OntarioTech Science

Kourosh Davoudi
kourosh@uoit.ca

Cluster Analysis: Basic Ideas and Concepts

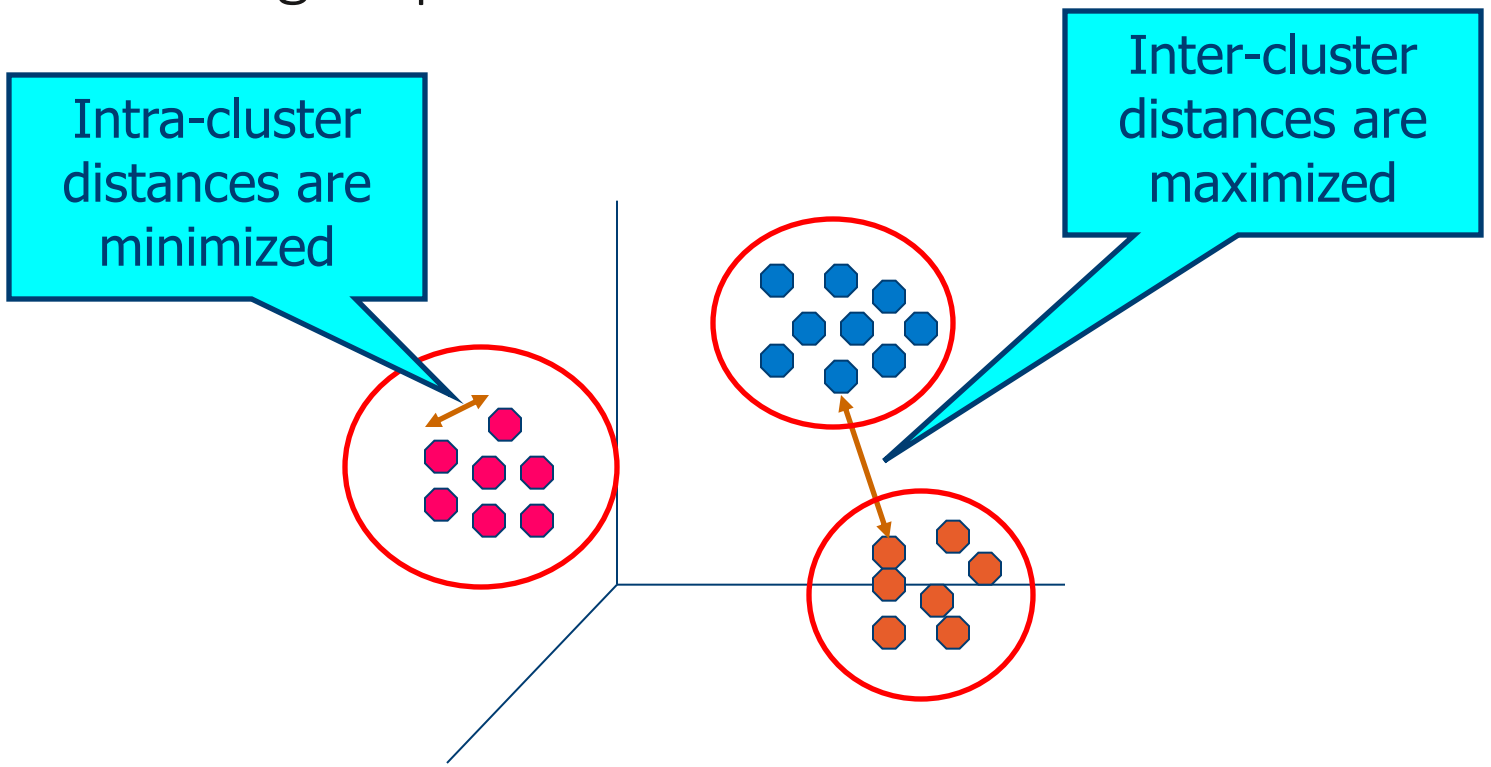
CSCI 4150U: Data Mining

Learning Outcomes

- Understand the basic concepts in clustering
- Explain different clustering approaches
  - Center-based Clustering
  - Hierarchical Clustering
  - Density Based Clustering
- Clustering Validation

What is Cluster Analysis?

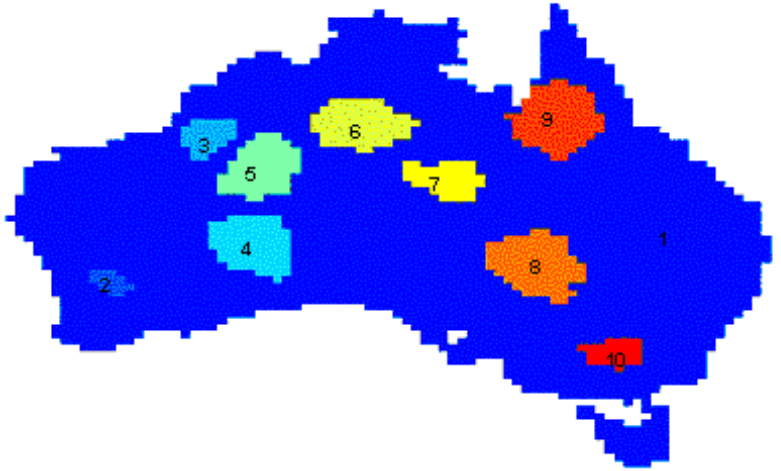
- Finding **groups of objects** such that the objects in a group will be **similar** (or related) to one another and **different** from (or unrelated to) the objects in other groups



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

- Summarization
  - Reduce the size of large data sets



Clustering precipitation in Australia

What is not Cluster Analysis?

- Simple segmentation
  - Dividing students into different registration groups alphabetically, by last name
- Results of a query
  - Groupings are a result of an external specification
  - Clustering is a grouping of objects based on similarity in the data
- Supervised classification
  - Have class label information

Grouping students by their GPAs is a clustering task

A. True
B. False

Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

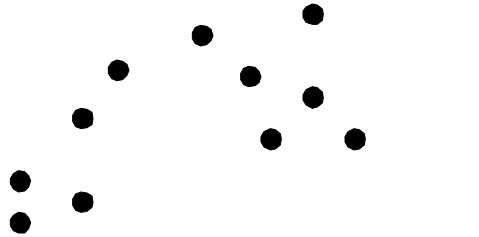
Two Clusters

Four Clusters

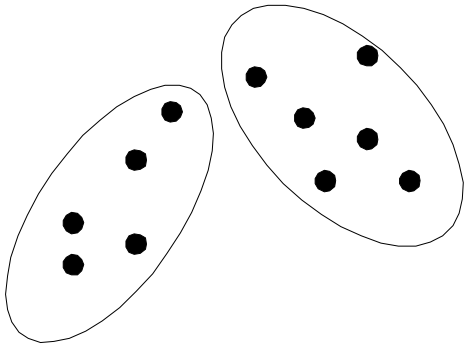
Types of Clusterings

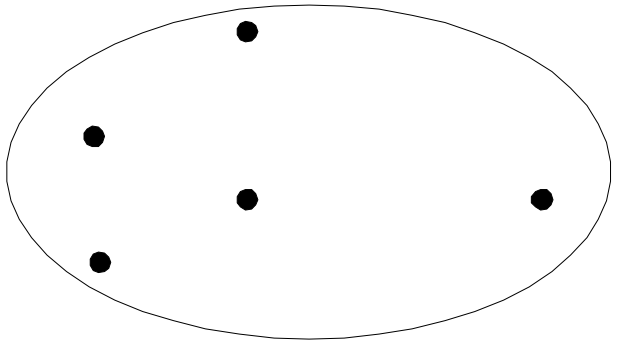
- Important distinction between hierarchical and partitional sets of clusters
  - Partitional Clustering
    - A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset
  - Hierarchical clustering
    - A set of nested clusters organized as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

Partitional Clustering



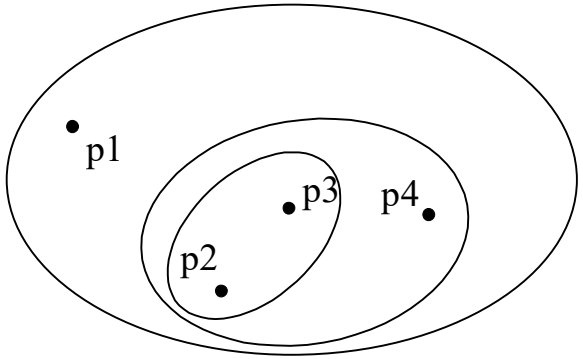
Original Points



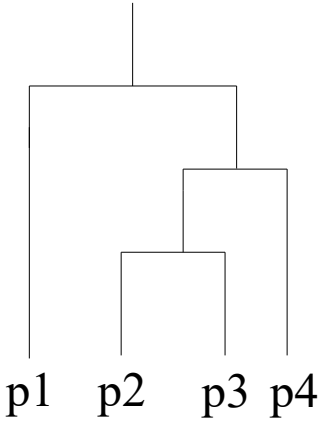


A Partitional Clustering

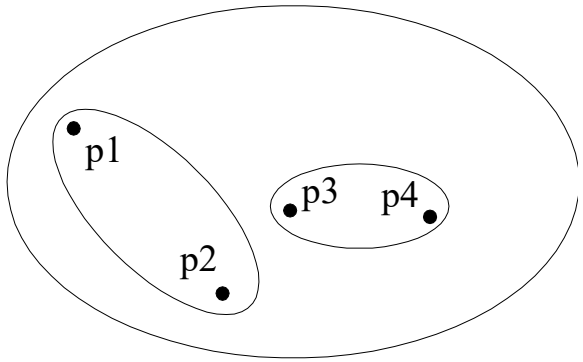
Hierarchical Clustering



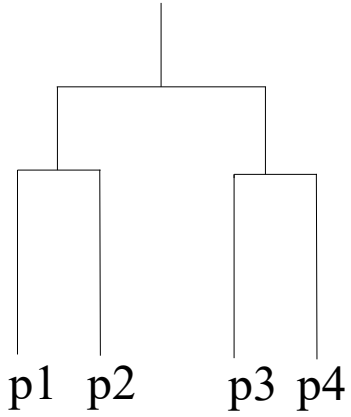
Traditional Hierarchical Clustering



Traditional Dendrogram



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.

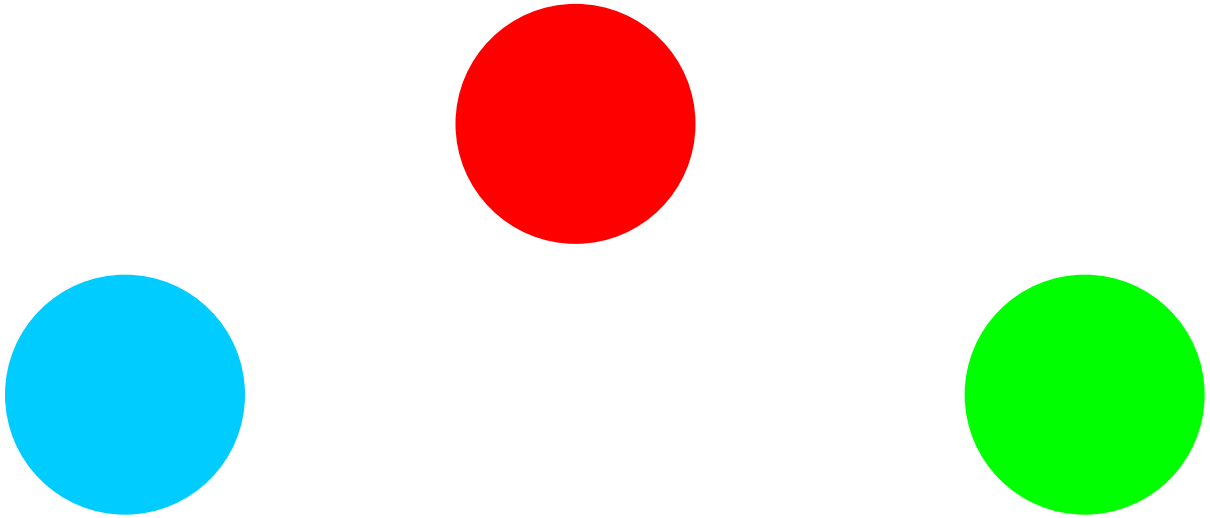
- Fuzzy versus non-fuzzy
  - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
  - Weights must sum to 1
  - Probabilistic clustering has similar characteristics

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

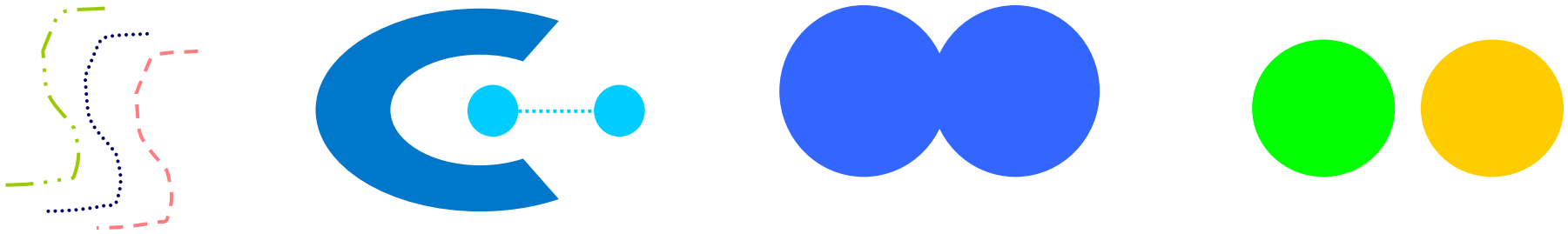
Types of Clusters: Center-Based

- 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

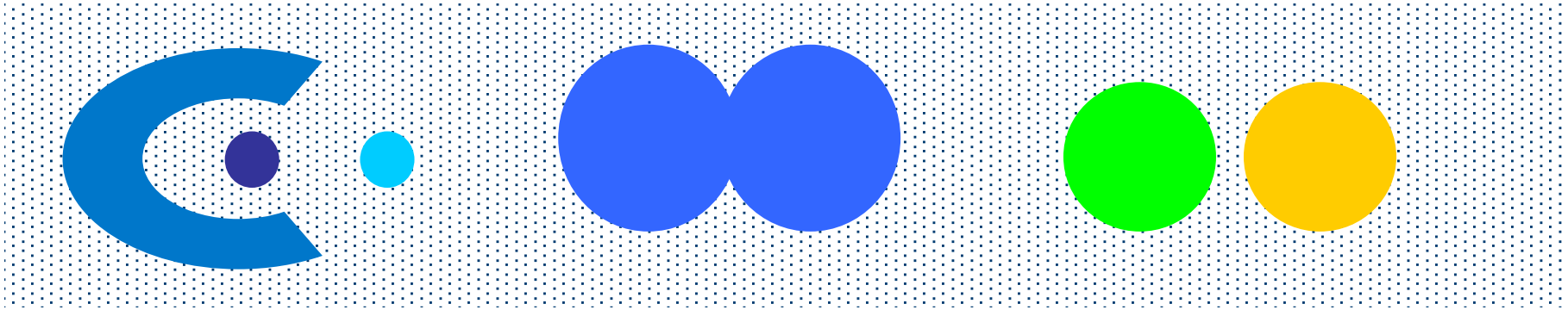
- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.



8 contiguous clusters

Types of Clusters: Density-Based

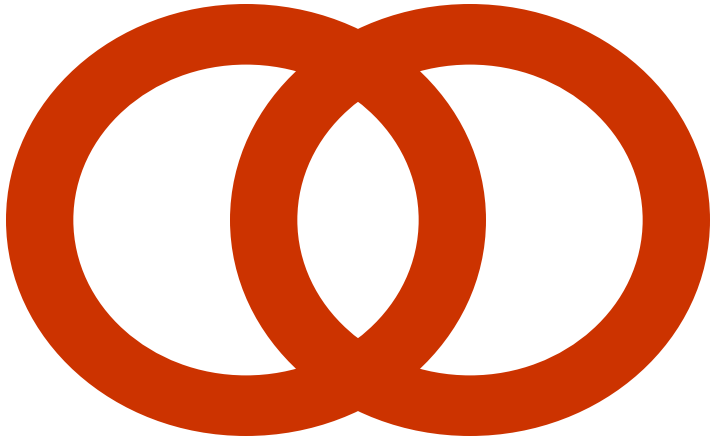
- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

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

Types of Clusters: Objective Function

- Clusters Defined by an Objective Function
  - Finds clusters that minimize or maximize an objective function.
  - Enumerate all possible ways of dividing the points into clusters and evaluate the `goodness' of each potential set of clusters by using the given objective function. (NP Hard)

  - A variation of the global objective function approach is to fit the data to a parameterized model.
    - Parameters for the model are determined from the data.
    - Mixture models assume that the data is a ‘mixture' of a number of statistical distributions.

Clustering Algorithms

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

K-means Clustering

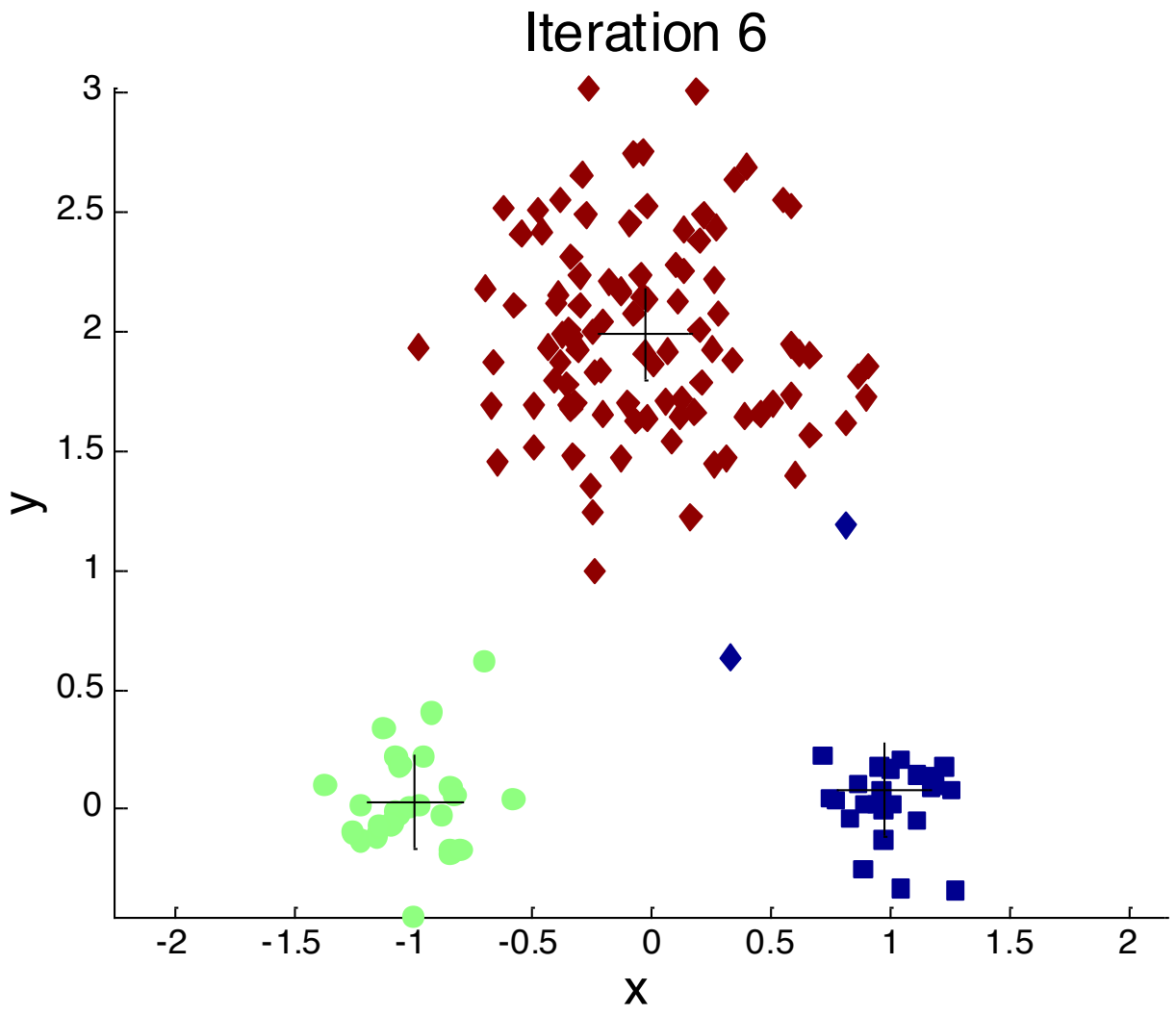
- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

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

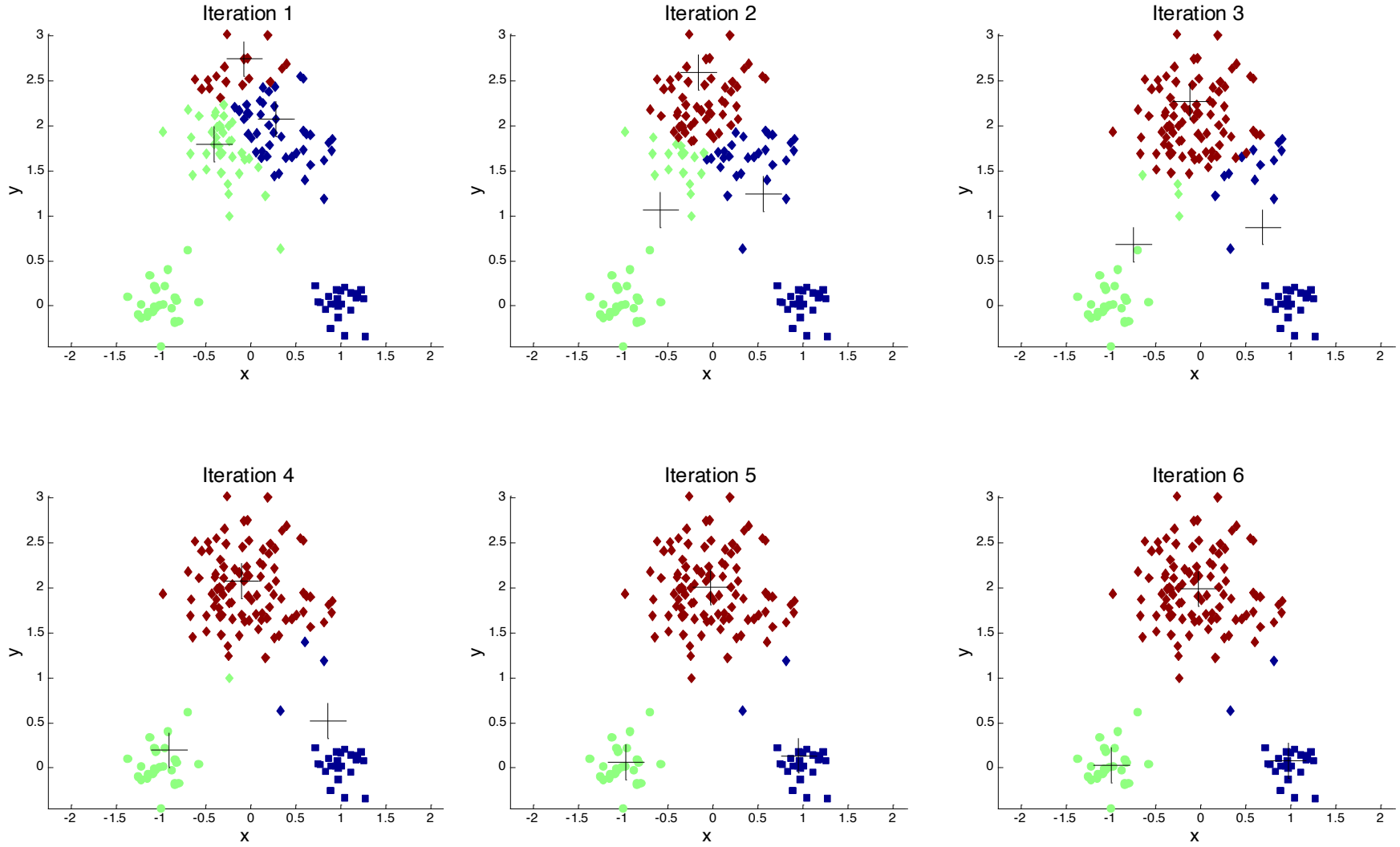
Finding the number of clusters is a challenge task in k-means algorithm.

A. True
B. False

Example of K-means Clustering



Example of K-means Clustering



K-means Clustering – Details

- Initial centroids are often chosen **randomly**.
- 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

The clustering result in k-means does NOT depend on what we choose as the initial cluster centers?

A. True
B. False

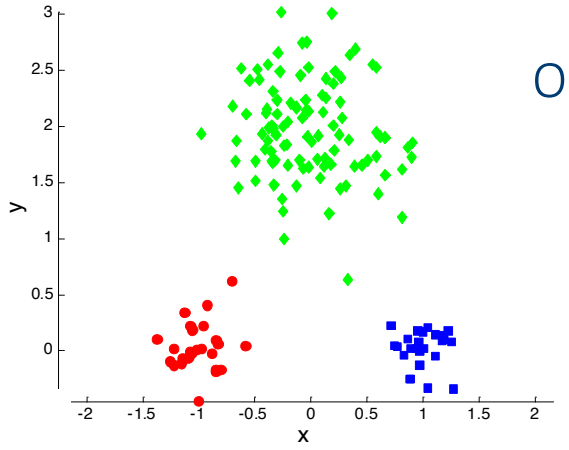
Evaluating K-means Clusters

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

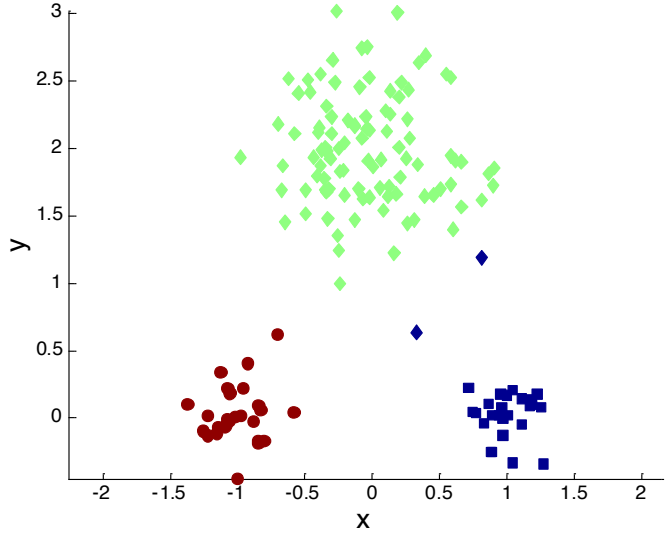
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

  - 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 sets of clusters, we prefer 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

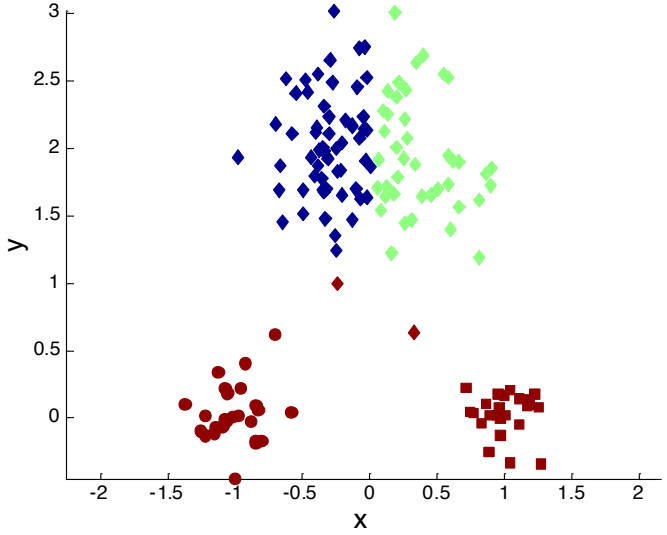
Two different K-means Clusterings



Original Points



Optimal Clustering



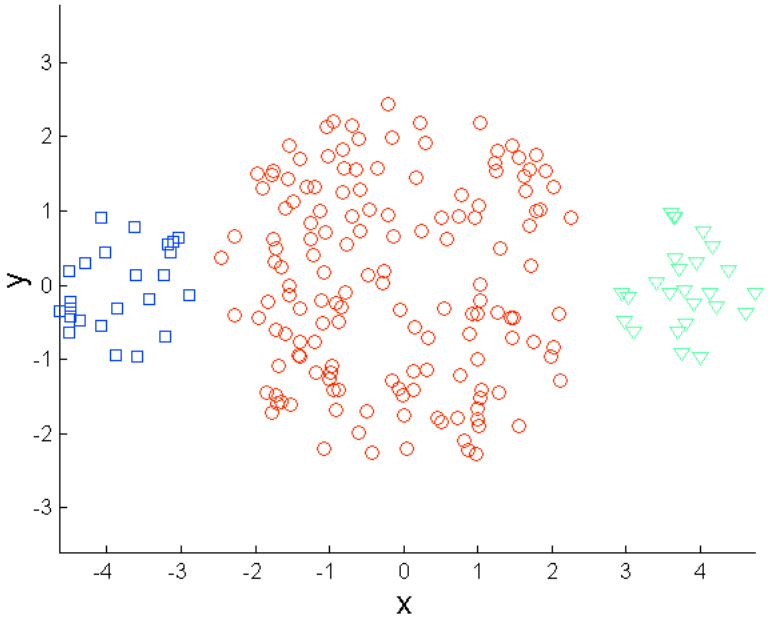
Sub-optimal Clustering

Limitations of K-means

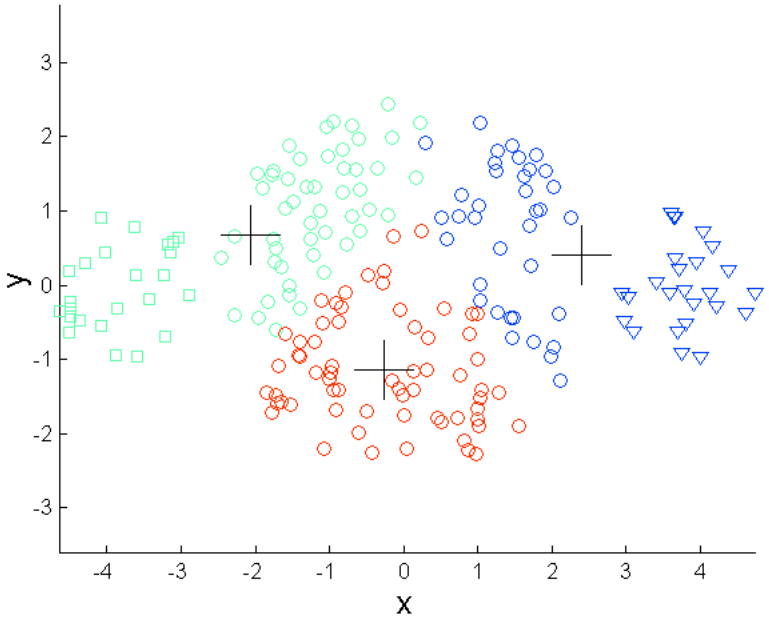
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

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

Limitations of K-means: Differing Sizes

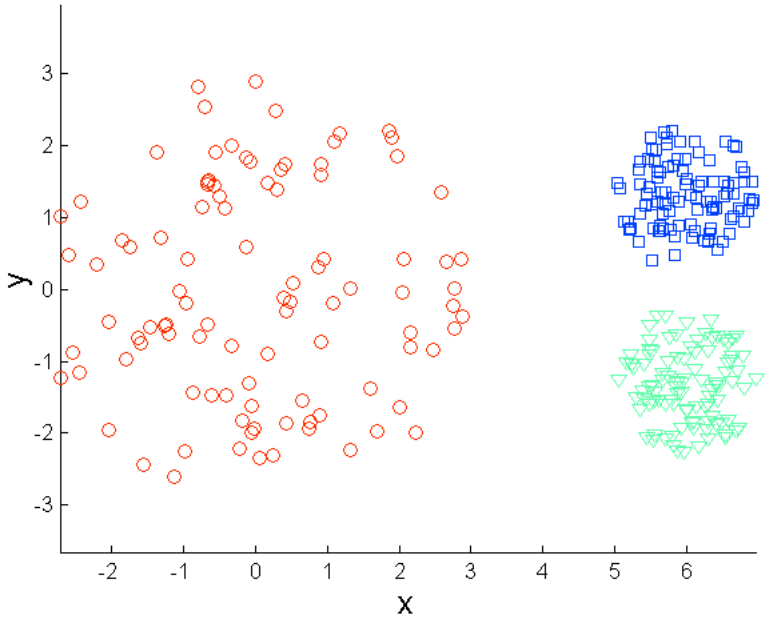


Original Points

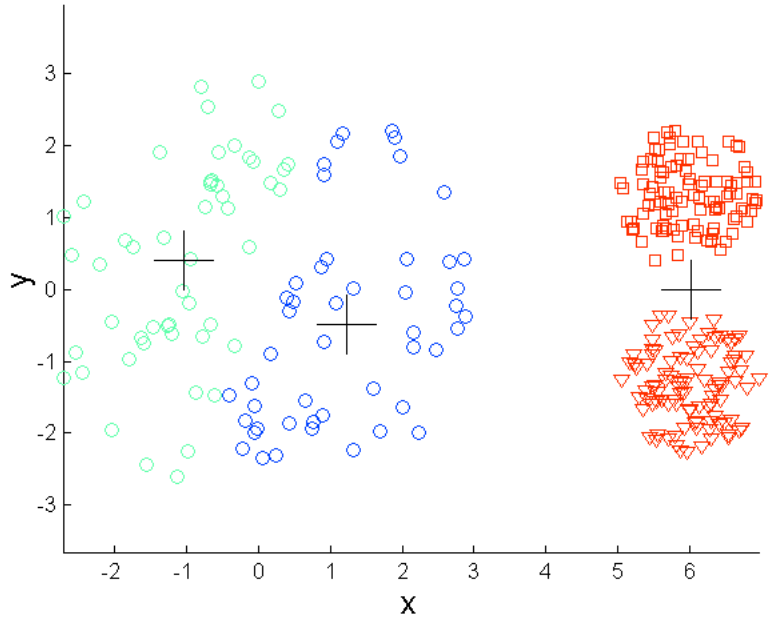


K-means (3 Clusters)

Limitations of K-means: Differing Density

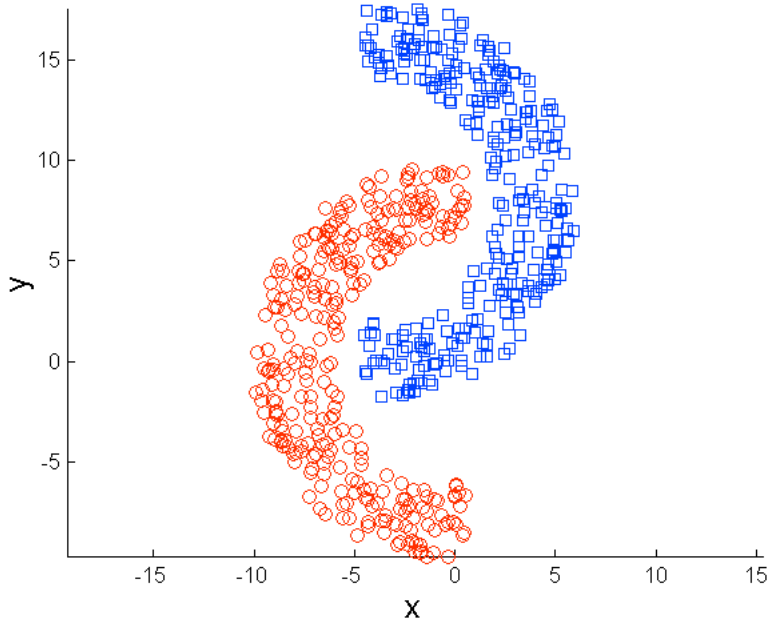


Original Points

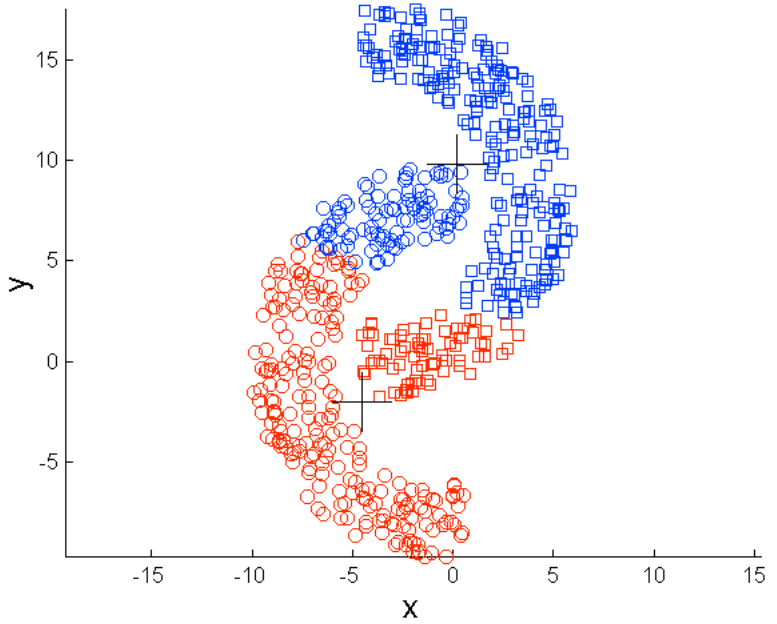


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

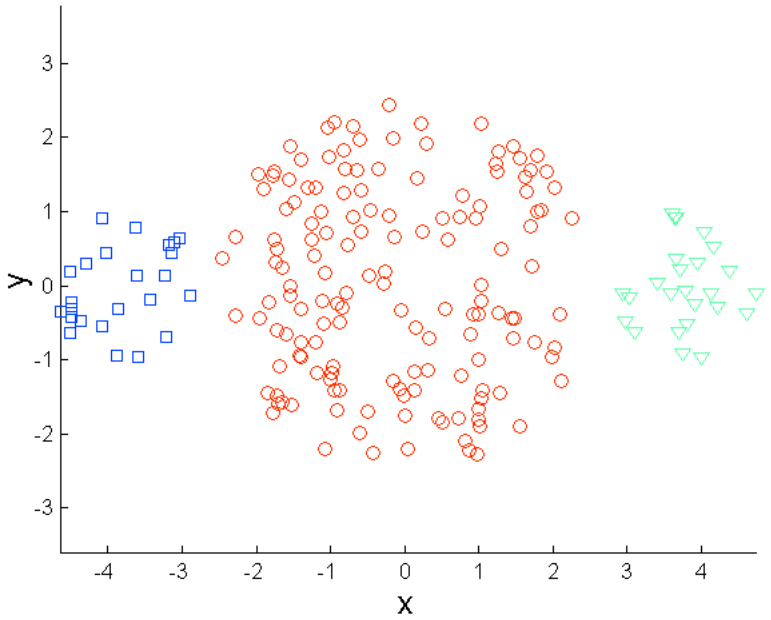


Original Points

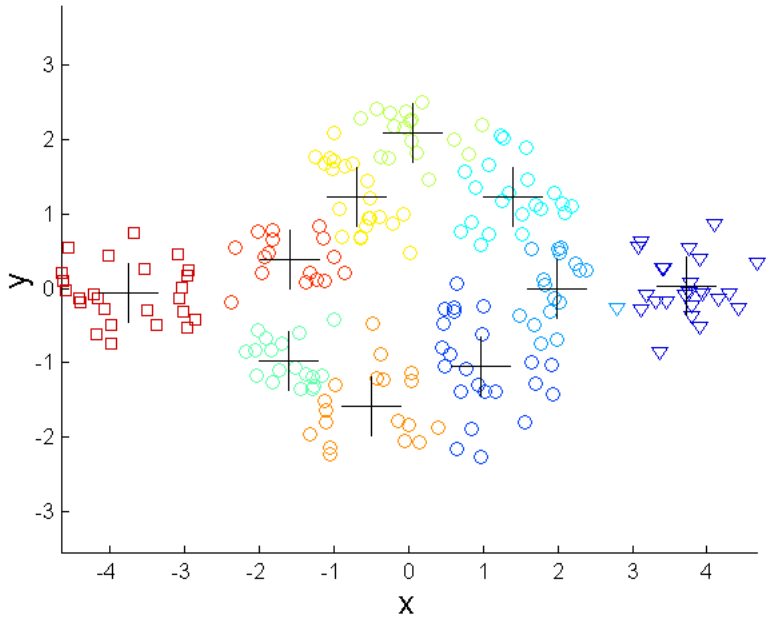


K-means (2 Clusters)

Overcoming K-means Limitations



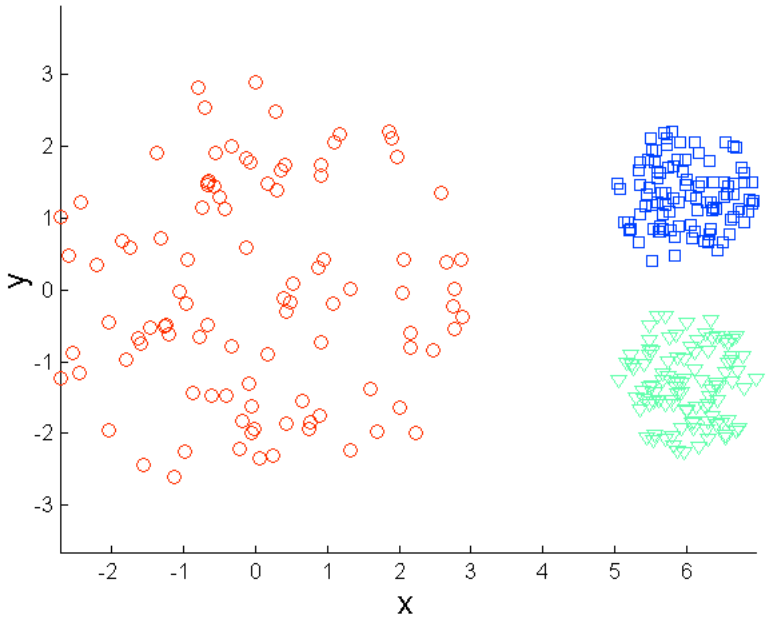
Original Points



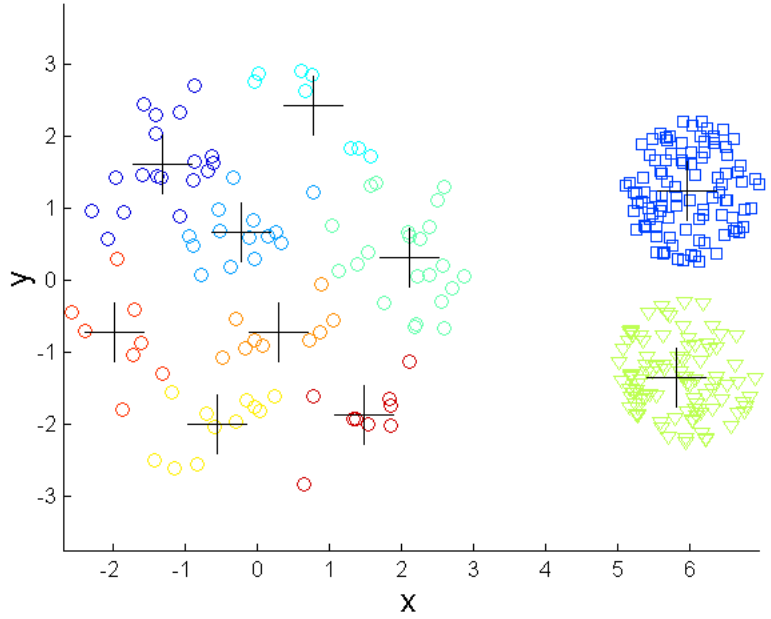
K-means Clusters

One **solution** is to use many clusters.
We can find parts of clusters, but need to put together.

Overcoming K-means Limitations



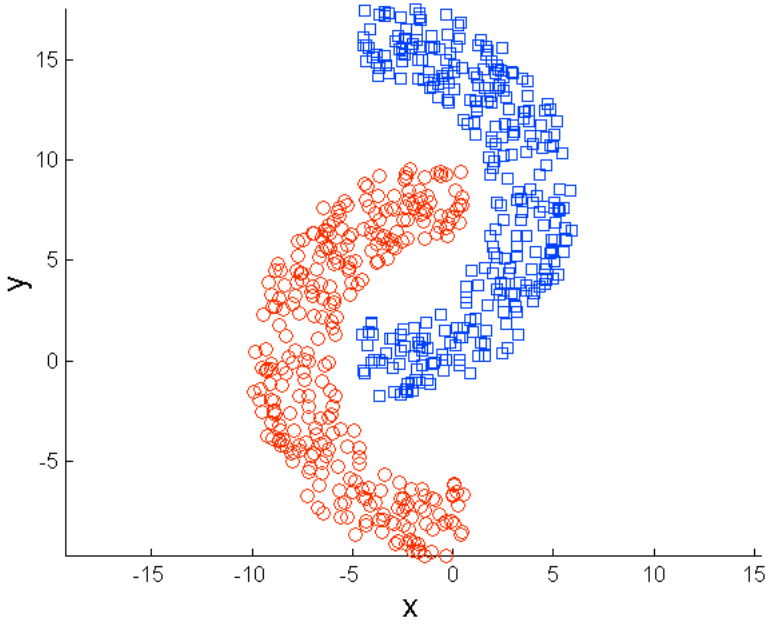
Original Points



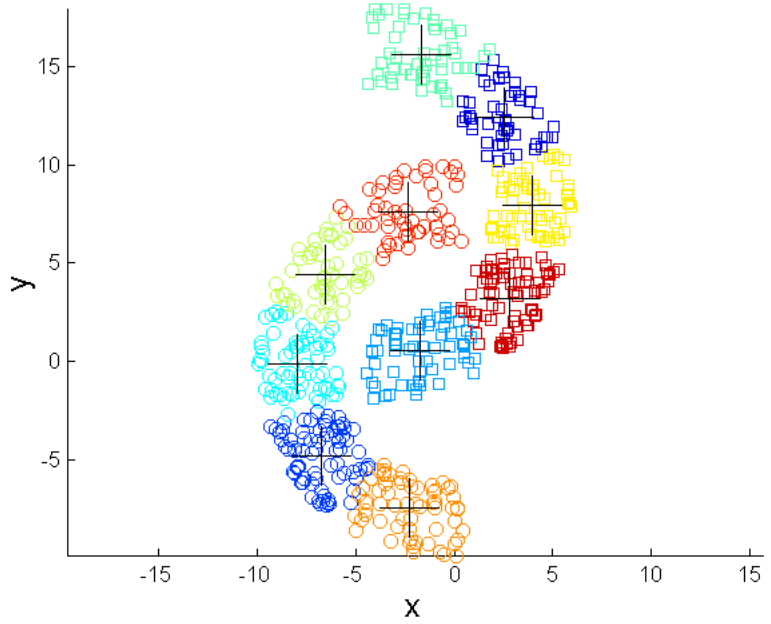
K-means Clusters

One **solution** is to use many clusters.
We can find parts of clusters, but need to put together.

Overcoming K-means Limitations

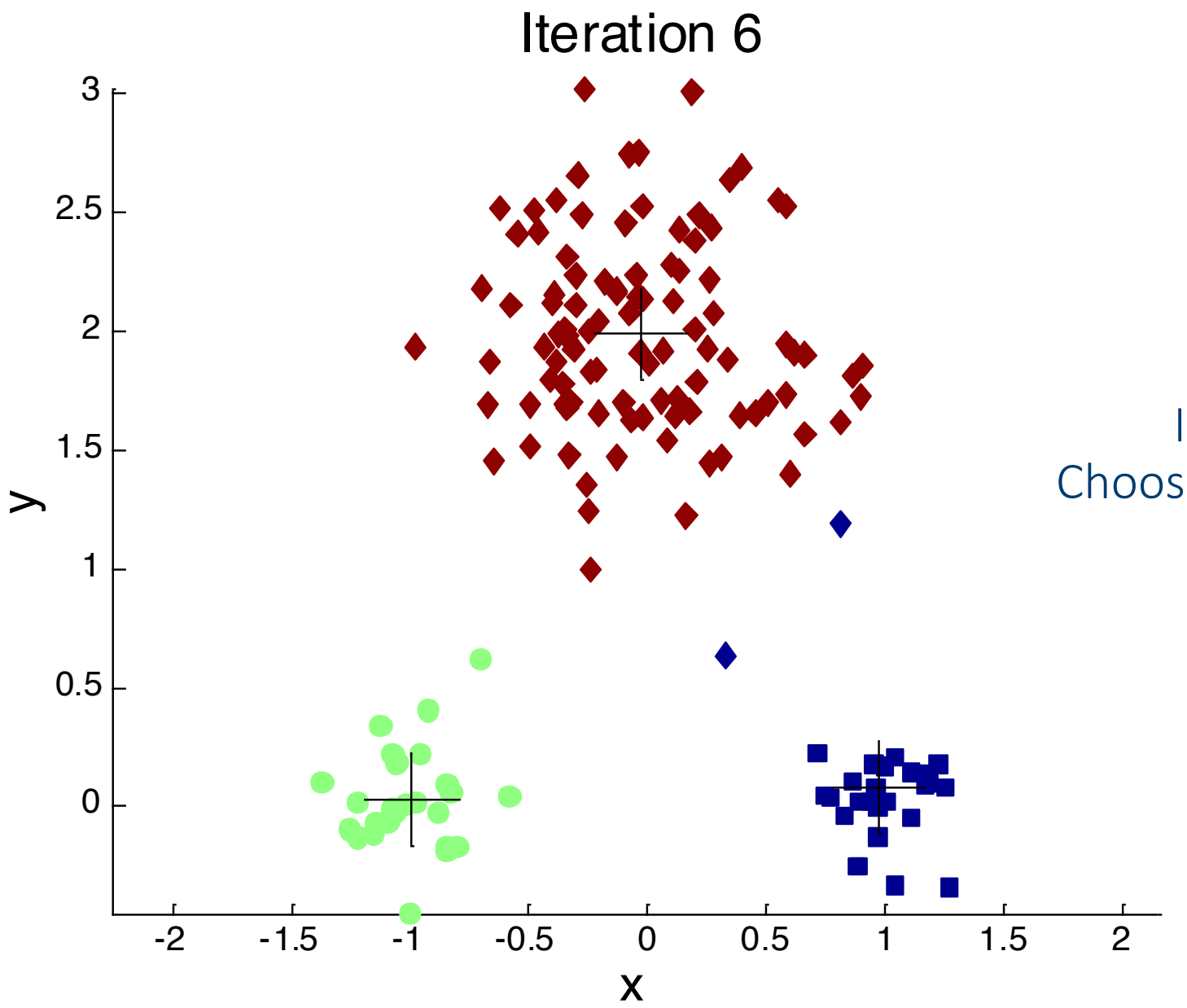


Original Points



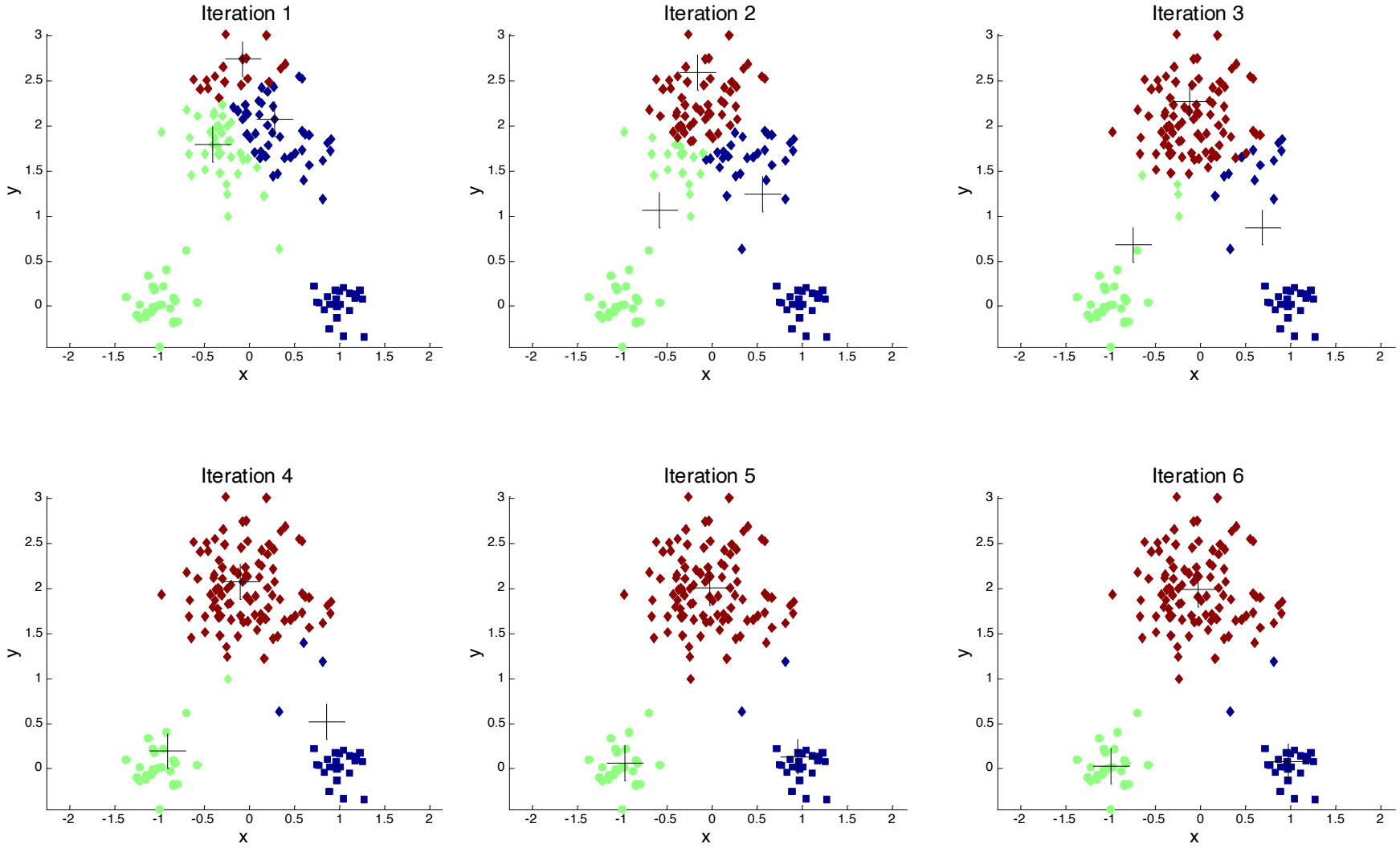
K-means Clusters

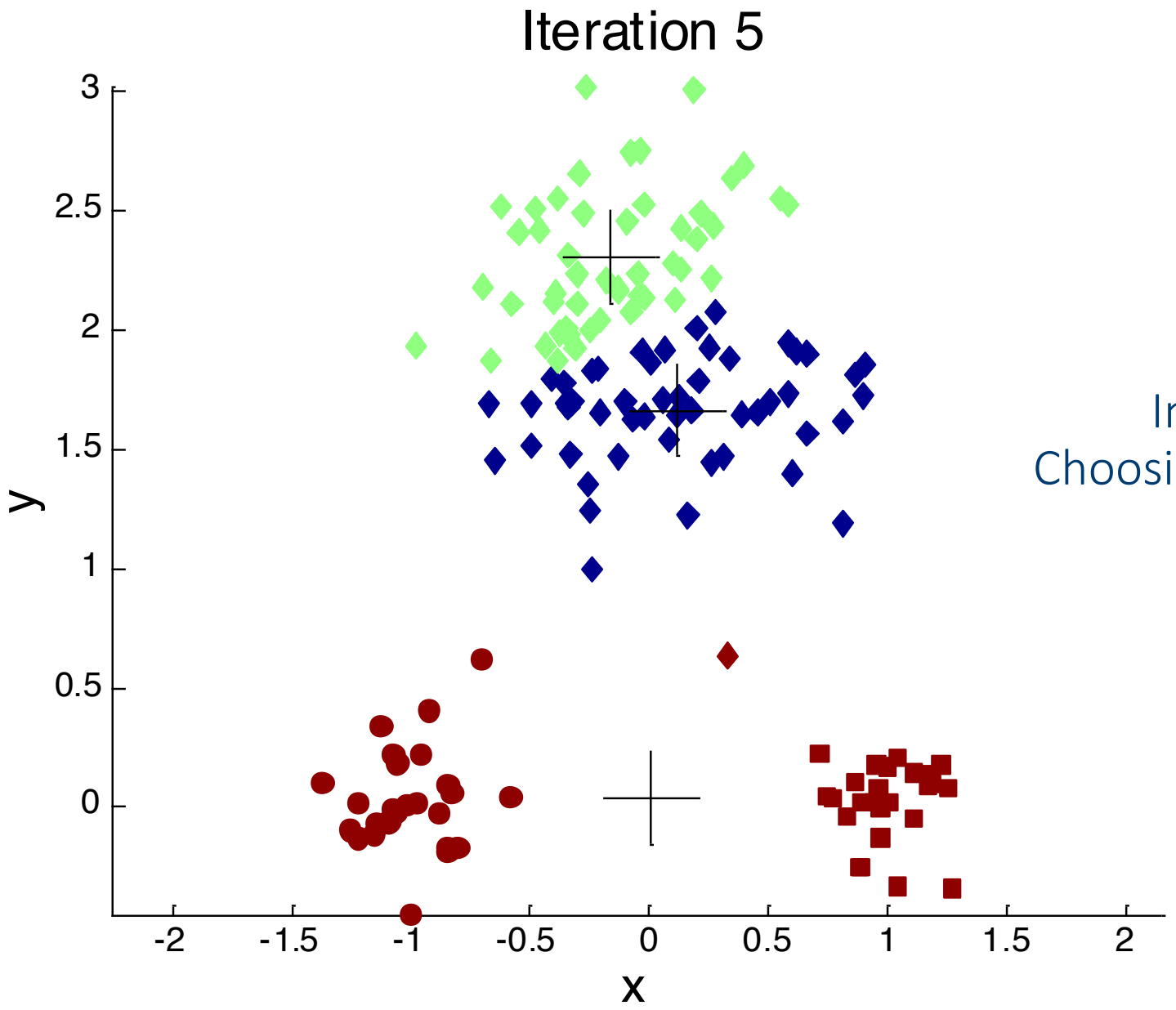
One **solution** is to use many clusters.
We can find parts of clusters, but need to put together.



Importance of Choosing Initial Centroids

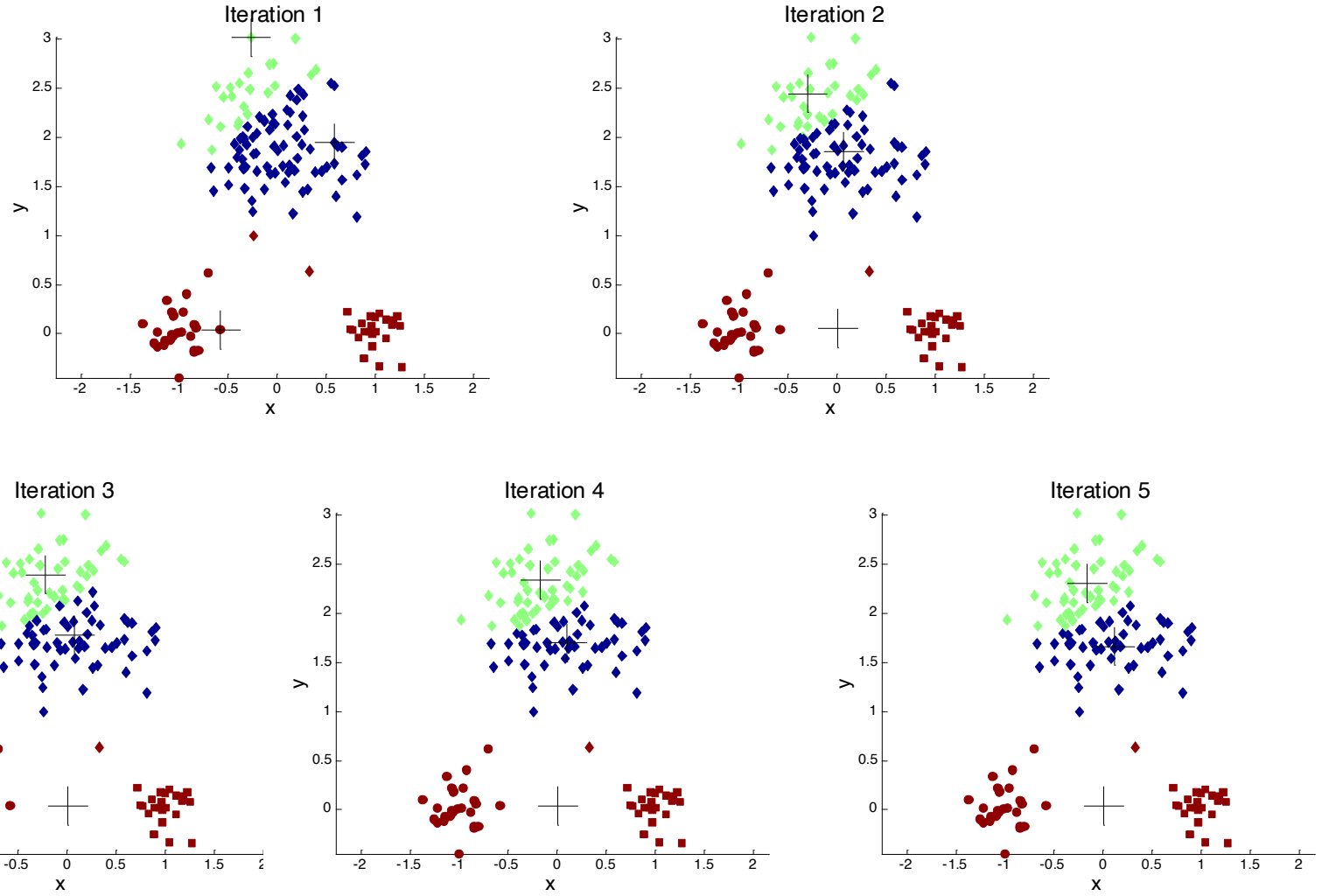
Importance of Choosing Initial Centroids





Importance of Choosing Initial Centroids

Importance of Choosing Initial Centroids …



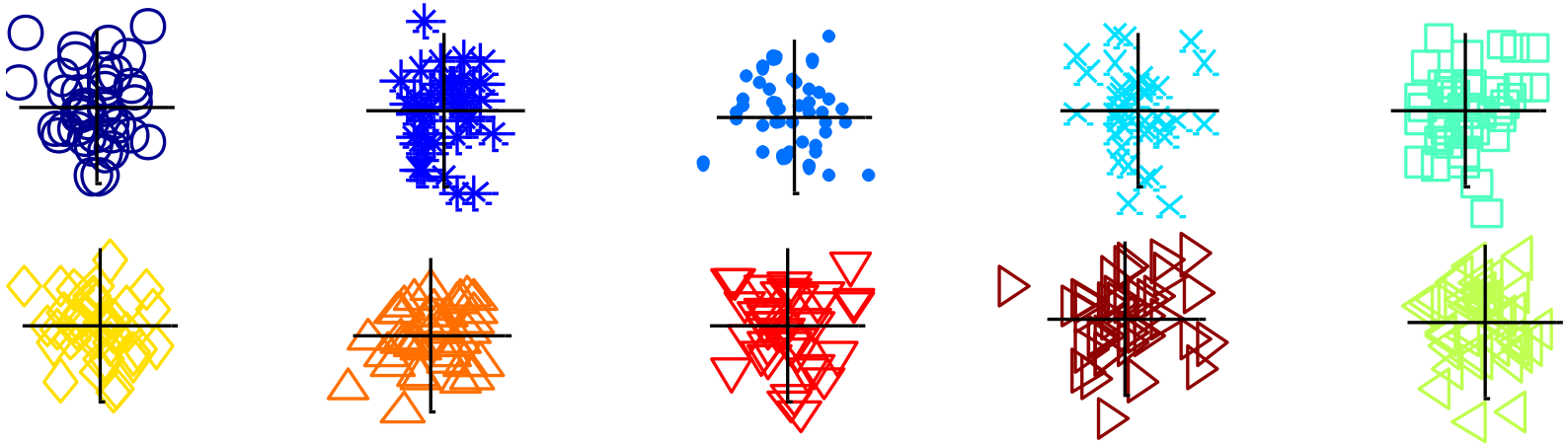
Problems with Selecting Initial Points

- 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

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

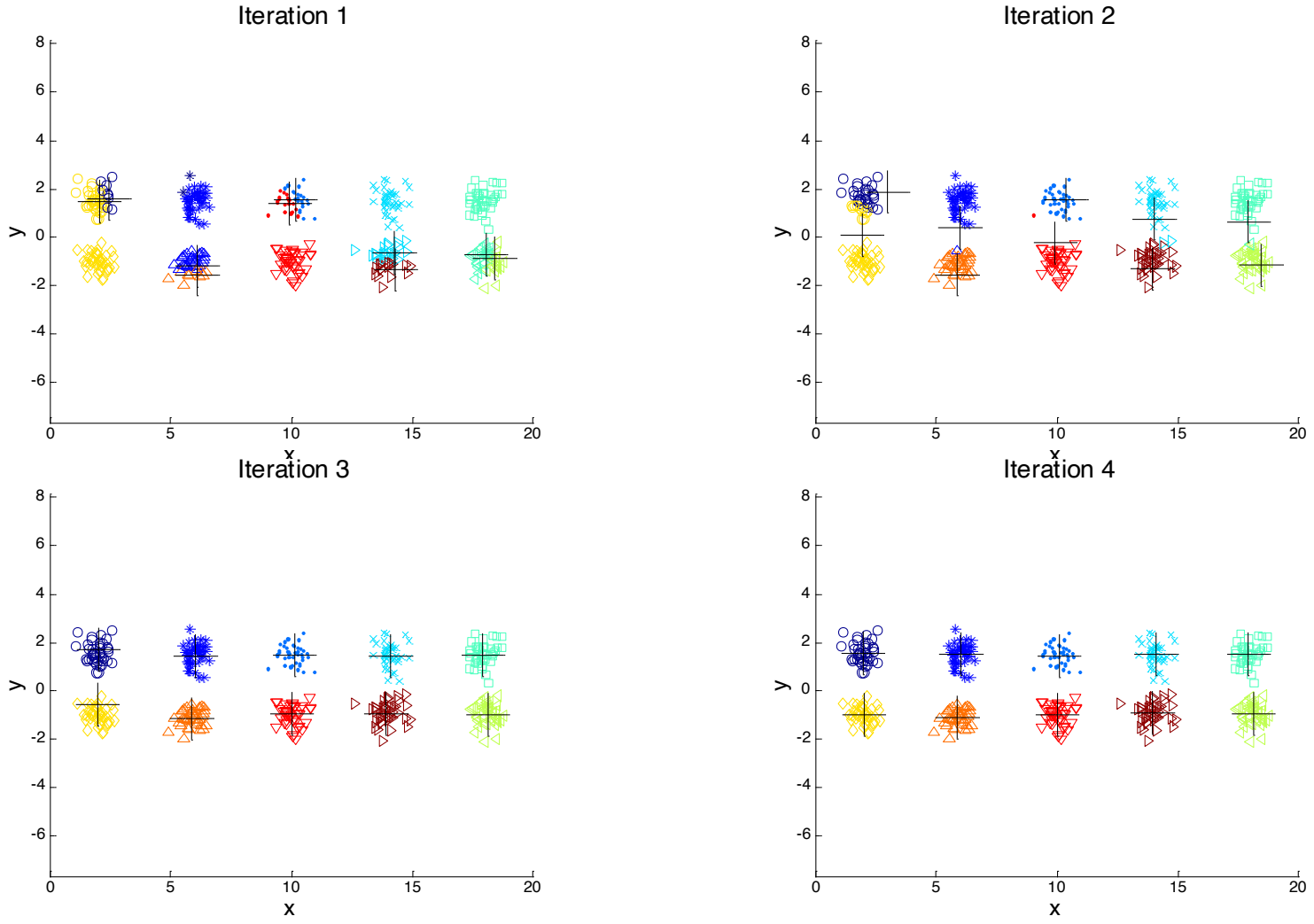
  - For example, if $K = 10$, then probability = 10!/1010 = 0.00036

10 Clusters Example



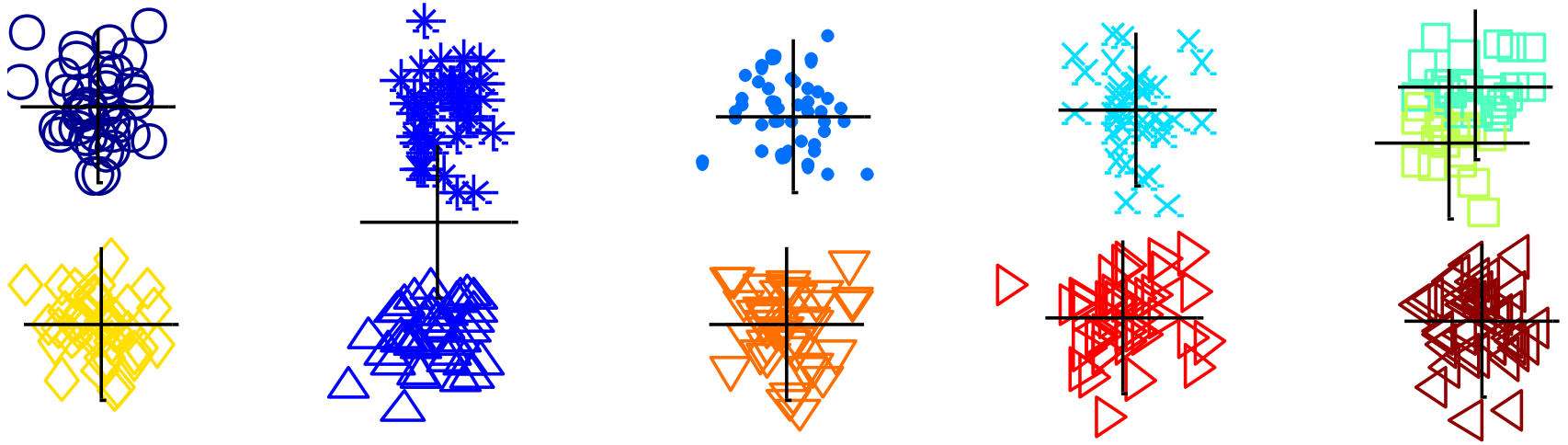
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



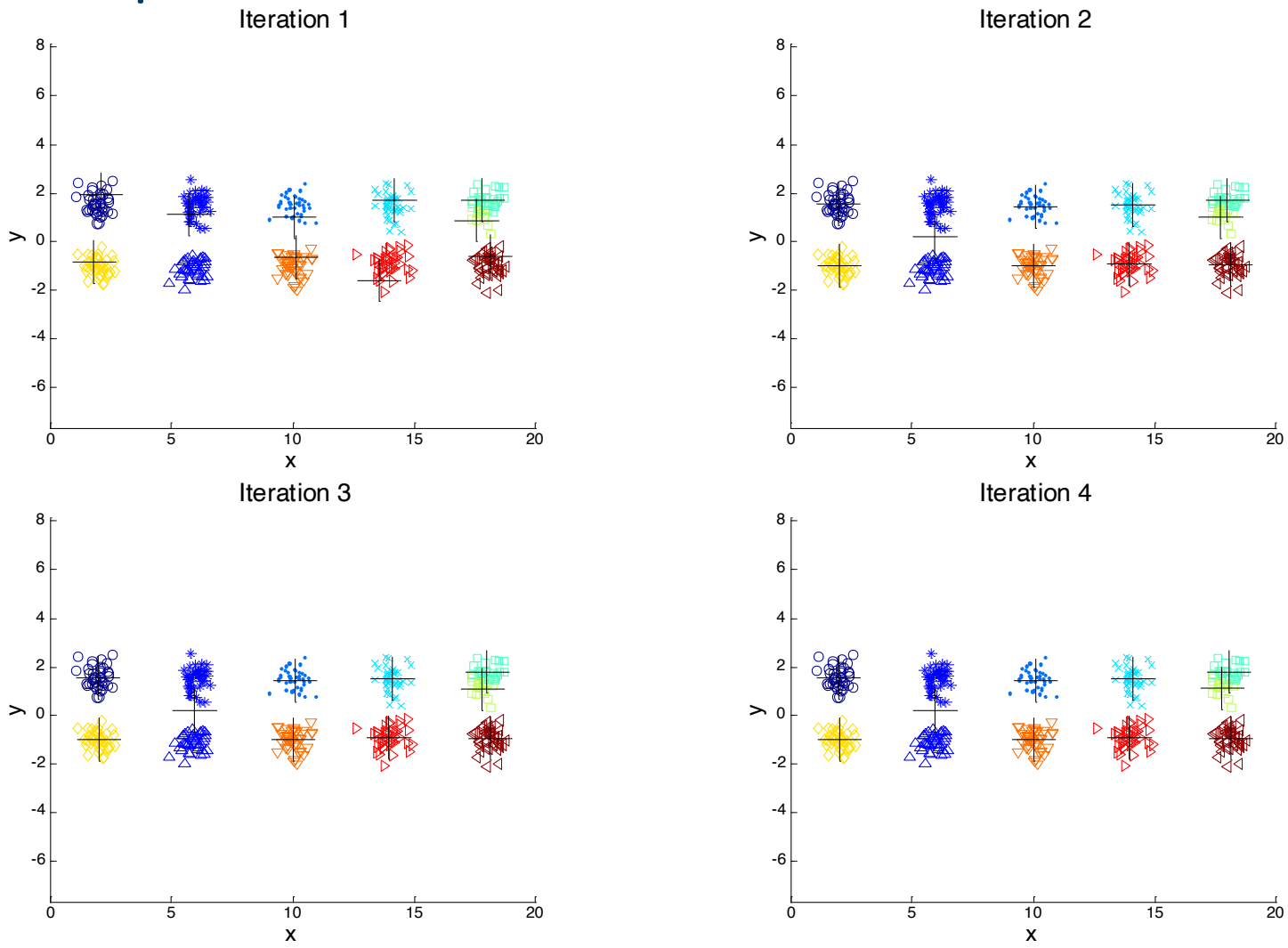
Starting with two initial centroids in one cluster of each pair of clusters

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
- Generate a larger number of clusters and then perform a hierarchical clustering
- Bisecting K-means
  - Not as susceptible to initialization issues

Pre-processing and Post-processing

- Pre-processing
  - Normalize the data
  - Eliminate outliers

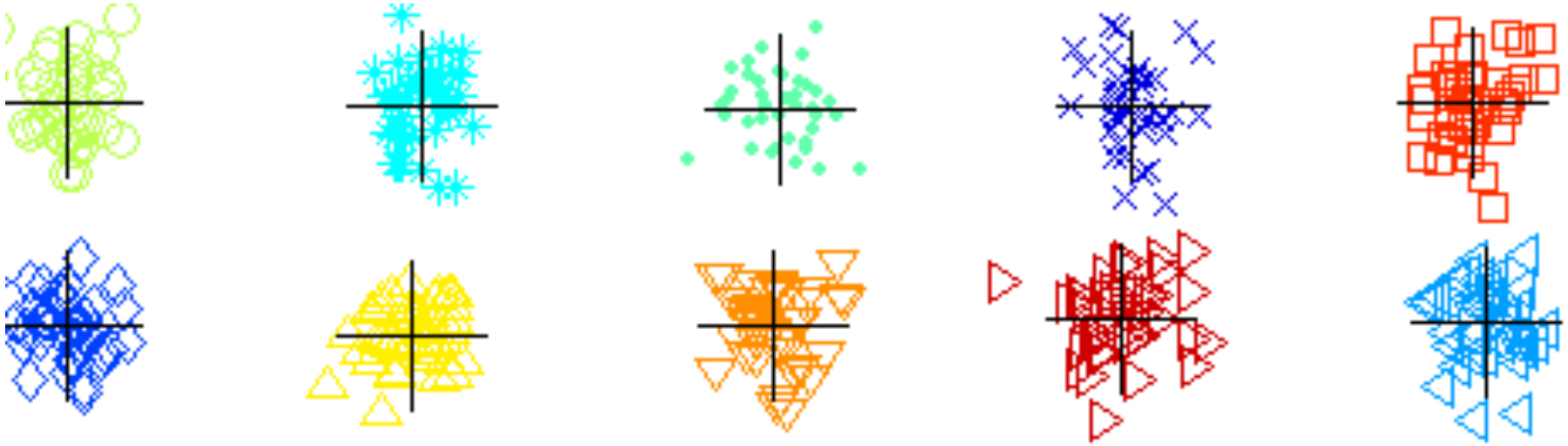
- Post-processing
  - Eliminate small clusters that may represent outliers
  - Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  - Merge clusters that are ‘close’ and that have relatively low SSE

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

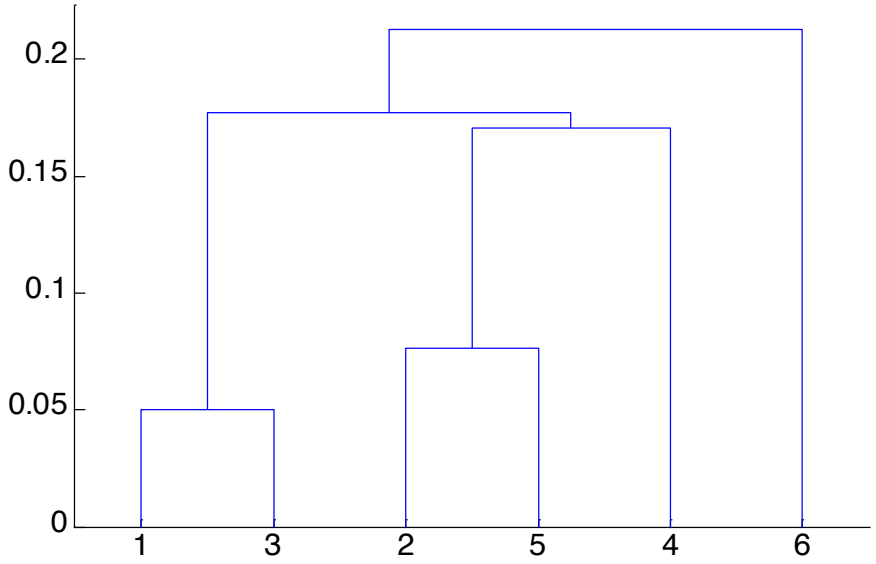


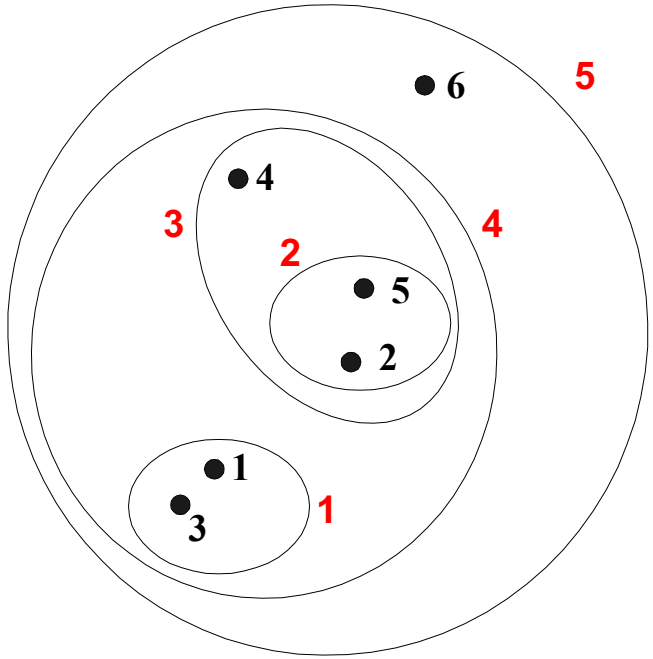
Clustering Algorithms

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

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

Agglomerative Clustering Algorithm

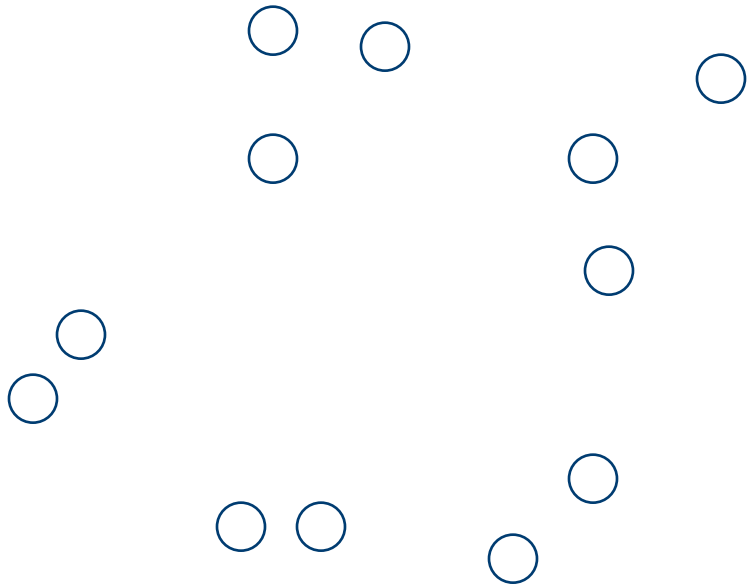
- Most popular hierarchical clustering technique

- Basic algorithm is straightforward:
  1. Compute the proximity matrix
  2. Let each data point be a cluster
  3. **Repeat**
  4. Merge the two closest clusters
  5. Update the proximity matrix
  6. **Until** only a single cluster remains

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

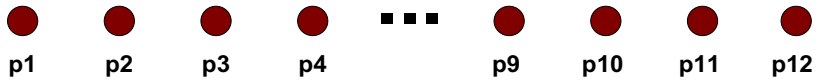
Starting Situation

- Start with clusters of individual points and a proximity matrix



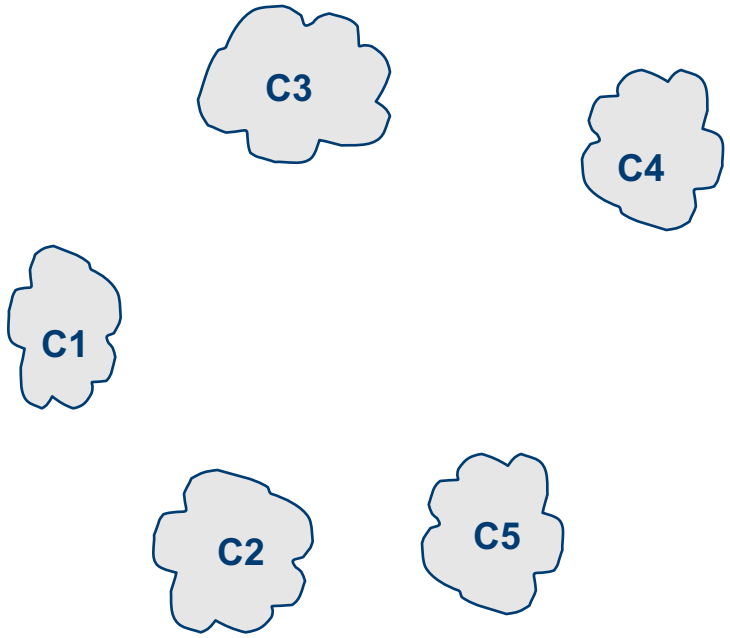
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix



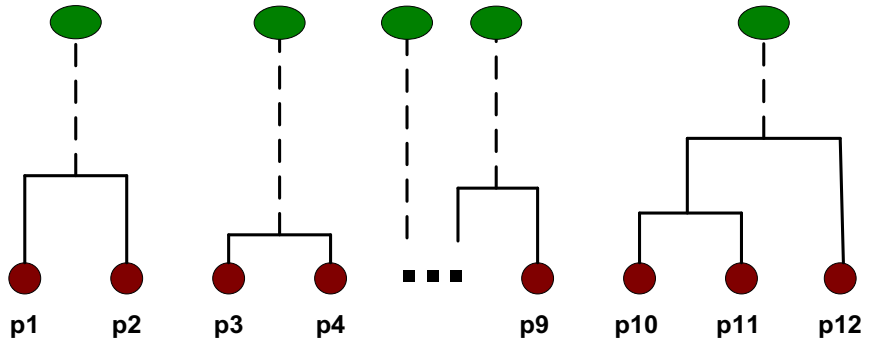
Intermediate Situation

- After some merging steps, we have some clusters



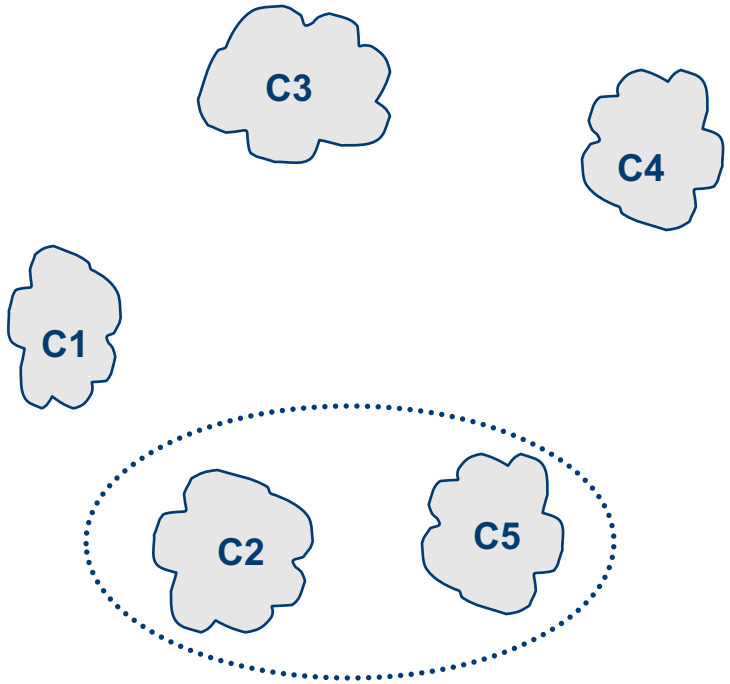
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



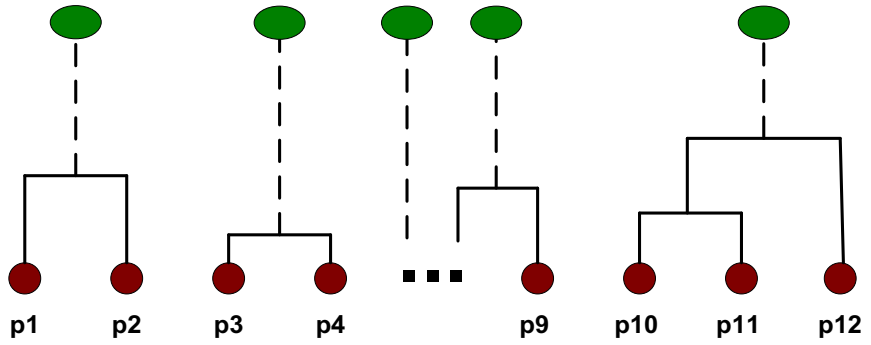
Intermediate Situation

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



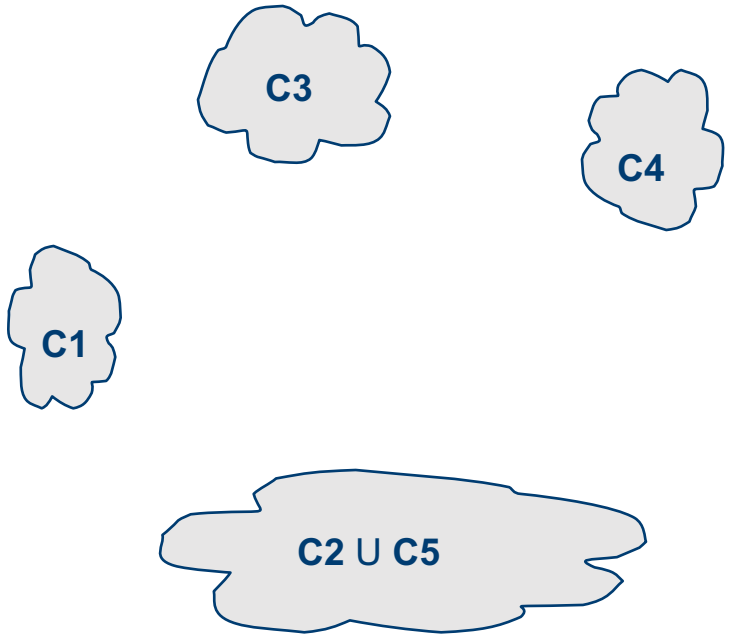
	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix



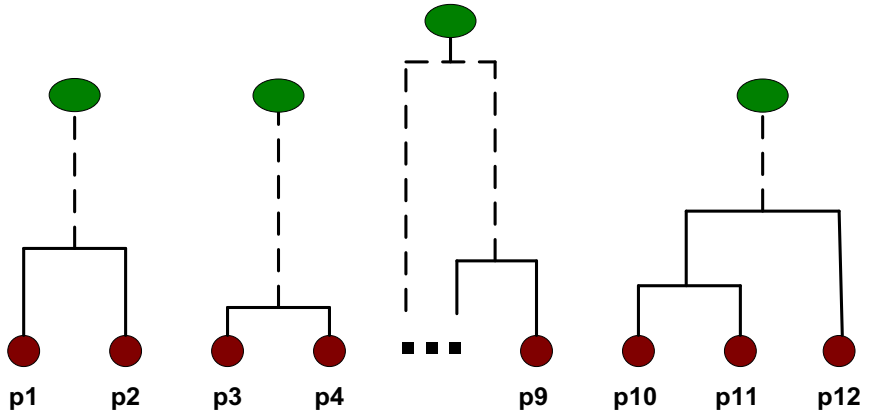
After Merging

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

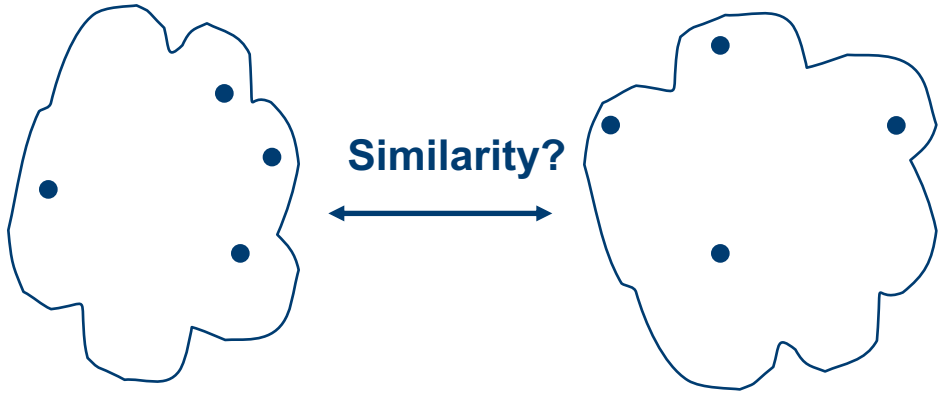


	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



How to Define Inter-Cluster Distance

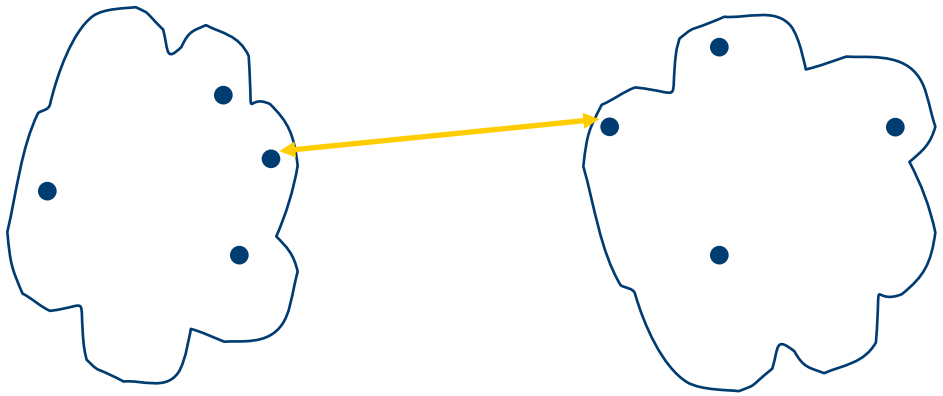


- MIN
- MAX
- Group Average
- Distance Between Centroids

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

Proximity Matrix

How to Define Inter-Cluster Similarity

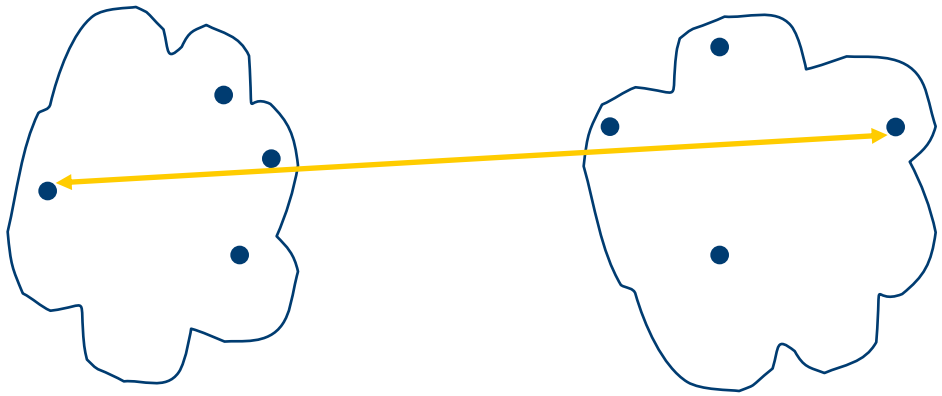


- MIN
- MAX
- Group Average
- Distance Between Centroids

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

Proximity Matrix

How to Define Inter-Cluster Similarity

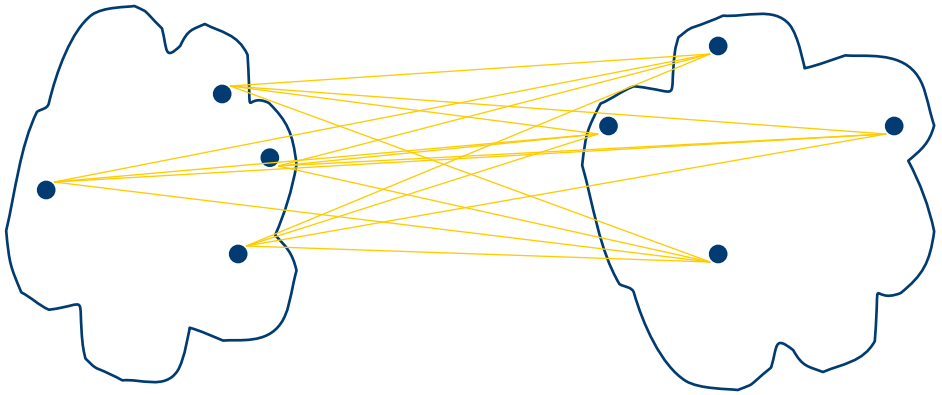


- MIN
- MAX
- Group Average
- Distance Between Centroids

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

Proximity Matrix

How to Define Inter-Cluster Similarity

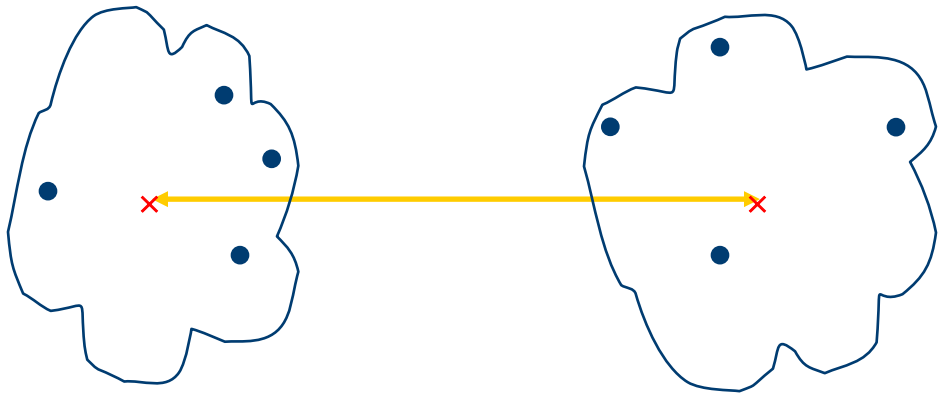


- MIN
- MAX
- Group Average
- Distance Between Centroids

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

Proximity Matrix

How to Define Inter-Cluster Similarity



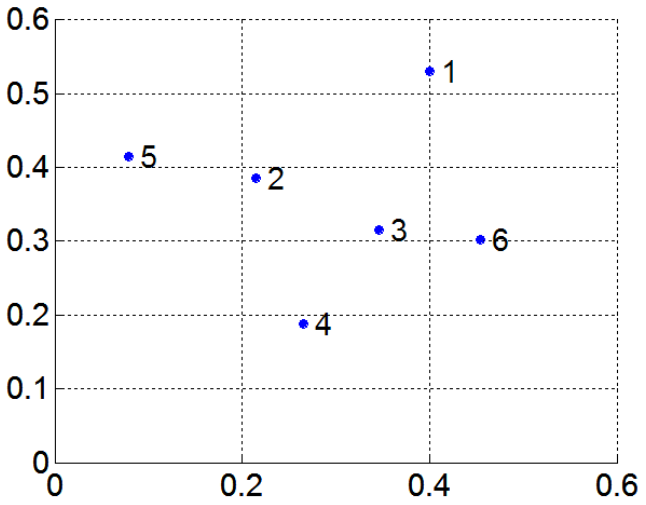
- MIN
- MAX
- Group Average
- Distance Between Centroids

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

Proximity Matrix

MIN or Single Link

- Proximity of two clusters is based on the two closest points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph
- Example:



Distance Matrix:

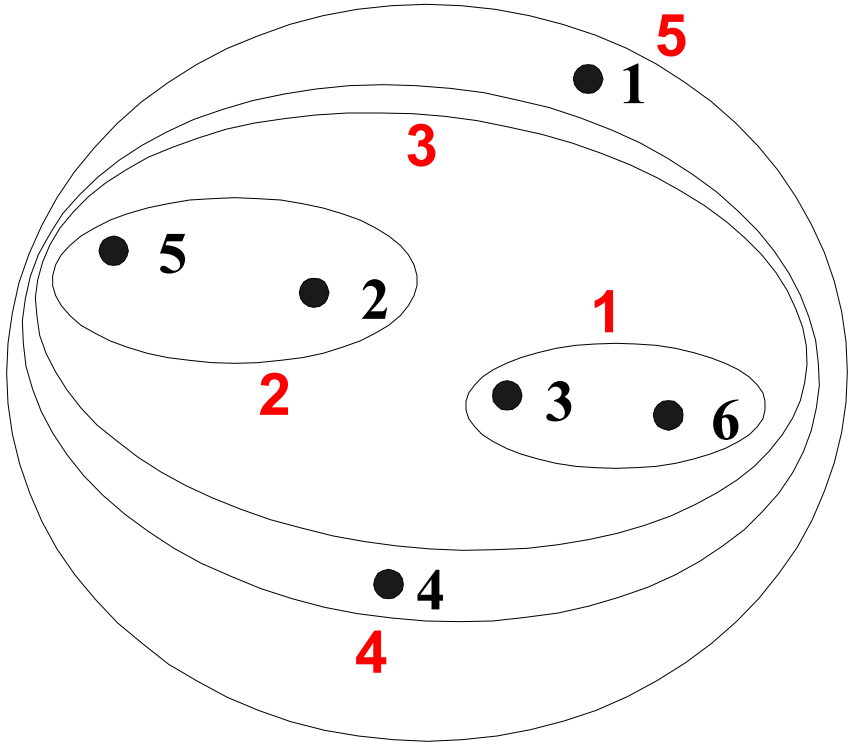
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

In the single link approach, initially ,assume that each point is a cluster, which clusters do you select for merge.

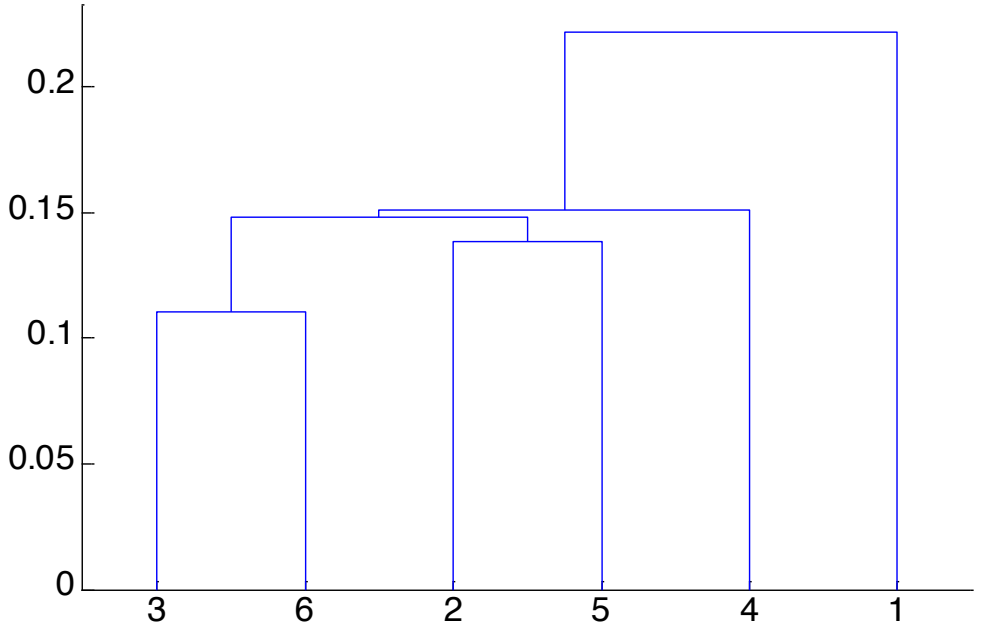
A. 4,5
B. 1,4
C. 2,5
D. 3,6

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

Hierarchical Clustering: MIN

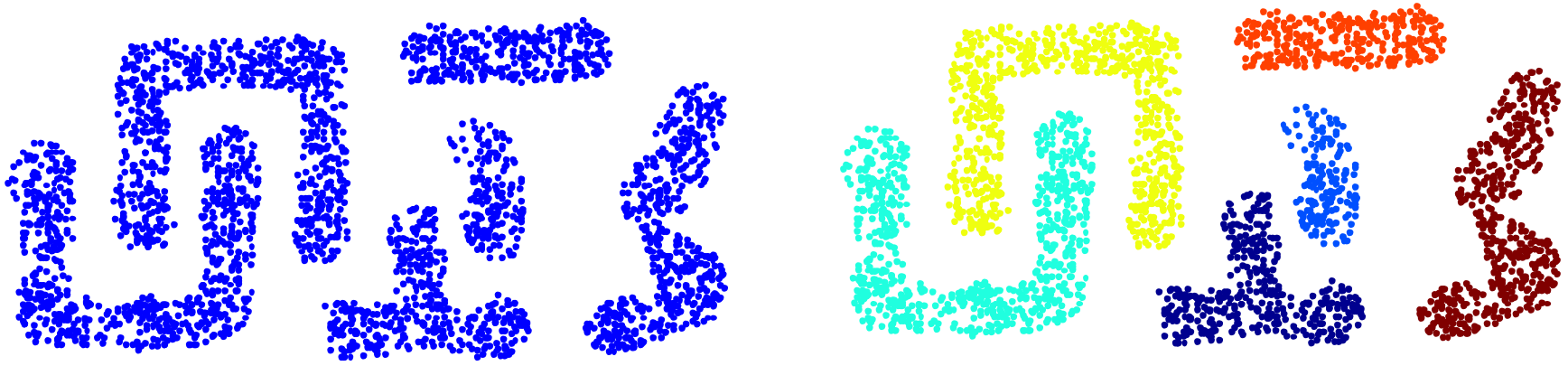


Nested Clusters



Dendrogram

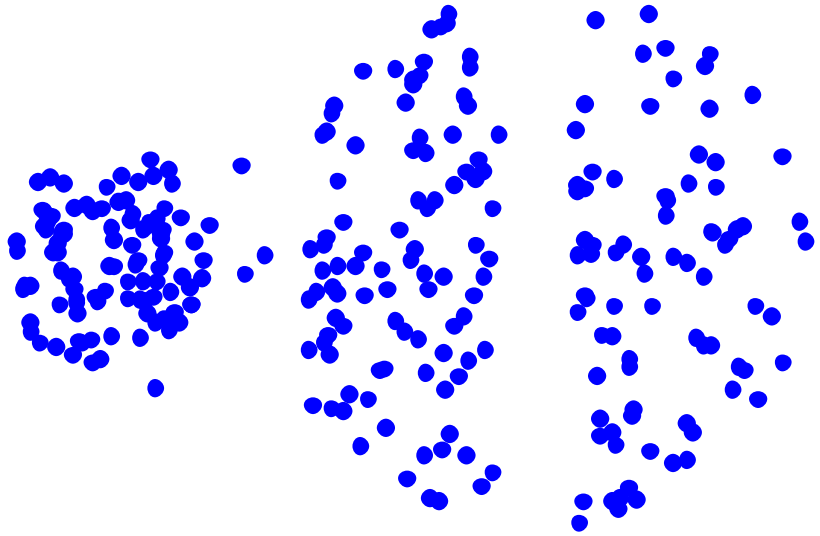
Strength of MIN



Original Points **Six Clusters**

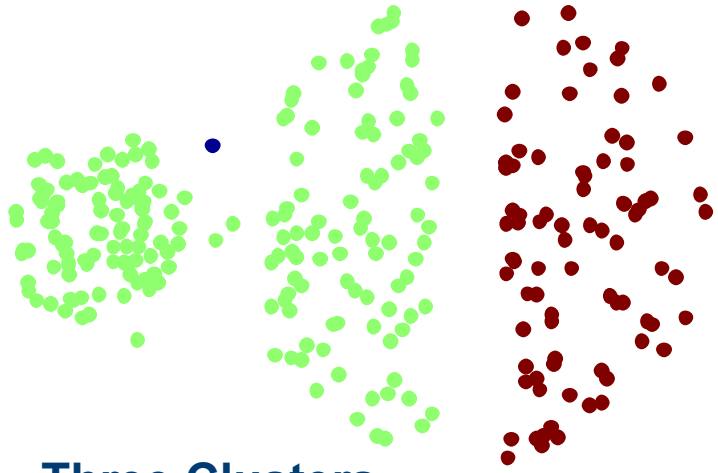
Can handle non-elliptical shapes

Limitations of MIN



Original Points

Two Clusters

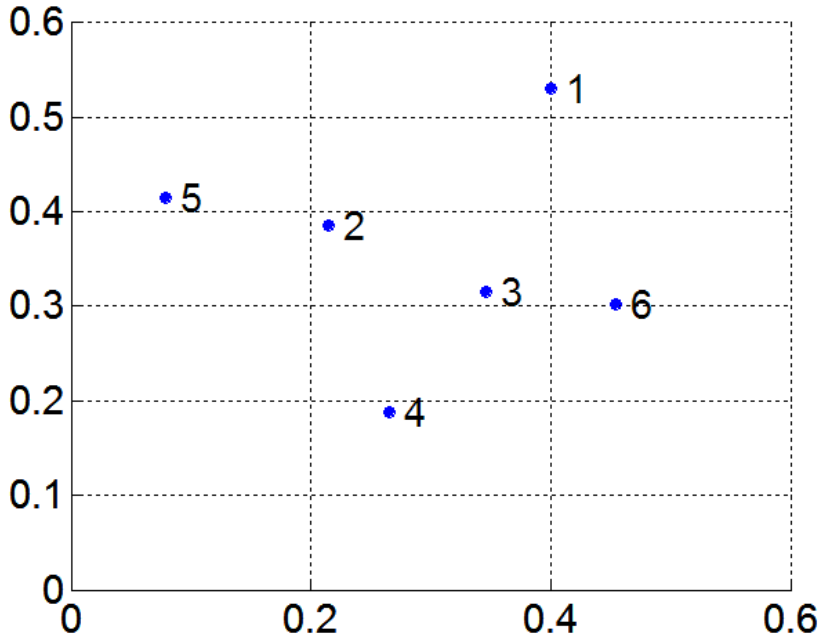


Three Clusters

- Sensitive to noise and outliers

MAX or Complete Linkage

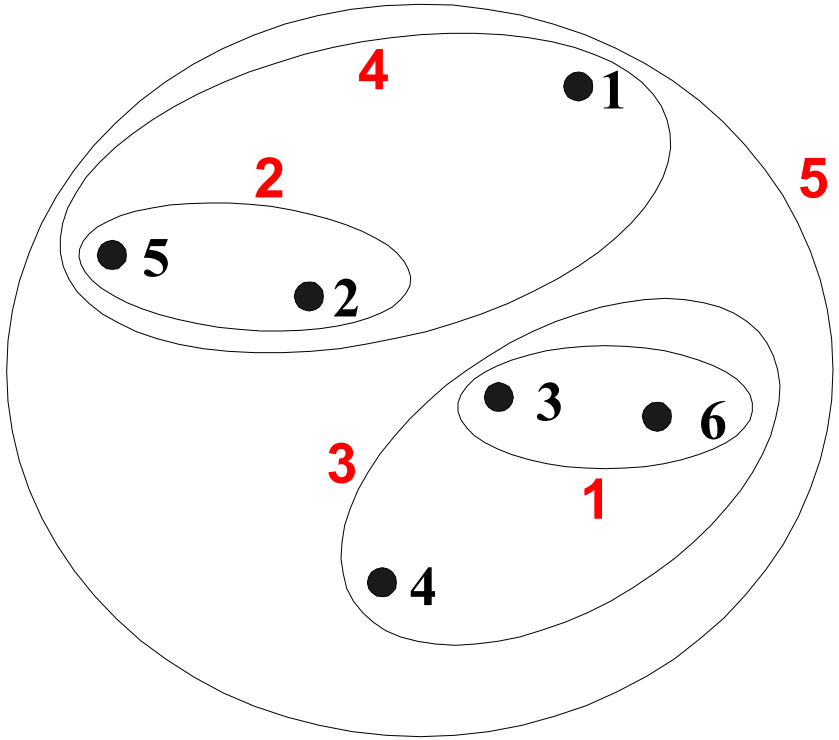
- Proximity of two clusters is based on the two most distant points in the different clusters
  - Determined by all pairs of points in the two clusters



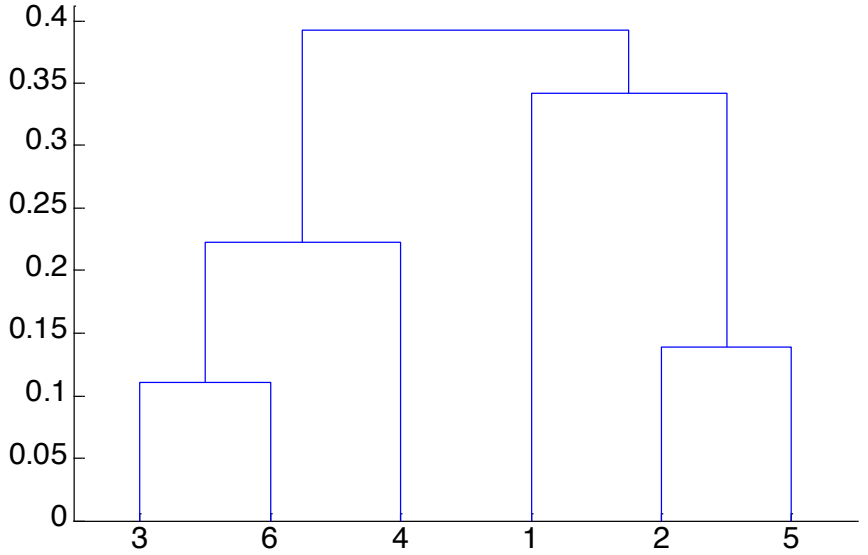
Distance Matrix:

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

Hierarchical Clustering: MAX

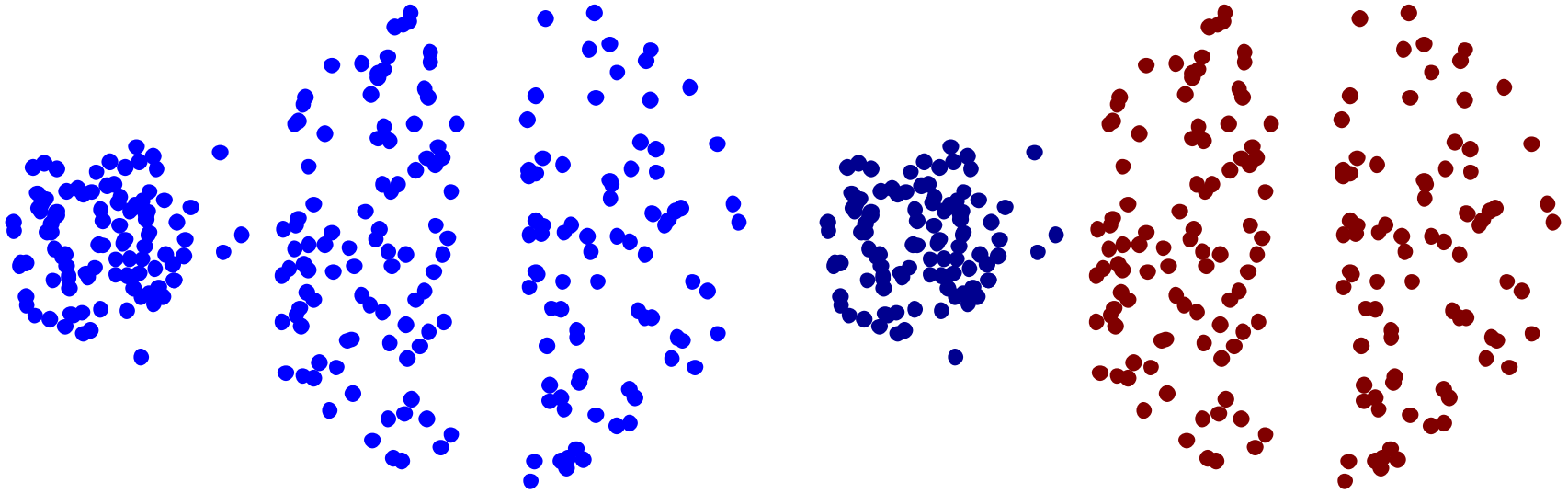


Nested Clusters



Dendrogram

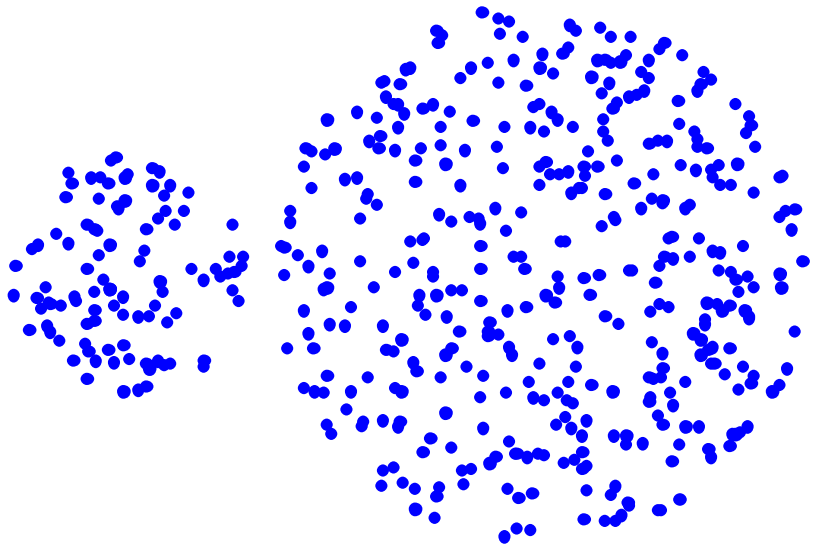
Strength of MAX



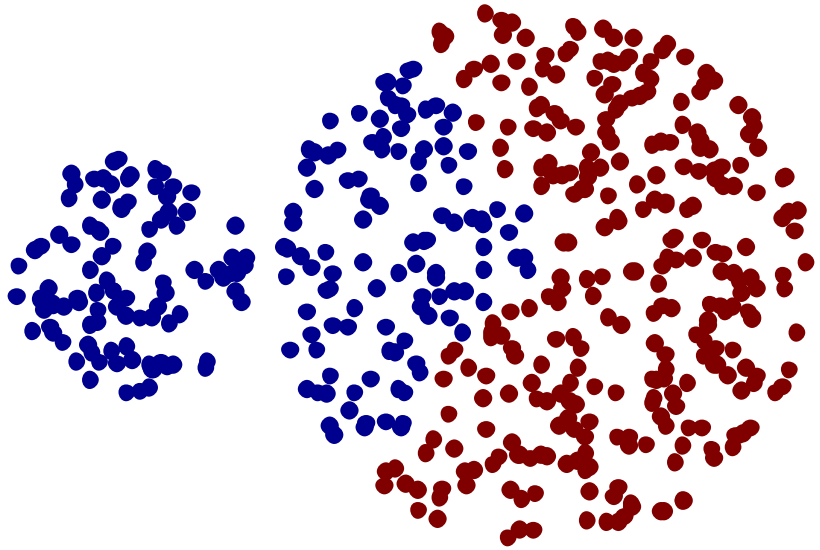
Original Points **Two Clusters**

- Less susceptible to noise and outliers

Limitations of MAX



Original Points



Two Clusters

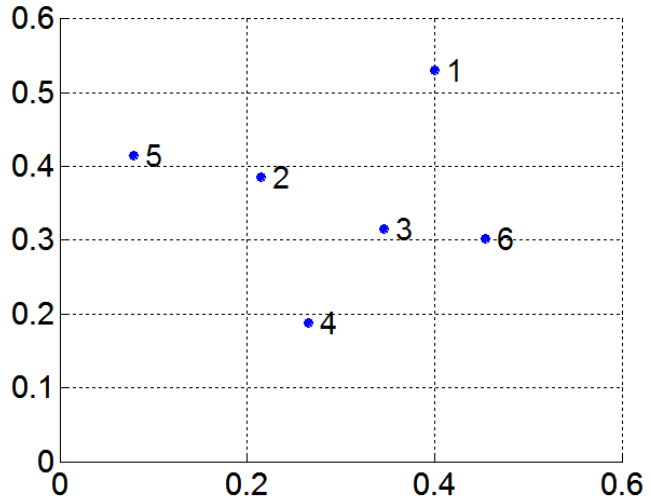
- Tends to break large clusters
- Biased towards globular clusters

Group Average

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

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

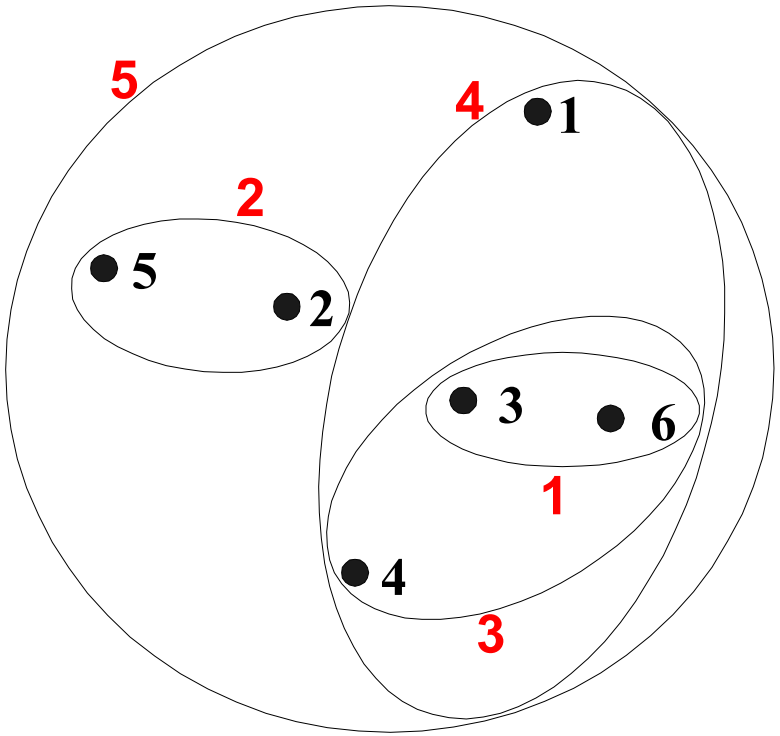
- Need to use average connectivity for scalability since total proximity favors large clusters



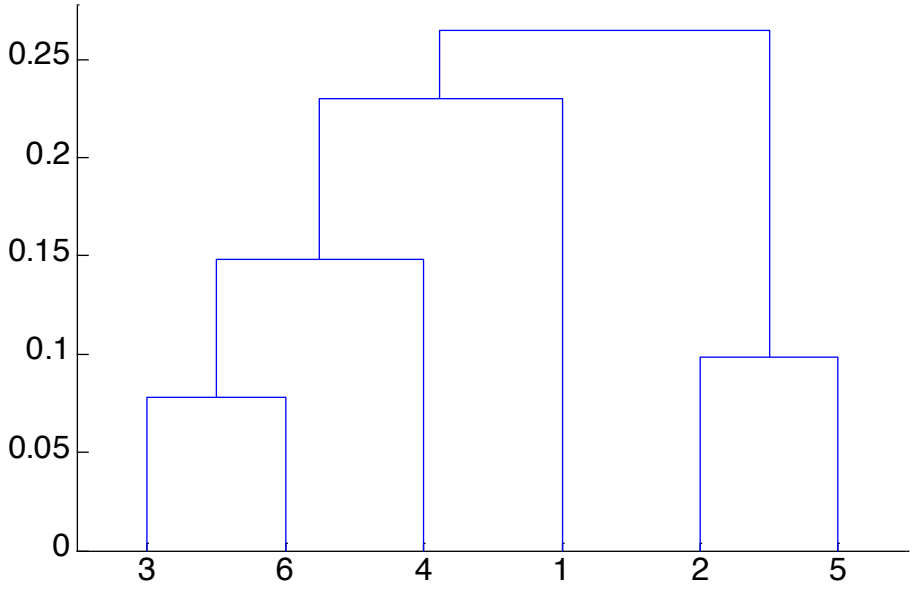
Distance Matrix:

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

Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

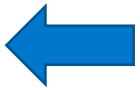
Hierarchical Clustering: Group Average

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

Hierarchical Clustering: Problems and Limitations

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

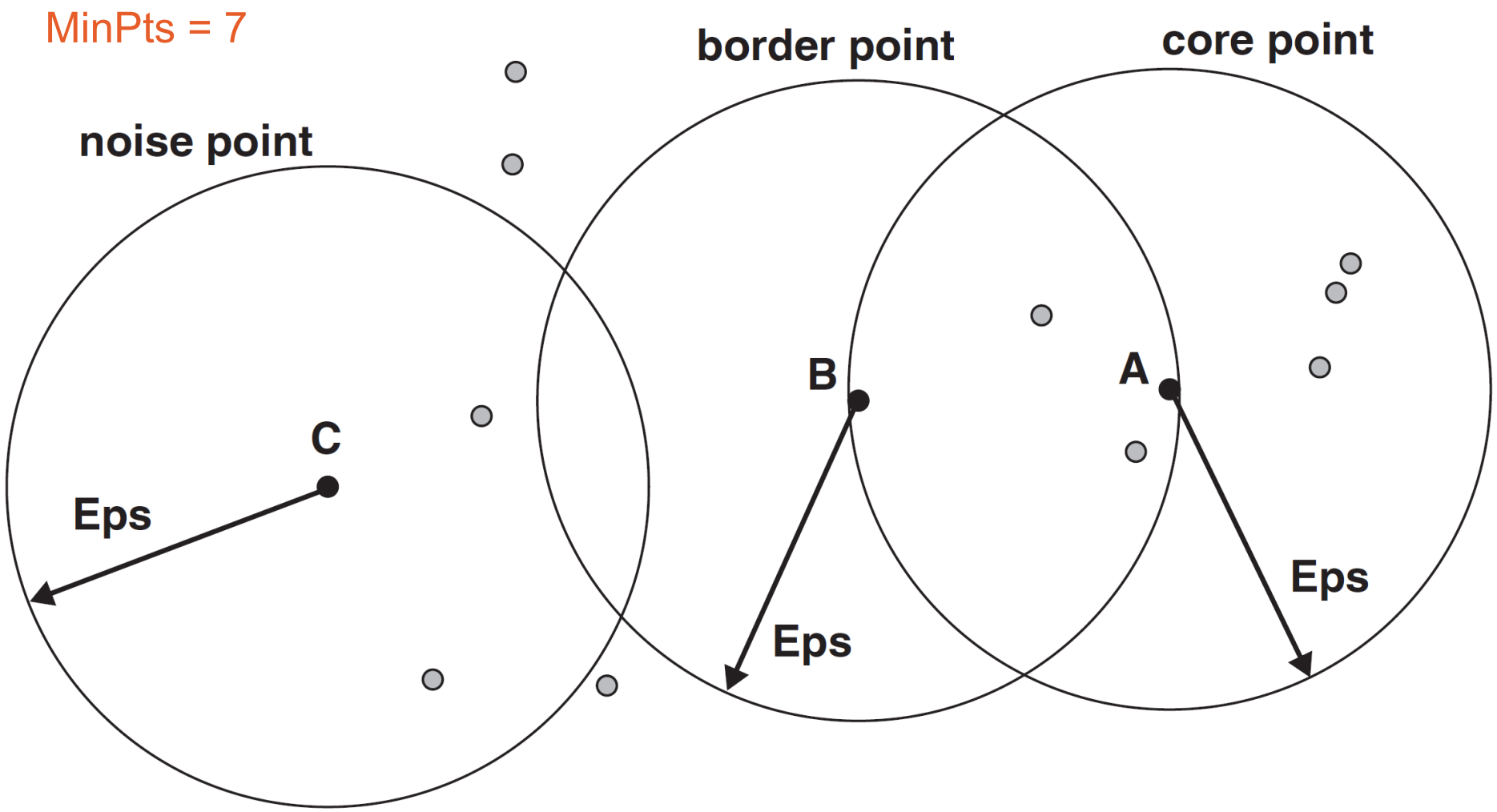
Clustering Algorithms

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

DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has at least a specified number of points (MinPts) within Eps (distance =< Eps)
    - These are points that are at the interior of a cluster
    - Counts the point itself
  - A border point is not a core point, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point

DBSCAN: Core, Border, and Noise Points



DBSCAN Algorithm

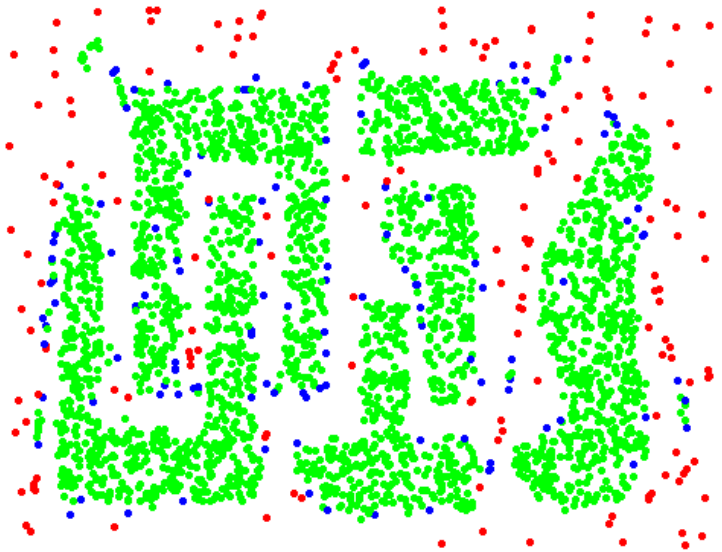
- Eliminate noise points
- Perform clustering on the remaining points

Algorithm 8.4 DBSCAN algorithm.

1: Label all points as core, border, or noise points.
2: Eliminate noise points.
3: Put an edge between all core points that are within *Eps* of each other.
4: Make each group of connected core points into a separate cluster.
5: Assign each border point to one of the clusters of its associated core points.

DBSCAN: Core, Border and Noise Points

Original Points

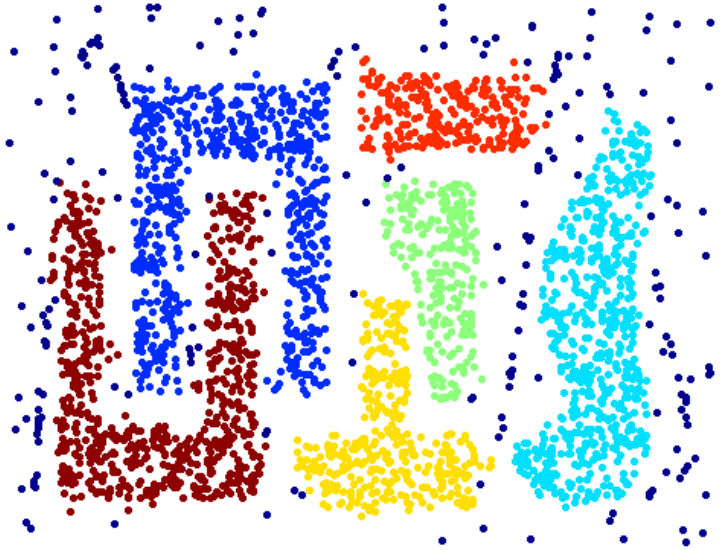


Point types: core, border and noise

Eps = 10, MinPts = 4

When DBSCAN Works Well

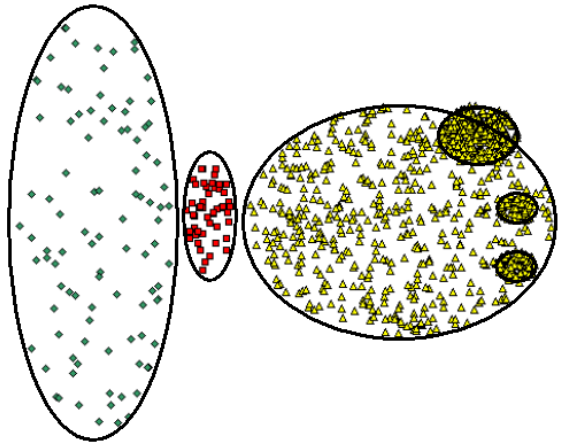
Original Points



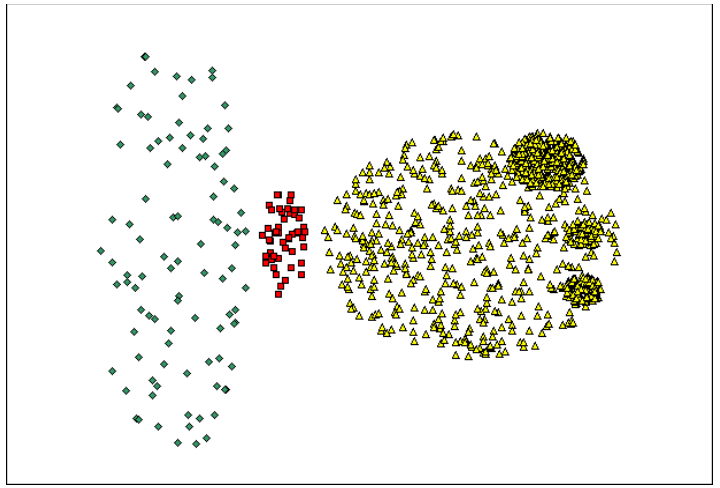
Clusters

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

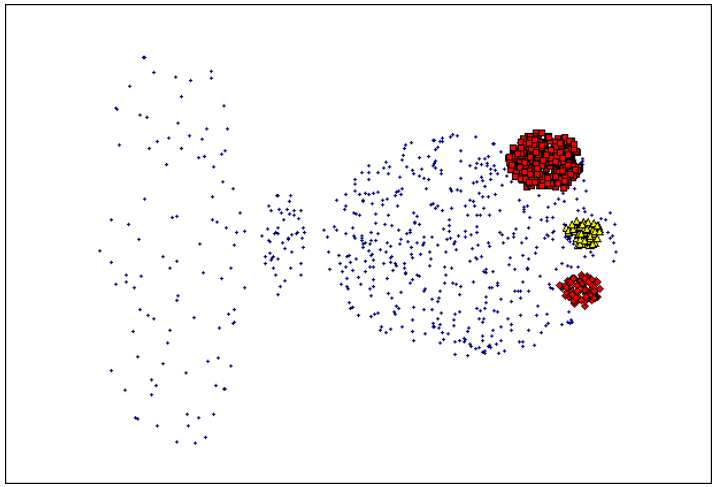
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.75).

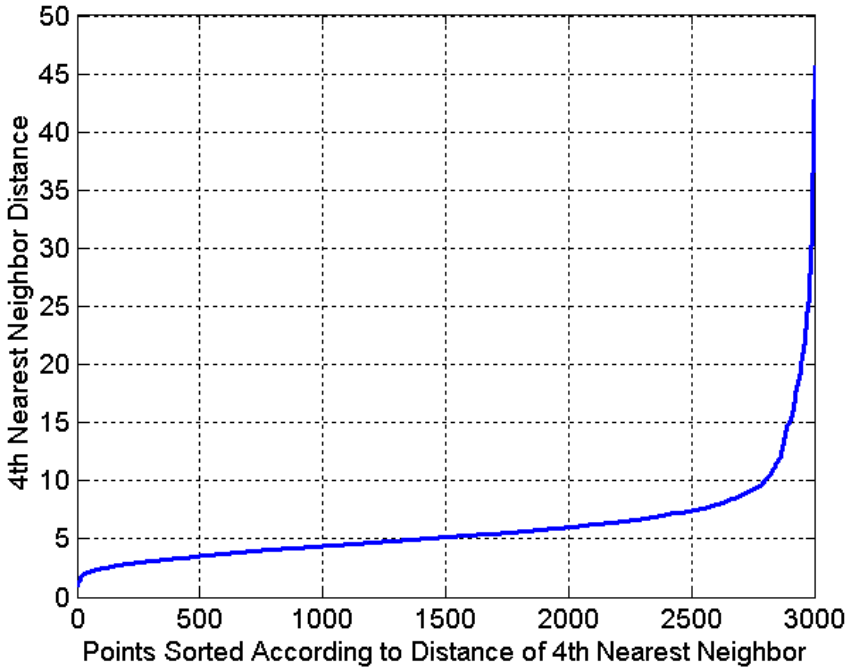


(MinPts=4, Eps=9.92)

- 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

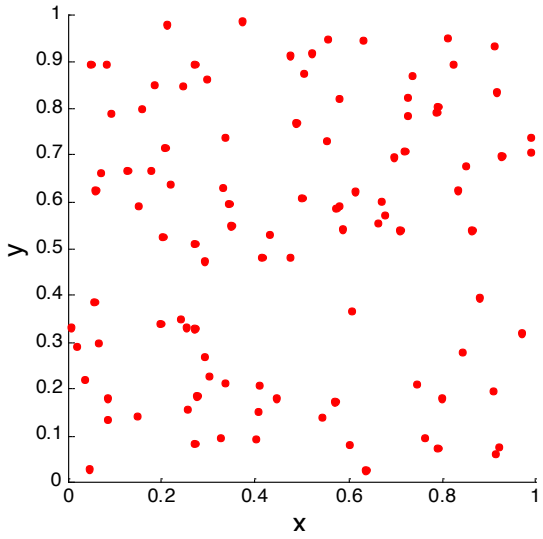


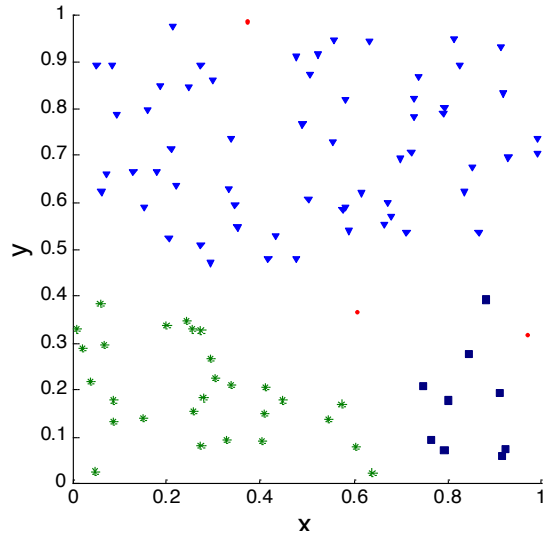
Cluster Validity

- 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

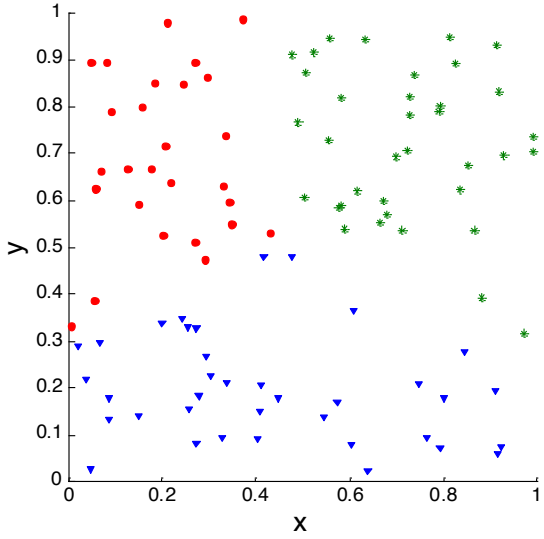
Random Points

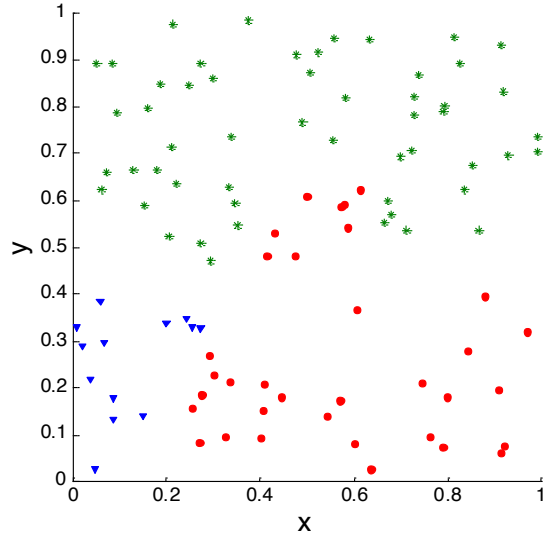




DBSCAN

K-means





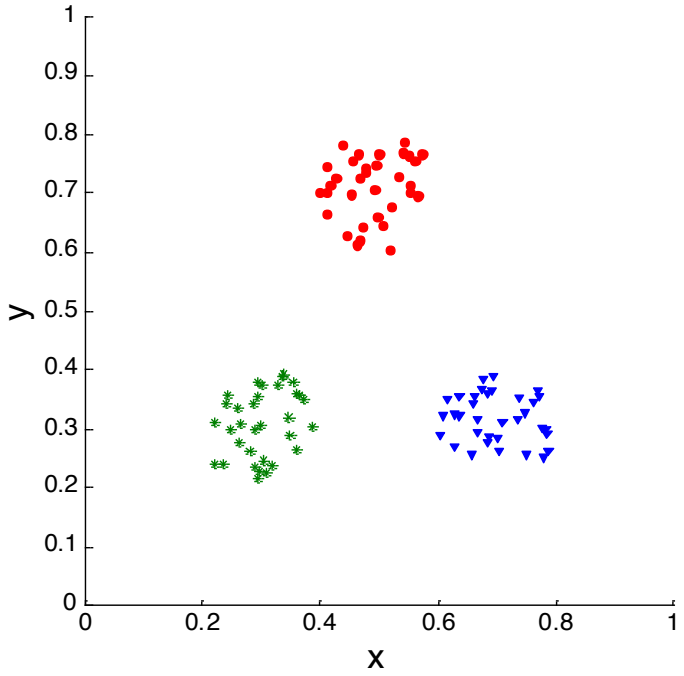
Complete Link

Measuring Cluster Validity Via Correlation

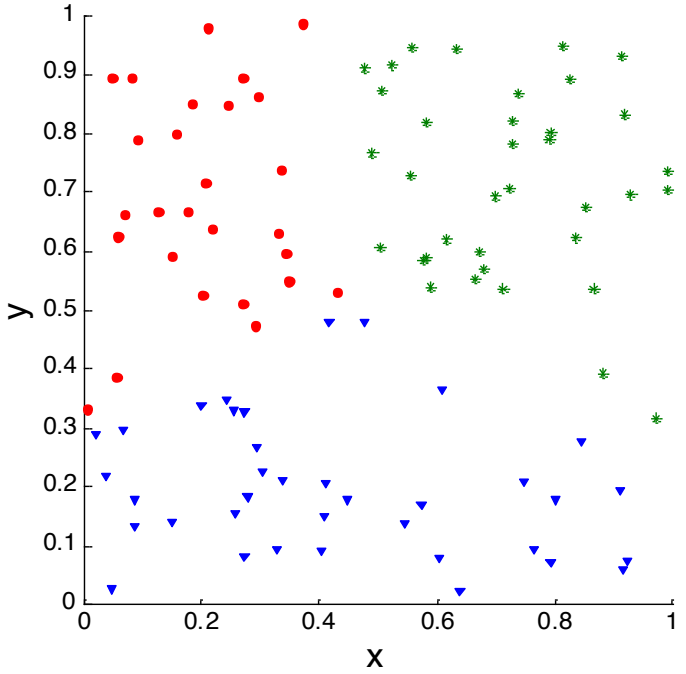
- Two matrices
  - Proximity Matrix
  - Ideal Similarity Matrix
    - One row and one column for each data point
    - An entry is 1 if the associated pair of points belong to the same cluster
    - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
  - Since the matrices are symmetric, only the correlation between n(n-1) / 2 entries needs to be calculated.
- High 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 ideal similarity and proximity matrices for the K-means clustering of the following two data sets.



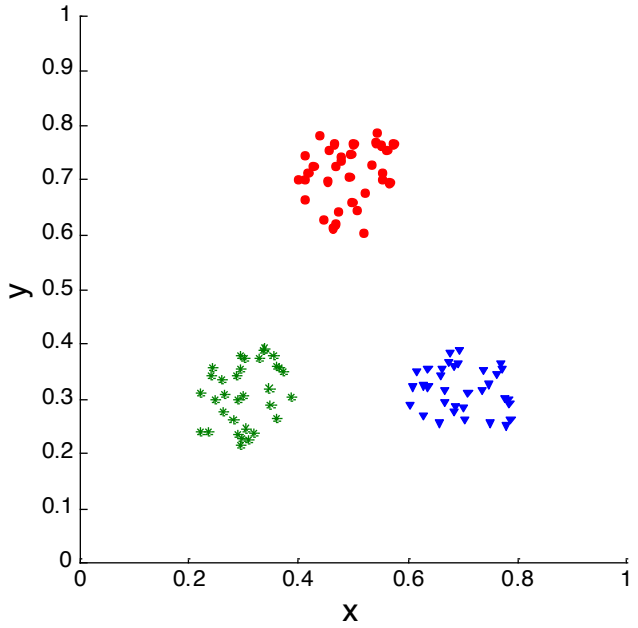
Corr = 0.9235

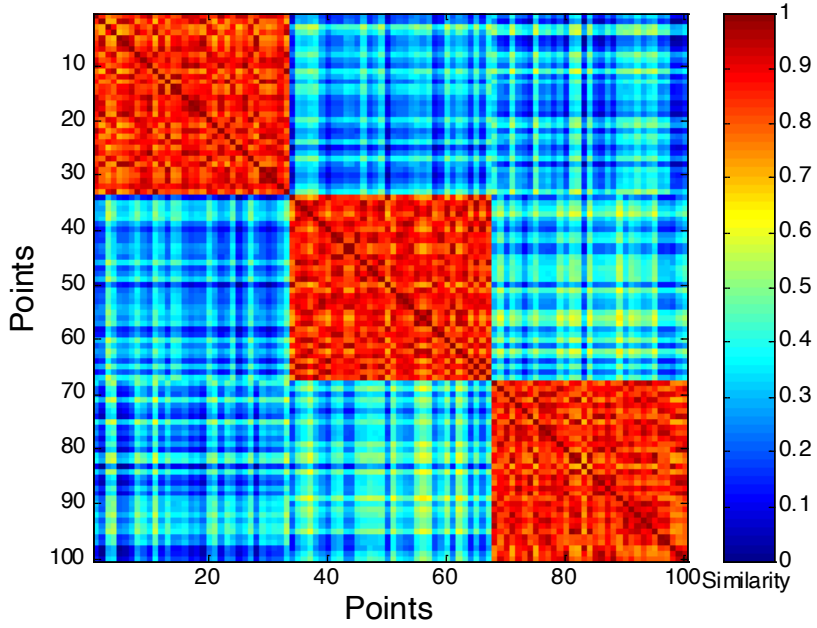


Corr = 0.5810

Using Similarity Matrix for Cluster Validation

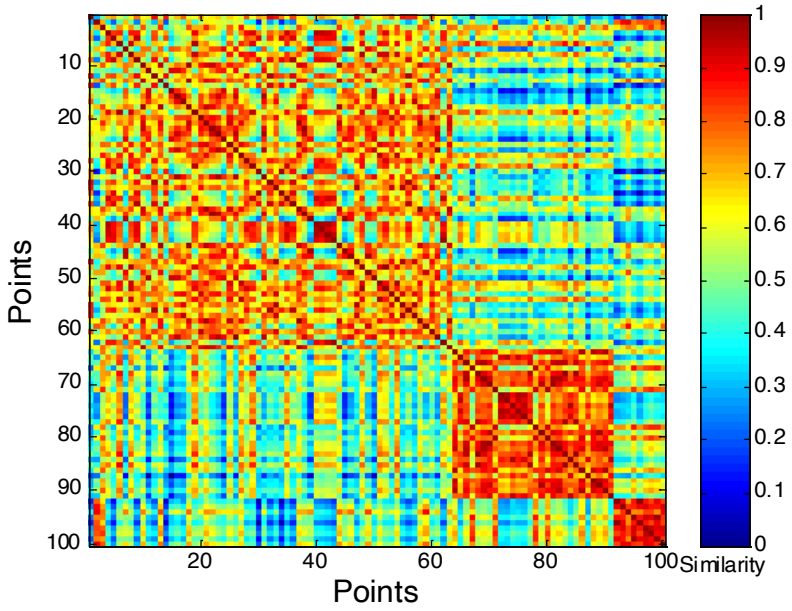
- Order the similarity matrix with respect to cluster labels and inspect visually.

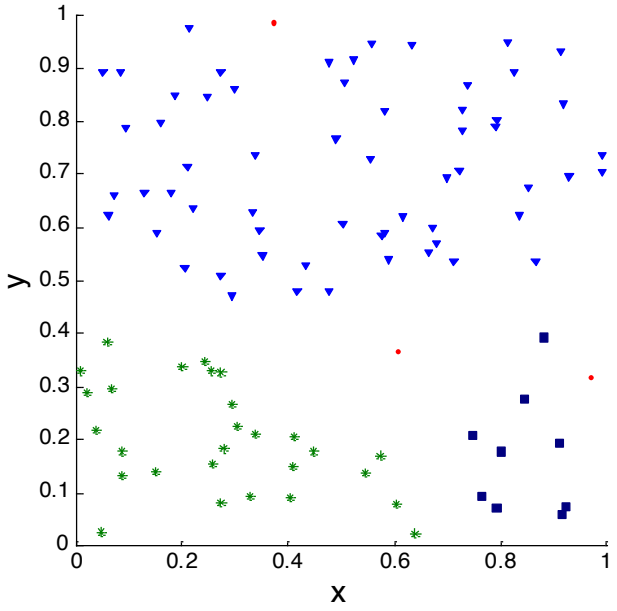




Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

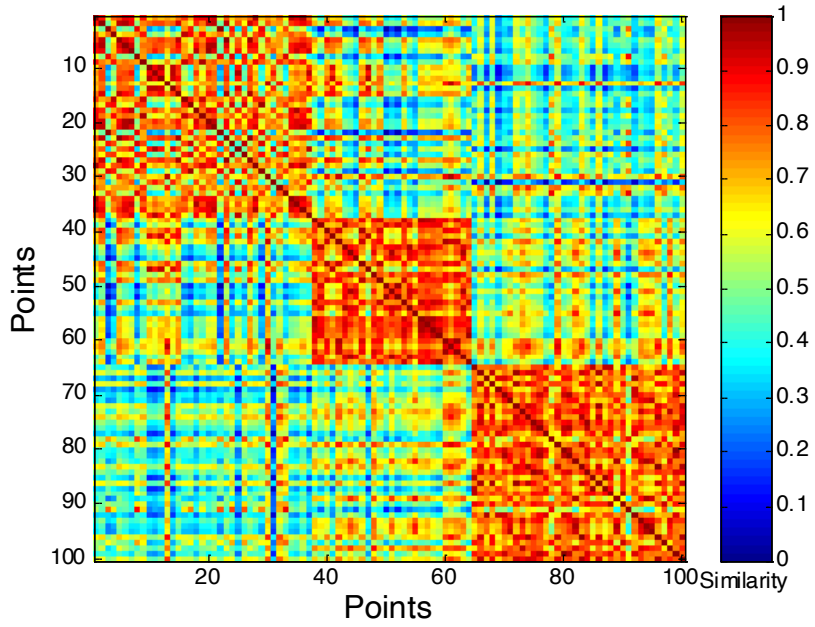


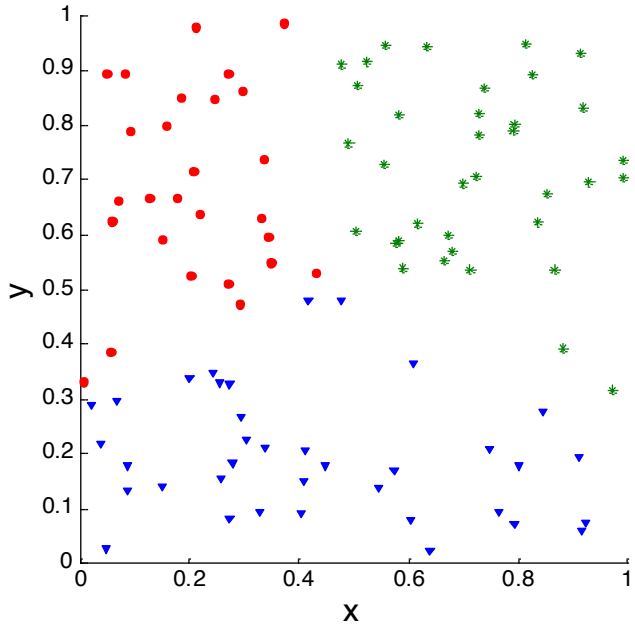


DBSCAN

Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

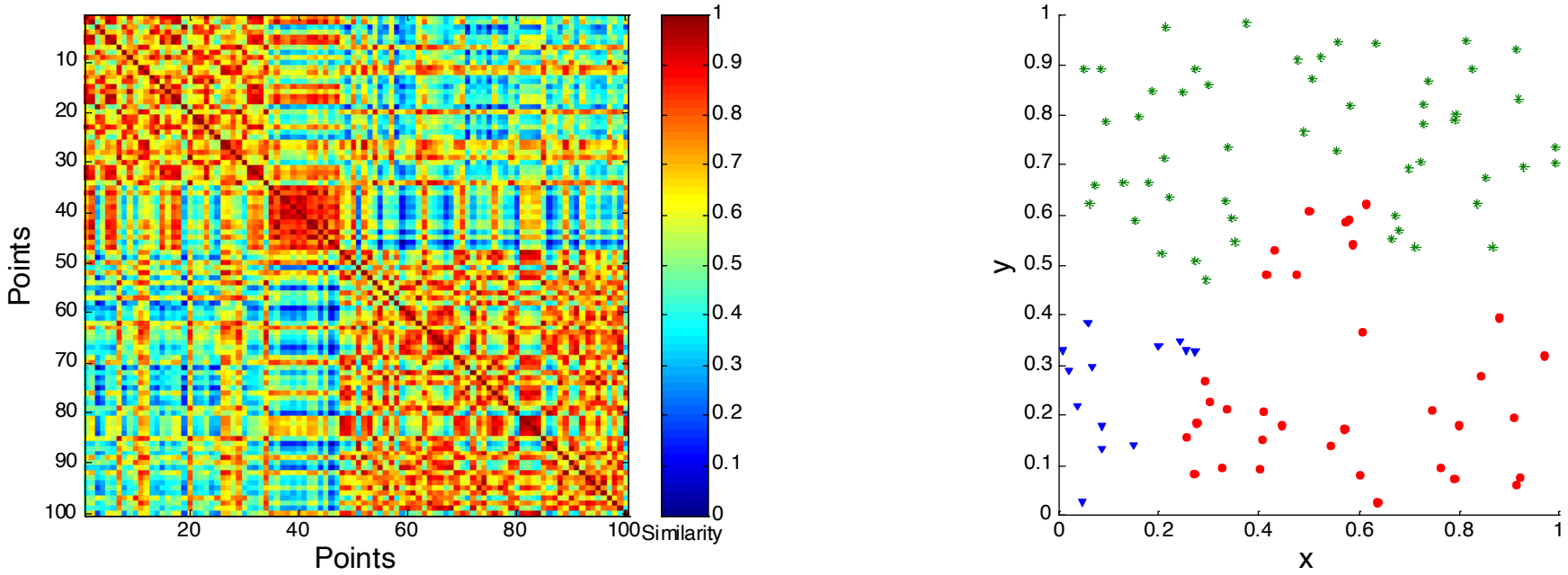




K-means

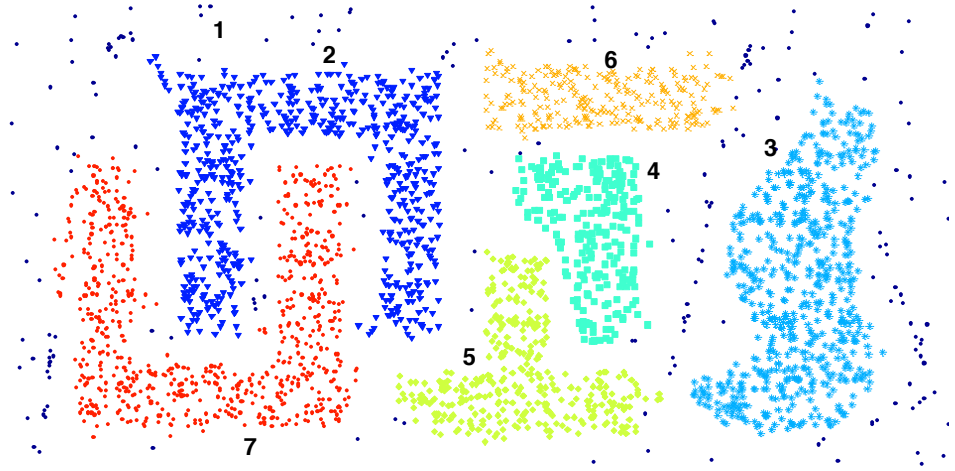
Using Similarity Matrix for Cluster Validation

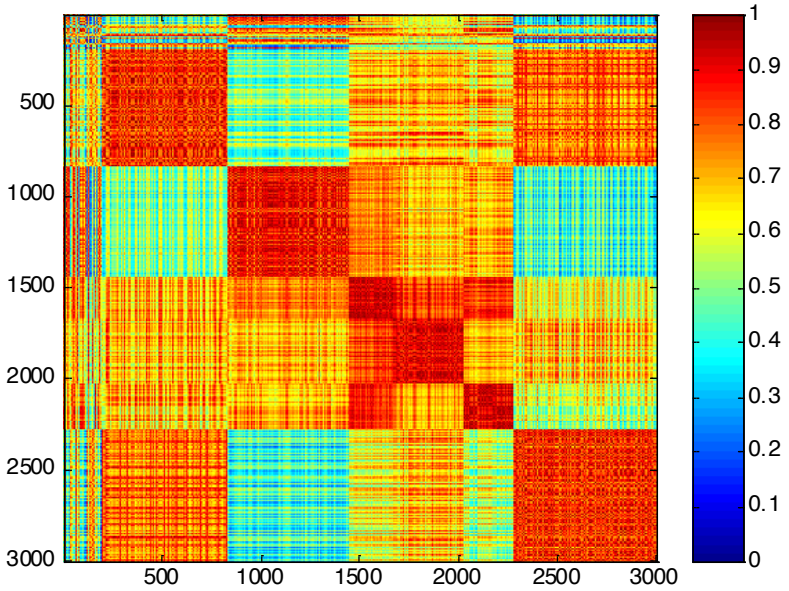
- Clusters in random data are not so crisp



Complete Link

Using Similarity Matrix for Cluster Validation



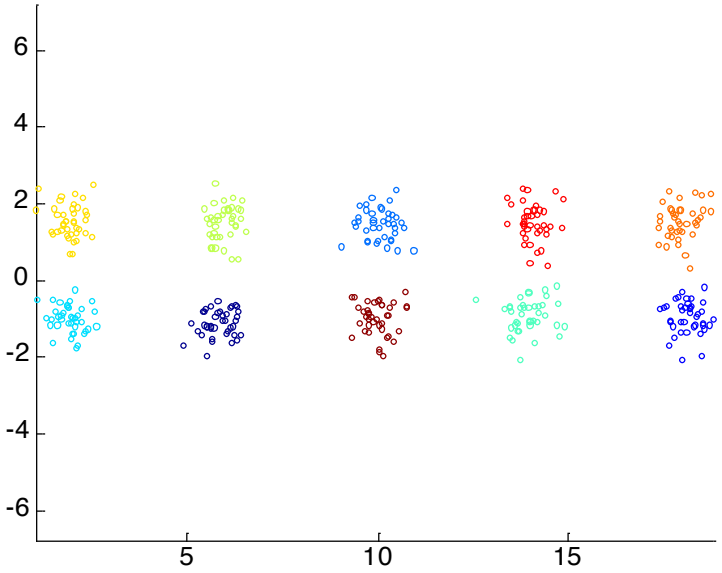


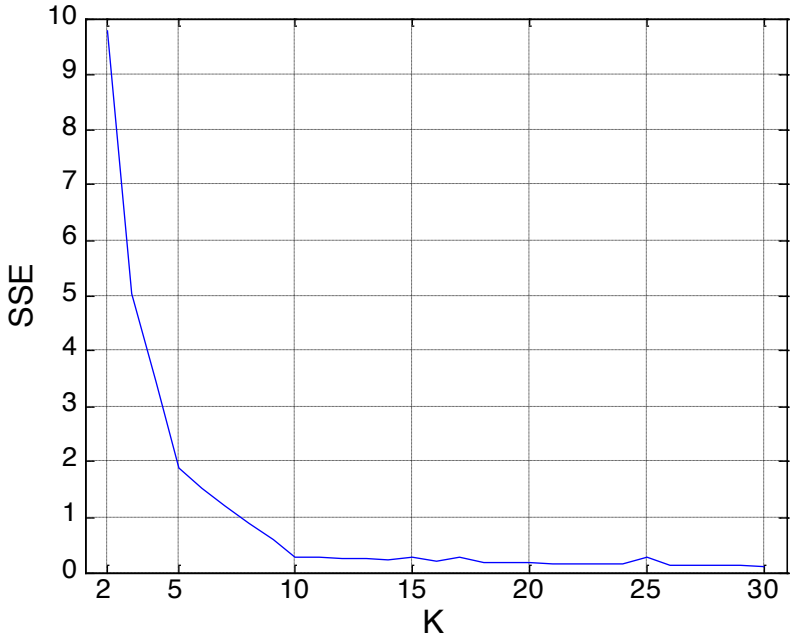
DBSCAN

Internal Measures: SSE

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

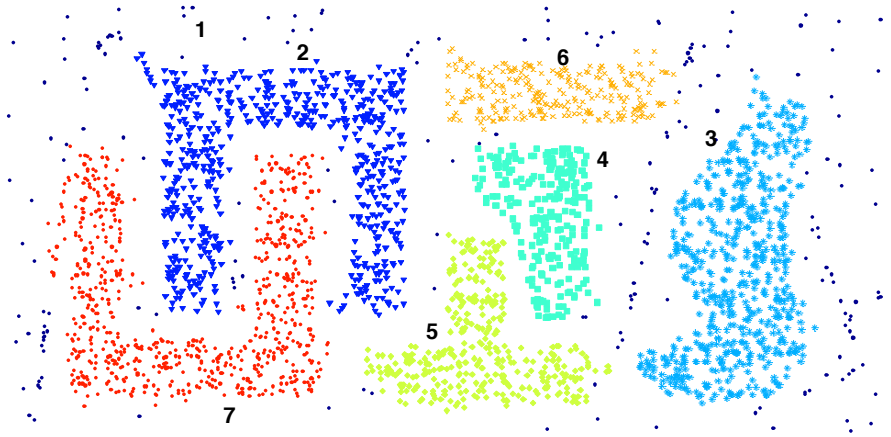
- Internal Index: Used to measure the goodness of a clustering structure without respect to external information
  - SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters

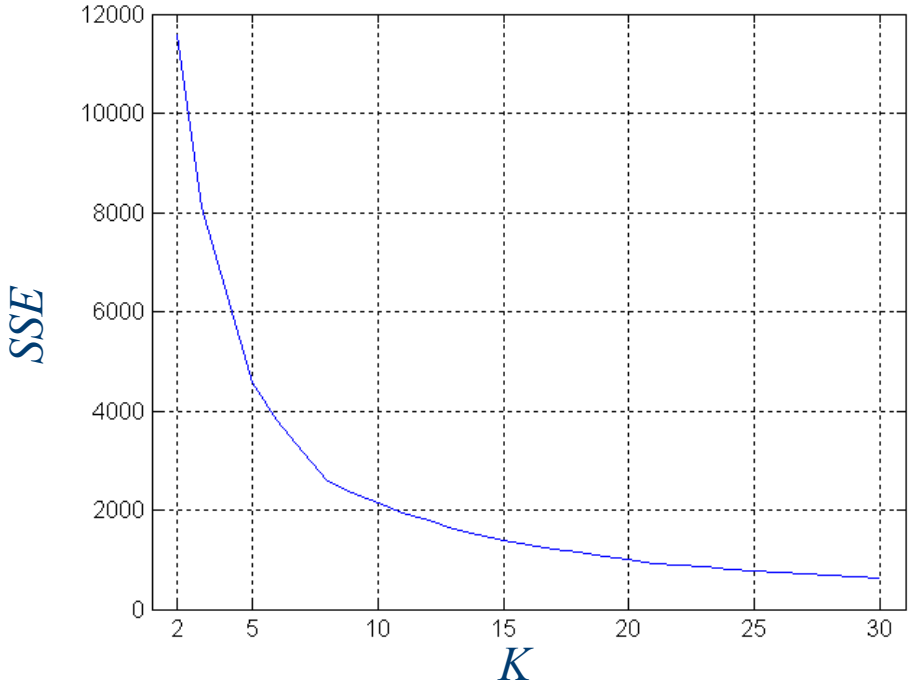




Internal Measures: SSE

- SSE curve for a more complicated data set





SSE of clusters found using K-means

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)

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

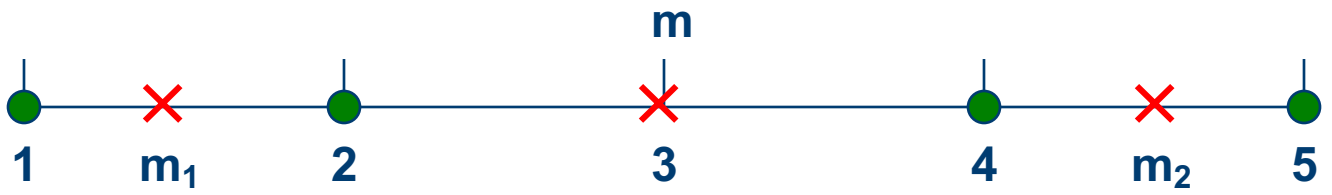
  - Separation is measured by the between cluster sum of squares

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

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

Internal Measures: Cohesion and Separation

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



K=1 cluster:

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

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

$$Total = 10 + 0 = 10$$

K=2 clusters:

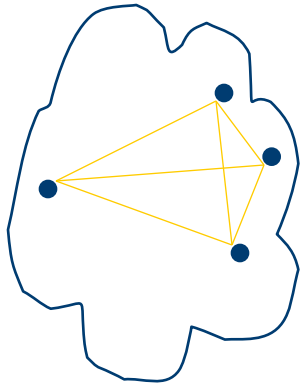
$$SSE = WSS = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

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

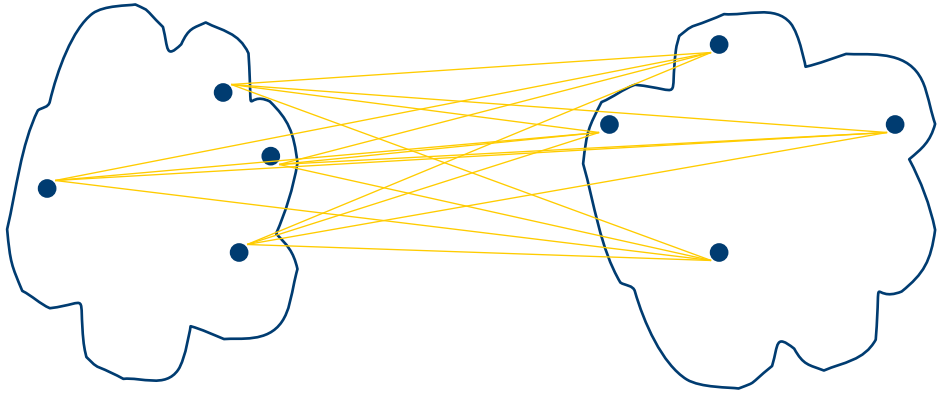
$$Total = 1 + 9 = 10$$

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

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

Participant Leaders

Points	Participant	Points	Participant