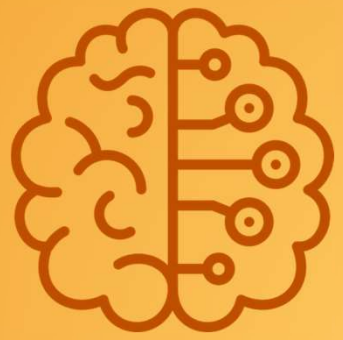
CS 4375 – Introduction to Machine Learning

Unsupervised Learning & Clustering

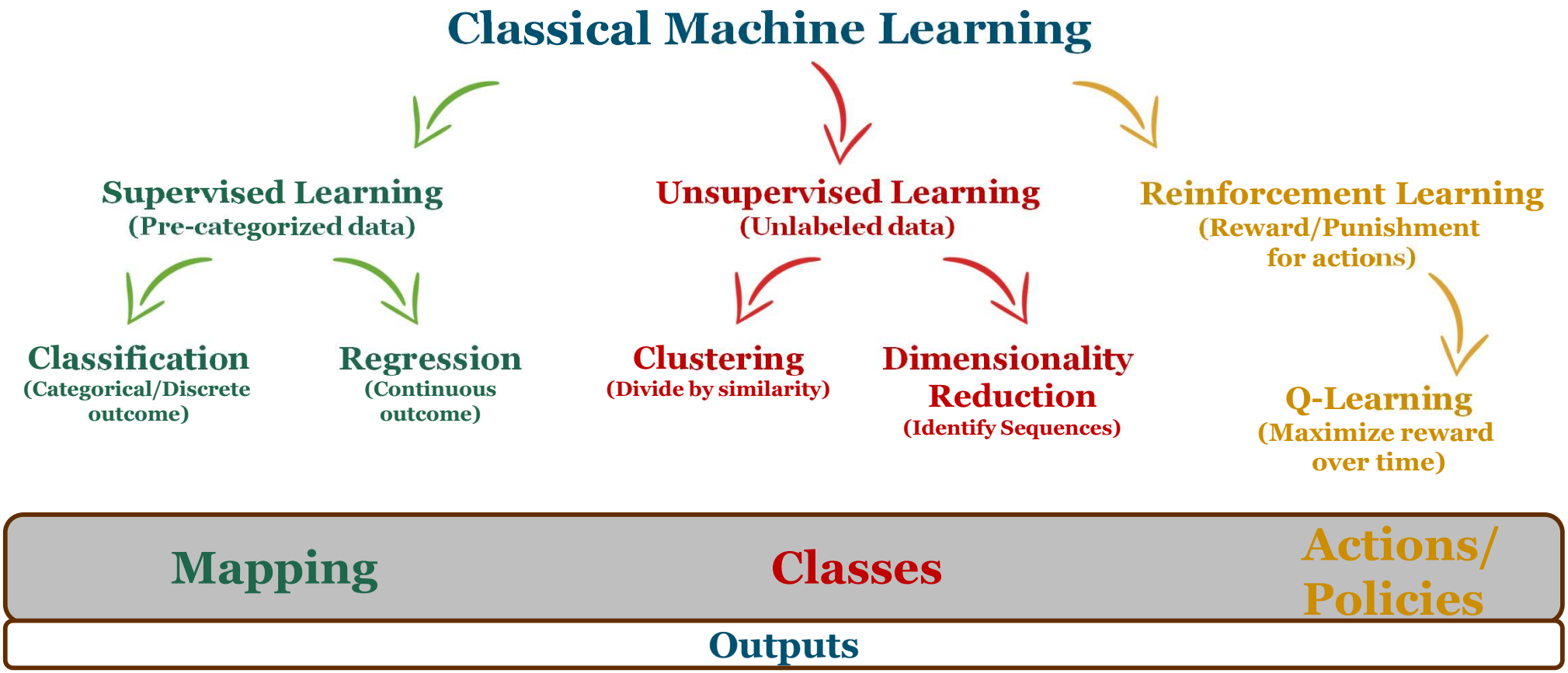
Erick Parolin

THE UNIVERSITY OF TEXAS AT DALLAS



[Based on the slides of Tan, Steinbach, Kumar]

Classical Machine Learning



Classical Machine Learning

Type of Supervision

What is Provided to Learn

What is Being Learned	Labeled Examples	Reward	Nothing
Discrete Function	Classification		Clustering
Continuous Function	Regression		
Policy		Reinforcement Learning	

Classical Machine Learning

Type of Supervision

What is Provided to Learn

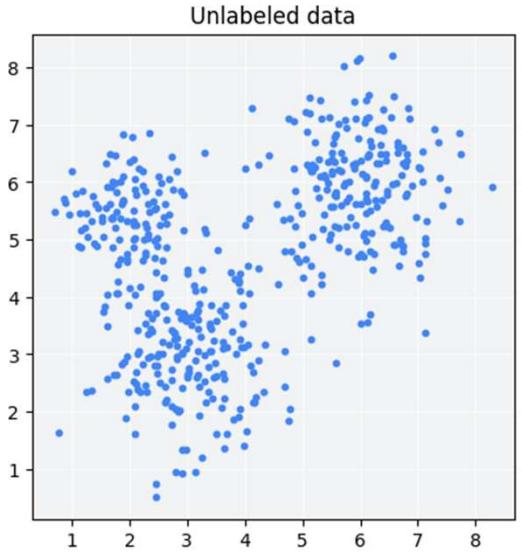
What is Being Learned	Labeled Examples	Reward	Nothing
Discrete Function	Classification		Clustering
Continuous Function	Regression		
Policy		Reinforcement Learning	

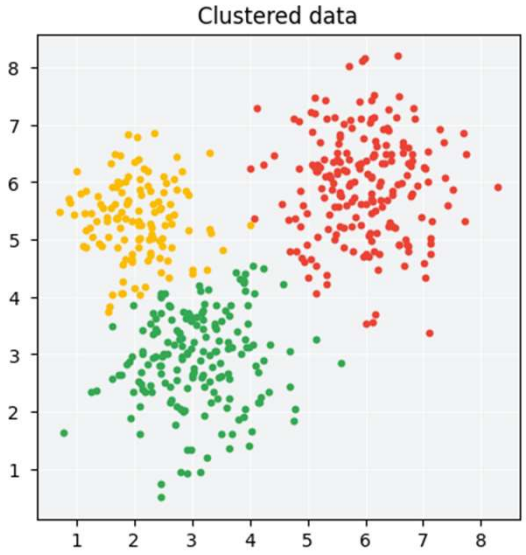
Clustering

- Requires data, but no labels
- Main task is to **detect patterns**
  - **Customer Segmentation:** Group customers based on purchasing behavior, demographics, or interests to tailor marketing strategies for each segment.
  - **Document Clustering**: Organize large collections of documents (e.g., news articles) into thematic clusters for easy retrieval or recommendation systems.
  - **Anomaly Detection**: Identify unusual patterns in data, such as fraud detection in transactions, by clustering typical behavior and flagging outliers.
  - **Image Segmentation:** Group pixels in images to identify and separate different regions, like segmenting objects in photos for computer vision tasks.
  - **Social Network Analysis:** Identify communities or friend groups within social networks by clustering users with similar interaction patterns.
  - **Genomics**: Cluster gene expression data to find genes with similar expression patterns, which can reveal biological pathways or disease markers.

Clustering

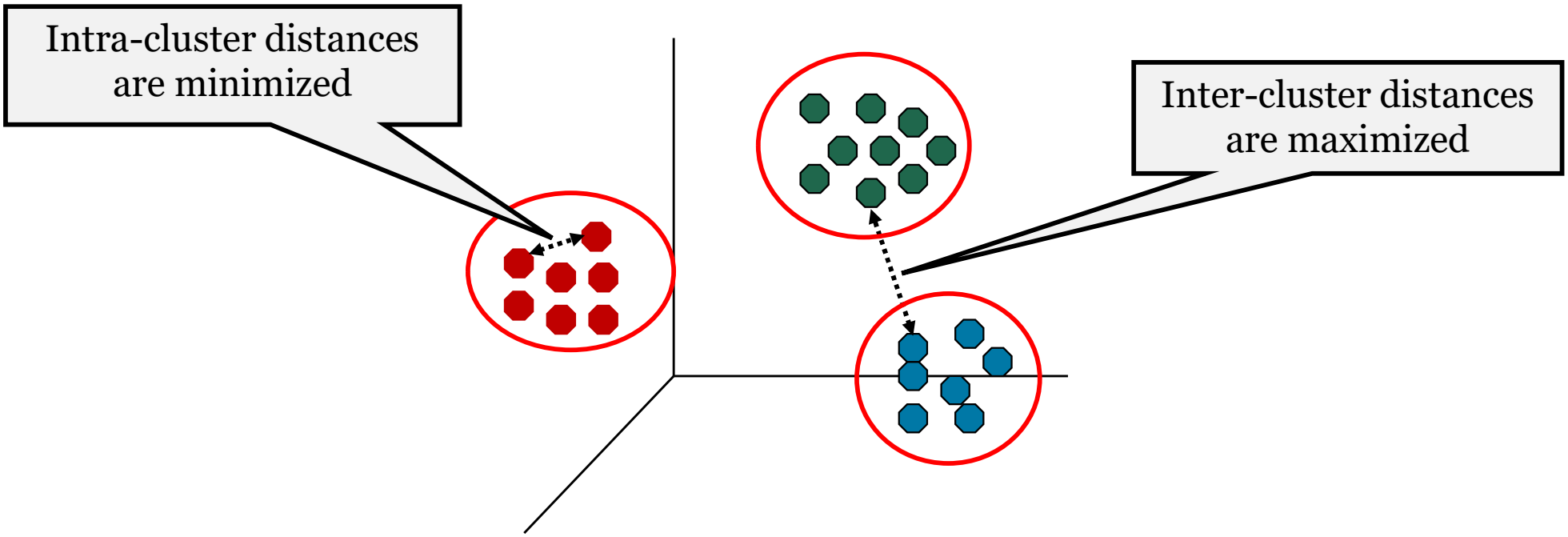
- **Basic idea:** group together similar instances
- Useful for finding the important parameters/features of a dataset





Clustering

- **Goal:** 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



Clustering

Notion of a Cluster can be Ambiguous

How many clusters?

Six Clusters

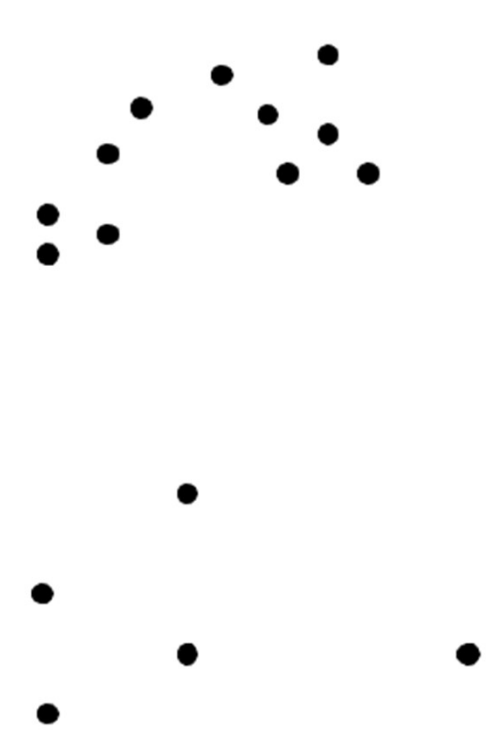
Two Clusters

Four Clusters

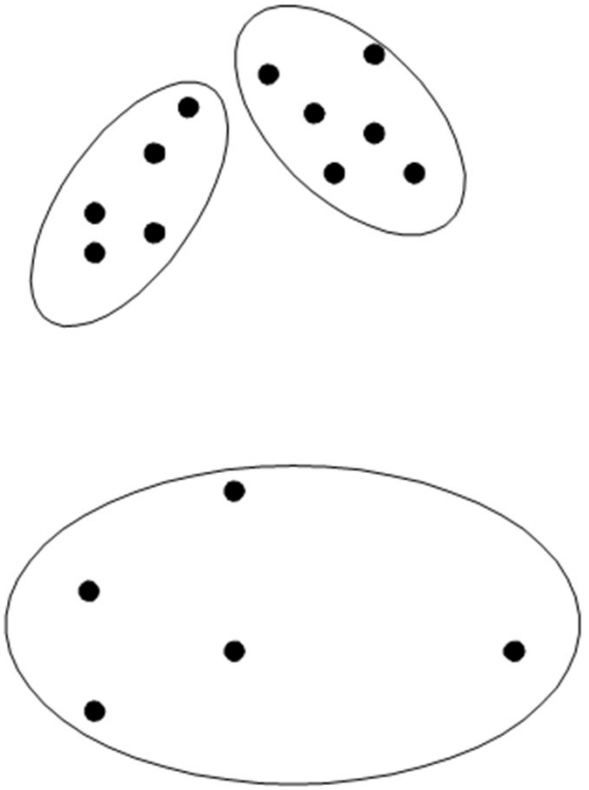
Types of Clustering

- 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

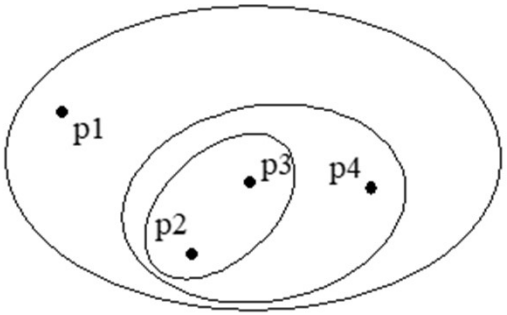


Original Points

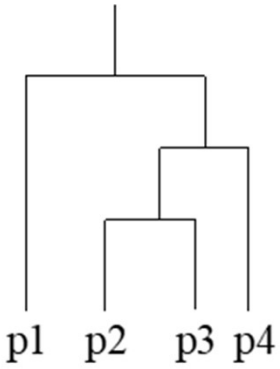


A Partitional Clustering

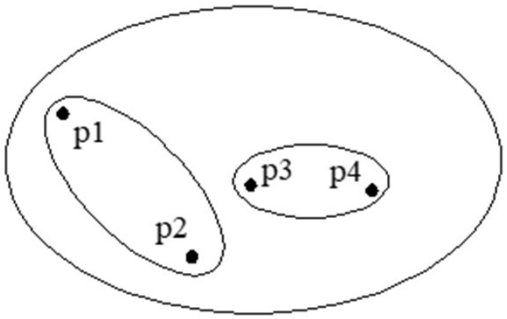
Hierarchical Clustering



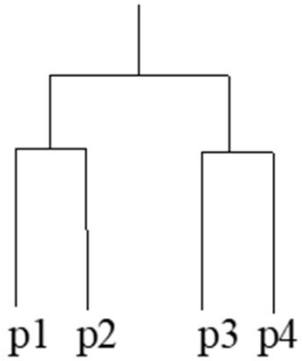
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Clustering – Definition

Formal Task

- **Input:** a collection of points $x^{(1)}, \ldots, x^{(m)} \in \mathbb{R}^n$, an integer k
- **Output**: A partitioning of the input points into k sets that minimizes some metric of closeness.

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

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

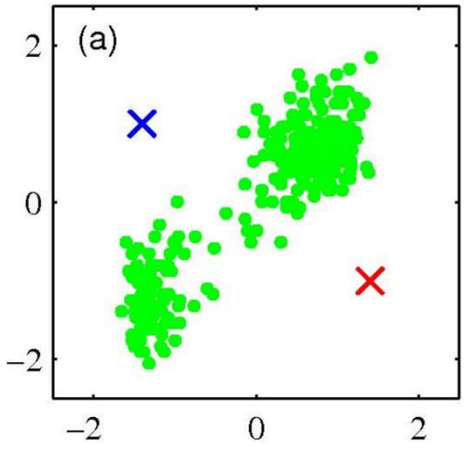
Evaluating K-means Clusters

- 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.

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

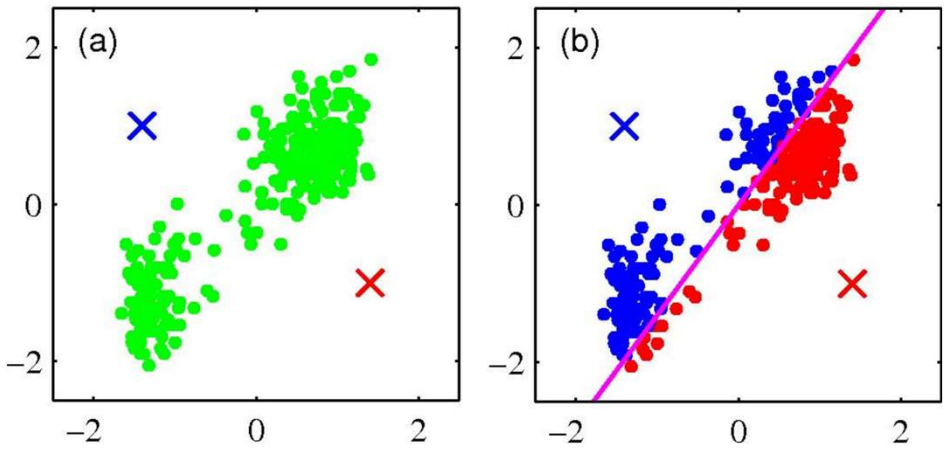
  - x is a data point in cluster C_i and m_i is the representative point for cluster C_i
  - Given two clusters, we can choose the one with the smallest error
  - One easy way to reduce SSE is to increase K, the number of clusters
    - A good clustering with smaller K can have a lower SSE than a poor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

K-means Clustering – Illustration



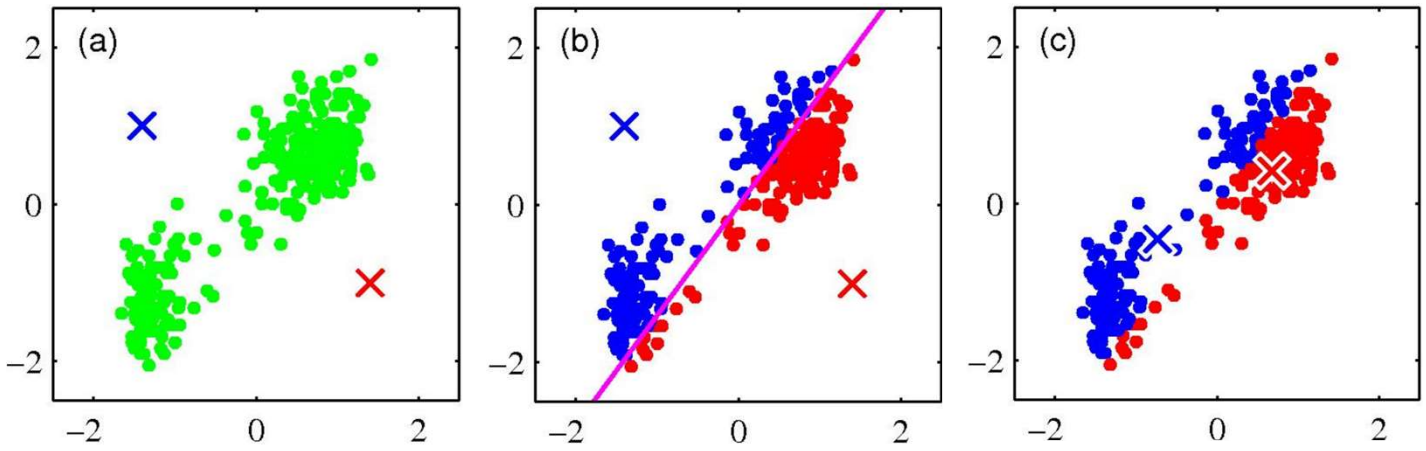
Pick *k* random points as cluster centers (means)

K-means Clustering – Illustration



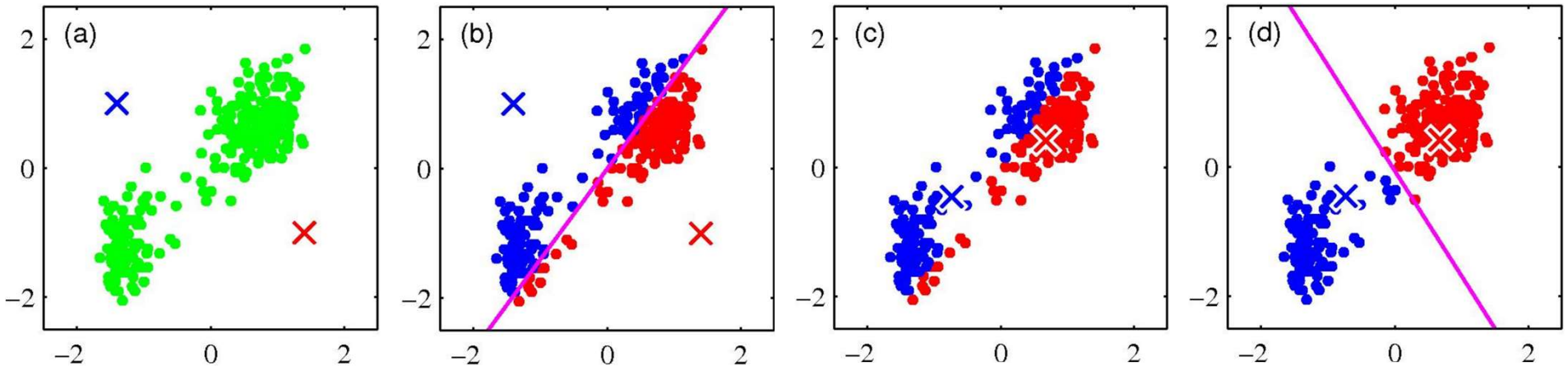
Iterative Step 1: Assign data instances to closest cluster center

K-means Clustering – Illustration



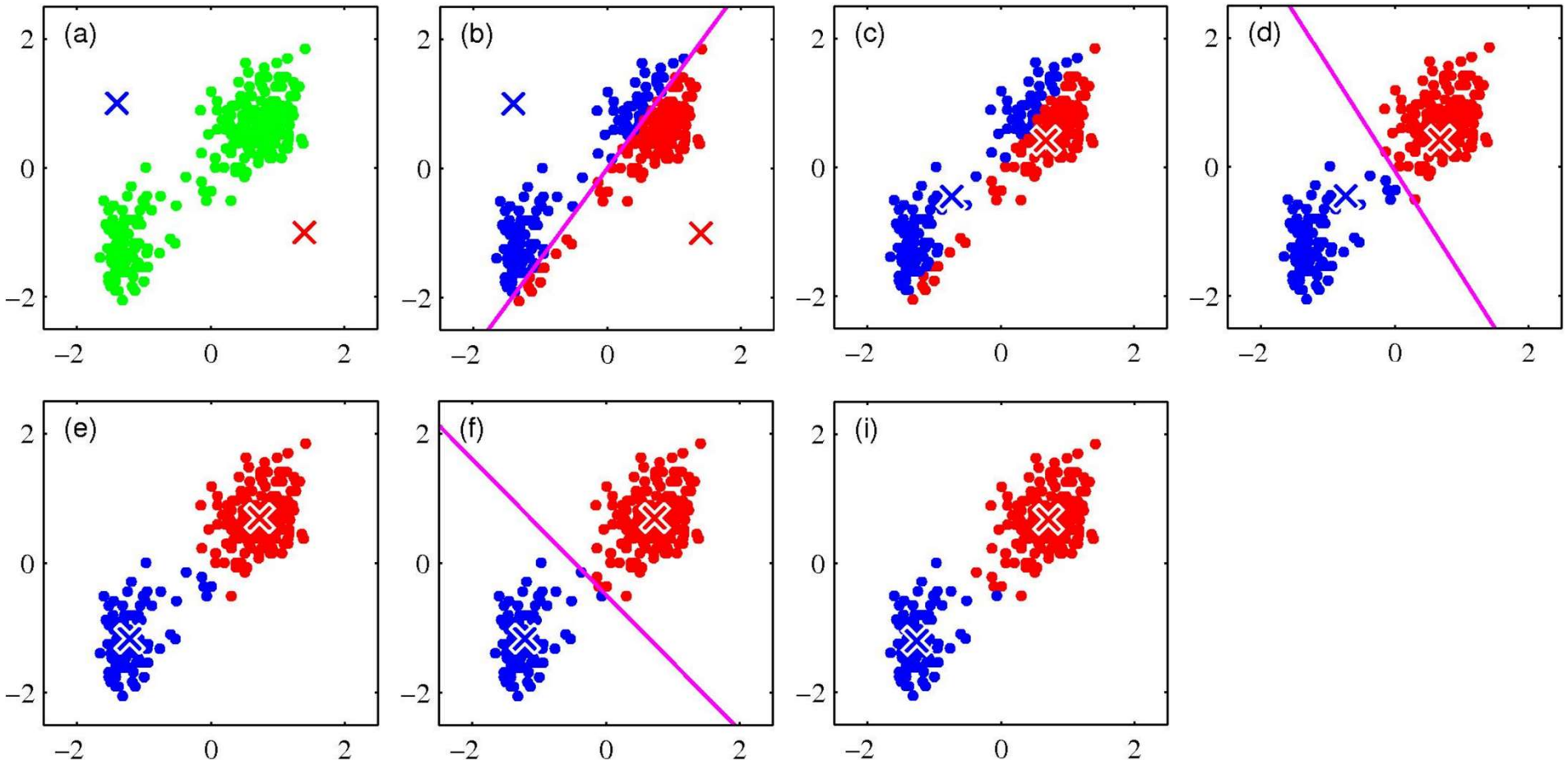
Iterative Step 2: Change the cluster center to the average of the assigned points

K-means Clustering – Illustration

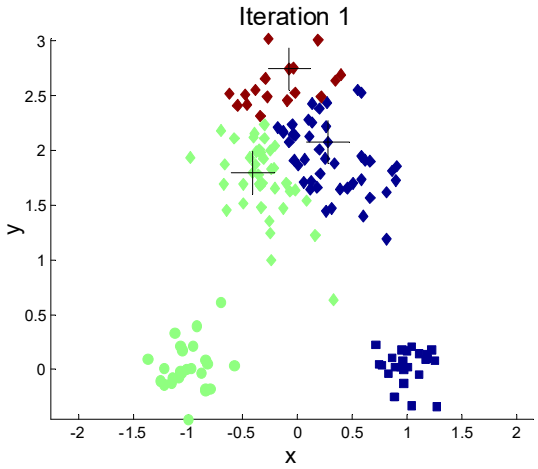


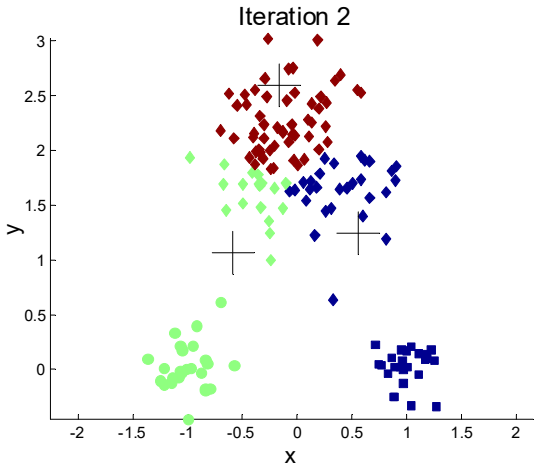
Repeat until convergence…

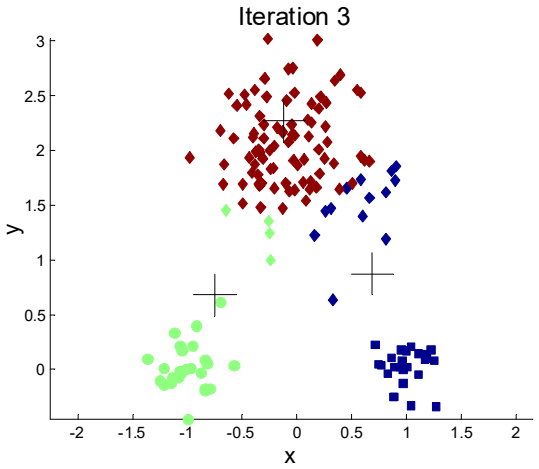
K-means Clustering – Illustration

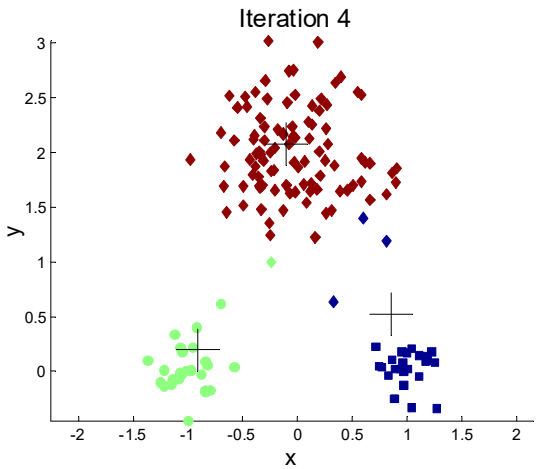


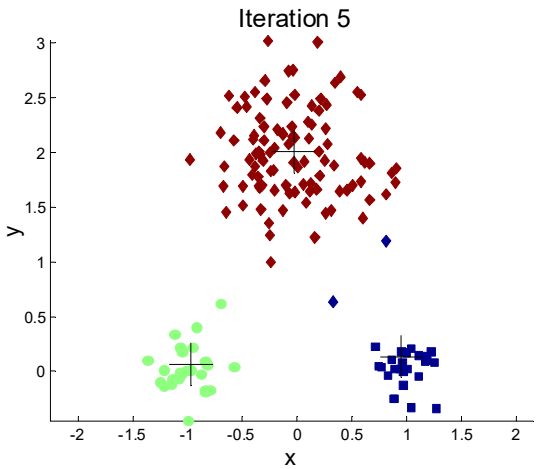
K-means Clustering – Another Illustration

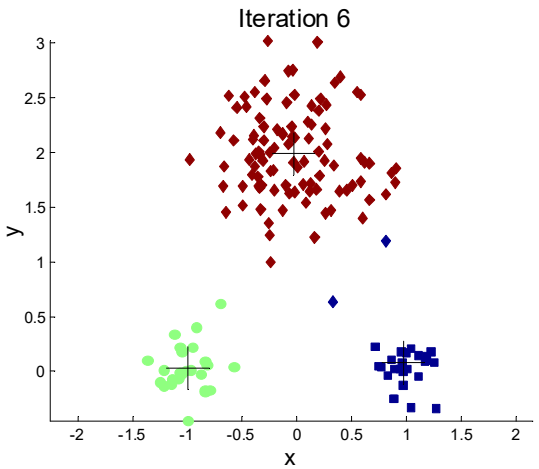






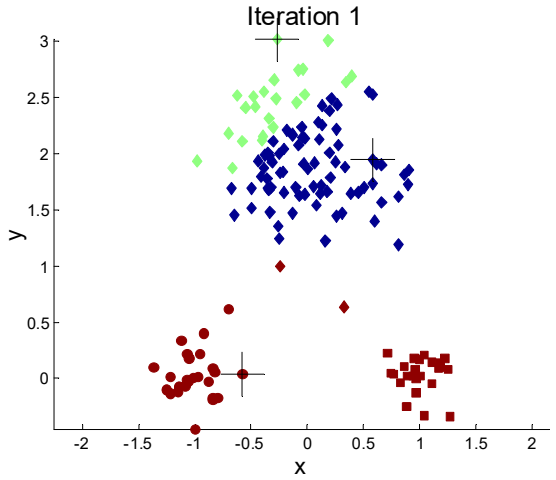


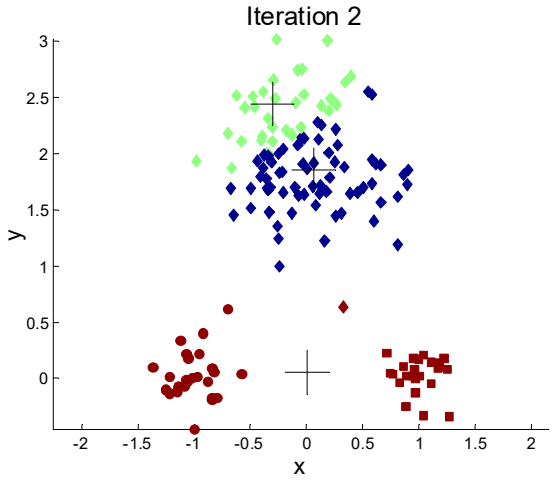


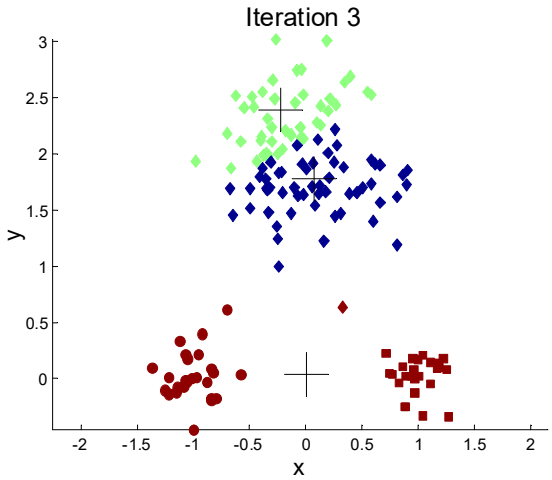


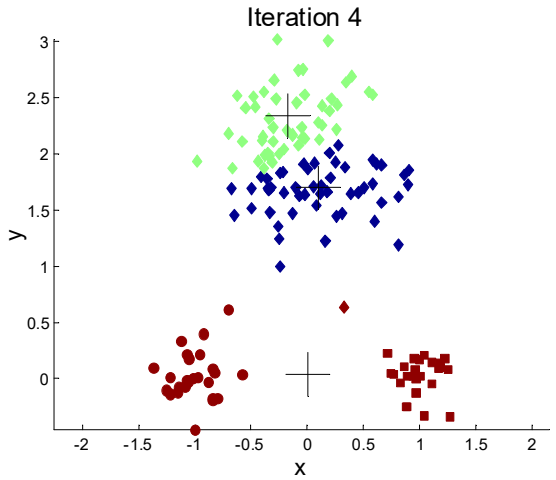
K-means Clustering – Another Illustration

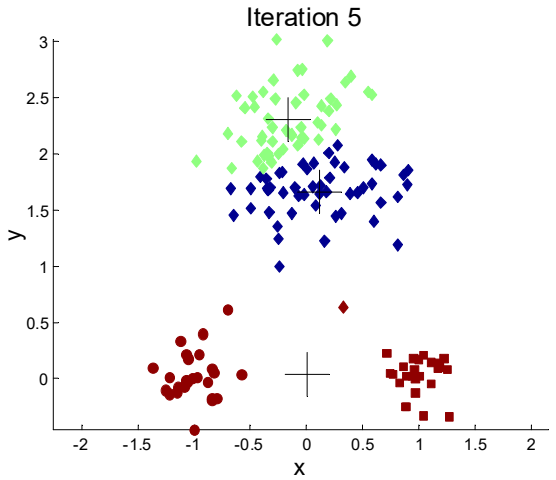
What if we choose another initial centroids?











K-means Clustering

Problems with Selecting Initial Points

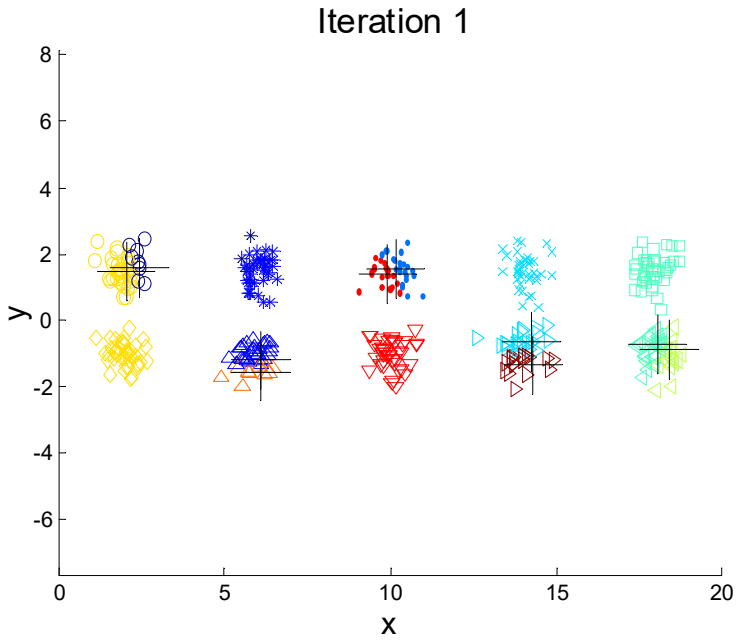
- If there are K "*real*" clusters then the chance of selecting one centroid from each cluster is small.
  - Chance is relatively small when K is large
  - If clusters are the same size, n, then

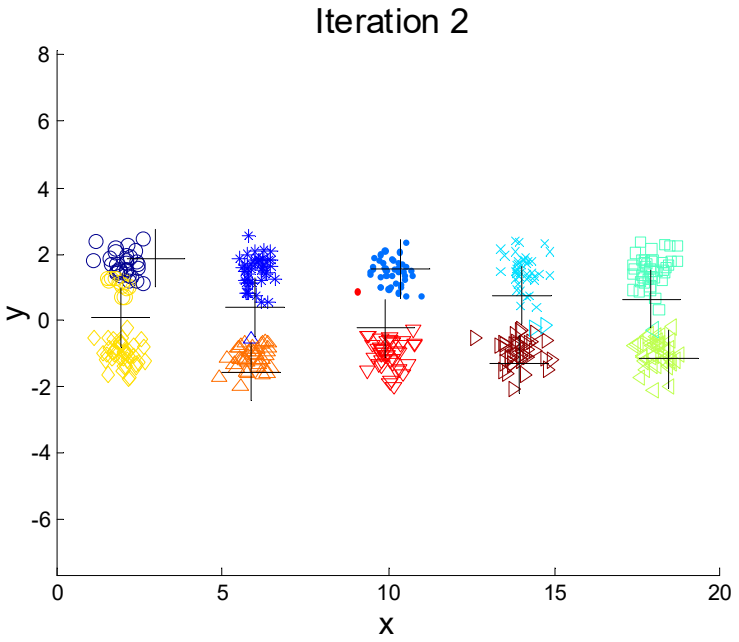
$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K! n^K}{(Kn)^K} = \frac{K!}{K^K}$$

- For example, if K = 10, then probability = 10!/1010 = 0.00036
  - Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't
  - Consider an example of five pairs of clusters

K-means Clustering

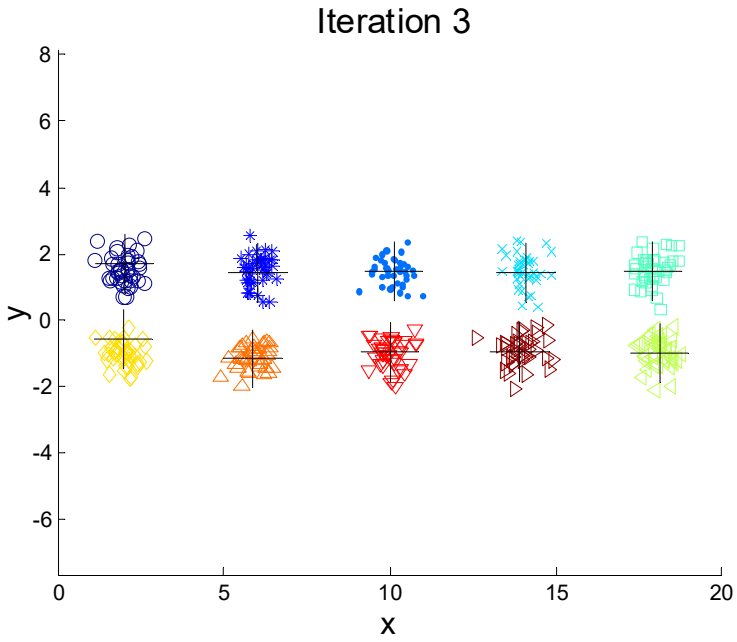
Example with 10 clusters: Starting with two initial centroids in one cluster of each pair of clusters

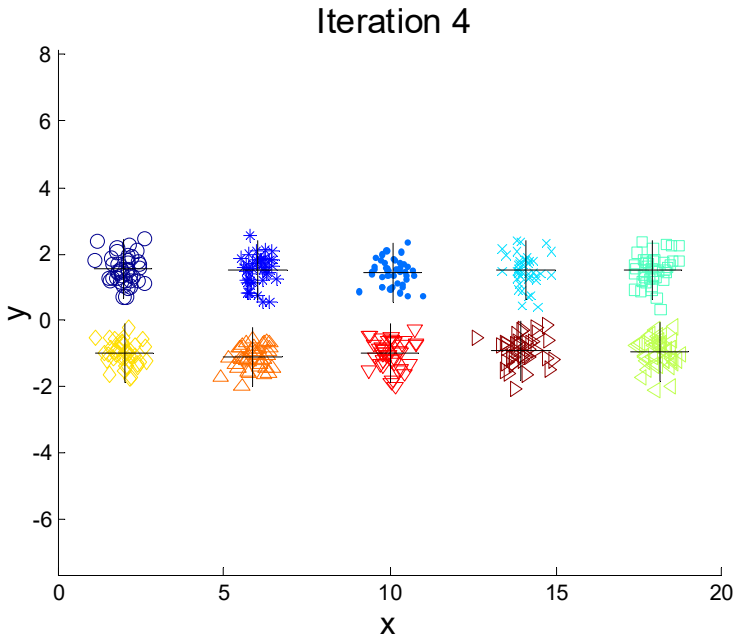




K-means Clustering

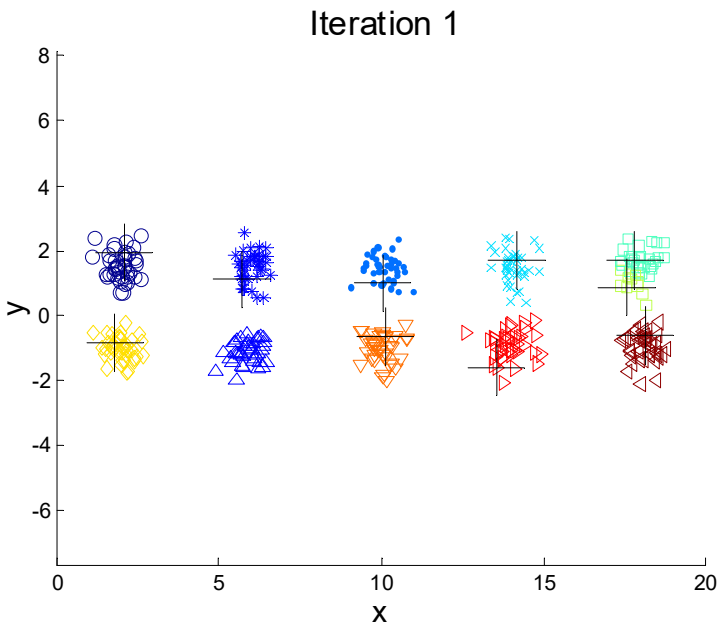
Example with 10 clusters: Starting with two initial centroids in one cluster of each pair of clusters

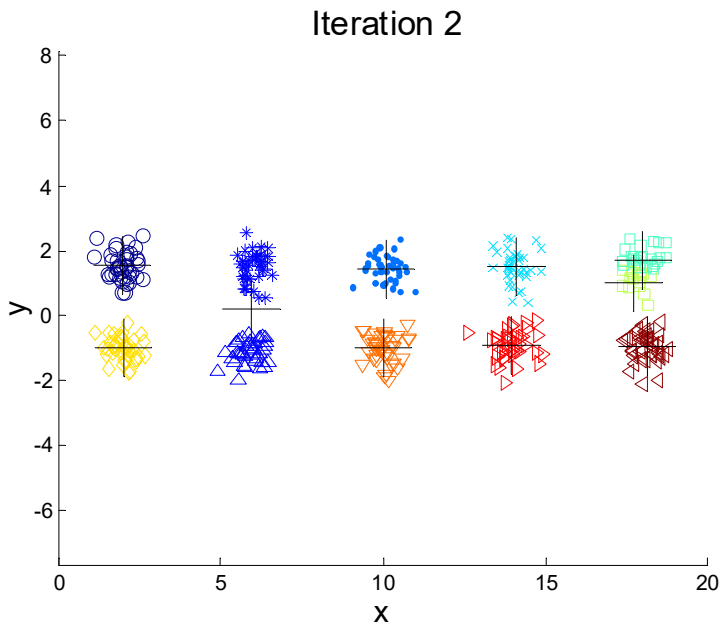




K-means Clustering

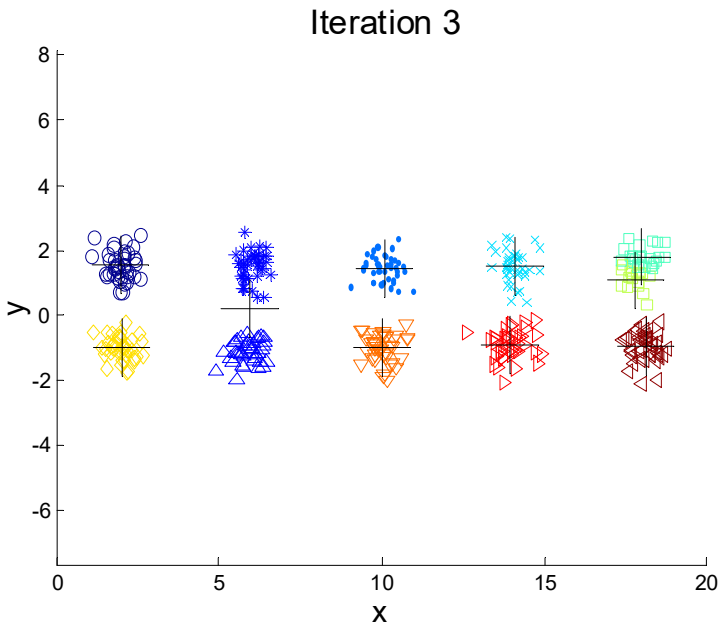
Example with 10 clusters: Starting with some pairs of clusters having three initial centroids, while other have only one.

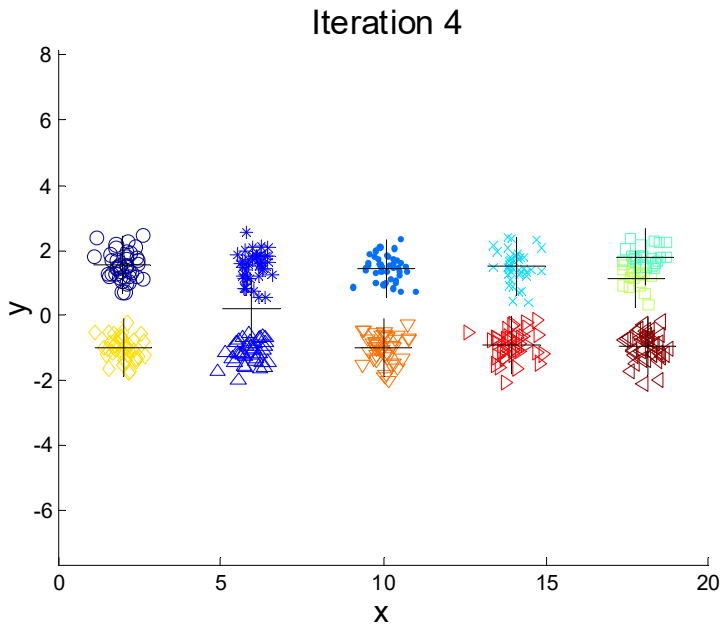




K-means Clustering

Example with 10 clusters: Starting with some pairs of clusters having three initial centroids, while other have only one.





K-means Clustering

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
- **K-mean++**

K-means Clustering

K-means++

- Initialization process is improved by following a more strategic approach:
  - **Step 1: Choose the first centroid randomly from the data points.**
  - **Step 2:** For each remaining data point, calculate its **distance to the nearest centroid already chosen**. The farther a point is from any of the chosen centroids, the higher its probability of being selected as the next centroid.
  - **Step 3: Select the next centroid** from the data points with a **probability proportional to the squared distance from its closest existing centroid**. This means points that are far from the existing centroids have a higher chance of being chosen.
  - **Step 4: Repeat Step 2 and Step 3** until k centroids have been chosen.
  - **Step 5: Proceed with the standard K-Means** algorithm using these k centroids as the initial seeds.
- Complexity of Initialization: $O(k \cdot n)$

K-means Clustering

Advantages of K-Means++

- **Better Initialization:** K-Means++ selects initial centroids that are more likely to be closer to the true cluster centers, reducing the chances of poor clustering due to unlucky random initialization.
- **Faster Convergence:** By starting with better initial centroids, the algorithm often converges faster, requiring fewer iterations.
- **Improved Clustering Quality:** It generally produces more accurate and stable clustering results compared to standard K-Means.

K-means Clustering

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

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

K-means Clustering

K-means algorithm

- For sum of squared error, K-means **converges** to a solution;
  - i.e., K-means reaches a state in which no points are shifting from one cluster to another, and hence, the centroids don't change.

- **Two stages each iteration:**
  - **Update assignments:** Each point is assigned to the nearest centroid.
  - **Update means:** The centroids are recalculated as the mean of the points in each cluster.

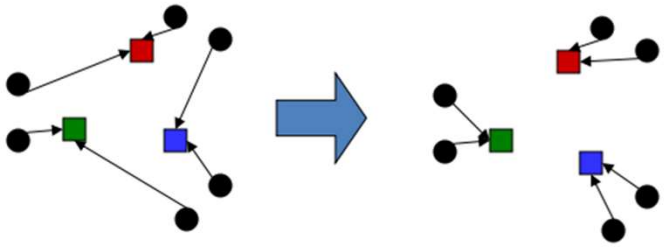
K-means Clustering

K-means algorithm

- **K-Means Iteratively Reduces the Objective Function**

  - **Stage 1 - Update assignments**

  The objective function SSE either **decreases or remains** the same because each data point is assigned to the cluster with the **minimum squared distance** to the centroid.
  By reassigning a point to a closer centroid, we decrease or maintain the sum of squared distances.



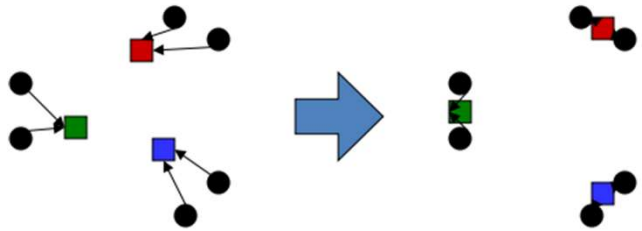
K-means Clustering

K-means algorithm

- **K-Means Iteratively Reduces the Objective Function**

  - **Stage 2 - Update means**

  The new centroid minimizes the sum of squared distances for the points in its cluster. This is because the arithmetic mean is the point that minimizes the sum of squared distances in Euclidean space.



- After both steps, the value of the objective function SSE either decreases or remains the same. It never increases.

K-means Clustering

K-means algorithm

- At each iteration, the objective function **SSE decreases or remains constant**.
- The **sum of squared distances is non-negative** (i.e., SSE ≥0), since it is the sum of squared terms.
- Since **SSE is non-increasing and bounded below by 0**, it converges to a limit.
- The K-Means algorithm operates on a **finite dataset** with a **finite number of possible cluster assignments**.
  - The number of ways n data points can be assigned to k clusters is finite (though very large), approximately k^n.
- Because the objective function **SSE strictly decreases** whenever an assignment or update changes, K-Means **cannot revisit the same clustering configuration** twice unless SSE remains unchanged.
- This implies that the algorithm will eventually reach a point where the **assignments and centroids no longer change**.
- When the assignments no longer change, the K-Means algorithm has **converged**, and no further updates will decrease the objective function.
- **Note**: K-Means converges to a local minimum, not necessarily the global minimum, due to its sensitivity to the initial centroid placement. (K-Means++ initialization is often used to improve the quality of the solution)

K-means Clustering

Limitations of K-means

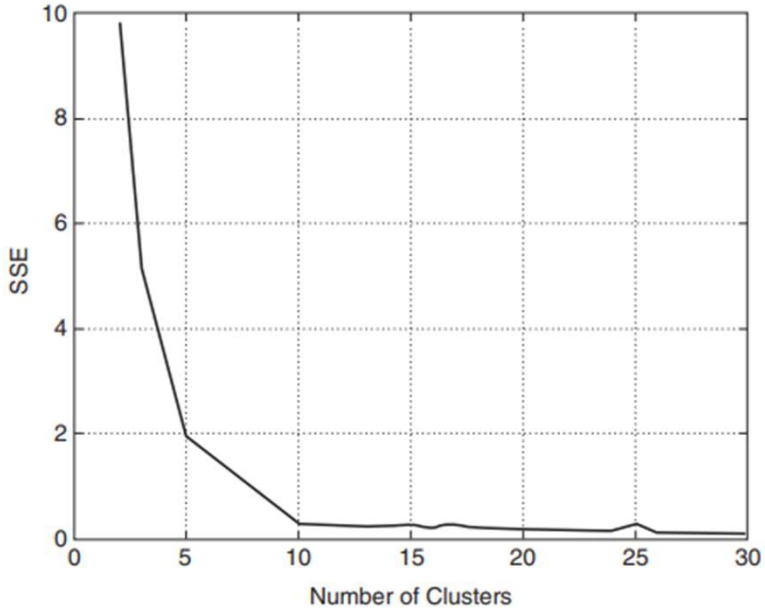
- Defining value for *K*
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

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

K-means Clustering

Limitations of K-means: Defining Value for K

- **Elbow method:** evaluates the sum of squared error (SSE) between data points and their assigned cluster centroids for different values of K and look for an inflection point in the plot.
- **Steps**:
  - Run K-Means for a range of *k* values (e.g., from 1 to 10).
  - Calculate SSE for each *k*.
  - Plot SSE against *k*.
  - Look for an "elbow" in the plot, where the rate of decrease sharply slows down. This point suggests an “optimal” *k*.



K-means Clustering

Limitations of K-means: Defining Value for K

- **Silhouette score:** measures how similar each data point is to its assigned cluster compared to other clusters. It ranges from −1 to 1:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))} \quad \text{where} \quad a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, j \neq i} \|x_i - x_j\| \quad \text{and} \quad b(i) = \min_{C_j \neq C_i} \left(\frac{1}{|C_j|} \sum_{j \in C_j} \|x_i - x_j\| \right)$$

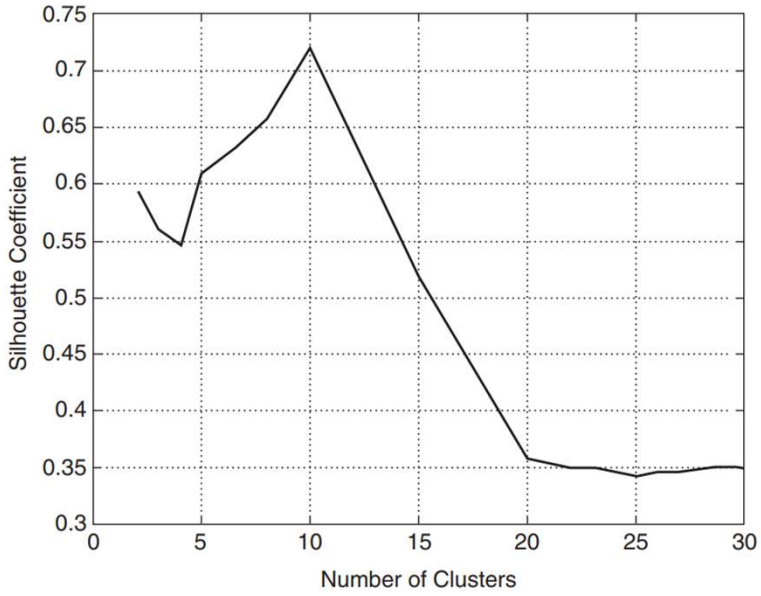
a(i) is the mean intra-cluster distance for point *i*: the average distance between *i* and all other points in the same cluster.
***b*(*i*)** is the mean nearest-cluster distance for point *i*: the average distance between *i* and all points in the nearest (or most similar) cluster that it is not a part of.

- s(i)≈1: The point is well-clustered and far from neighboring clusters.
- $s(i) \approx 0$: The point is on or very close to the decision boundary between two clusters.
- $s(i) \approx -1$: The point is likely misclassified and assigned to the wrong cluster.

K-means Clustering

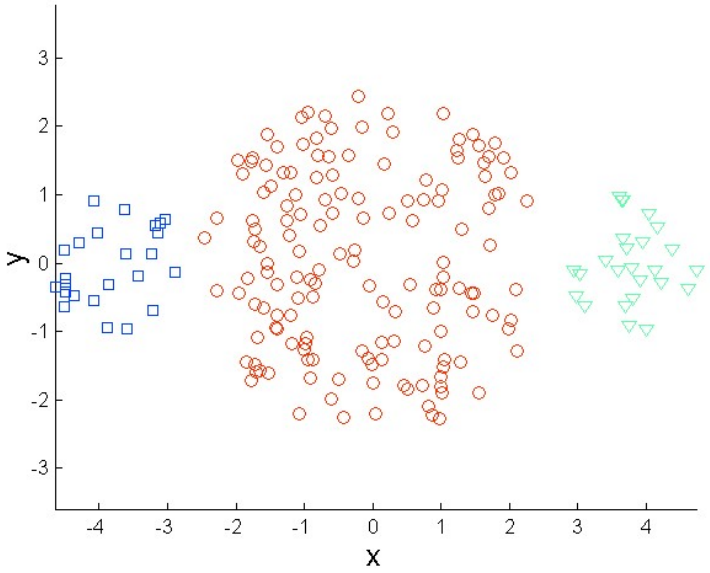
Limitations of K-means: Defining Value for K

- **Silhouette score:** measures how similar each data point is to its assigned cluster compared to other clusters.
- **Steps:**
  - Compute the Silhouette Score for different *k* values.
  - Choose the *k* that maximizes the Silhouette Score.

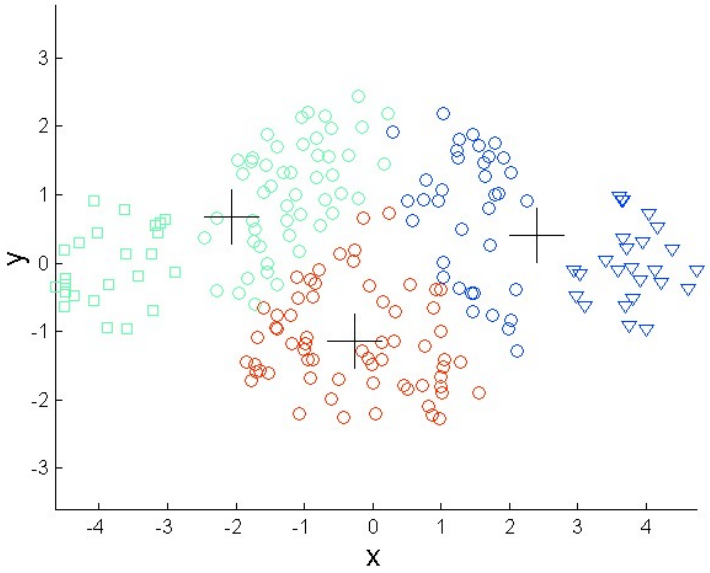


K-means Clustering

Limitations of K-means: Differing Sizes



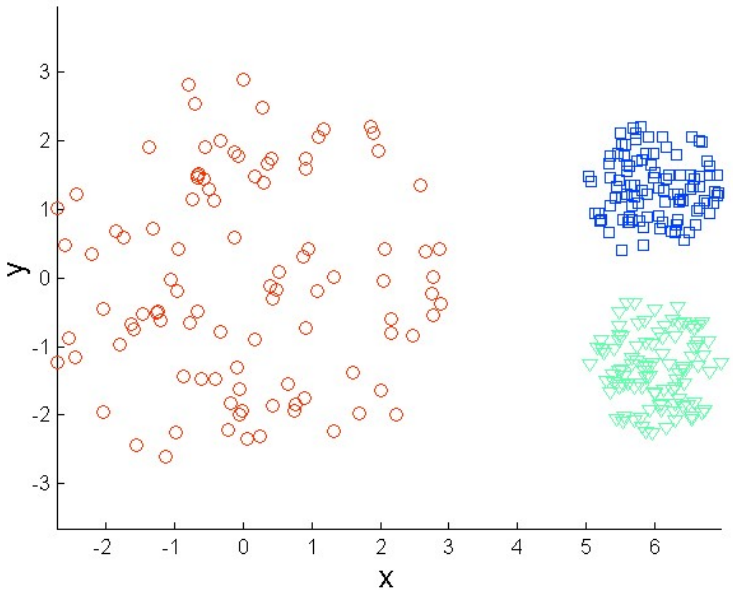
Original Points



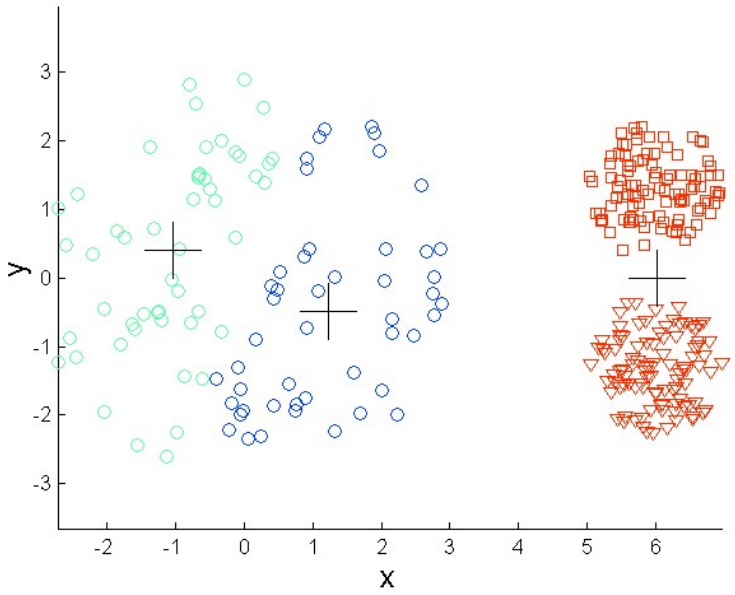
K-means (3 Clusters)

K-means Clustering

Limitations of K-means: Differing Density



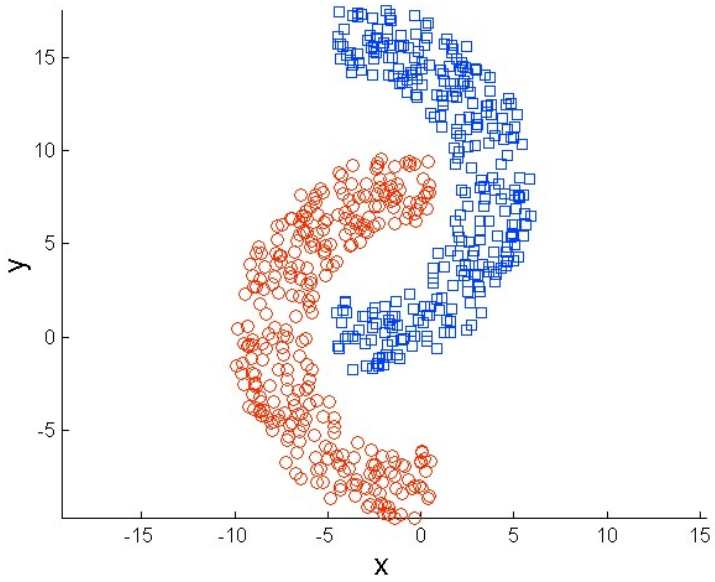
Original Points



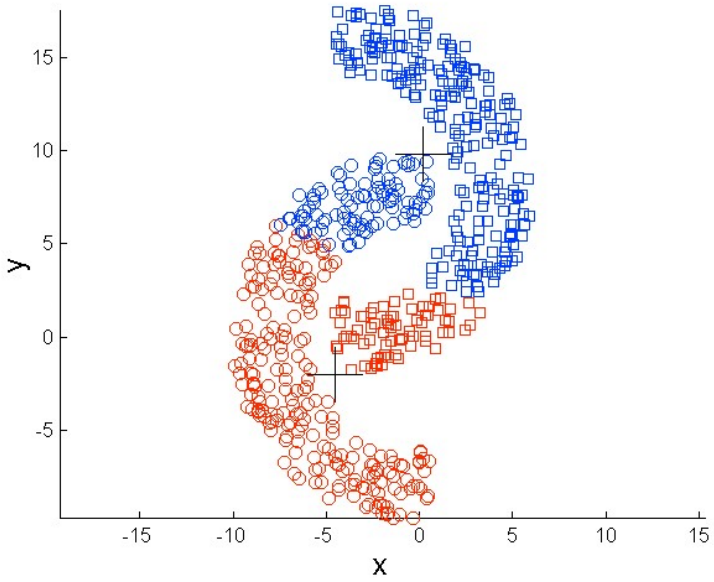
K-means (3 Clusters)

K-means Clustering

Limitations of K-means: Non-globular Shapes



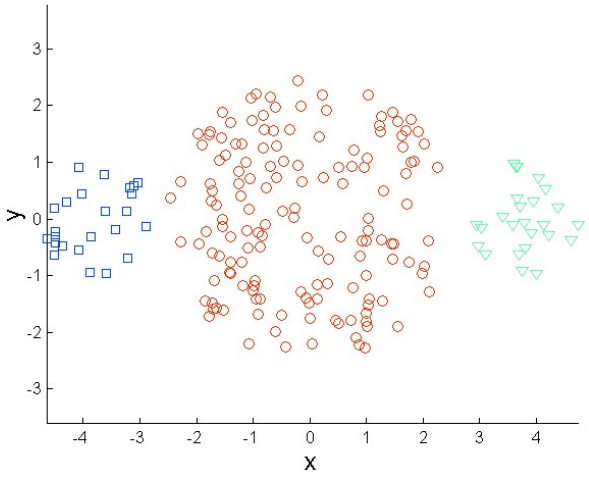
Original Points



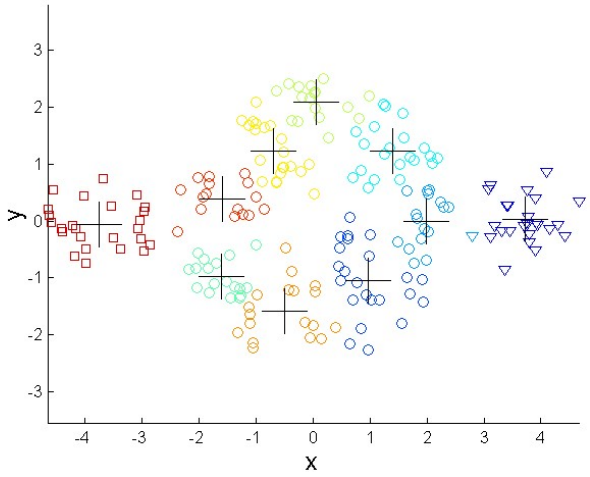
K-means (2 Clusters)

K-means Clustering

Overcoming K-means Limitations



Original Points

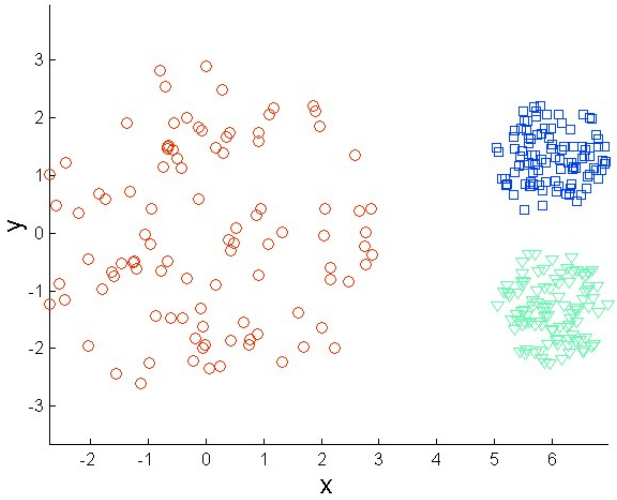


K-means Clusters

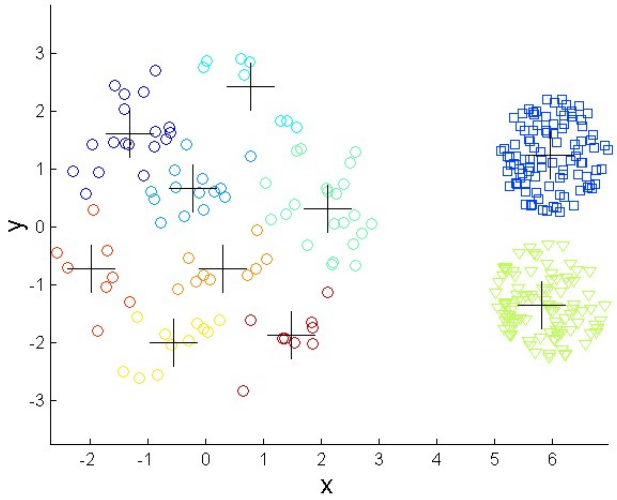
Use many clusters.
Find parts of clusters
but need to put together.

K-means Clustering

Overcoming K-means Limitations



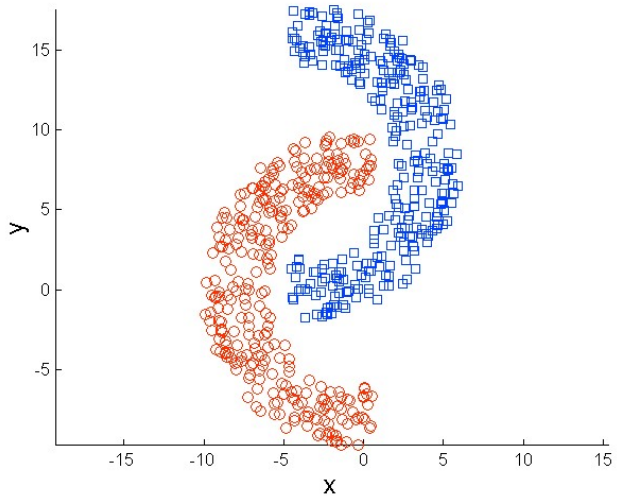
Original Points



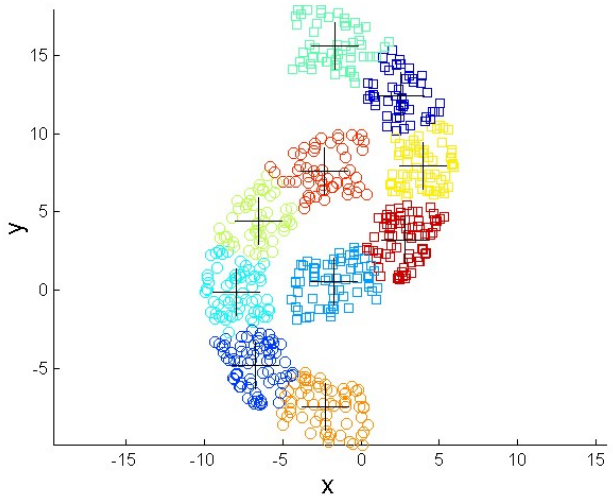
K-means Clusters

K-means Clustering

Overcoming K-means Limitations



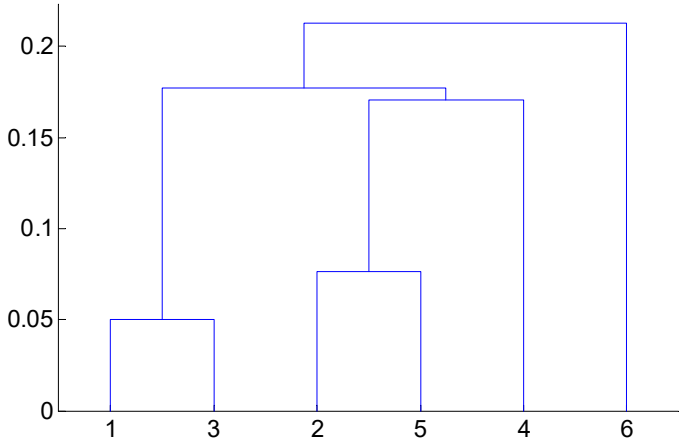
Original Points

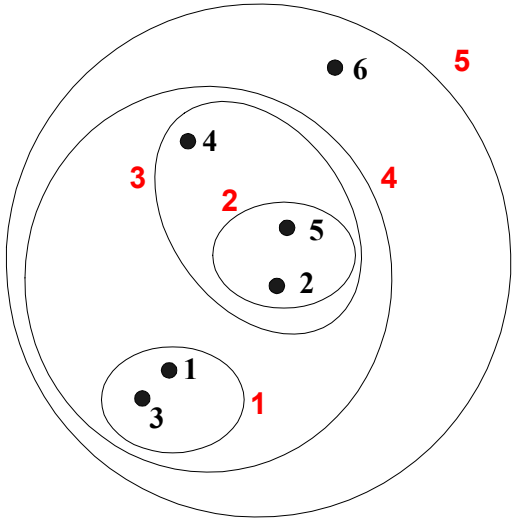


K-means Clusters

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





Hierarchical Clustering

Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level

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

Hierarchical Clustering

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

Hierarchical Clustering

Agglomerative Clustering Algorithm

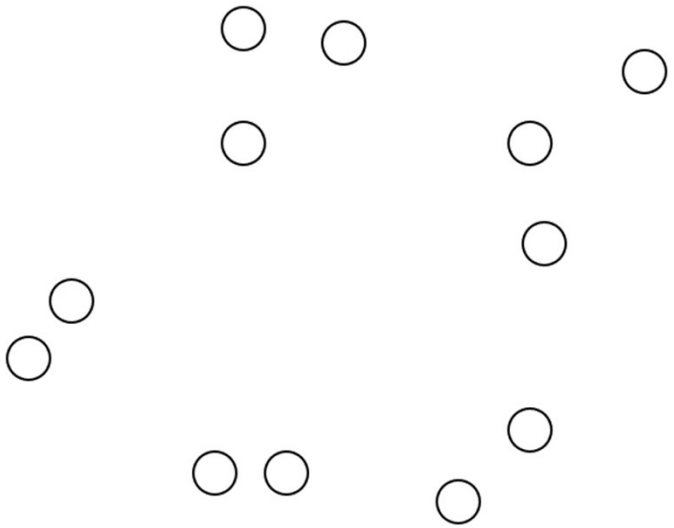
Compute the proximity matrix
Let each data point be a cluster
Repeat
    Merge the two closest clusters
    Update the proximity matrix
Until only a single cluster remains

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

Hierarchical Clustering

Starting Situation

- Start with clusters of individual points and a proximity matrix



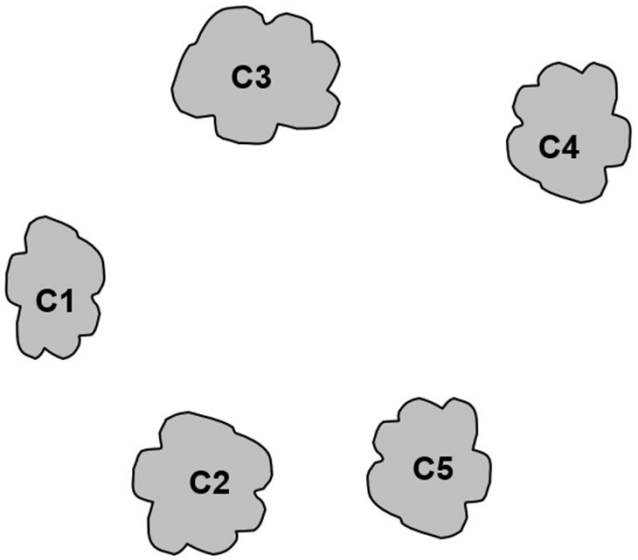
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Hierarchical Clustering

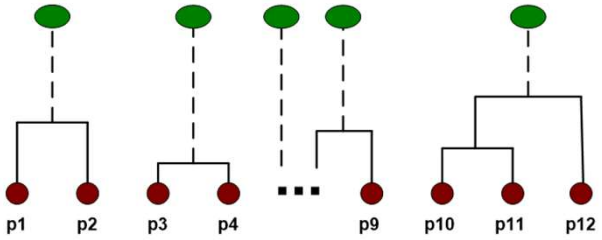
Intermediate Situation

- After some merging steps, we have some clusters



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

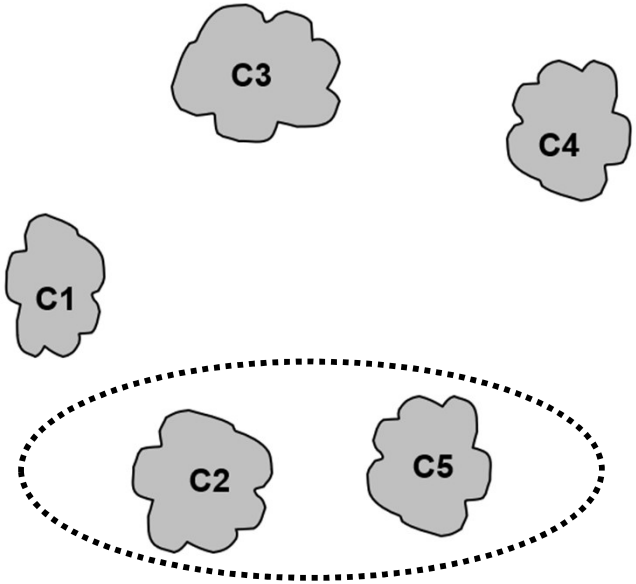
Proximity Matrix



Hierarchical Clustering

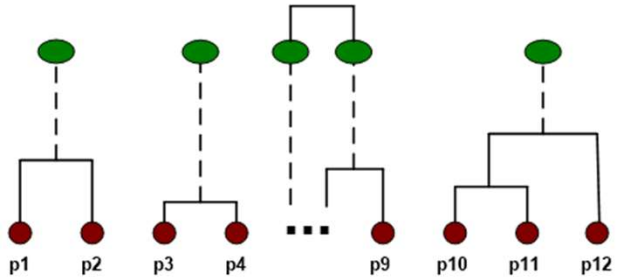
Intermediate Situation

- 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					

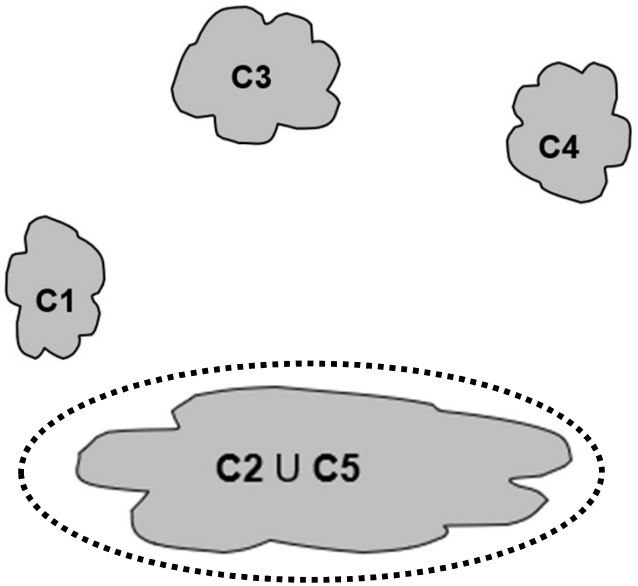
Proximity Matrix



Hierarchical Clustering

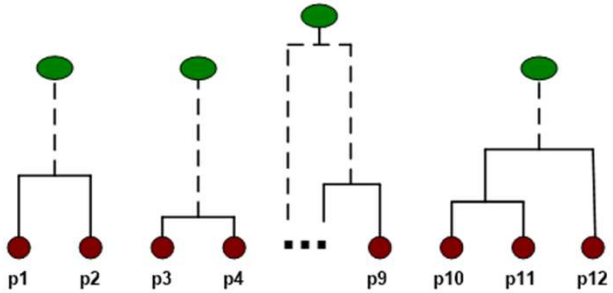
After Merging

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



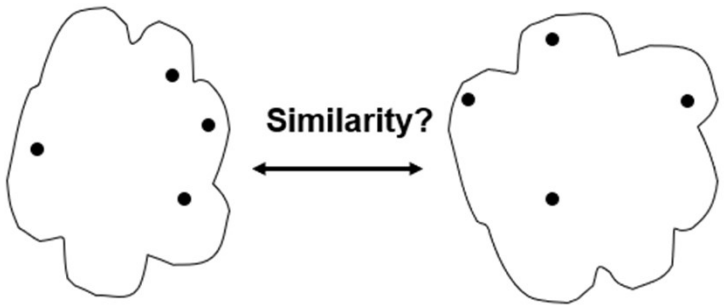
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



Hierarchical Clustering

How to Define Inter-Cluster Similarity



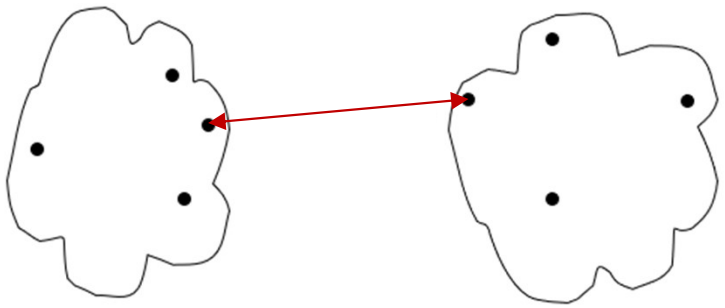
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

Hierarchical Clustering

How to Define Inter-Cluster Similarity



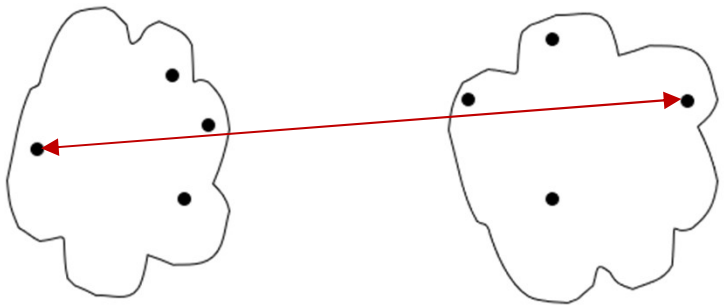
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

Hierarchical Clustering

How to Define Inter-Cluster Similarity



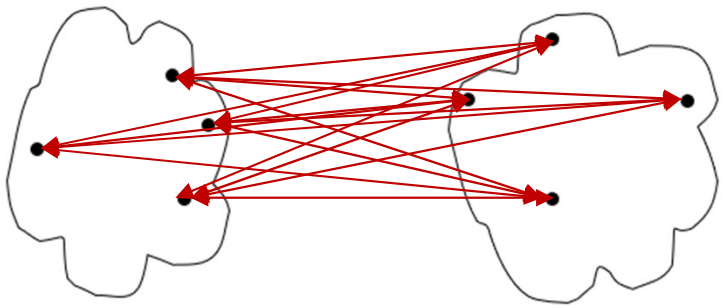
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

Hierarchical Clustering

How to Define Inter-Cluster Similarity



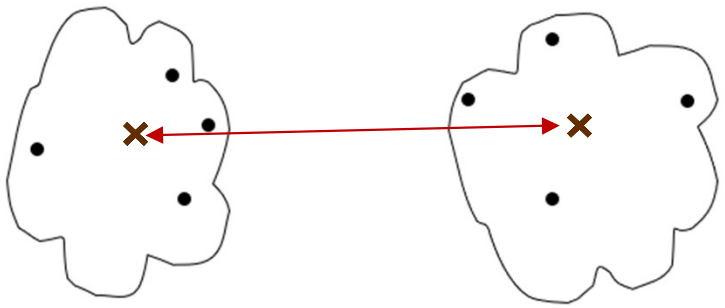
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

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

Hierarchical Clustering

How to Define Inter-Cluster Similarity



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

Proximity Matrix

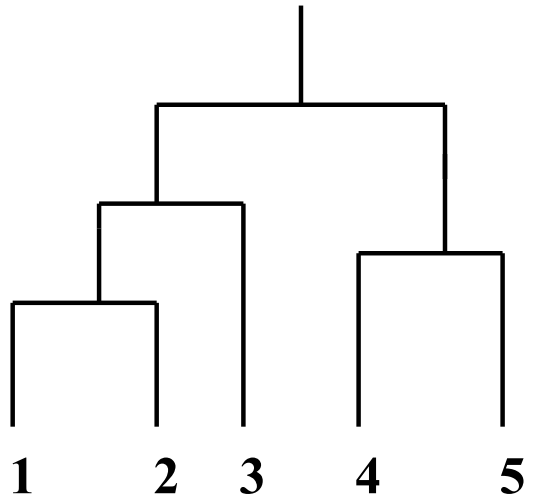
- MIN
- MAX
- Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function
  - Ward's Method uses squared error

Hierarchical Clustering

Cluster Similarity: MIN or Single Link

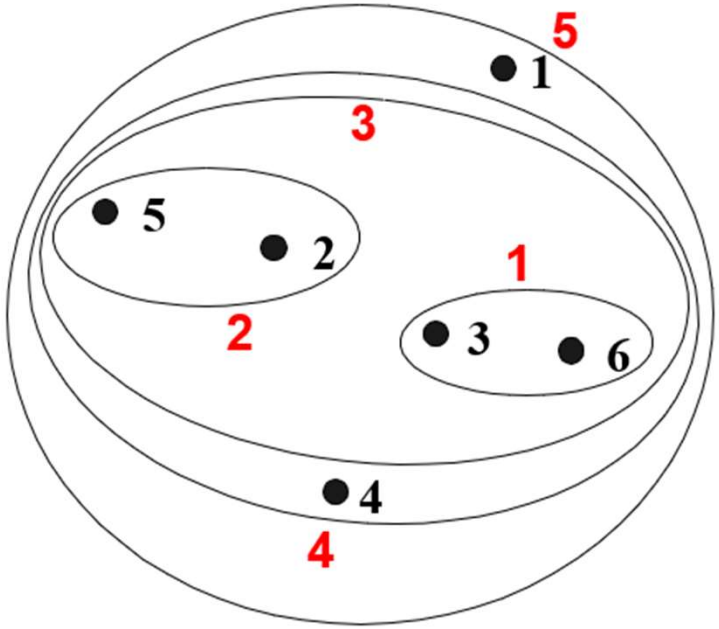
- Similarity of two clusters is based on the two most similar (closest) points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph.

	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

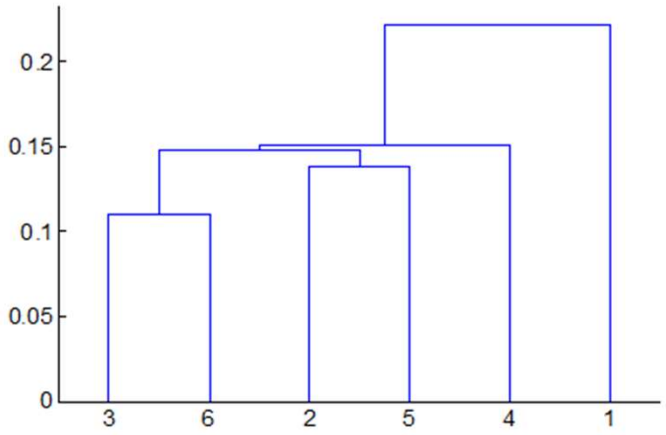


Hierarchical Clustering

Hierarchical Clustering: MIN



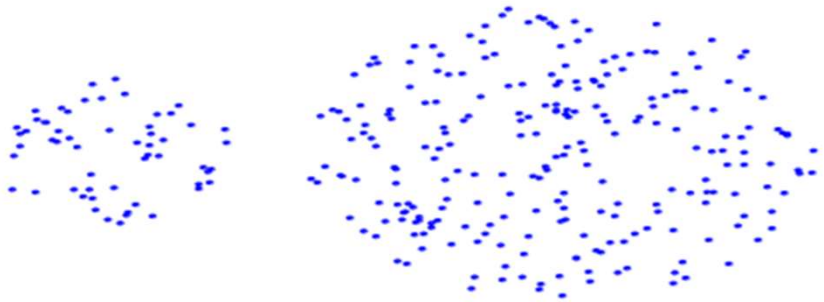
Nested Clusters



Dendrogram

Hierarchical Clustering

Strength of MIN



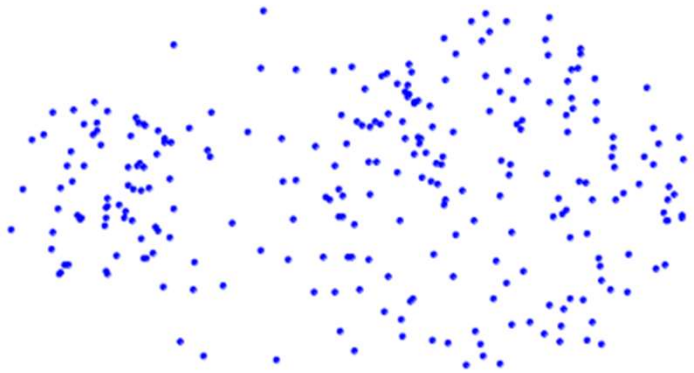
Original Points

Two Clusters

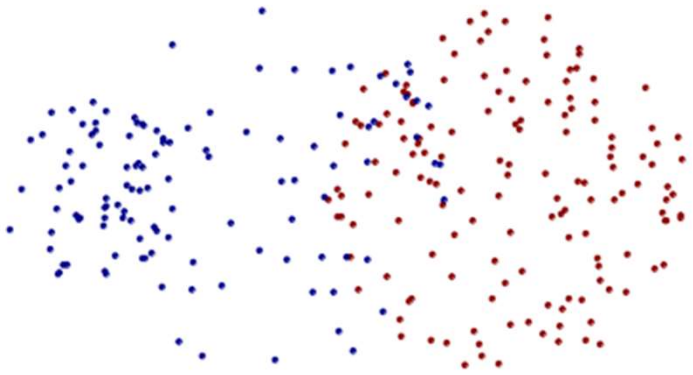
- Can handle non-elliptical shapes

Hierarchical Clustering

Limitations of MIN



Original Points



Two Clusters

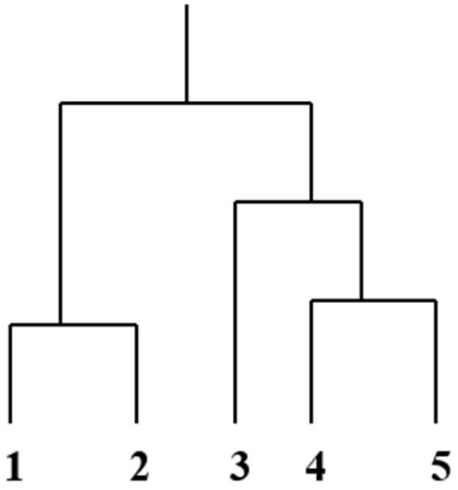
- Sensitive to noise and outliers

Hierarchical Clustering

Cluster Similarity: MAX or Complete Linkage

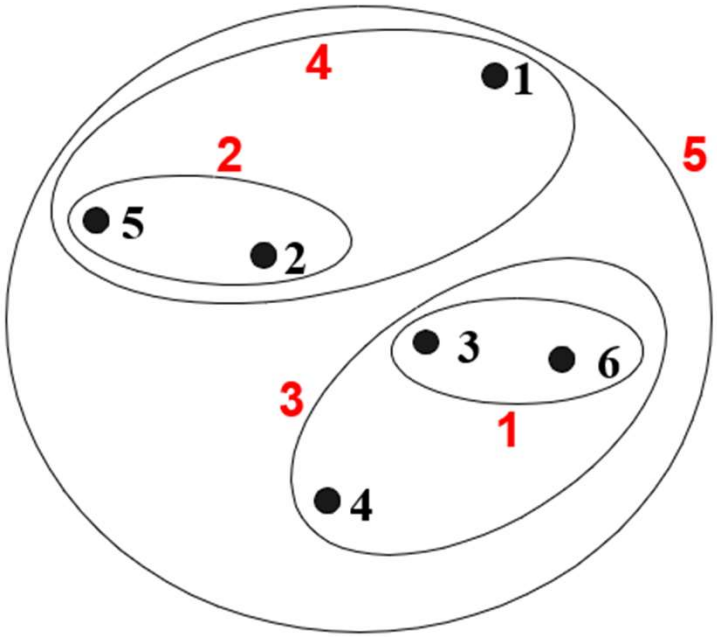
- Similarity of two clusters is based on the two least similar (most distant) points in the different clusters
  - Determined by all pairs of points in the two clusters

	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

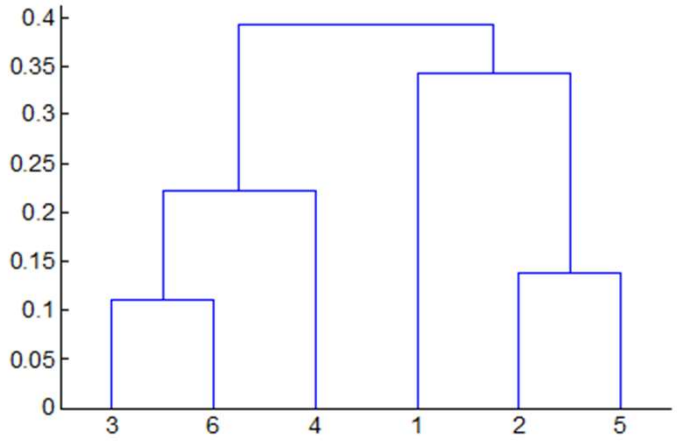


Hierarchical Clustering

Hierarchical Clustering: MAX



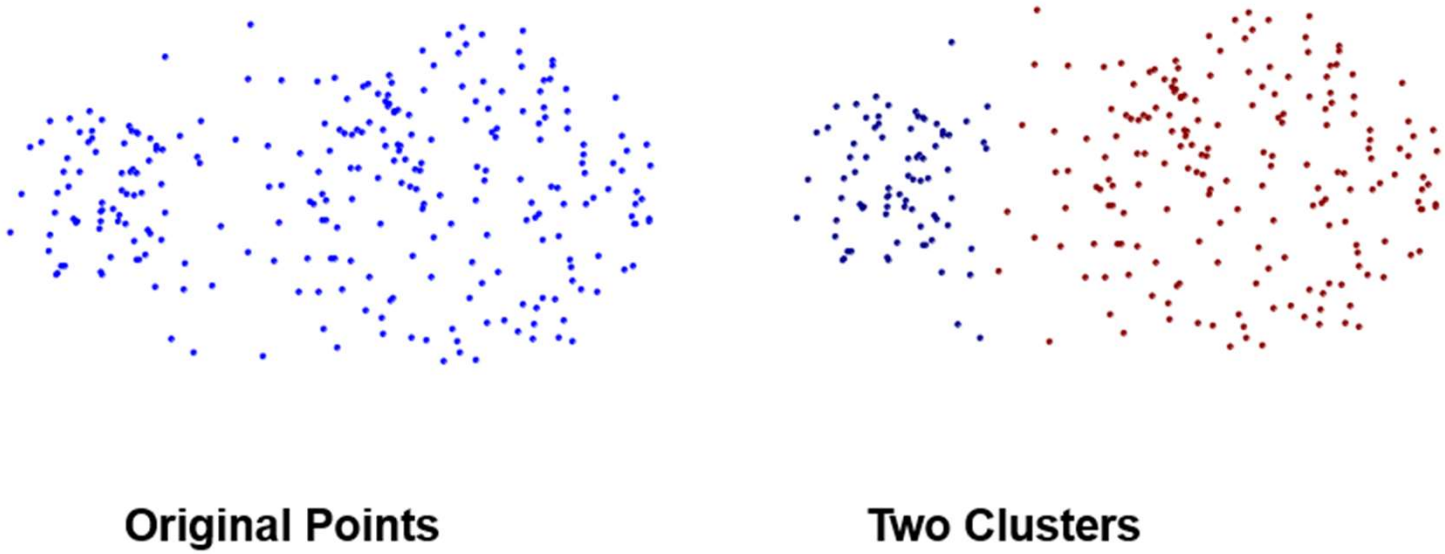
Nested Clusters



Dendrogram

Hierarchical Clustering

Strength of MAX



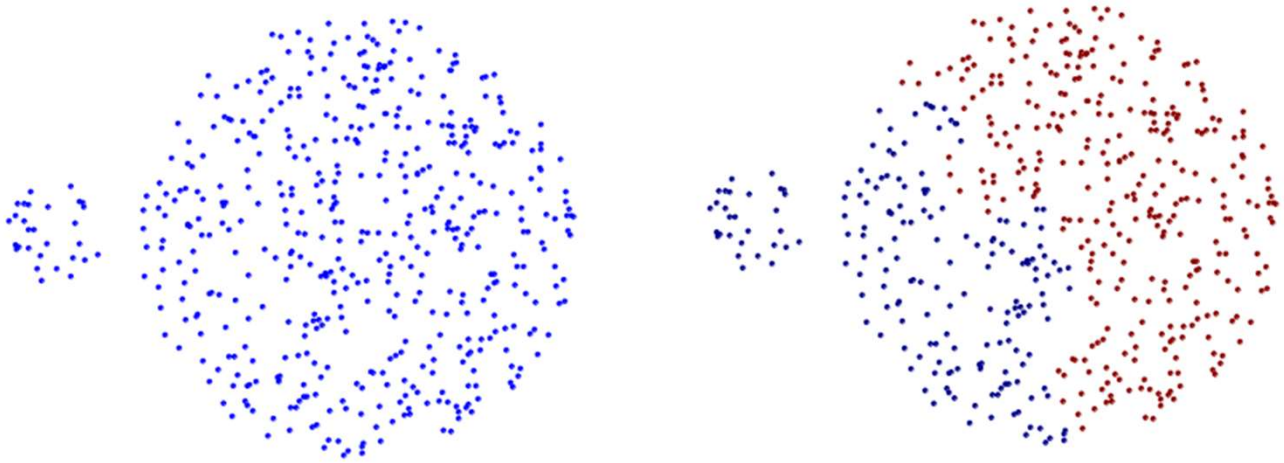
Original Points

Two Clusters

Less susceptible to noise and outliers

Hierarchical Clustering

Limitations of MAX



Original Points

Two Clusters

Tends to break large clusters

Biased towards globular clusters

Hierarchical Clustering

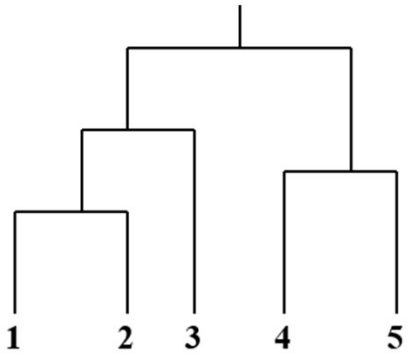
Cluster Similarity: Group Average

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

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

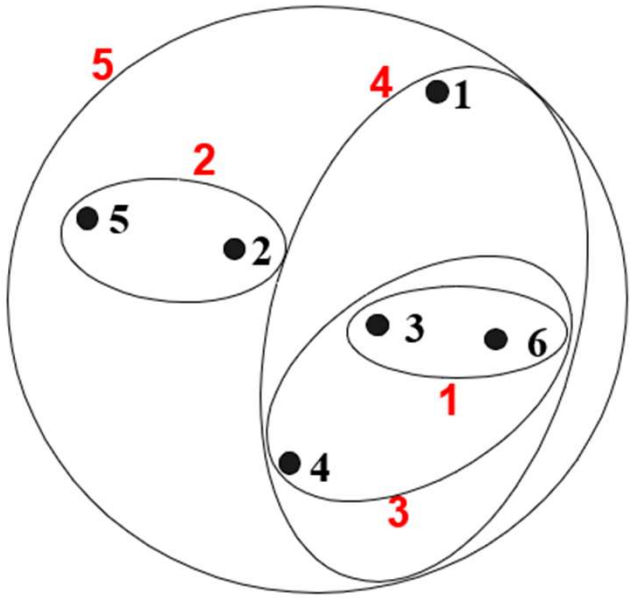
- Need to use average connectivity for scalability since total proximity favors large clusters

	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

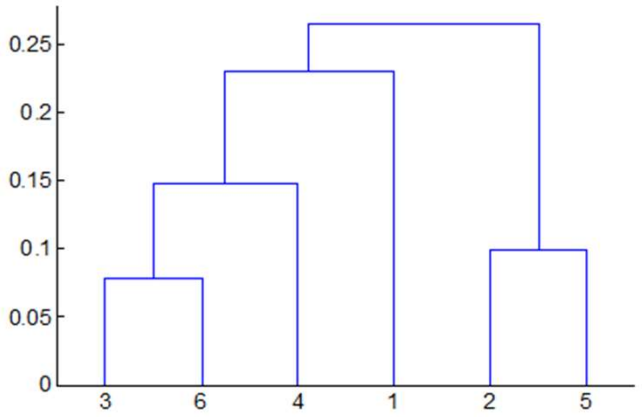


Hierarchical Clustering

Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

Hierarchical Clustering

Hierarchical Clustering: Group Average

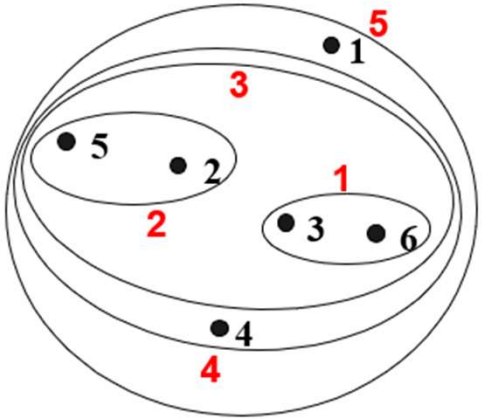
- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise and outliers
- Limitations
  - Biased towards globular clusters

Hierarchical Clustering

Hierarchical Clustering: Ward's Method

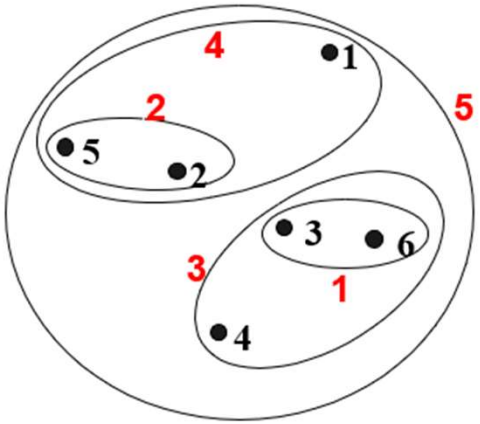
- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise and outliers
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used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

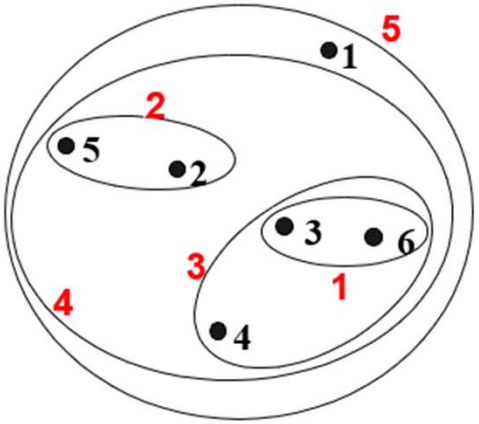
Hierarchical Clustering: Comparison



MIN

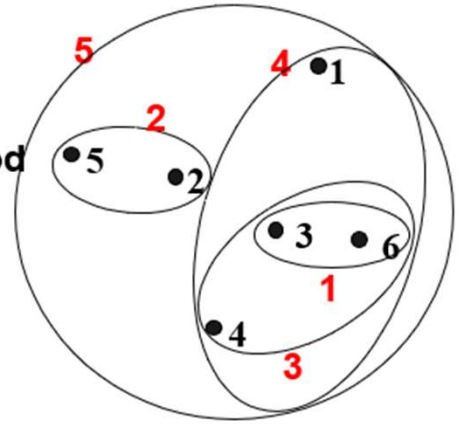
MAX





Group Average

Ward's Method



Hierarchical Clustering

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

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

Clustering

Clustering 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

Clustering

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**. (Using 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.

Clustering

Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
  - **External Index (Supervised)**: Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
  - **Internal Index (Unsupervised)**: Used to measure the goodness of a clustering structure without respect to external information.
    - Sum of Squared Error (SSE)

Clustering

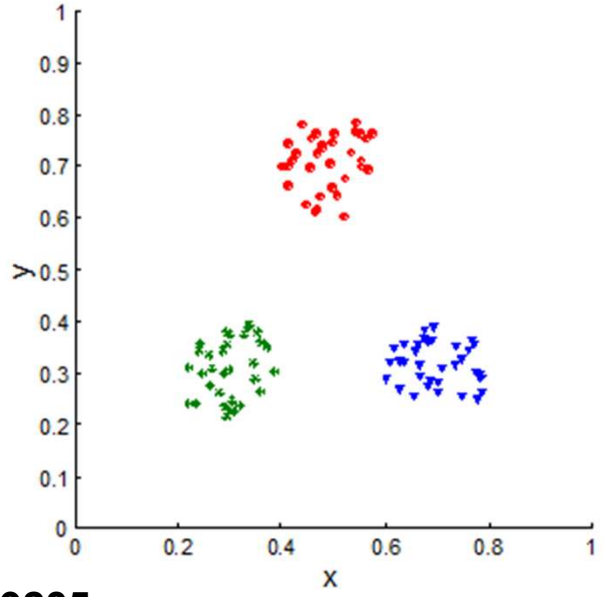
Measuring Cluster Validity Via Correlation

- 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.

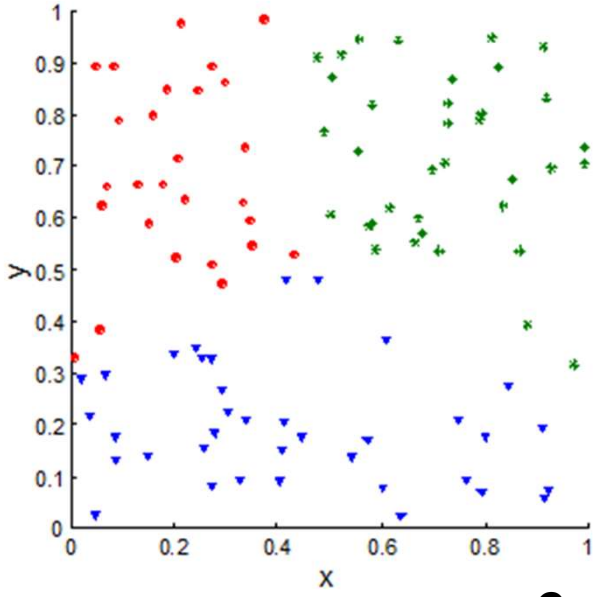
Clustering

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

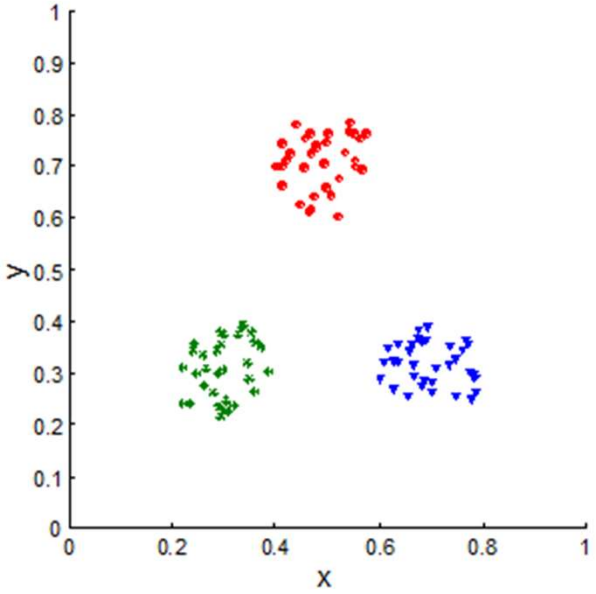


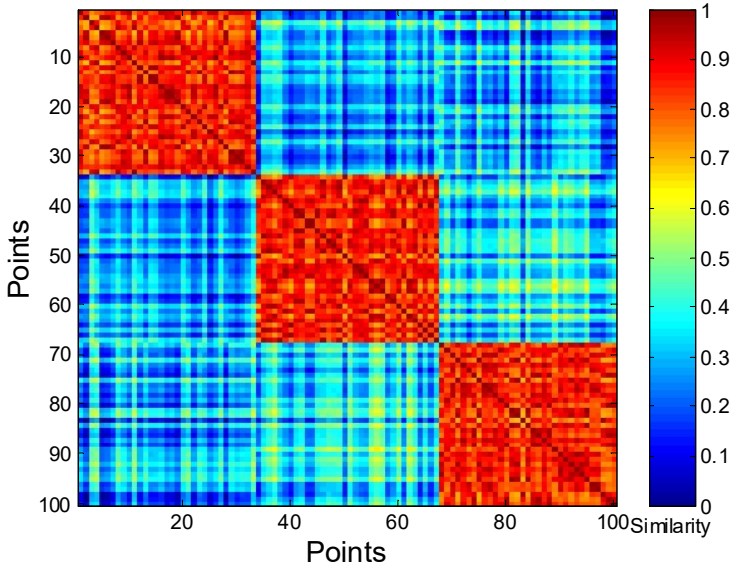
Corr = -0.5810

Clustering

Measuring Cluster Validity Via Correlation

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

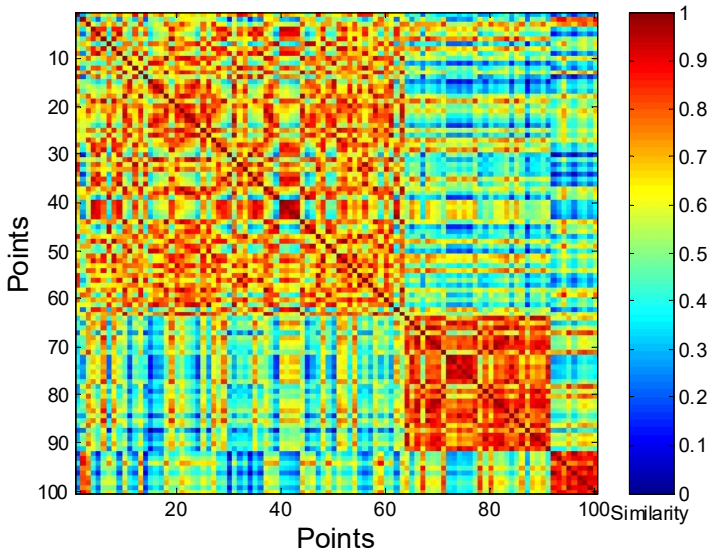


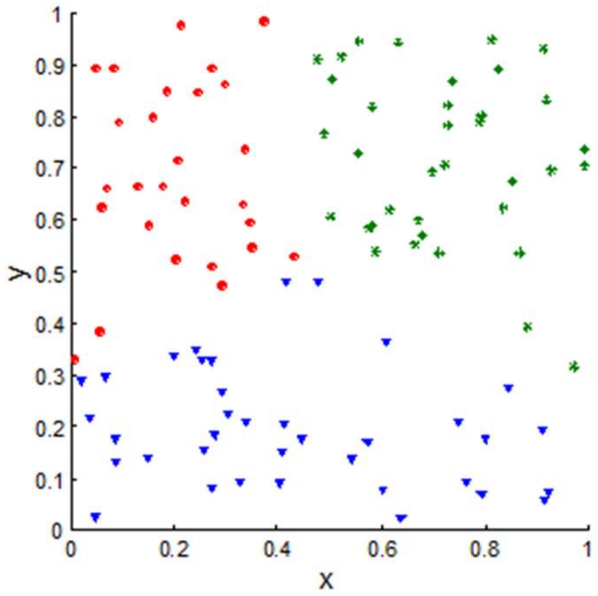


Clustering

Measuring Cluster Validity Via Correlation

- Clusters in random data are not so crisp

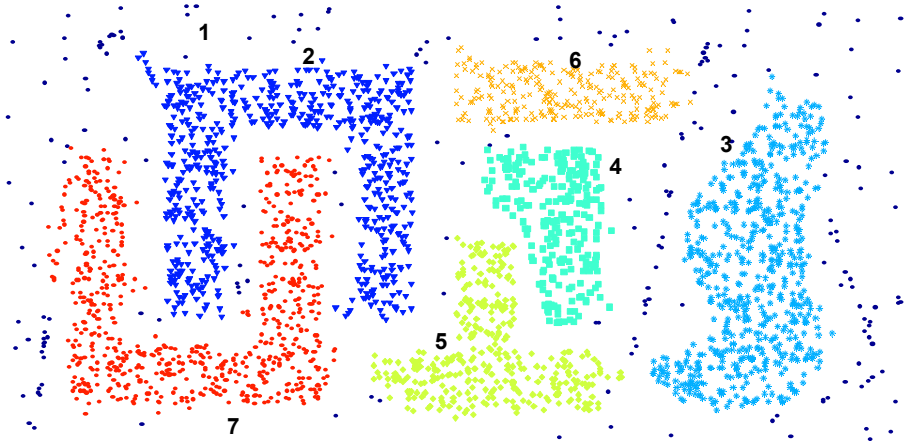


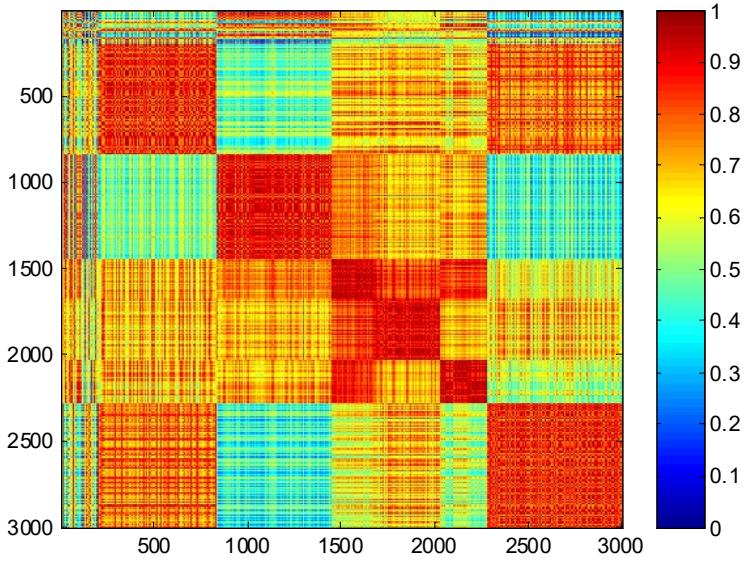


Clustering

Measuring Cluster Validity Via Correlation

- Correlation Analysis for a more complicated data set

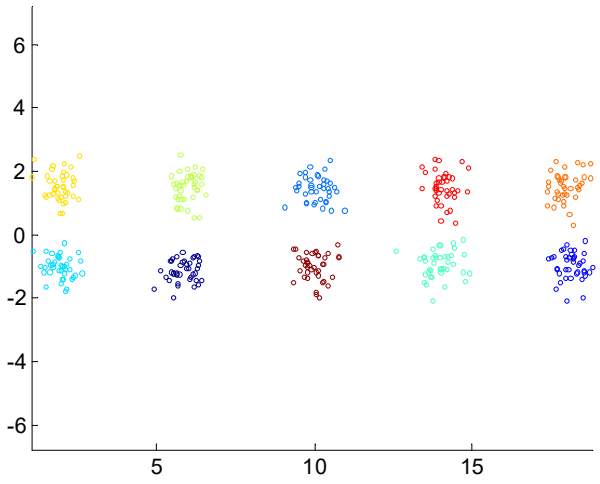


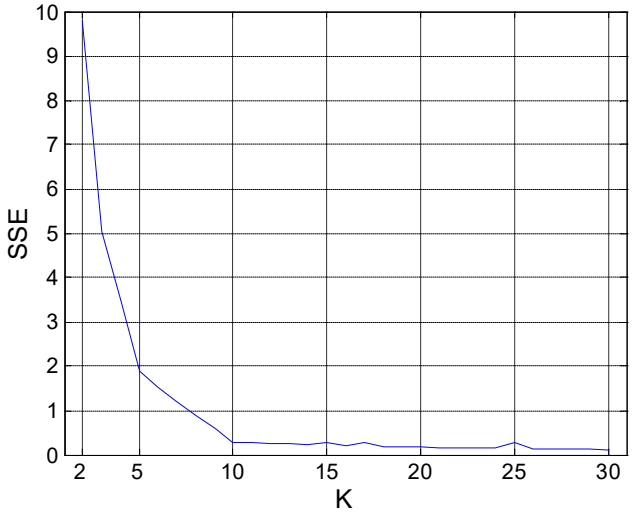


Clustering

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 clustering or two clusters (average SSE).
- Can also be used to estimate the number of clusters

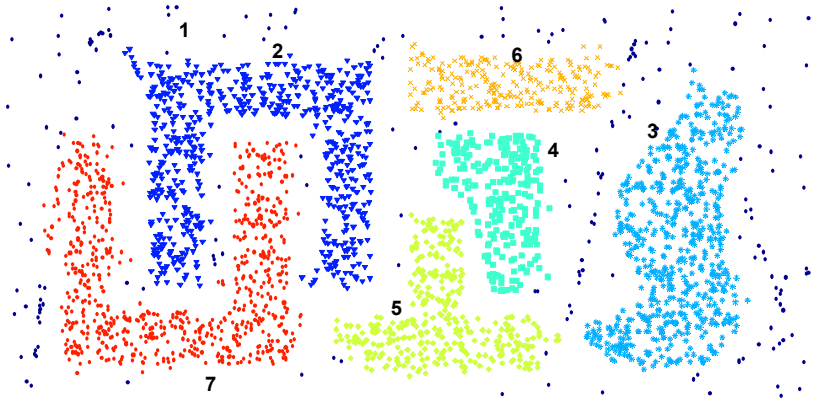


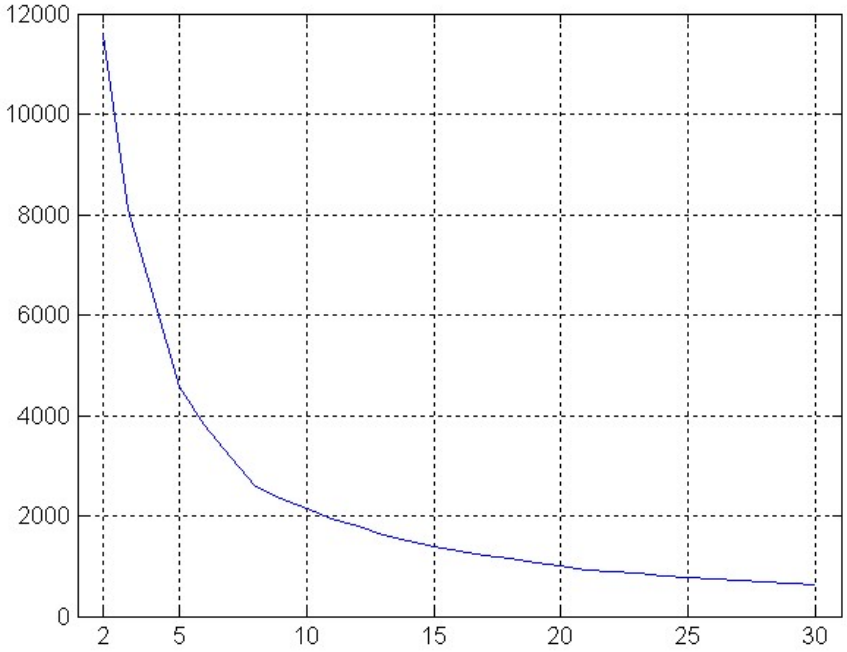


Clustering

Internal Measures: SSE

- SSE curve for a more complicated data set





SSE of clusters found using K-means

Clustering

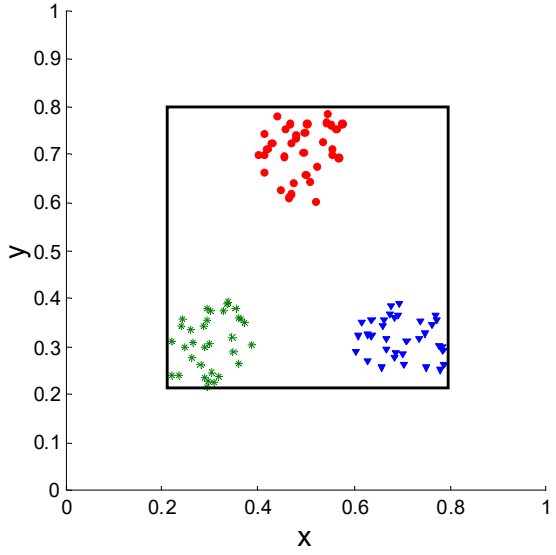
Framework for Cluster Validity

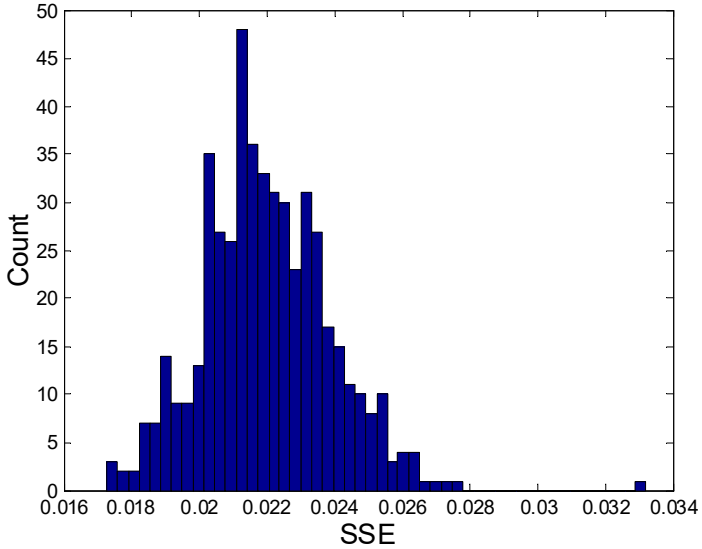
- Need a framework to interpret any measure.
  - For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?
- Statistics provide a framework for cluster validity
  - The more "atypical" a clustering result is, the more likely it represents valid structure in the data
  - Can **compare** the values of an index that result from **random data** or clustering to those of a clustering result.
    - If the value of the index is unlikely, then the cluster results are valid
- 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.

Clustering

Framework for Cluster Validity – 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

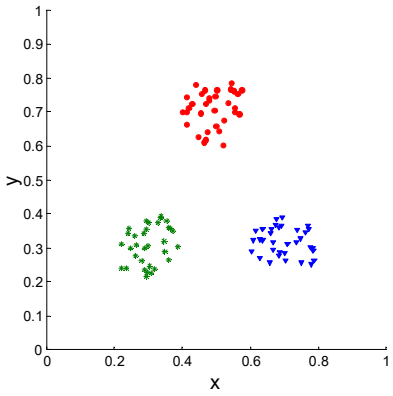




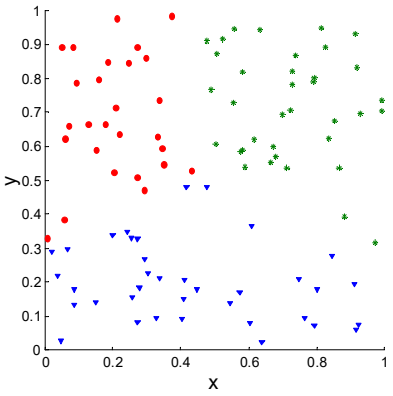
Clustering

Statistical Framework for Correlation

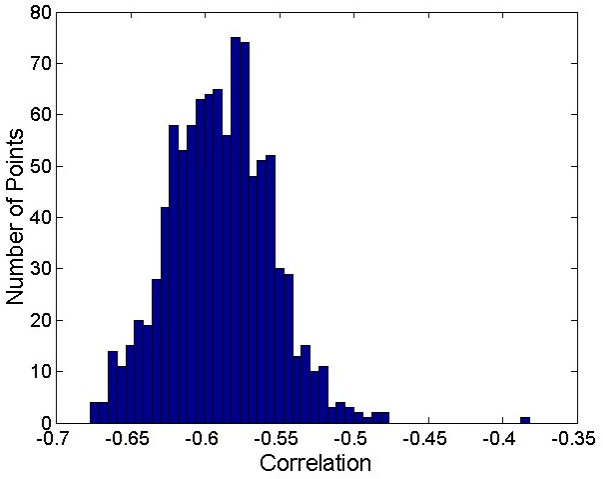
- Correlation of incidence and proximity matrices for the K-means clustering of the following two data sets.



Corr = -0.9235



Corr = -0.5810



Clustering

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

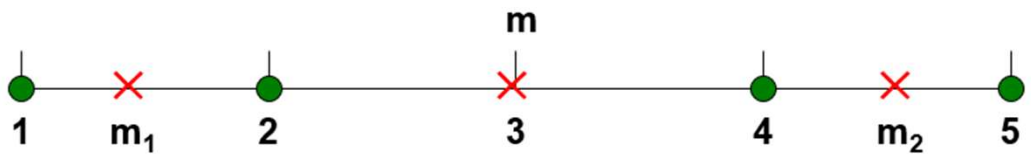
$$BSS = \sum_{i} |C_i| (m - m_i)^2$$

where |Ci| is the size of cluster i and m is the overall centroid (for all points)

Clustering

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:

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

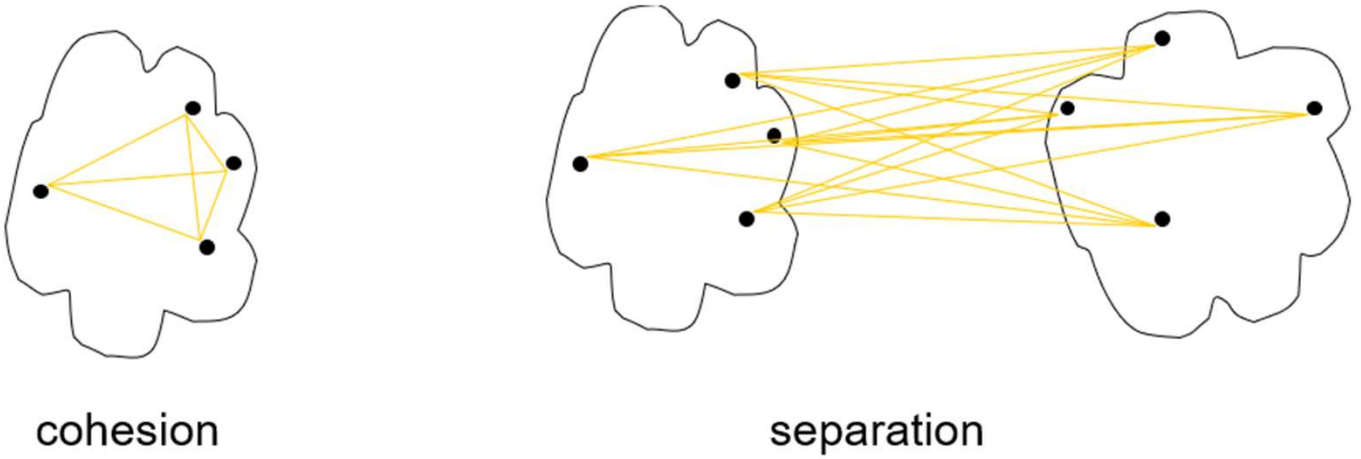
$$BSS = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

Clustering

Internal Measures: Cohesion and Separation

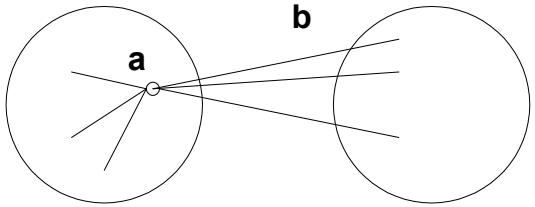
- A proximity graph-based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



Clustering

Internal Measures: Silhouette Coefficient

- Silhouette Coefficient combine ideas of both cohesion and separation, but for individual points, as well as clusters and clustering
- The Silhouette Score for a single data point *i* is defined as



$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

Where:

a(i) is the mean intra-cluster distance for point *i*: the average distance between *i* and all other points in the same cluster.

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, j \neq i} \|x_i - x_j\|$$

***b*(*i*)** is the mean nearest-cluster distance for point *i*: the average distance between *i* and all points in the nearest (or most similar) cluster that it is not a part of.

$$b(i) = \min_{C_j \neq C_i} \left(\frac{1}{|C_j|} \sum_{j \in C_j} \|x_i - x_j\| \right)$$

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

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 $purity = \sum_{i=1}^{K} \frac{m_i}{m} purity_j$.

Readings

- **Introduction to Data Mining, by Pang-Ning Tan et al. – Chapters 7**