Limitations of MIN



Original Points



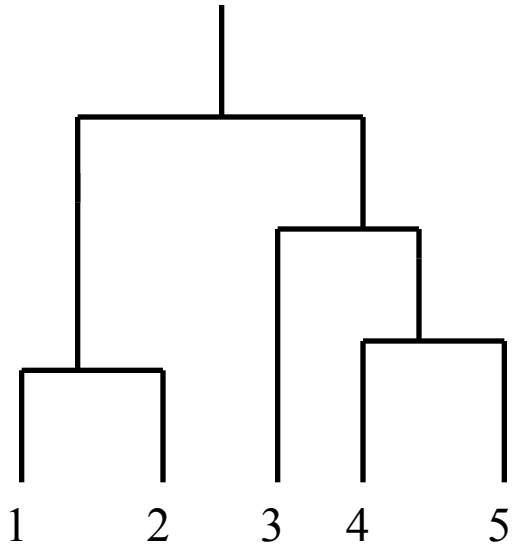
Two Clusters

- Sensitive to noise and outliers

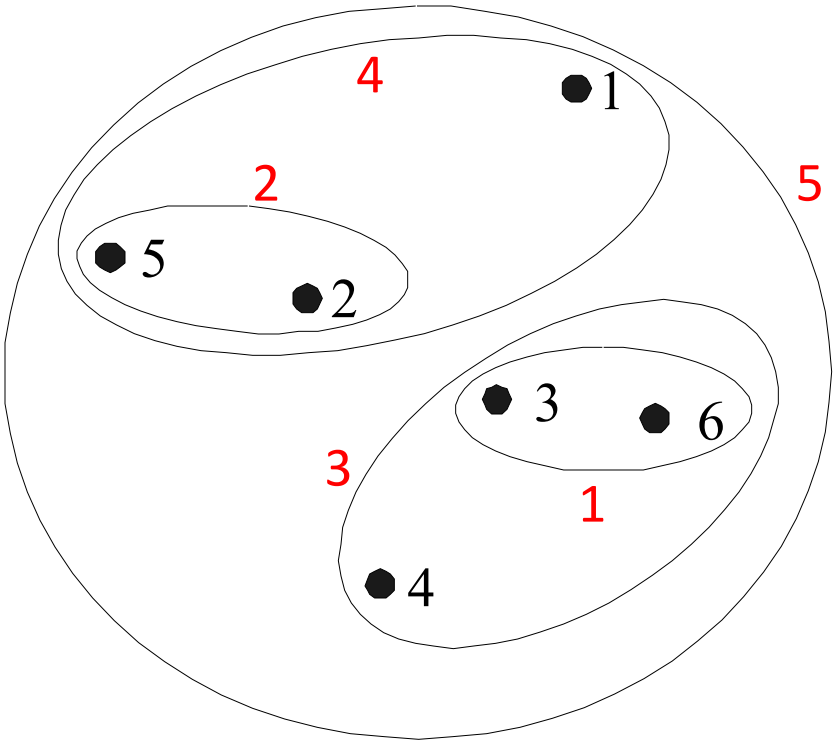
# Cluster Similarity: MAX or Complete Linkage

- **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**

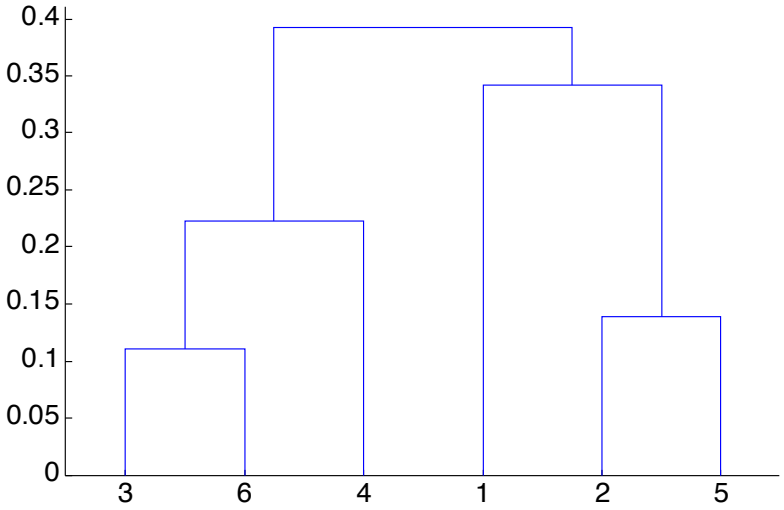
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Hierarchical Clustering: MAX

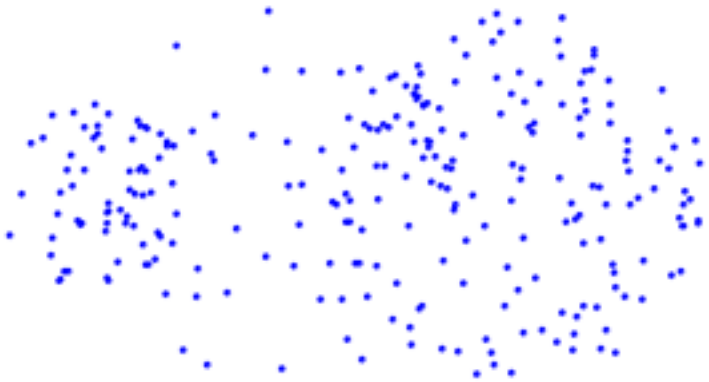



Nested Clusters

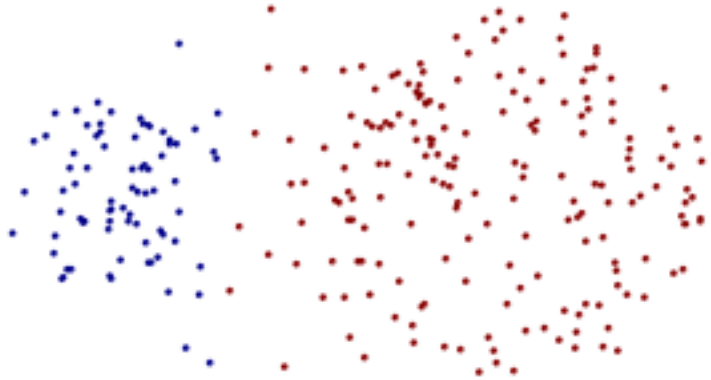



Dendrogram

# Strength of MAX



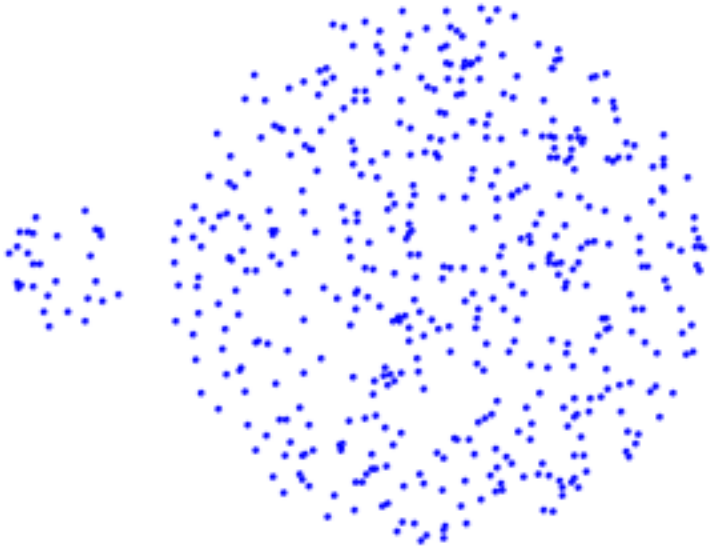
Original Points



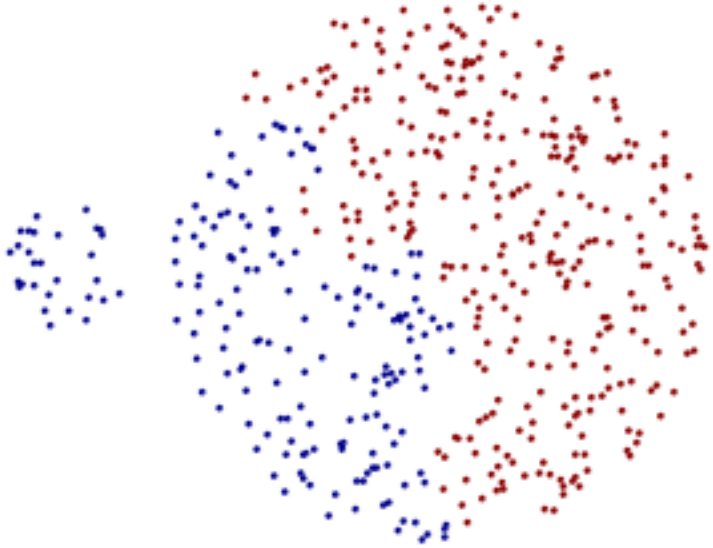
Two Clusters

- Less susceptible to noise and outliers

# Limitations of MAX



Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

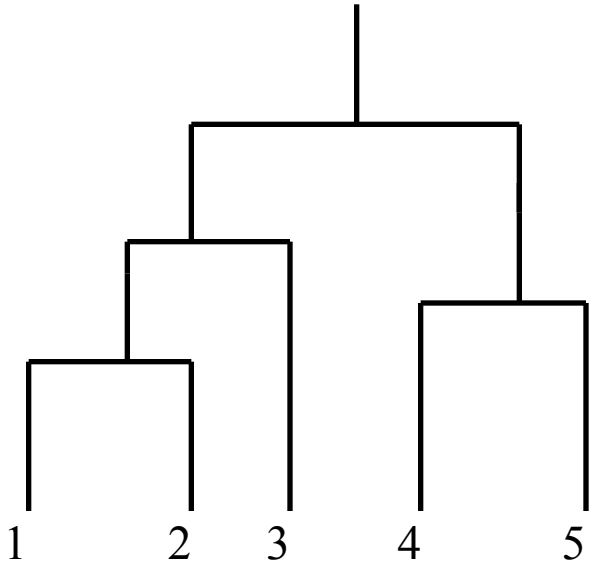
# Cluster Similarity: Group Average

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

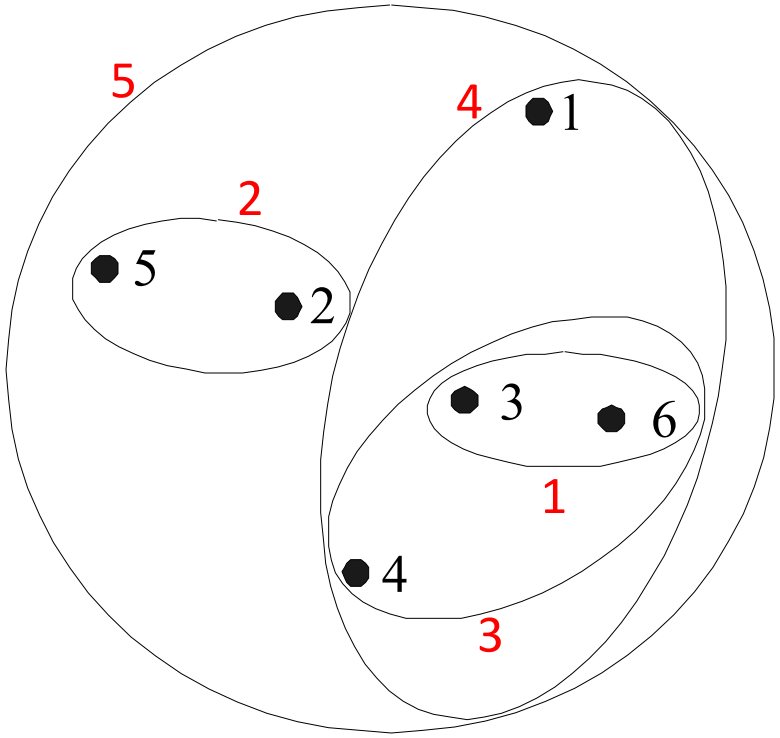
$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

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

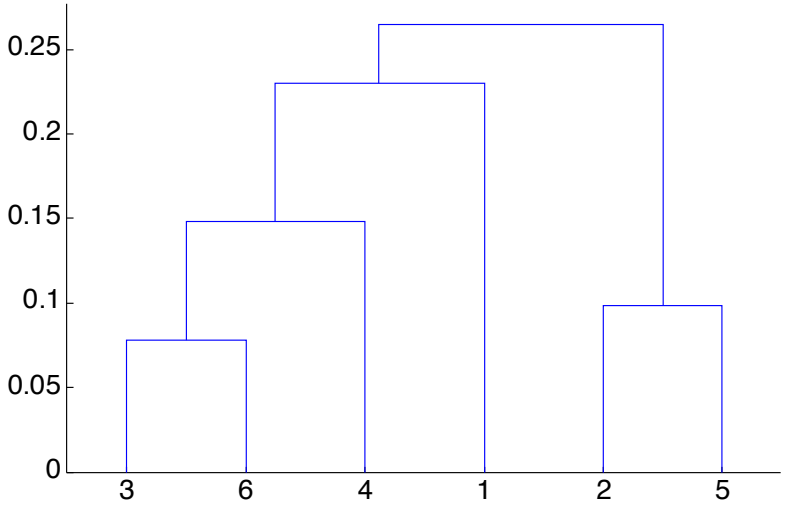
| | I1 | I2 | I3 | I4 | I5 |
|---|---|---|---|---|---|
| I1 | 1.00 | 0.90 | 0.10 | 0.65 | 0.20 |
| I2 | 0.90 | 1.00 | 0.70 | 0.60 | 0.50 |
| I3 | 0.10 | 0.70 | 1.00 | 0.40 | 0.30 |
| I4 | 0.65 | 0.60 | 0.40 | 1.00 | 0.80 |
| I5 | 0.20 | 0.50 | 0.30 | 0.80 | 1.00 |



# Hierarchical Clustering: Group Average




Nested Clusters




Dendrogram

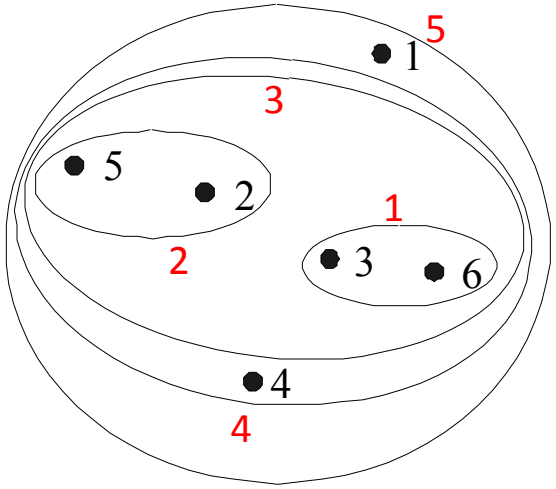
# Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

# Cluster Similarity: Ward's Method

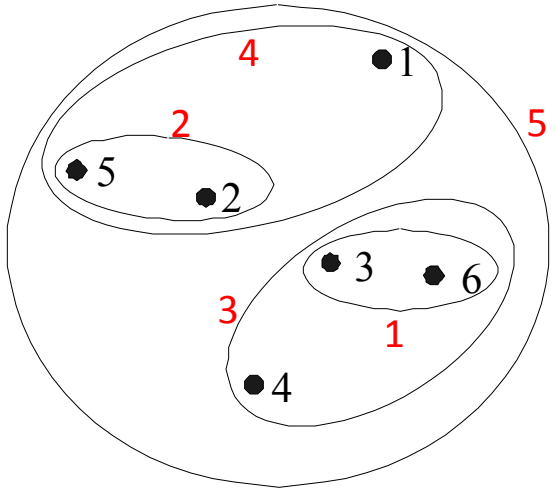
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used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

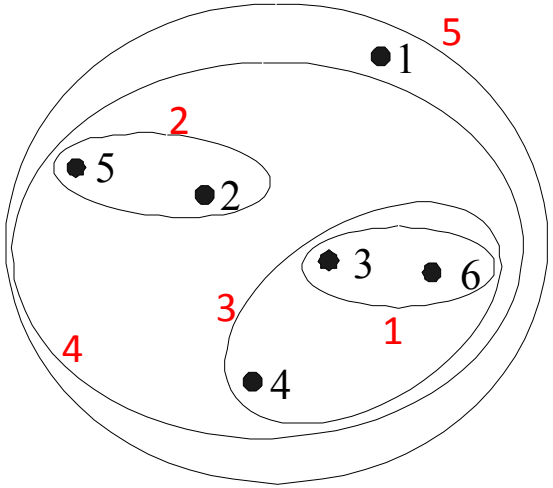
# Hierarchical Clustering: Comparison




MIN

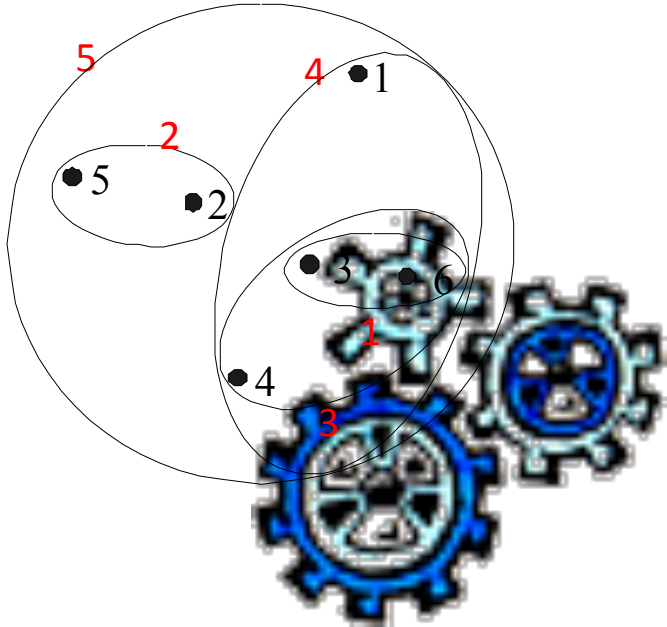
MAX




Group Average

Ward's Method



## Hierarchical Clustering: Time and Space requirements

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

# Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise and outliers
  - Difficulty handling different sized clusters and convex shapes
  - Breaking large clusters

# MST: Divisive Hierarchical Clustering

- **Use MST for constructing hierarchy of clusters**

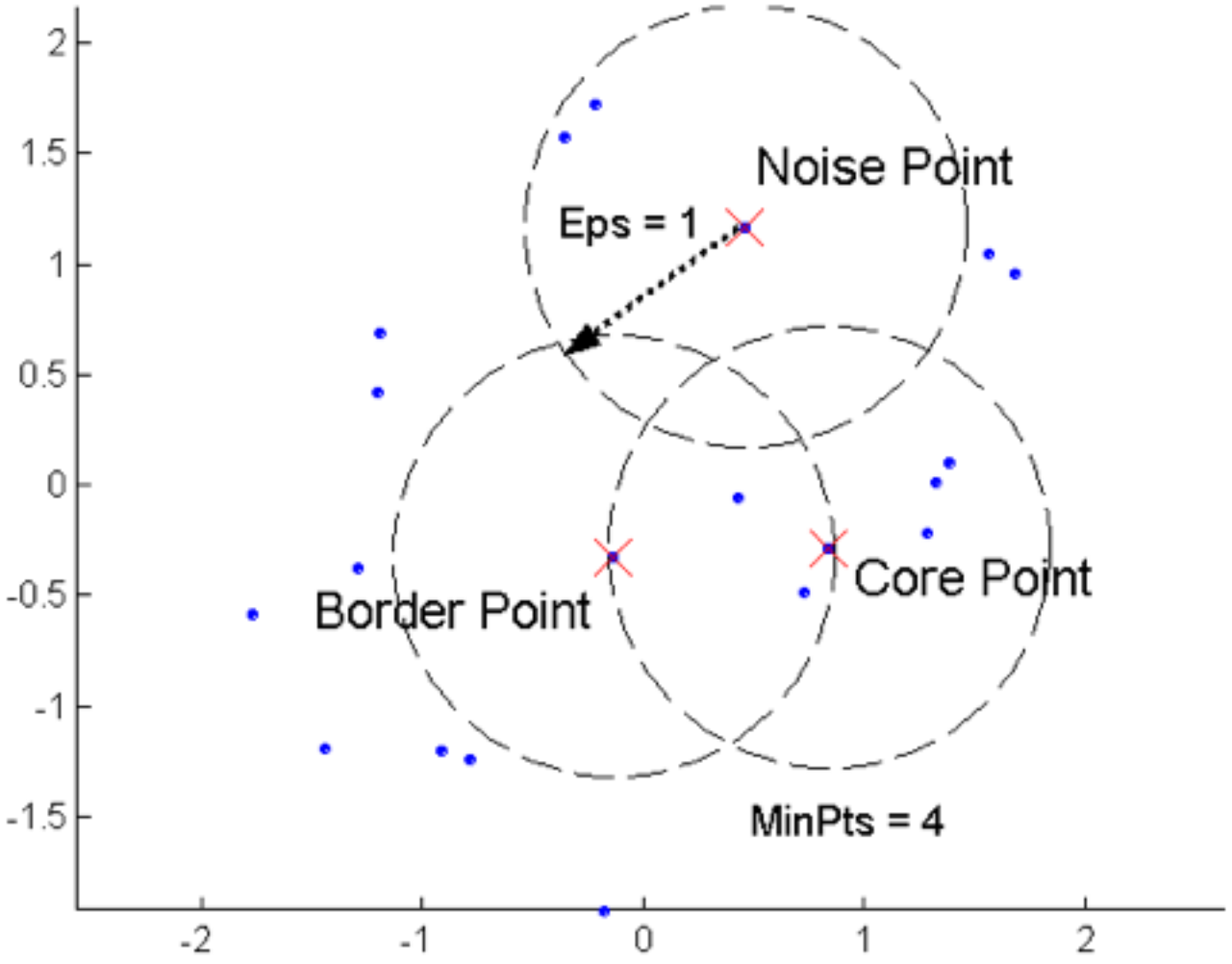
**Algorithm 7.5** MST Divisive Hierarchical Clustering Algorithm

1: Compute a minimum spanning tree for the proximity graph.
2: **repeat**
3: Create a new cluster by breaking the link corresponding to the largest distance (smallest similarity).
4: **until** Only singleton clusters remain

# 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 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 fewer 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



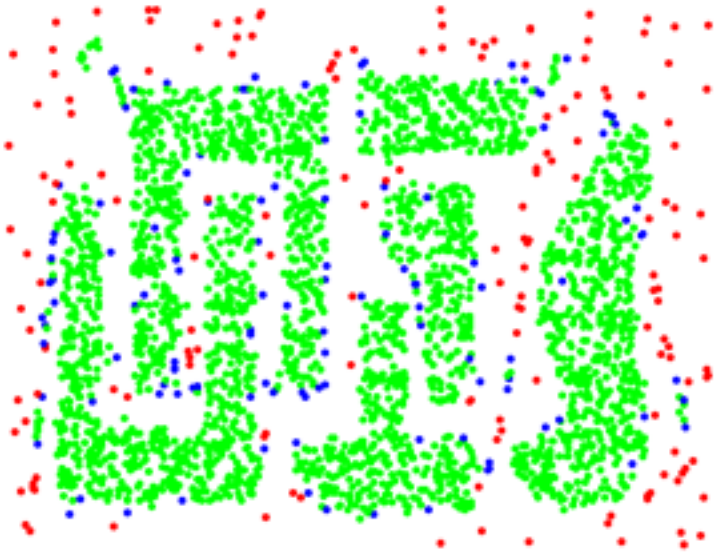
# DBSCAN Algorithm

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

```
current_cluster_label ← 1
for all core points do
  if the core point has no cluster label then
    current_cluster_label ← current_cluster_label + 1
    Label the current core point with cluster label current_cluster_label
  end if
  for all points in the Eps-neighborhood, except i^th the point itself do
    if the point does not have a cluster label then
      Label the point with cluster label current_cluster_label
    end if
  end for
end for
```

# DBSCAN: Core, Border and Noise Points

Original Points

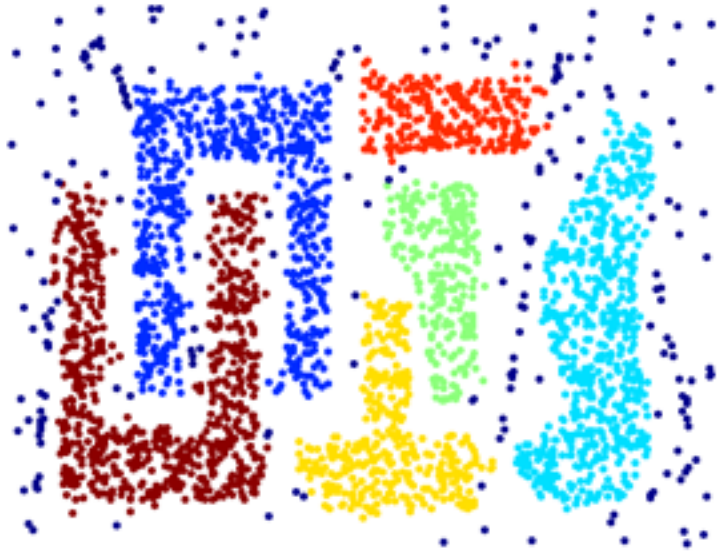


Point types: core, border and noise

Eps = 10, MinPts = 4

# When DBSCAN Works Well

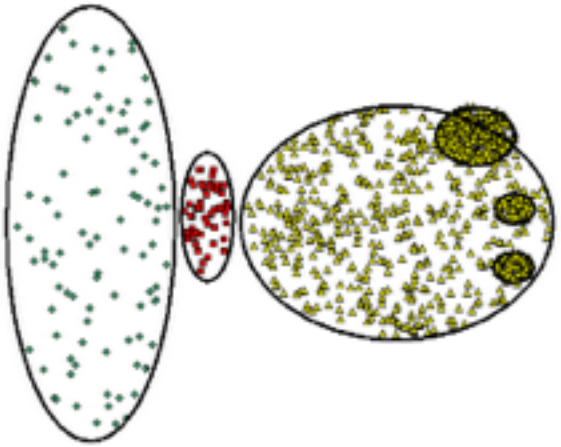
Original Points



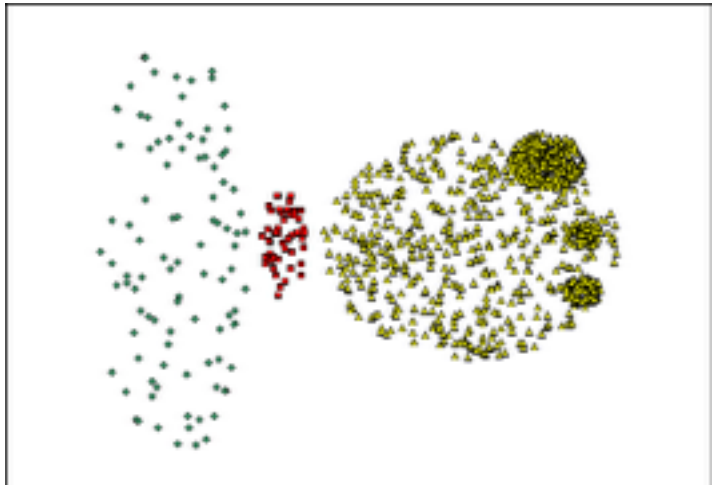
Clusters

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

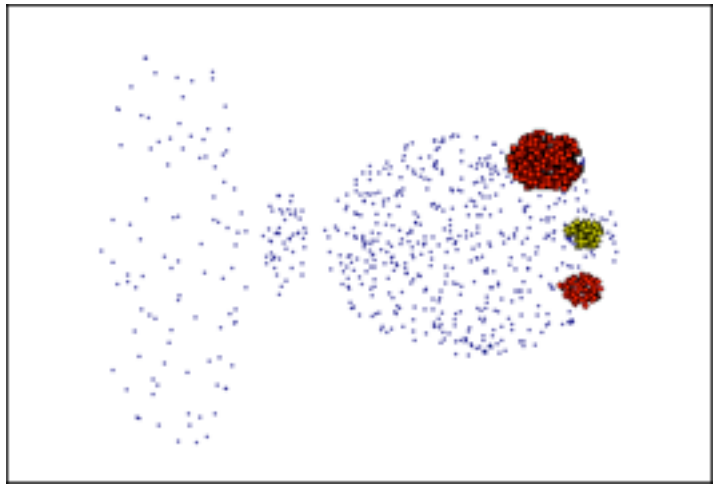
# When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.75).

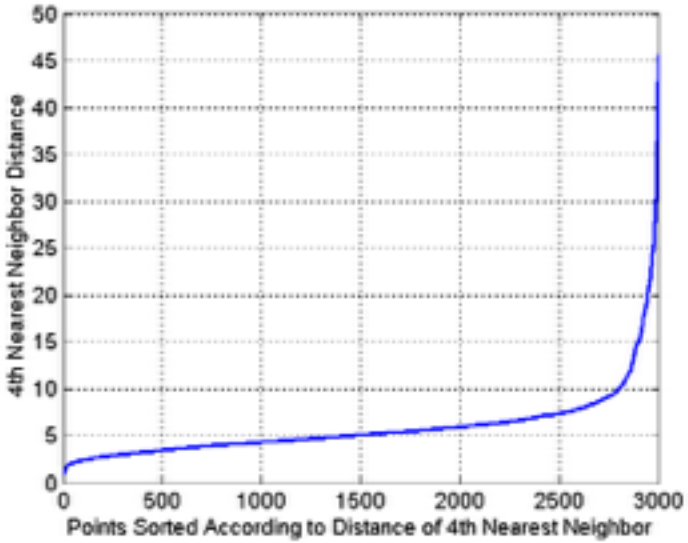


(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

# DBSCAN: Determining EPS and MinPts

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

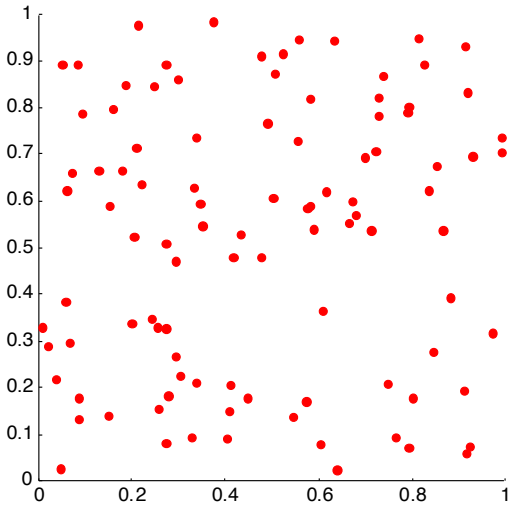


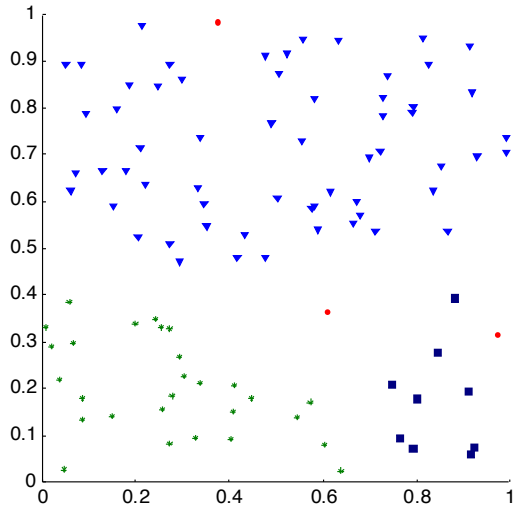
# Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall
- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?
- But "clusters are in the eye of the beholder"!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

# Clusters found in Random Data

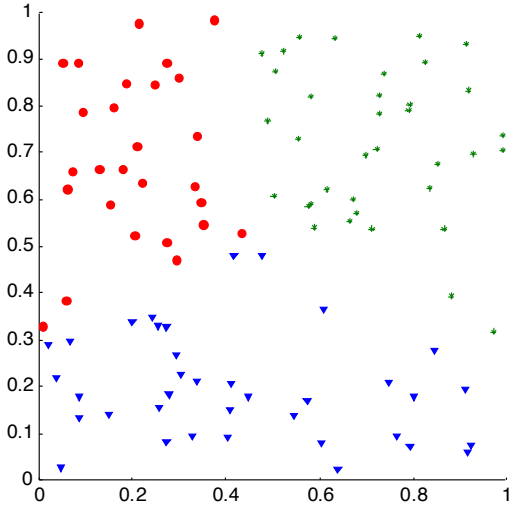
Random Points

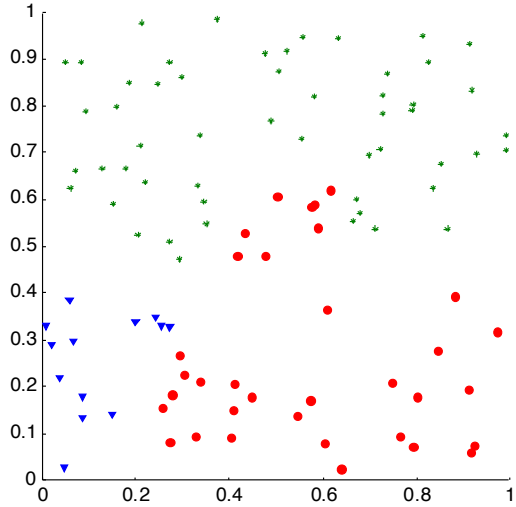



DBSCAN

K-means




Complete Link

# 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.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.

# Measures of Cluster Validity

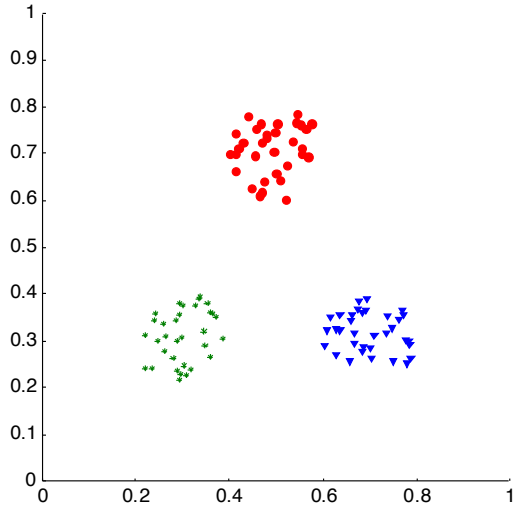
- **Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.**
  - **External Index: Used to measure the extent to which cluster labels match externally supplied class labels.**
    - Entropy
  - **Internal Index: Used to measure 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
- **Sometimes these are referred to as criteria instead of indices**
  - **However, sometimes criterion is the general strategy and index is the numerical measure that implements the criterion.**

# Measuring Cluster Validity Via Correlation

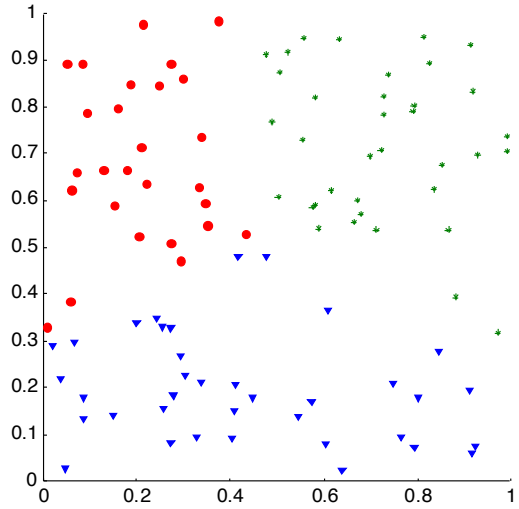
- **Two matrices**
  - **Proximity Matrix**
  - **"Incidence" Matrix**
    - One row and one column for each data point
    - An entry is 1 if the associated pair of points belong to the same cluster
    - An entry is 0 if the associated pair of points belongs to different clusters
- **Compute the correlation between the two matrices**
  - **Since the matrices are symmetric, only the correlation between n(n-1) / 2 entries needs to be calculated.**
- **High correlation indicates that points that belong to the same cluster are close to each other.**
- **Not a good measure for some density or contiguity based clusters.**

# Measuring Cluster Validity Via Correlation

- **Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets**



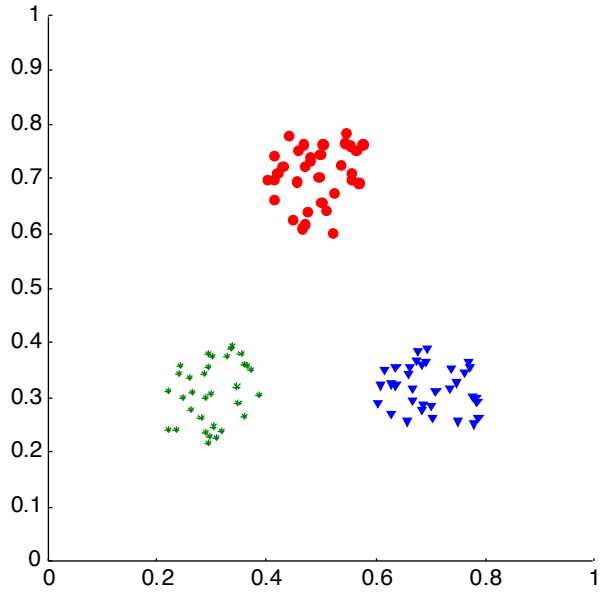

Corr = -0.9235

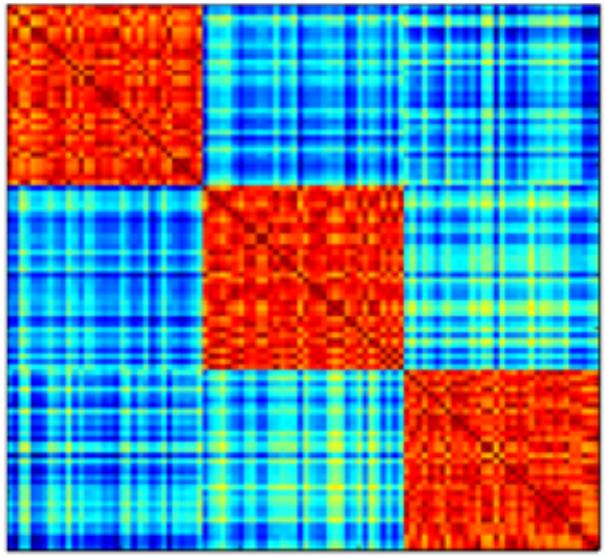



Corr = -0.5810

# Using Similarity Matrix for Cluster Validation

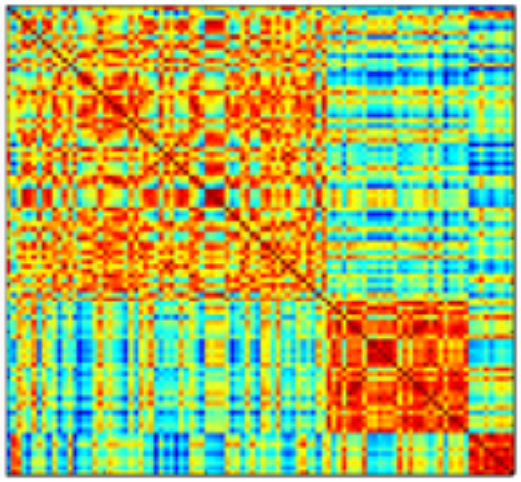
- **Order the similarity matrix with respect to cluster labels and inspect visually.**

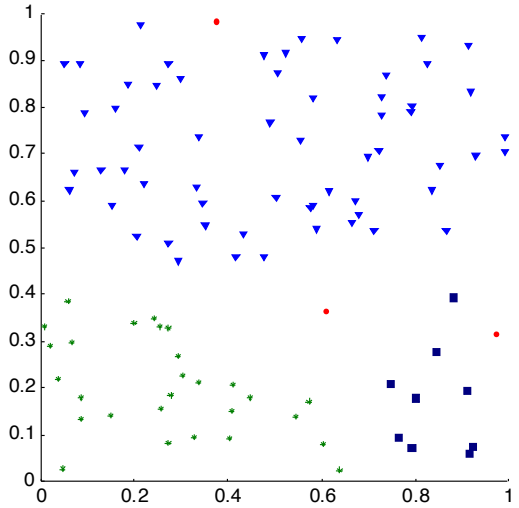


# Using Similarity Matrix for Cluster Validation

- **Clusters in random data are not so crisp**

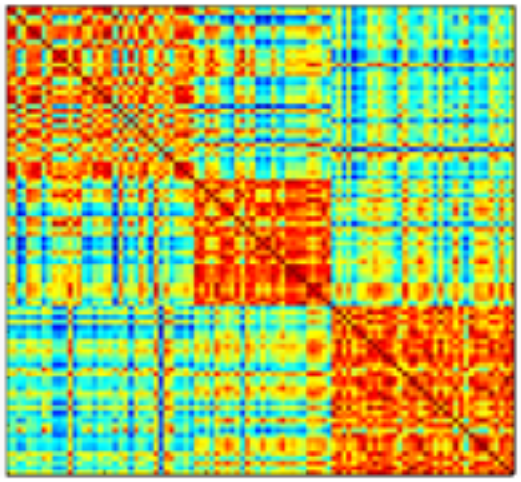


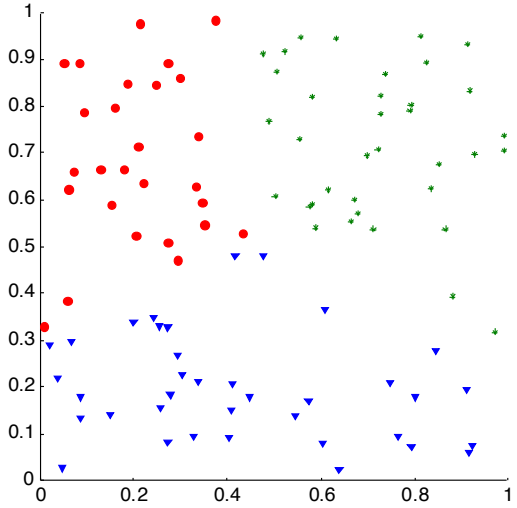



DBSCAN

# Using Similarity Matrix for Cluster Validation

- **Clusters in random data are not so crisp**

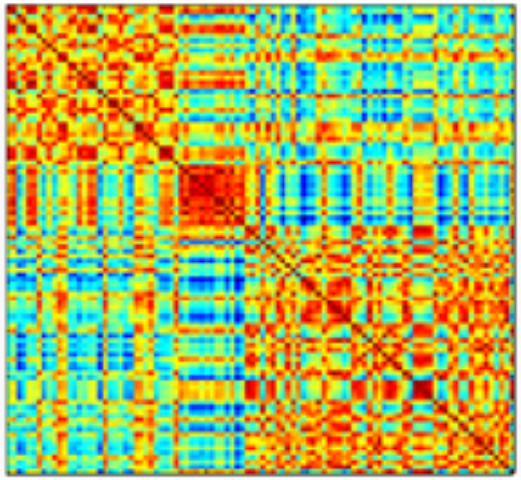


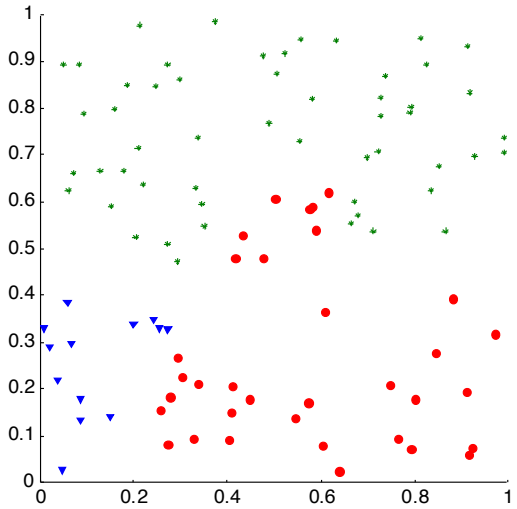



K-means

# Using Similarity Matrix for Cluster Validation

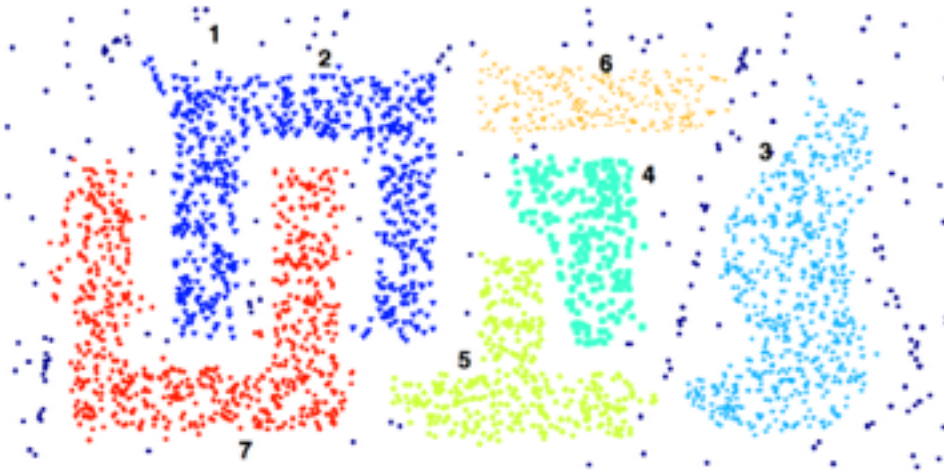
- **Clusters in random data are not so crisp**

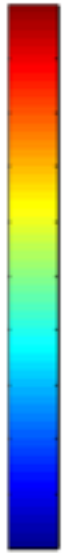





Complete Link

# Using Similarity Matrix for Cluster Validation

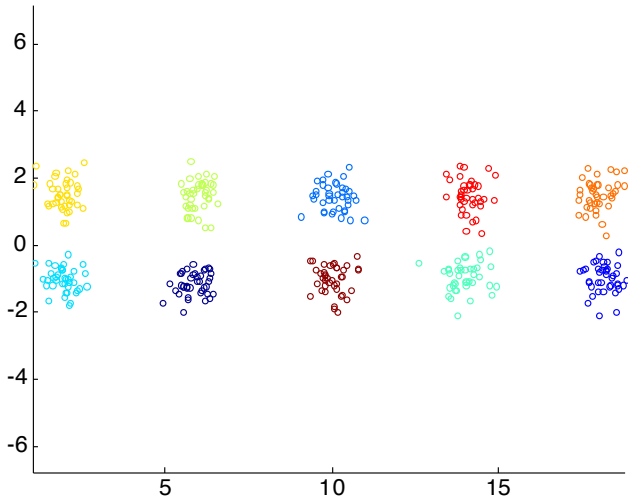


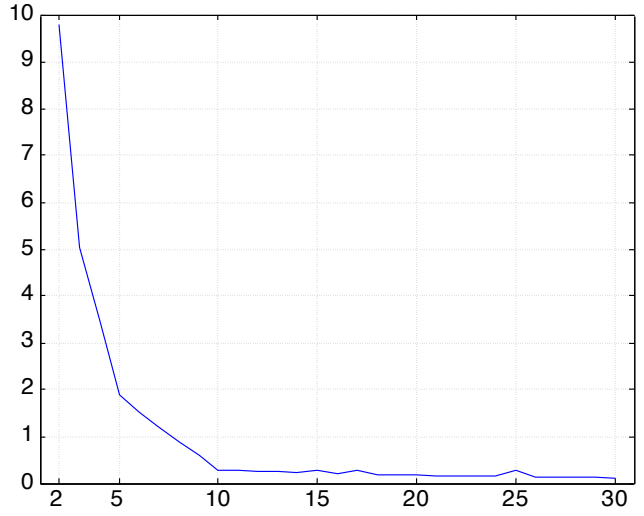



DBSCAN

# Internal Measures: SSE

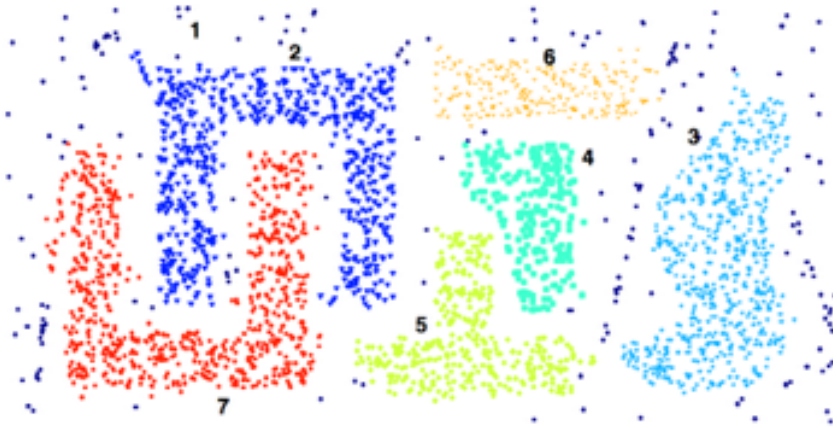
- Clusters in more complicated figures aren't well separated
- Internal Index: Used to measure the goodness of a clustering structure without respect to external information
  - SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of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

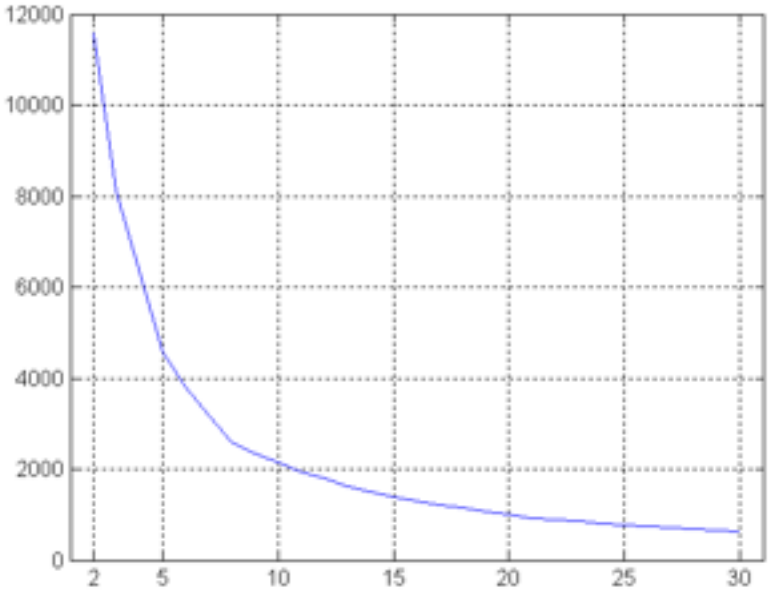


# Internal Measures: SSE

- SSE curve for a more complicated data set




SSE of clusters found using K-means

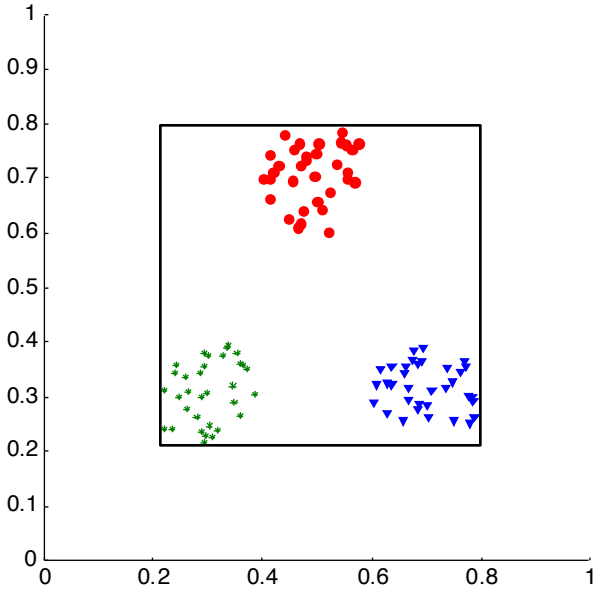
# Framework for Cluster Validity

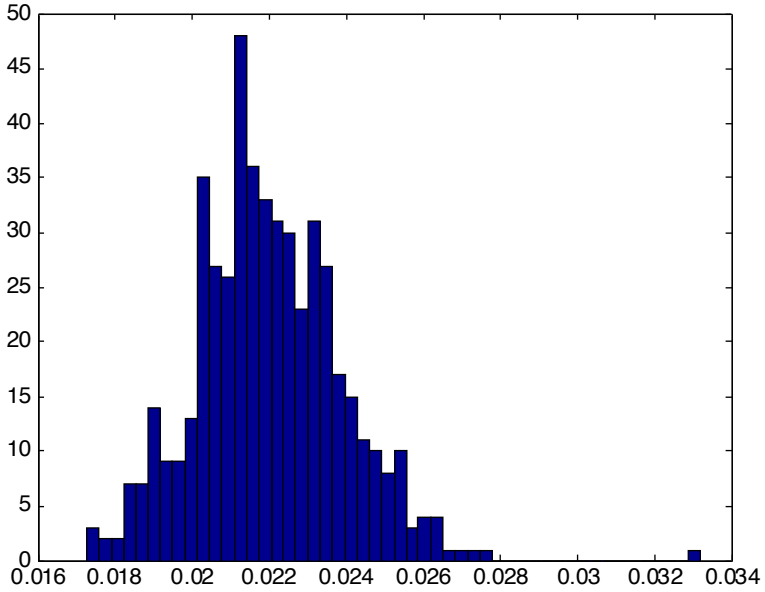
- **Need a framework to interpret any measure.**
  - **For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?**
- **Statistics provide a framework for cluster validity**
  - **The more "atypical" a clustering result is, the more likely it represents valid structure in the data**
  - **Can compare the values of an index that result from random data or clusterings to those of a clustering result.**
    - If the value of the index is unlikely, then the cluster results are valid
  - **These approaches are more complicated and harder to understand.**
- **For comparing the results of two different sets of cluster analyses, a framework is less necessary.**
  - **However, there is the question of whether the difference between two index values is significant**

# Statistical Framework for SSE

## Example

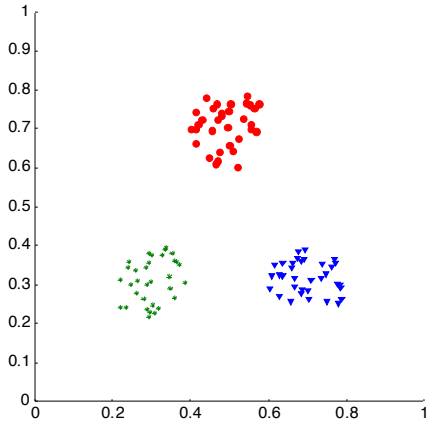
- Compare SSE of 0.005 against three clusters in random data
- Histogram shows SSE of three clusters in 500 sets of random data points of size 100 distributed over the range 0.2 – 0.8 for x and y values

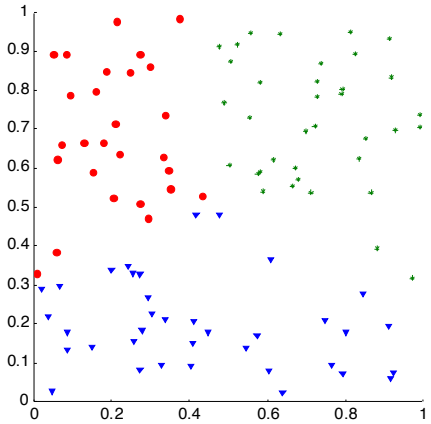


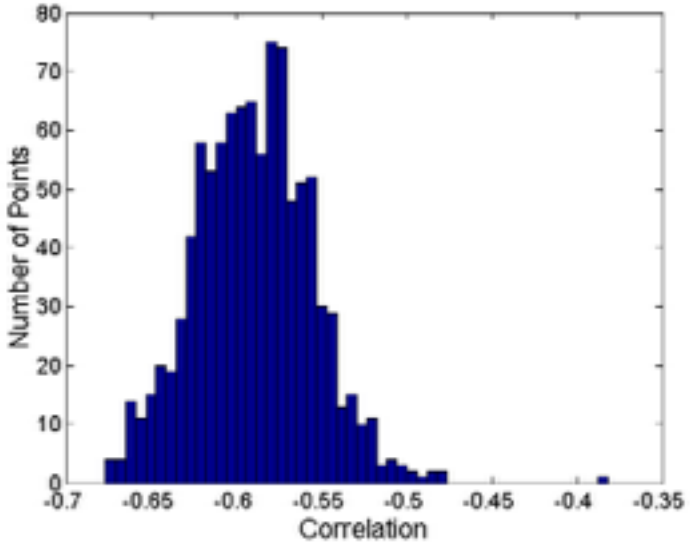



# Statistical Framework for Correlation

- **Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets.**




Corr = -0.9235

Corr = -0.5810

# 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)**

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

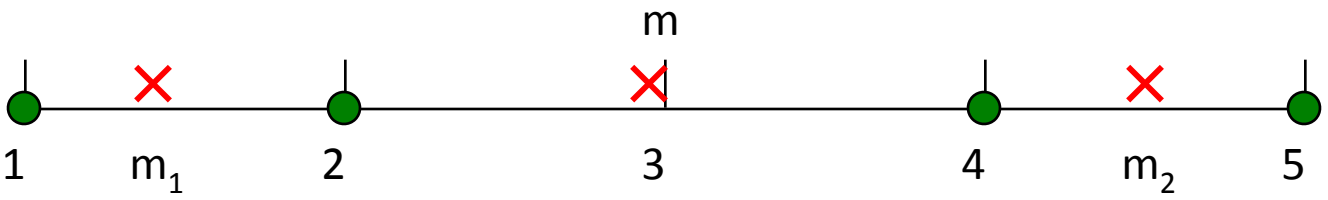
  - **Separation is measured by the between cluster sum of squares**

$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

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

# Internal Measures: Cohesion and Separation

- **Example: SSE**
  - **BSS + WSS = constant**




K=1 cluster:

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

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

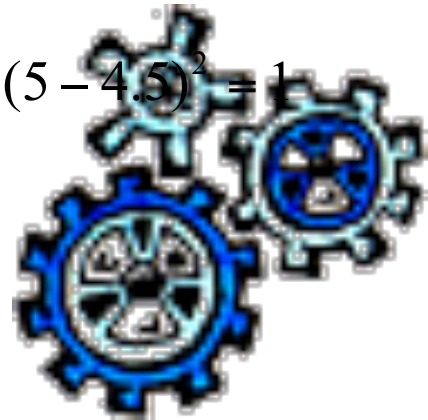
$$Total = 10 + 0 = 10$$

K=2 clusters:

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

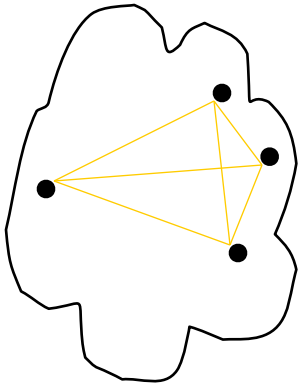
$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

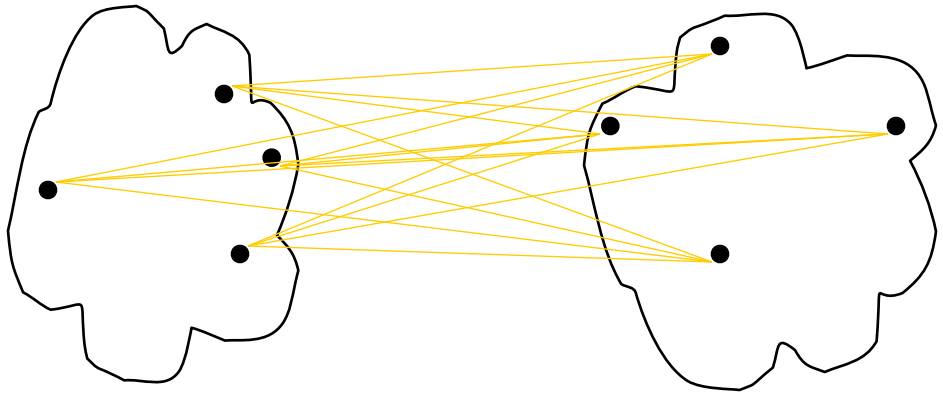


# Internal Measures: Cohesion and Separation

- **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



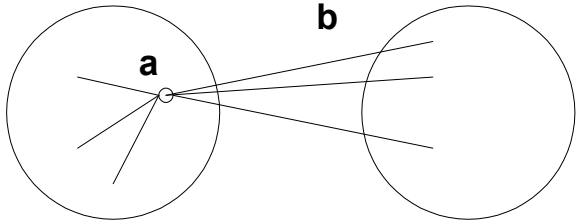
separation

# Internal Measures: Silhouette Coefficient

- Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points, 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

    $s = 1 - a/b$ if $a < b$, (or $s = b/a - 1$ if $a \geq b$, not the usual case)

  - Typically between 0 and 1.
  - The closer to 1 the better.




- Can calculate the Average Silhouette width for a cluster or a clustering

# 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., $e = \sum_{i=1}^{K} \frac{m_i}{m} e_j$, 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 $purity_j = \max p_{ij}$ and the overall purity of a clustering by $purity = \sum_{i=1}^{K} \frac{m_i}{m} purity_j$.

# 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**