CS 412 Intro. to Data Mining

Chapter 9. Cluster Analysis

Arindam Banerjee, Computer Science, UIUC, Fall 2021

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Spectral Clustering
- Evaluation of Clustering
- Summary

Cluster Analysis: An Introduction

- What Is Cluster Analysis?
- Applications of Cluster Analysis
- Cluster Analysis: Requirements and Challenges
- Cluster Analysis: A Multi-Dimensional Categorization
- An Overview of Typical Clustering Methodologies
- An Overview of Clustering Different Types of Data
- An Overview of User Insights and Clustering

What Is Cluster Analysis?

- **What is a cluster?**
  - A cluster is a collection of data objects which are
    - Similar (or related) to one another within the same group (i.e., cluster)
    - Dissimilar (or unrelated) to the objects in other groups (i.e., clusters)
- **Cluster analysis** (or ***clustering*, *data segmentation***, …)
  - Given a set of data points, partition them into a set of groups (i.e., clusters) which are as similar as possible
- Cluster analysis is **unsupervised learning** (i.e., no predefined classes)
  - This contrasts with *classification* (i.e., *supervised learning*)
- Typical ways to use/apply cluster analysis
  - As a stand-alone tool to get insight into data distribution, or
  - As a preprocessing (or intermediate) step for other algorithms

What Is Good Clustering?

- A good clustering method will produce high quality clusters which should have
  - **High intra-class similarity:** Cohesive within clusters
  - **Low inter-class similarity:** Distinctive between clusters
- **Quality function**
  - There is usually a separate "quality" function that measures the "goodness" of a cluster
  - It is hard to define "similar enough" or "good enough"
    - The answer is typically highly subjective
- There exist many similarity measures and/or functions for different applications
- Similarity measure is critical for cluster analysis

Cluster Analysis: Applications

- A key intermediate step for other data mining tasks
  - Generating a compact summary of data for classification, pattern discovery, hypothesis generation and testing, etc.
  - Outlier detection: Outliers—those “far away” from any cluster
- Data summarization, compression, and reduction
  - Ex. Image processing: Vector quantization
- Collaborative filtering, recommendation systems, or customer segmentation
  - Find like-minded users or similar products
- Dynamic trend detection
  - Clustering stream data and detecting trends and patterns
- Multimedia data analysis, biological data analysis and social network analysis
  - Ex. Clustering images or video/audio clips, gene/protein sequences, etc.

Considerations for Cluster Analysis

- **Partitioning criteria**
  - Single level vs. hierarchical partitioning (often, multi-level hierarchical partitioning is desirable, e.g., grouping topical terms)
- **Separation of clusters**
  - Exclusive (e.g., one customer belongs to only one region) vs. non-exclusive (e.g., one document may belong to more than one class)
- **Similarity measure**
  - Distance-based (e.g., Euclidean, road network, vector) vs. connectivity-based (e.g., density or contiguity)
- **Clustering space**
  - Full space (often when low dimensional) vs. subspaces (often in high-dimensional clustering)

Requirements and Challenges

- **Quality**
  - Ability to deal with different types of attributes: Numerical, categorical, text, multimedia, networks, and mixture of multiple types
  - Discovery of clusters with arbitrary shape
  - Ability to deal with noisy data
- **Scalability**
  - Clustering all the data instead of only on samples
  - High dimensionality
  - Incremental or stream clustering and insensitivity to input order
- **Constraint-based clustering**
  - User-given preferences or constraints; domain knowledge; user queries
- **Interpretability and usability**
  - The final generated clusters should be semantically meaningful and useful

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Spectral Clustering
- Evaluation of Clustering
- Summary

Partitioning-Based Clustering Methods

- Basic Concepts of Partitioning Algorithms
- The K-Means Clustering Method
- Initialization of K-Means Clustering
- The K-Medoids Clustering Method
- The K-Medians and K-Modes Clustering Methods
- The Kernel K-Means Clustering Method

Partitioning Algorithms: Basic Concepts

- Partitioning method: Discovering the groupings in the data by optimizing a specific objective function and iteratively improving the quality of partitions
- *K*-partitioning method: Partitioning a dataset ***D*** of ***n*** objects into a set of ***K*** clusters so that an objective function is optimized (e.g., the sum of squared distances is minimized, where c_k is the centroid or medoid of cluster C_k)
  - A typical objective function: **Sum of Squared Errors** (**SSE**)

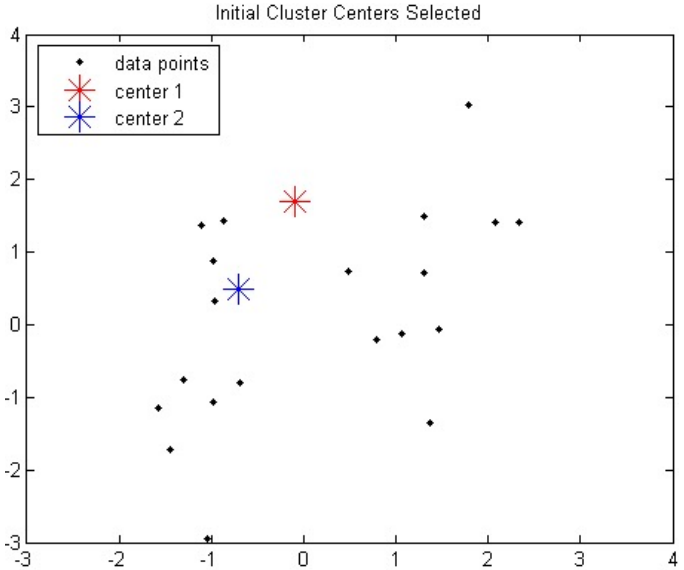
$$SSE(C) = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} \| x_i - c_k \|^2$$

- Problem definition: Given *K*, find a partition of *K clusters* that optimizes the chosen partitioning criterion
  - Global optimal: Needs to exhaustively enumerate all partitions
  - Heuristic methods (i.e., greedy algorithms): *K-Means*, *K-Medians*, *K-Medoids*, etc.

The *K-Means* Clustering Method

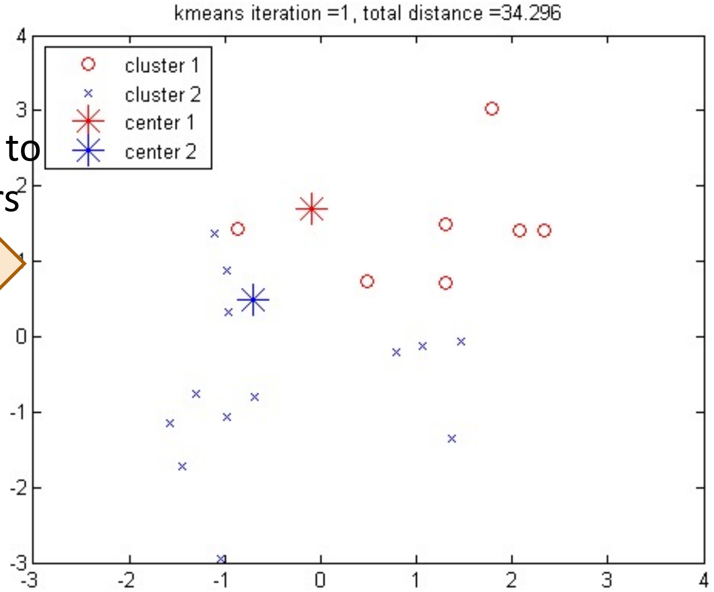
- *K-Means* (MacQueen'67, Lloyd'57/'82)
  - Each cluster is represented by the center of the cluster
- Given K, the number of clusters, the *K-Means* clustering algorithm is outlined as follows
  - Select *K* points as initial centroids
  - **Repeat**
    - Form *K* clusters by assigning each point to its closest centroid
    - Re-compute the centroids (i.e., *mean point*) of each cluster
  - **Until** convergence criterion is satisfied
- Different kinds of measures can be used
  - Manhattan distance (L_1 norm), square Euclidean distance, Cosine similarity

Example: *K-Means* Clustering

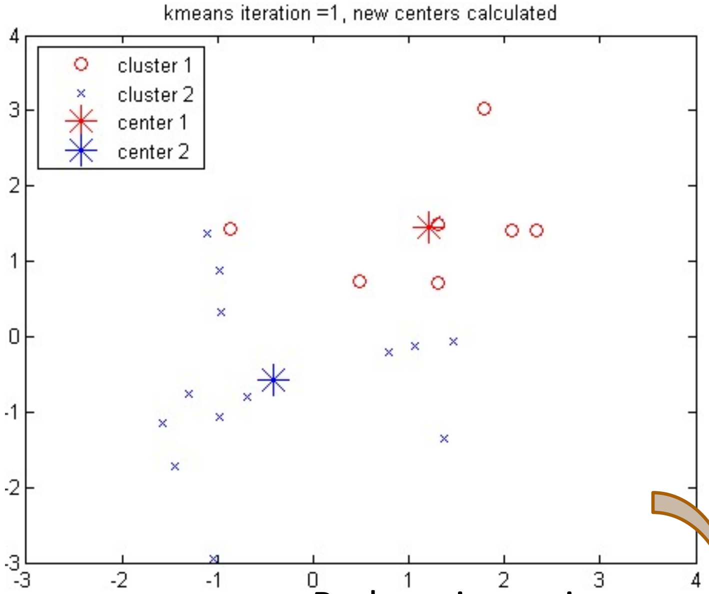


The original data points & randomly select K = 2 centroids

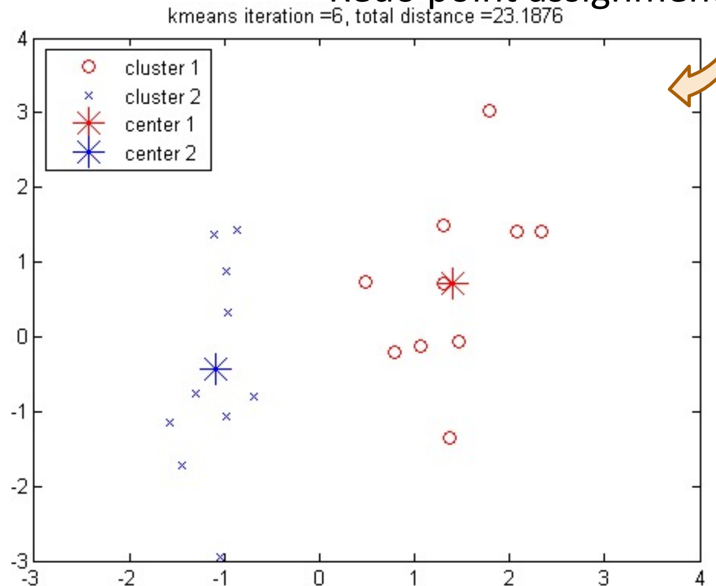
Assign points to clusters



Recompute cluster centers



Redo point assignment



Execution of the K-Means Clustering Algorithm

Select K points as initial centroids

Repeat

- Form K clusters by assigning each point to its closest centroid
- Re-compute the centroids (i.e., *mean point*) of each cluster

Until convergence criterion is satisfied

Discussion on the *K-Means* Method

- **Efficiency**: $O(tKn)$ where n: # of objects, K: # of clusters, and t: # of iterations
  - Normally, K, $t \ll n$; thus, an efficient method
- K-means clustering often ***terminates at* a *local optimal***
  - Initialization can be important to find high-quality clusters
- **Need to specify *K*,** the *number* of clusters, in advance
  - There are ways to automatically determine the "*best*" *K*
  - In practice, one often runs a range of values and selected the "*best*" *K* value
- **Sensitive to noisy data and *outliers***
  - Variations: Using K-medians, K-medoids, etc.
- K-means is applicable only to objects in a continuous n-dimensional space
  - Using the K-modes for ***categorical data***
- Not suitable to discover clusters with ***non-convex shapes***
  - Using density-based clustering, kernel *K*-means, etc.

Variations of *K-Means*

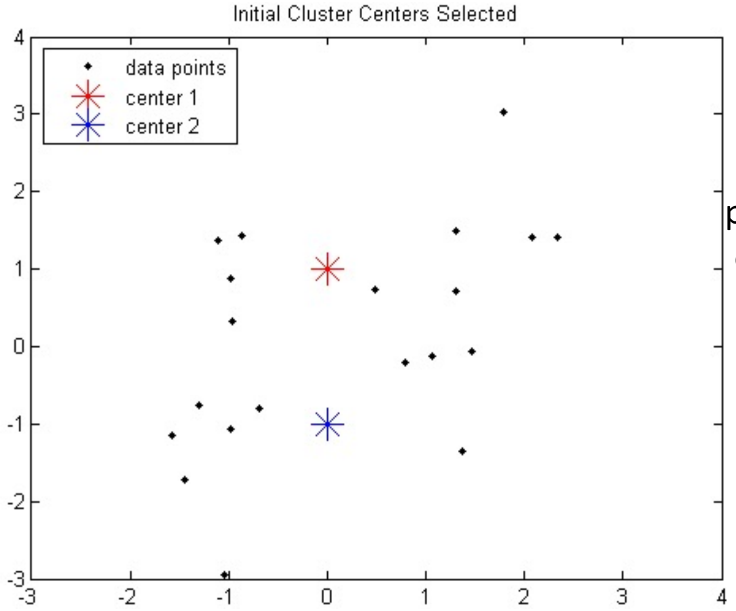
- There are many variants of the *K-Means* method, varying in different aspects
  - Choosing better initial centroid estimates
    - *K-means++, Intelligent K-Means, Genetic K-Means*

      To be discussed in this lecture
  - Choosing different representative prototypes for the clusters
    - *K-Medoids, K-Medians, K-Modes*

      To be discussed in this lecture
  - Applying feature transformation techniques
    - *Weighted K-Means, Kernel K-Means*

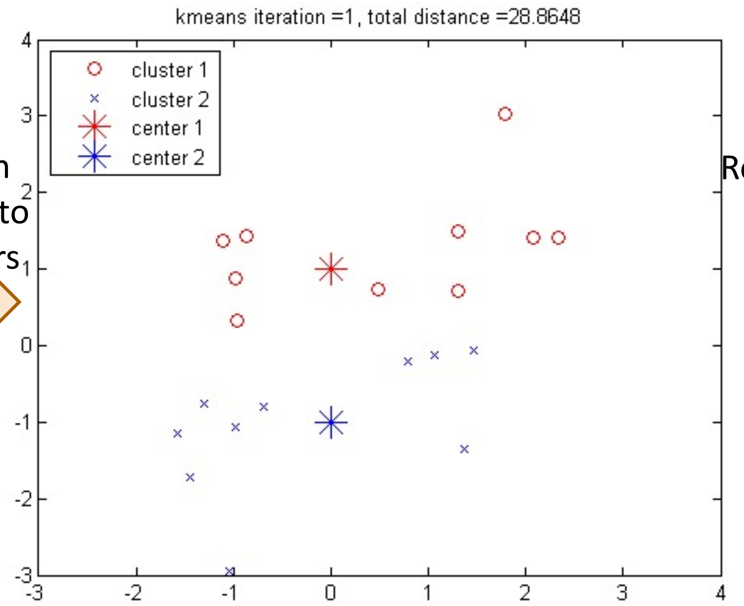
      To be discussed in this lecture

Poor Initialization in K-Means May Lead to Poor Clustering

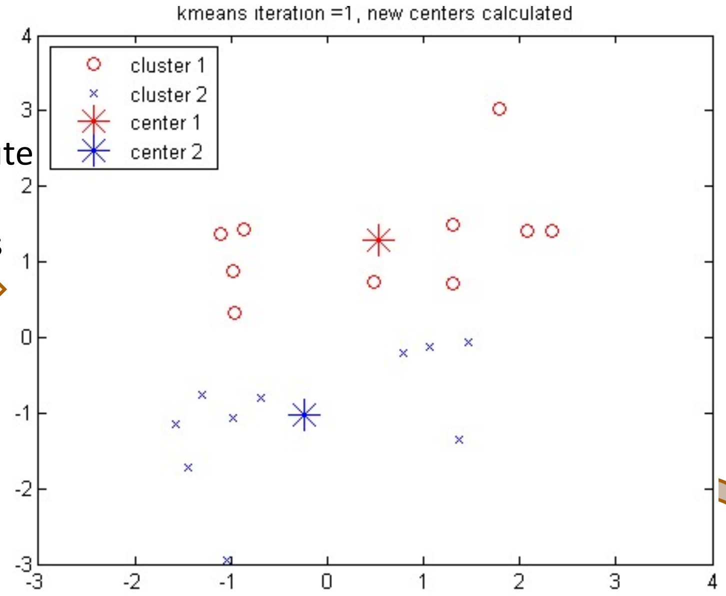


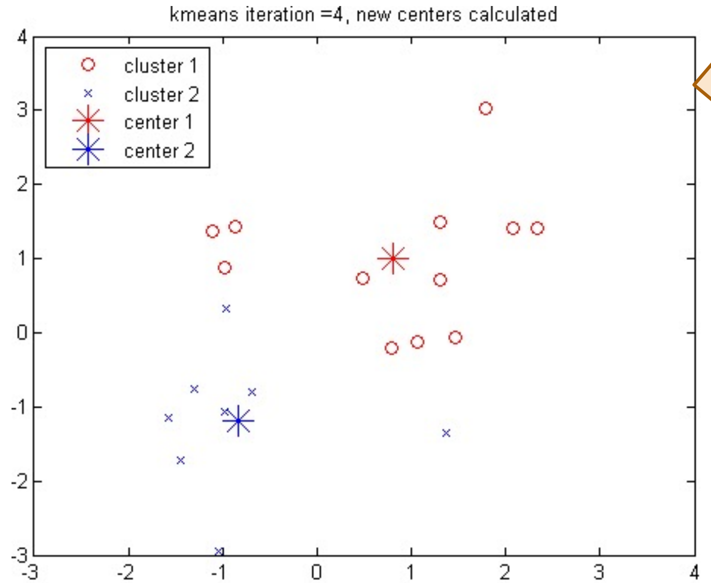
Another random selection of k centroids for the same data points

Assign points to clusters



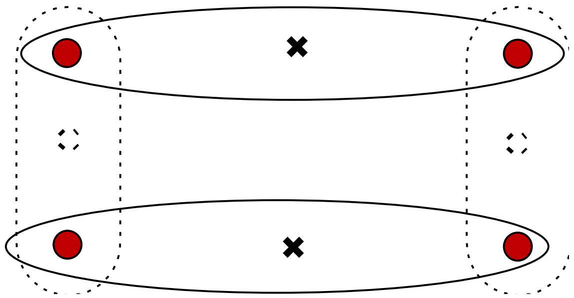
Recompute cluster centers





- Rerun of the *K-Means* using another random *K* seeds
- This run of *K*-Means generates a poor quality clustering

Initialization of K-Means: Problem and Solution

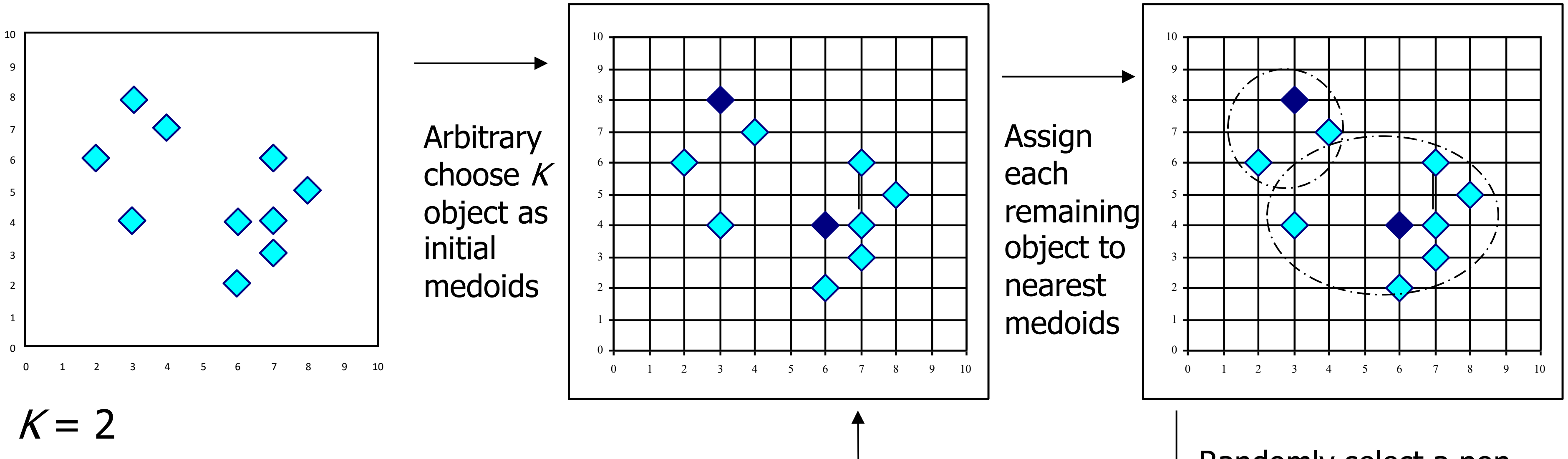


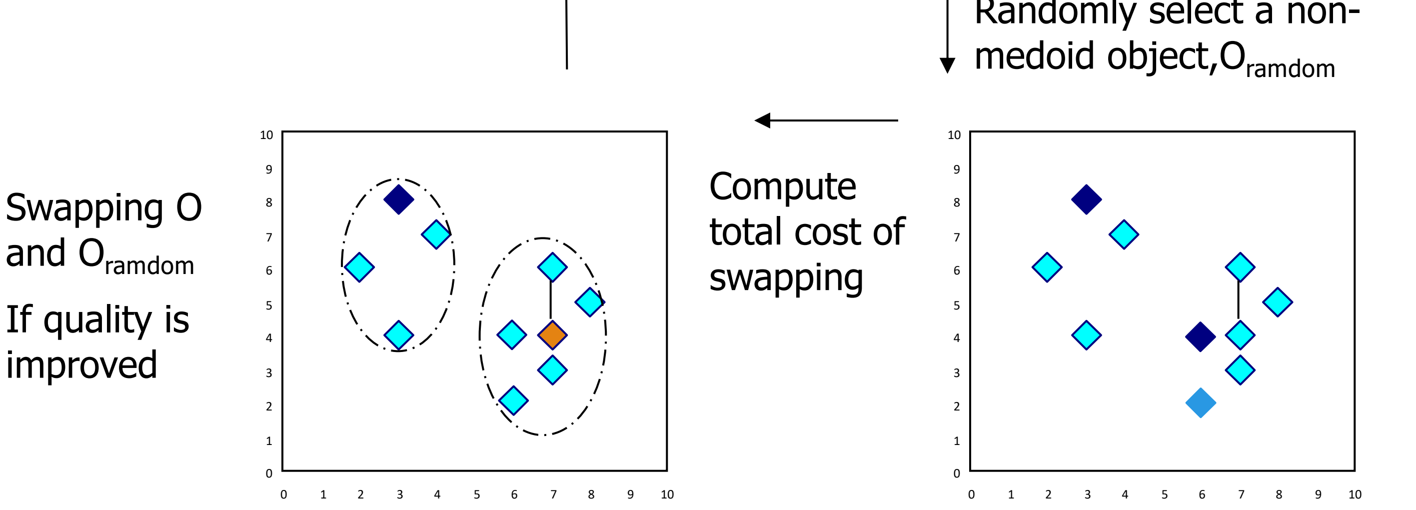
- Different initializations may generate rather different clustering results (some could be far from optimal)
- Original proposal (MacQueen'67): Select *K* seeds randomly
  - Need to run the algorithm multiple times using different seeds
- There are many methods proposed for better initialization of *k* seeds
  - ***K-Means++*** (Arthur & Vassilvitskii'07):
    - The first centroid is selected at random
    - The next centroid selected is the one that is farthest from the currently selected (selection is based on a weighted probability score)
    - The selection continues until *K* centroids are obtained

Handling Outliers: From *K-Means* to *K-Medoids*

- The *K-Means* algorithm is sensitive to outliers!—since an object with an extremely large value may substantially distort the distribution of the data
- *K-Medoids*: Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located** object in a cluster
- The *K-Medoids* clustering algorithm:
  - Select K points as the initial representative objects (i.e., as initial K *medoids*)
  - **Repeat**
    - Assigning each point to the cluster with the closest medoid
    - Randomly select a non-representative object o_i
    - Compute the total cost S of swapping the medoid m with o_i
    - If $S < 0$, then swap m with o_i to form the new set of medoids
  - **Until** convergence criterion is satisfied

PAM: A Typical *K-Medoids* Algorithm





Select initial *K medoids* randomly

Repeat

Object re-assignment

Swap medoid *m* with o_i if it improves the clustering quality

Until convergence criterion is satisfied

Discussion on *K-Medoids* Clustering

- *K-Medoids* Clustering: Find *representative* objects (<u>medoids</u>) in clusters
- *PAM* (Partitioning Around Medoids: Kaufmann & Rousseeuw 1987)
  - Starts from an initial set of medoids, and
  - Iteratively replaces one of the medoids by one of the non-medoids if it improves the total sum of the squared errors (SSE) of the resulting clustering
  - *PAM* works effectively for small data sets but does not scale well for large data sets (due to the computational complexity)
  - Computational complexity: PAM: $O(K(n - K)^2)$ (quite expensive!)
- Efficiency improvements on PAM
  - *CLARA* (Kaufmann & Rousseeuw, 1990):
    - PAM on samples; $O(Ks^2 + K(n - K))$, s is the sample size
  - *CLARANS* (Ng & Han, 1994): Randomized re-sampling, ensuring efficiency + quality

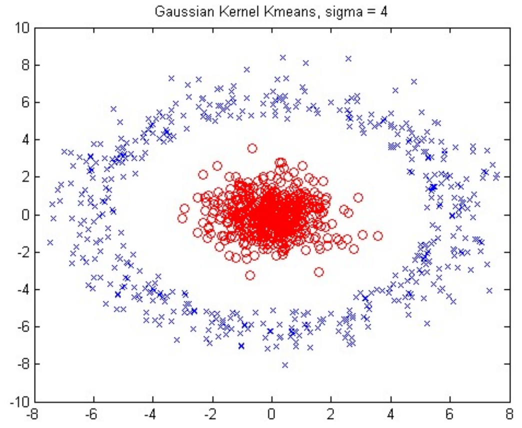
K-Medians: Handling Outliers by Computing Medians

- Medians are less sensitive to outliers than means
  - Think of the median salary vs. mean salary of a large firm when adding a few top executives!
- ***K-Medians***: Instead of taking the **mean** value of the object in a cluster as a reference point, **medians** are used (L_1-norm as the distance measure)
- The criterion function for the *K-Medians* algorithm: $S = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} | x_{ij} - med_{kj} |$
- The *K-Medians* clustering algorithm:
  - Select *K* points as the initial representative objects (i.e., as initial *K medians*)
  - **Repeat**
    - Assign every point to its nearest median
    - Re-compute the median using the median of each individual feature
  - **Until** convergence criterion is satisfied

K-Modes: Clustering Categorical Data

- *K-Means* cannot handle non-numerical (categorical) data
  - Mapping categorical value to 1/0 cannot generate quality clusters
- ***K-Modes***: An extension to *K-Means* by replacing means of clusters with ***modes***
  - Mode: The value that appears most often in a **set** of data values
- Dissimilarity measure between object X and the center of a cluster Z
  - $\Phi(x_j, z_j) = 1 - n_j^r/n_l$ when $x_j = z_j = r$; 1 when $x_j \neq z_j$
    - where z_j is the categorical value of attribute j in Z_l, n_l is the number of objects in cluster *l*, and n_j^r is the number of objects whose j^{th} attribute value is r
- This dissimilarity measure (distance function) is **frequency-based**
- Algorithm is still based on iterative *object cluster assignment* and *centroid update*
- A ***fuzzy K-Modes*** method is proposed to calculate a ***fuzzy cluster membership value*** for each object to each cluster
- A mixture of categorical and numerical data: Using a ***K-Prototype*** method

Kernel K-Means Clustering



- *Kernel K-Means* can be used to detect non-convex clusters
  - A region is **convex** if it contains all the line segments connecting any pair of its points.
  - *K-Means* can only detect clusters that are linearly separable
- Idea: Project data onto the high-dimensional kernel space, and then perform *K-Means* clustering
  - Map data points in the input space onto a high-dimensional feature space using the kernel function
  - Perform *K-Means* on the mapped feature space
- Computational complexity is higher than K-Means
  - Need to compute and store *n* x *n* kernel matrix generated from the kernel function on the original data, where *n* is the number of points
- *Spectral clustering* can be considered as a variant of Kernel K-Means clustering

Kernel Functions and Kernel K-Means Clustering

- Typical kernel functions:
  - Polynomial kernel of degree h: $K(\boldsymbol{X_i}, \boldsymbol{X_j}) = (\boldsymbol{X_i} \cdot \boldsymbol{X_j} + 1)^h$
  - Gaussian radial basis function (RBF) kernel: $K(\boldsymbol{X_i}, \boldsymbol{X_j}) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$
  - Sigmoid kernel: $K(\boldsymbol{X_i}, \boldsymbol{X_j}) = \tanh(\kappa \boldsymbol{X_i} \cdot \boldsymbol{X_j} - \delta)$
- The formula for kernel matrix K for any two points $x_i, x_j \in C_k$ is $K_{x_i x_j} = \phi(x_i) \bullet \phi(x_j)$
- The SSE criterion of *kernel K-means*: $$SSE(C) = \sum_{k=1}^{K} \sum_{x_{i \in C_k}} \| \phi(x_i) - c_k \|^2$$
  - The formula for the cluster centroid:

$$c_k = \frac{\sum_{x_{i \in C_k}} \phi(x_i)}{|C_k|}$$

- Clustering can be performed without the actual individual projections $\phi(x_i)$ and $\phi(x_j)$ for the data points $x_i, x_j \in C_k$

Example: Kernel Functions and Kernel K-Means Clustering

- Gaussian radial basis function (RBF) kernel: $K(\boldsymbol{X_i}, \boldsymbol{X_j}) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$
- Suppose there are 5 original 2-dimensional points:
  - $x_1(0, 0)$, $x_2(4, 4)$, $x_3(-4, 4)$, $x_4(-4, -4)$, $x_5(4, -4)$
- If we set σ to 4, we will have the following points in the kernel space
  - E.g., $||x_1 - x_2||^2 = (0-4)^2 + (0-4)^2 = 32$, thus, $K(x_1, x_2) = e^{-\frac{32}{2\cdot 4^2}} = e^{-1}$

Original Space

	x	y
x_1	0	0
x_2	4	4
x_3	-4	4
x_4	-4	-4
x_5	4	-4

RBF Kernel Space ($\sigma = 4$)

$K(x_i, x_1)$	$K(x_i, x_2)$	$K(x_i, x_3)$	$K(x_i, x_4)$	$K(x_i, x_5)$
0	$e^{-\frac{4^2+4^2}{2\cdot 4^2}} = e^{-1}$	e^{-1}	e^{-1}	e^{-1}
e^{-1}	0	e^{-2}	e^{-4}	e^{-2}
e^{-1}	e^{-2}	0	e^{-2}	e^{-4}
e^{-1}	e^{-4}	e^{-2}	0	e^{-2}
e^{-1}	e^{-2}	e^{-4}	e^{-2}	0

Kernel k-means

Minimize sum of squared error:

Kernel k-means: $\min \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij} \left\| x_i - c_j \right\|^2$

Replace with $\varphi(x)$

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij} \left\| \varphi(x_i) - \widetilde{c}_j \right\|^2$$

$u_{ij} \in \{0,1\}$ $\qquad \sum_{j=1}^{m} u_{ij} = 1$

www.cse.msu.edu/~cse802/Kernel k-means&Spectral_Clustering.pptx

Kernel k-means

- Cluster centers:

$$\tilde{c}_j = \frac{1}{n_j} \sum_{i=1}^{n} u_{ij} \phi(x_i)$$

- Substitute for centers:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij} \left\| \phi(x_i) - \tilde{c}_j \right\|^2$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{m} u_{ij} \left\| \phi(x_i) - \frac{1}{n_j} \sum_{l=1}^{n} u_{lj} \phi(x_l) \right\|^2$$

www.cse.msu.edu/~cse802/Kernel k-means&Spectral_Clustering.pptx

Kernel k-means

- Use kernel trick:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} u_{ij} \left\| \phi(x_i) - \tilde{c}_j \right\|^2 = trace(K) - trace(UKU')$$

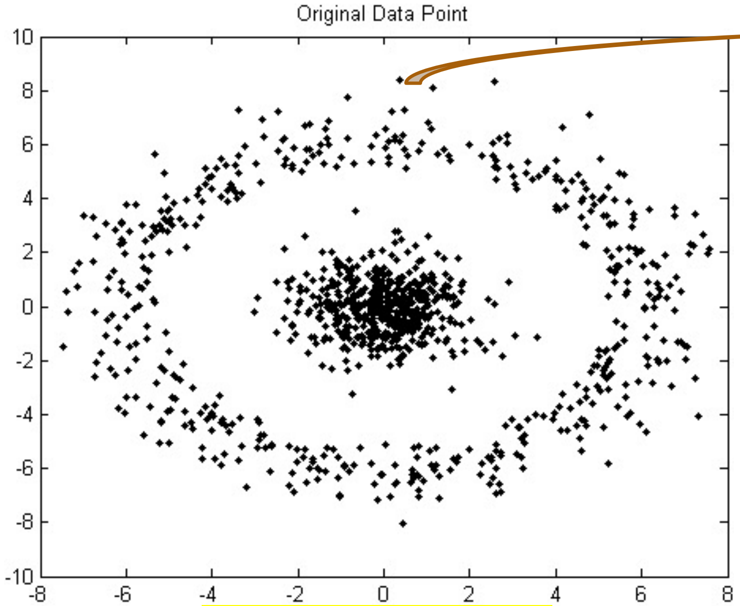
- Optimization problem:

$$\min \quad trace(K) - trace(UKU') \approx \max \quad trace(UKU')$$

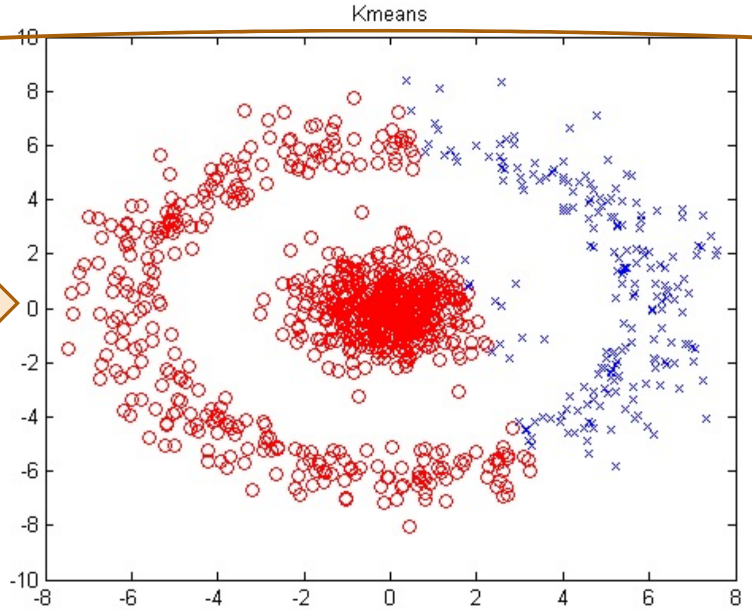
- K is the n x n kernel matrix, U is the optimal normalized cluster membership matrix

www.cse.msu.edu/~cse802/Kernel k-means&Spectral_Clustering.pptx

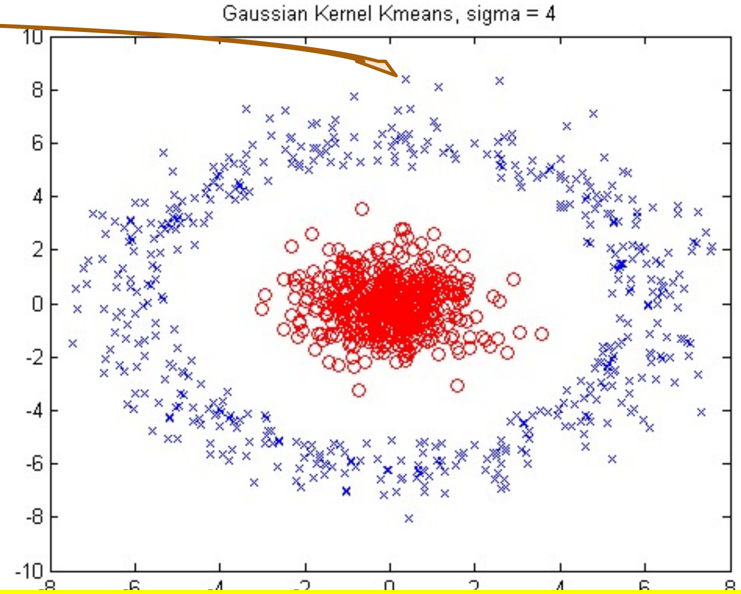
Example: Kernel K-Means Clustering



The original data set



The result of K-Means clustering



The result of Gaussian Kernel K-Means clustering

- The above data set cannot generate quality clusters by K-Means since it contains non-convex clusters
- Gaussian RBF Kernel transformation maps data to a kernel matrix K for any two points x_i, x_j: $K_{x_i x_j} = \phi(x_i) \bullet \phi(x_j)$ and Gaussian kernel: $K(\boldsymbol{X_i}, \boldsymbol{X_j}) = e^{-\|X_i - X_j\|^2 / 2\sigma^2}$
- K-Means clustering is conducted on the mapped data, generating quality clusters

Chapter 10. Cluster Analysis: Basic Concepts and Methods

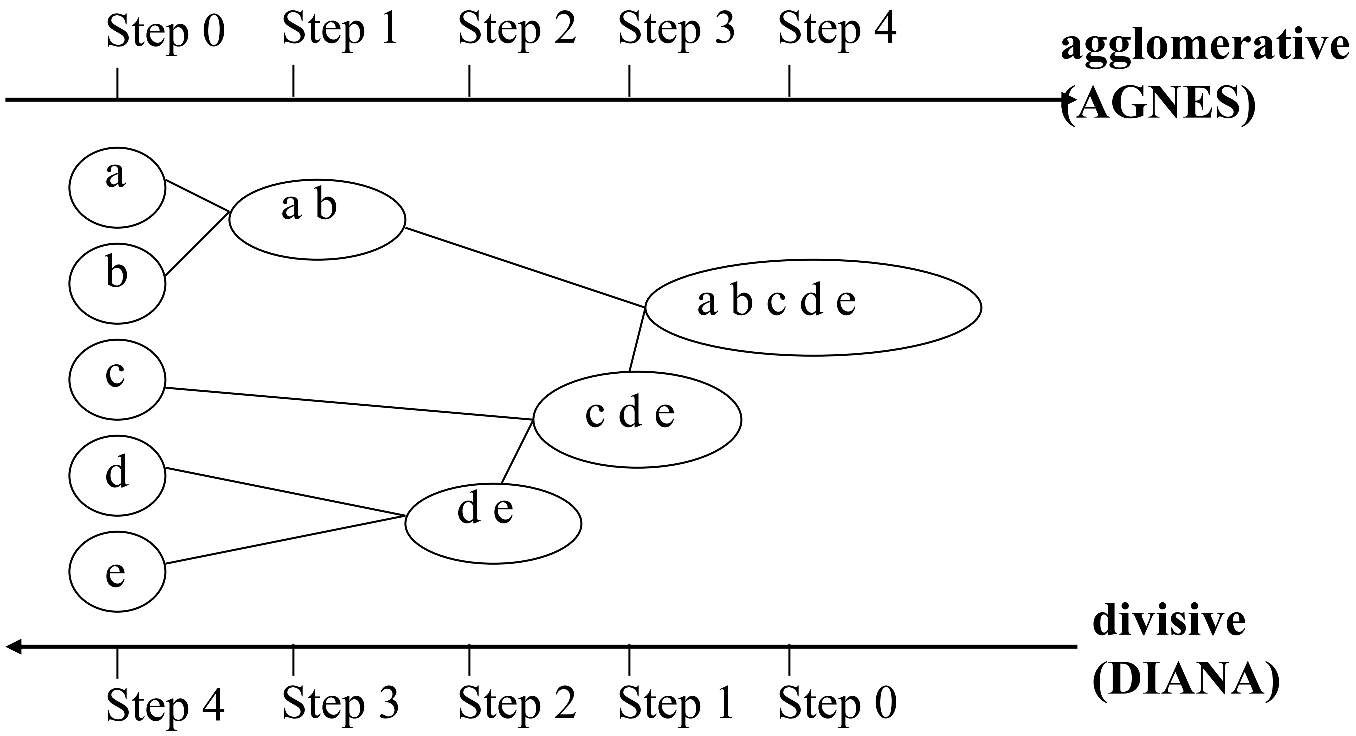
- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Spectral Clustering
- Evaluation of Clustering
- Summary

Hierarchical Clustering Methods

- Basic Concepts of Hierarchical Algorithms
- Agglomerative Clustering Algorithms
- Divisive Clustering Algorithms
- Extensions to Hierarchical Clustering

Hierarchical Clustering: Basic Concepts

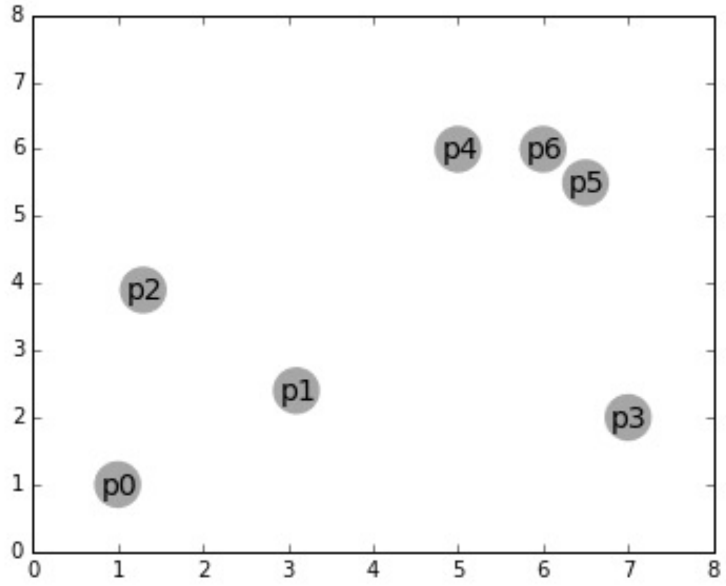
- Hierarchical clustering
  - Generate a clustering hierarchy (drawn as a **dendrogram**)
  - Not required to specify ***K*,** the number of clusters
  - More deterministic
  - No iterative refinement

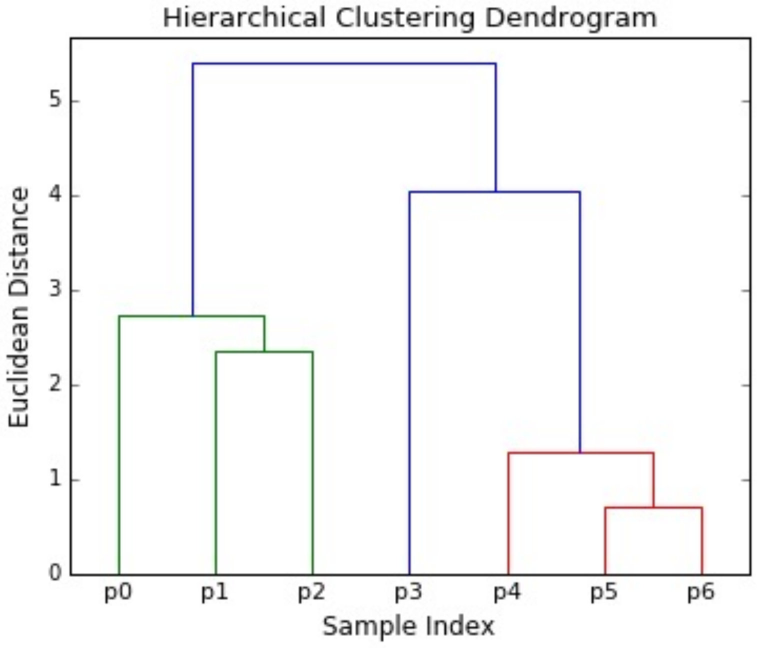


- Two categories of algorithms:
  - **Agglomerative**: Start with singleton clusters, continuously merge two clusters at a time to build a **bottom-up** hierarchy of clusters
  - **Divisive:** Start with a huge macro-cluster, split it continuously into two groups, generating a **top-down** hierarchy of clusters

Dendrogram: Shows How Clusters are Merged

- Dendrogram: Decompose a set of data objects into a tree of clusters by multi-level nested partitioning
- A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster



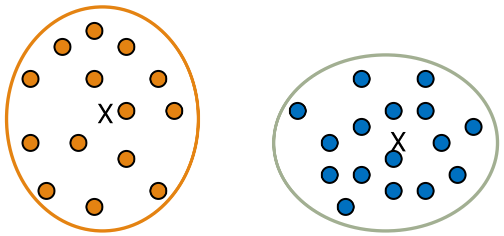


Hierarchical clustering generates a dendrogram (a hierarchy of clusters)

Agglomerative Clustering Algorithm

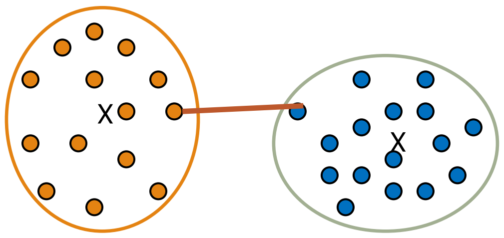
- AGNES (AGglomerative NESting) (Kaufmann and Rousseeuw, 1990)
  - Continuously merge nodes that have the least dissimilarity
  - Eventually all nodes belong to the same cluster

- Agglomerative clustering varies on different similarity measures among clusters
  - Single link (nearest neighbor)
  - Complete link (diameter)
  - Average link (group average)
  - Centroid link (centroid similarity)

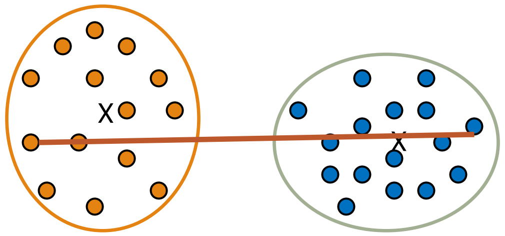


Single Link vs. Complete Link in Hierarchical Clustering

- **Single link** (nearest neighbor)
  - The similarity between two clusters is the similarity between their most similar (nearest neighbor) members
  - Local similarity-based: Emphasizing more on close regions, ignoring the overall structure of the cluster
  - Capable of clustering non-elliptical shaped group of objects
  - Sensitive to noise and outliers

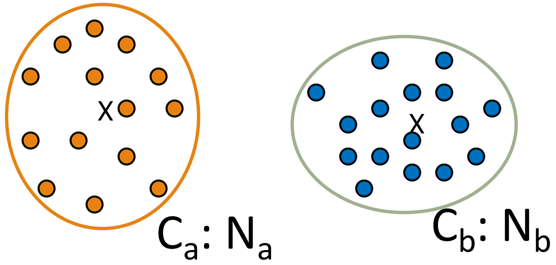


- **Complete link** (diameter)
  - The similarity between two clusters is the similarity between their most dissimilar members
  - Merge two clusters to form one with the smallest diameter
  - Nonlocal in behavior, obtaining compact shaped clusters
  - Sensitive to outliers

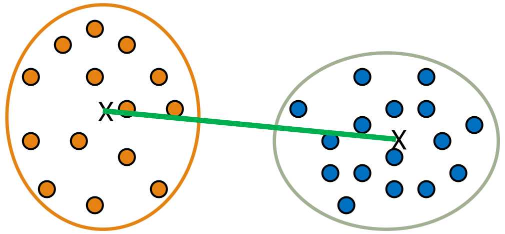


Agglomerative Clustering: Average vs. Centroid Links

- Agglomerative clustering with **average link**
  - **Average link**: The average distance between an element in one cluster and an element in the other (i.e., all pairs in two clusters)
    - Expensive to compute



- Agglomerative clustering with **centroid link**
  - **Centroid link**: The distance between the centroids of two clusters



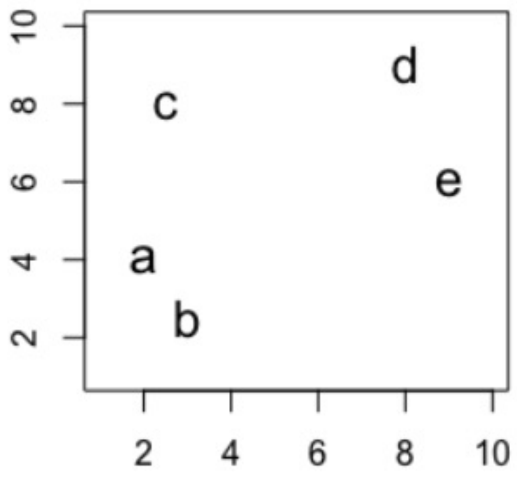
- **Group Averaged Agglomerative Clustering (GAAC)**
  - Let two clusters C_a and C_b be merged into $C_{a\cup b}$. The new centroid is:
    - N_a is the cardinality of cluster C_a, and c_a is the centroid of C_a

$$c_{a\cup b} = \frac{N_a c_a + N_b c_b}{N_a + N_b}$$

  - The similarity measure for GAAC is the average of their distances
- Agglomerative clustering with **Ward's criterion**
  - **Ward's criterion:** The increase in the value of the SSE criterion for the clustering obtained by merging them into C_a U C_b: $W(C_{a\cup b}, c_{a\cup b}) - W(C, c) = \frac{N_a N_b}{N_a + N_b} d(c_a, c_b)$

Agglomerative Clustering: Example of Single Link

- 2-D Data points
  - a(2,4)
  - b(3,2)
  - c(2,8)
  - d(8,9)
  - e(9,6)

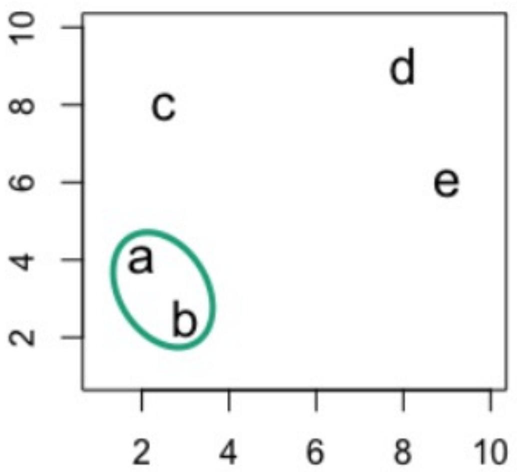


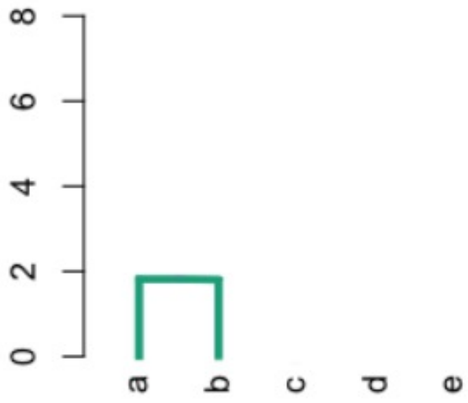
Distance Matrix

	a	b	c	d	e
a	0				
b	2.2	0			
c	4	6.1	0		
d	7.8	8.6	6.1	0	
e	7.3	7.2	7.3	3.2	0

Agglomerative Clustering: Example of Single Link

- 2-D Data points





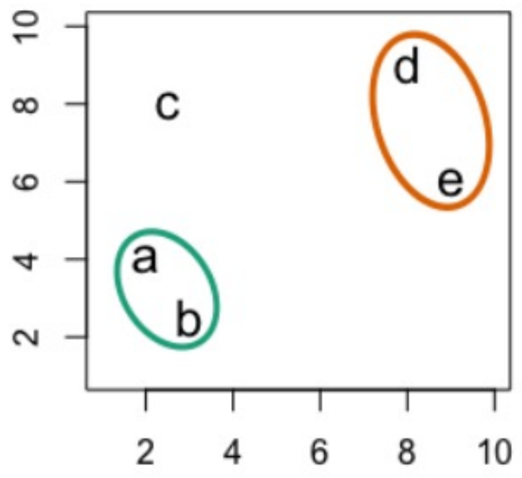
Distance Matrix

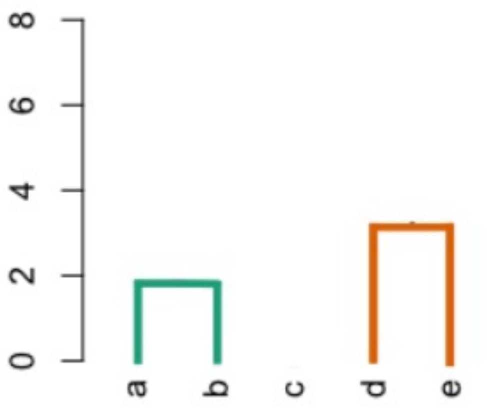
	a,b	c	d	e
a,b	0			
c	4	0		
d	7.8	6.1	0	
e	7.2	7.3	3.2	0

- Update distance
  - Distance((a,b), c) = min(Distance(a,c), Distance(b,c)) = min(4, 6.1)=4
  - Distance((a,b), d) = min(Distance(a,d), Distance(b,d)) = min(7.8, 8.6)=7.8
  - Distance((a,b), e) = min(Distance(a,e), Distance(b,e)) = min(7.3, 7.2)=7.2

Agglomerative Clustering: Example of Single Link

- 2-D Data points





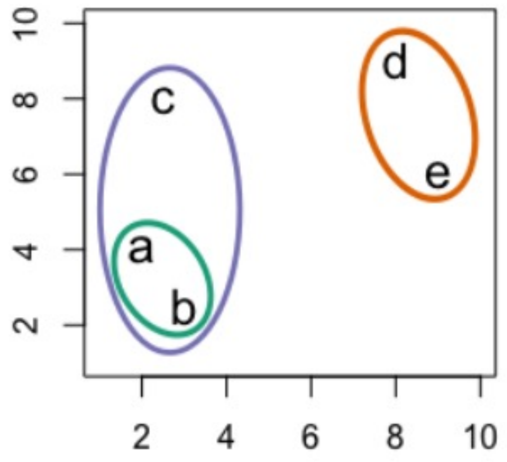
Distance Matrix

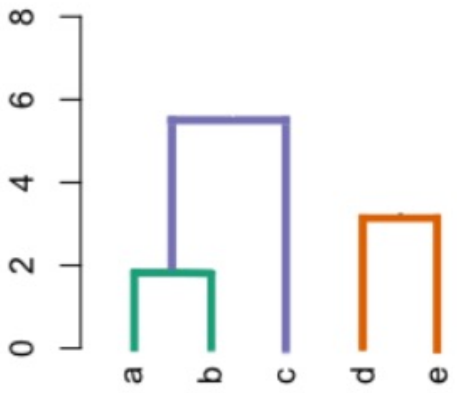
	a,b	c	d,e
a,b	0		
c	4	0	
d,e	7.2	6.1	0

- Update distance
  - Distance((d,e), (a,b)) = min(Distance(d,(a,b)), Distance(e,(a,b)) = 7.2
  - Distance((d,e), c) = min(Distance(d,c), Distance(e,c)) = 6.1

Agglomerative Clustering: Example of Single Link

- 2-D Data points





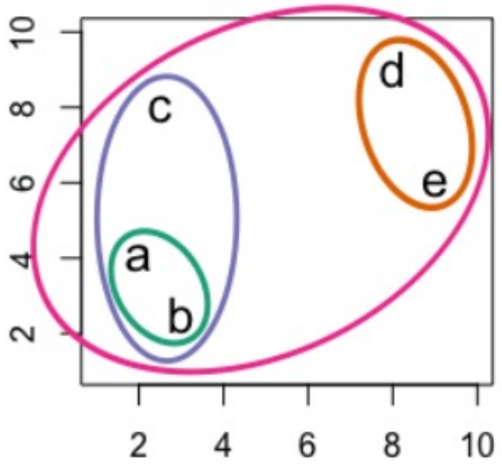
Distance Matrix

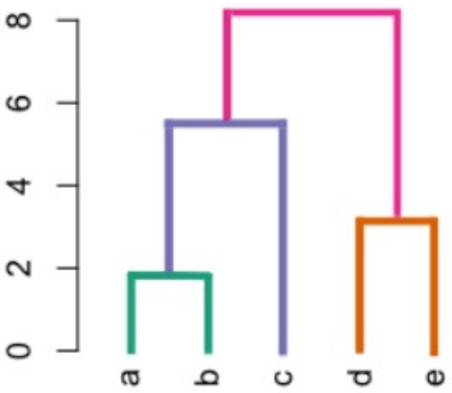
	a,b,c	d,e
a,b,c	0	
d,e	6.1	0

- Update distance
  - Distance((d,e), (c,(a,b)))= min(Distance((d,e),(a,b)), Distance((d,e),c) = 6.1

Agglomerative Clustering: Example of Single Link

- 2-D Data points

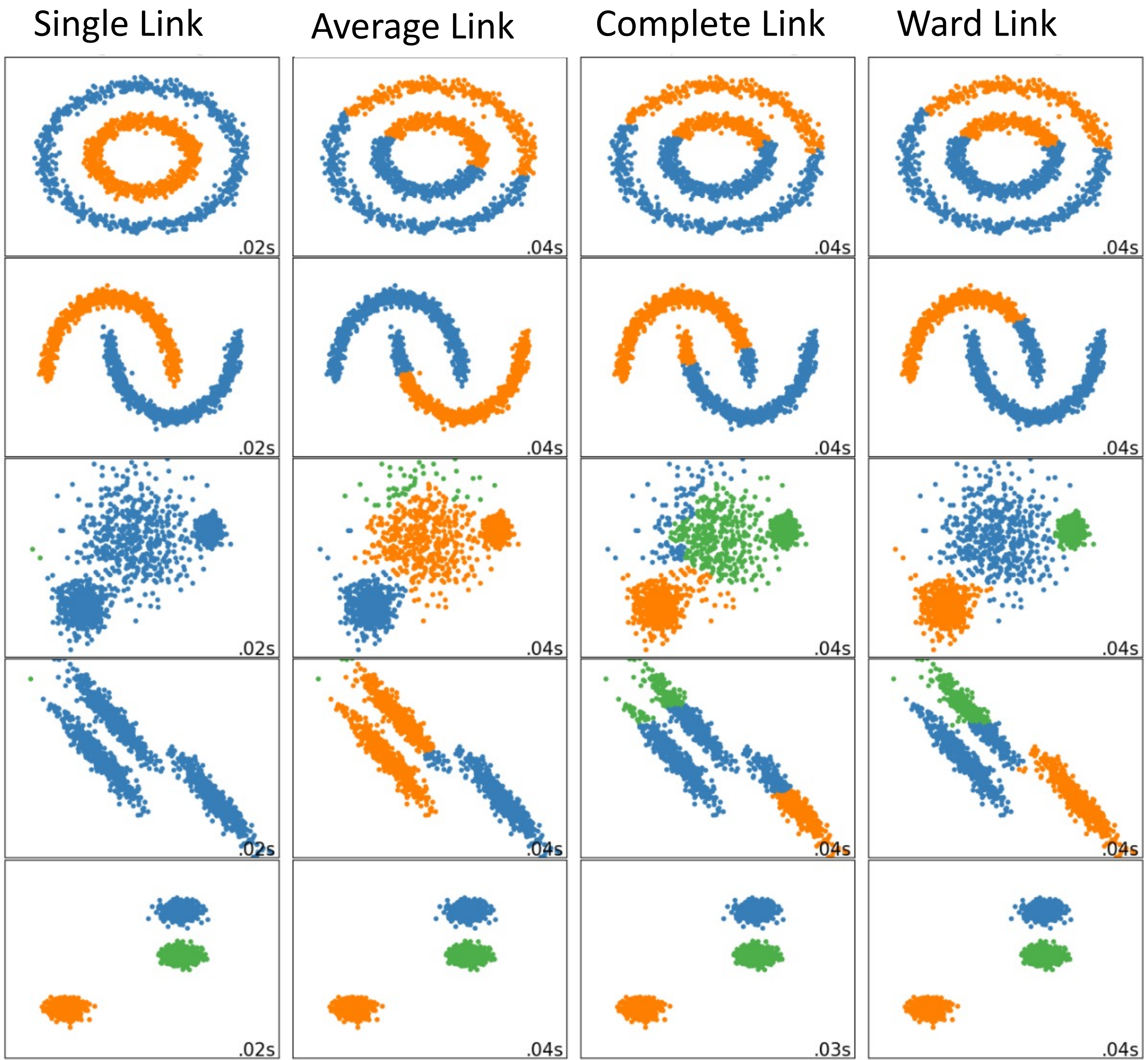




Distance Matrix

	a,b,c,d,e
a,b,c,d,e	0

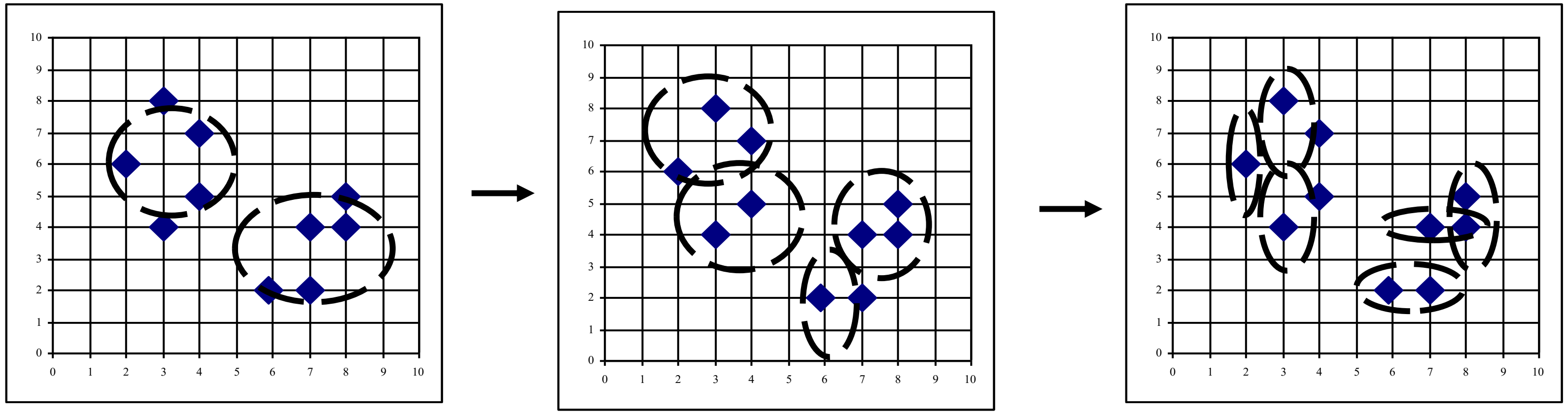
Comparison of different linkage methods



- Observations:
- Single link performs well on non-globular data, but it performs poorly in the presence of noise.
- Average and Complete Linkage performs well on globular data, but have mixed results otherwise.
- Ward is the most effective method for noisy data.

Divisive Clustering

- DIANA (Divisive Analysis) (Kaufmann and Rousseeuw,1990)
  - Implemented in some statistical analysis packages, e.g., Splus
- Inverse order of AGNES: Eventually each node forms a cluster on its own



- Divisive clustering is a top-down approach
  - The process starts at the root with all the points as one cluster
  - It recursively splits the higher level clusters to build the dendrogram
  - Can be considered as a global approach
  - More efficient when compared with agglomerative clustering

More on Algorithm Design for Divisive Clustering

- Choosing which cluster to split
  - Check the sums of squared errors of the clusters and choose the one with the largest value
- Splitting criterion: Determining how to split
  - One may use Ward's criterion to chase for greater reduction in the difference in the SSE criterion as a result of a split
  - For categorical data, Gini-index can be used
- Handling noise
  - Use a threshold to determine the termination criterion (do not generate clusters that are too small because they contain mainly noise)

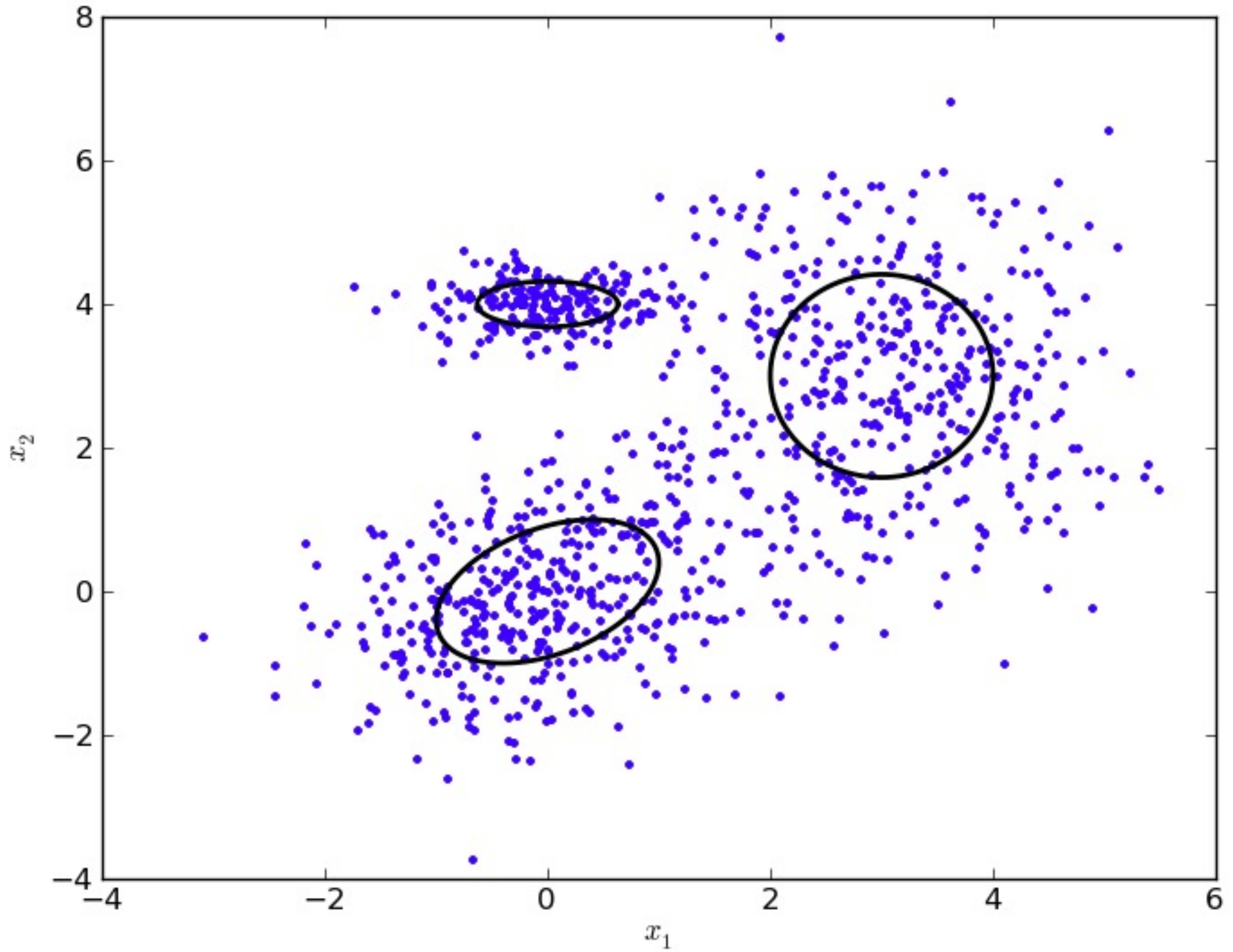
Extensions to Hierarchical Clustering

- Major weaknesses of hierarchical clustering methods
  - Can never undo what was done previously
  - Does not scale well
    - Time complexity of at least $O(n^2)$, where n is the number of total objects
- Other hierarchical clustering algorithms
  - BIRCH (1996): Use CF-tree and incrementally adjust the quality of sub-clusters
  - CURE (1998): Represent a cluster using a set of well-scattered representative points
  - CHAMELEON (1999): Use graph partitioning methods on the K-nearest neighbor graph of the data

Chapter 10. Cluster Analysis: Basic Concepts and Methods

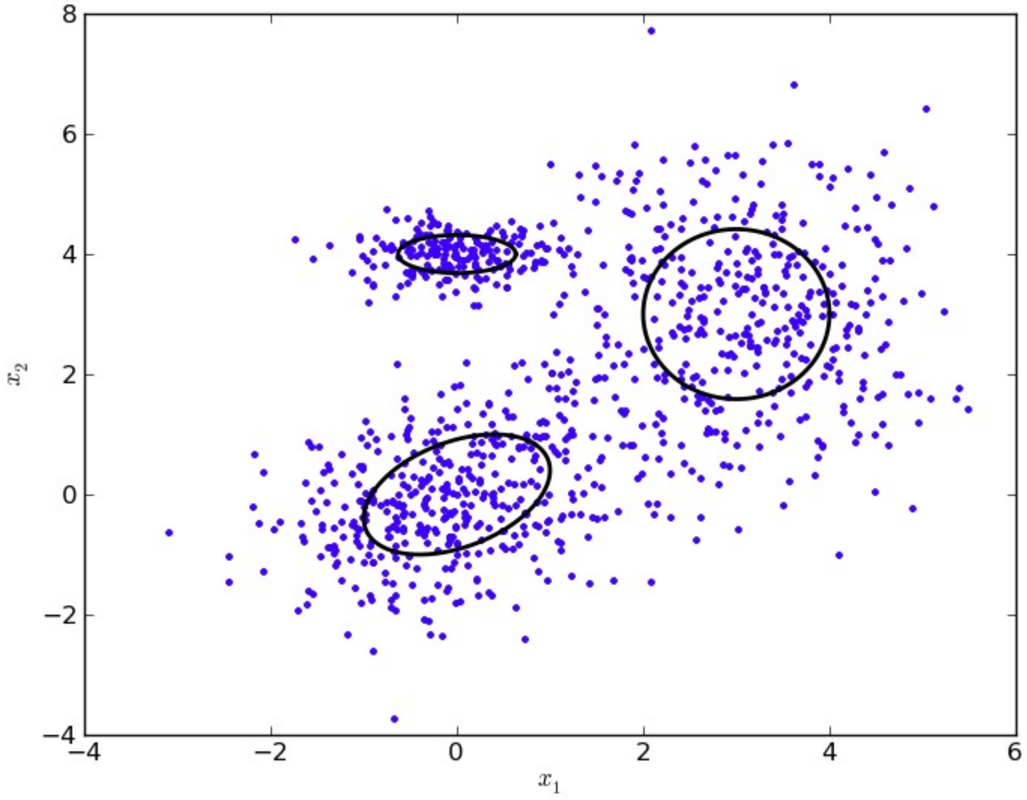
- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Evaluation of Clustering
- Summary

Hard Clustering Can Be Difficult

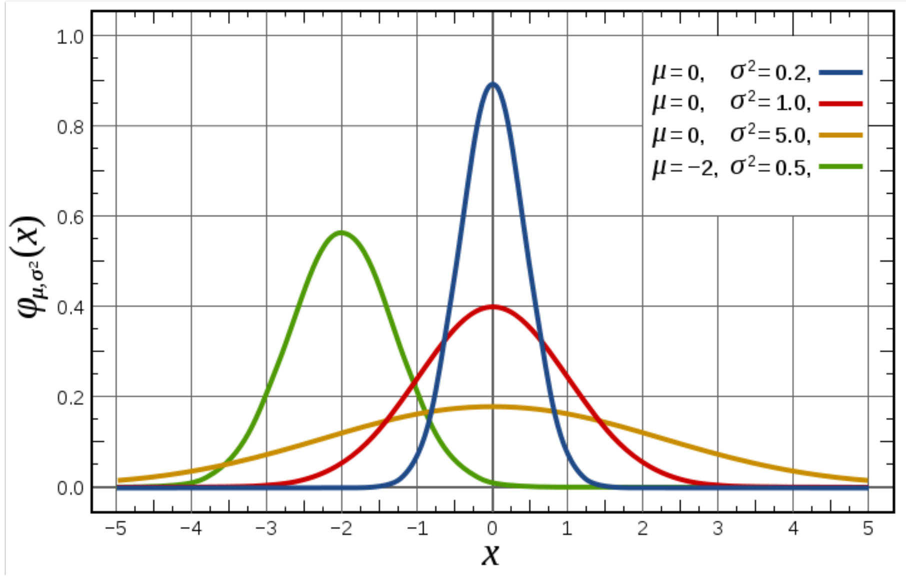


Soft Clustering

- Every object i is assigned to one cluster j with a probability
  - $P(z_i = j) \in [0,1]$ and $\Sigma_j P(z_i = j) = 1$
  - Where z_i is a hidden variable of which cluster x_i belongs to.



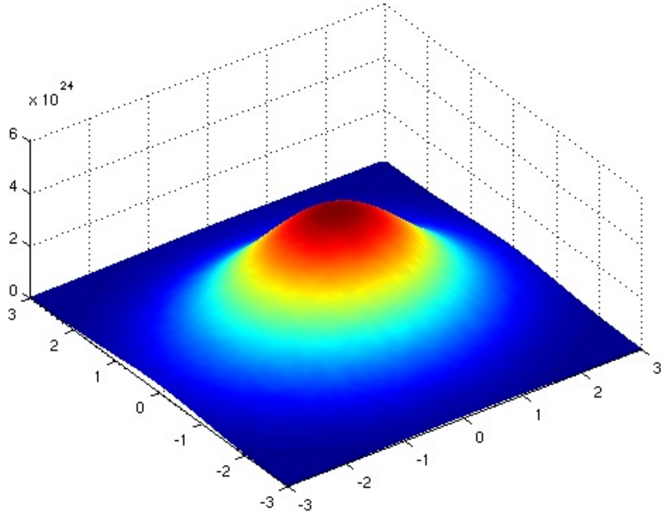
Gaussian Distribution



1-d Gaussian

$$\mathcal{N}(\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

2-d Gaussian



k-dimensional Gaussian

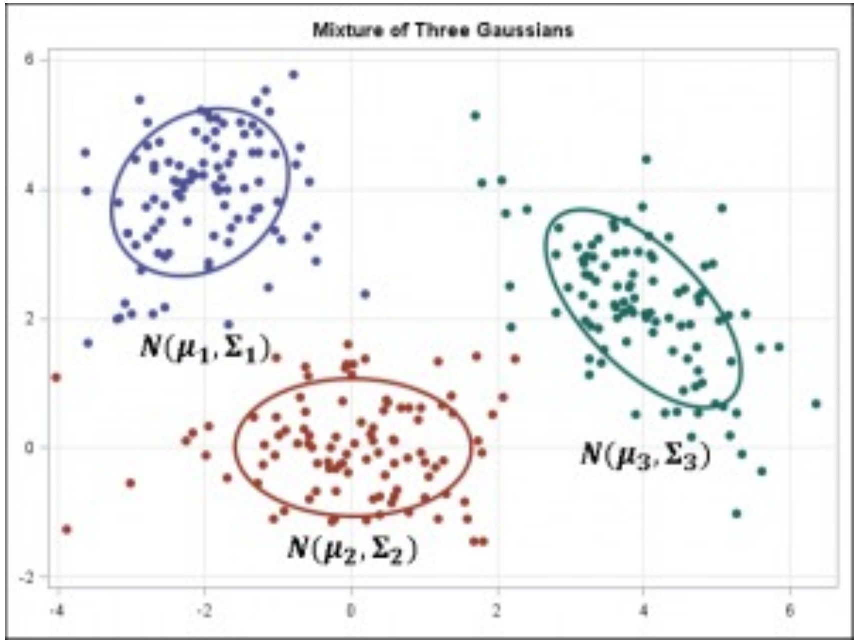
$$f_{\mathbf{x}}(x_1, \ldots, x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}}$$

Gaussian Mixture Model

- Assumptions
  - Each data point comes from one of K classes.
  - The cluster prior distribution w_j is *unknown*.
  - Each cluster c_j follows a Gaussian distribution:

$$P\left(x\middle|c_j,\theta_j\right) = \frac{1}{\sqrt{2\pi\sigma_j^2}} e^{-\frac{\left(x-\mu_j\right)^2}{2\sigma_j^2}}$$

  - The parameters for each class μ_j, σ_j are *unknown* (need to be learned)
  - The probability of x_i is the sum over all classes, $P(x_i|\theta) = \Sigma_{j=1}^{K} P\left(x_i\middle|c_j,\theta_j\right)P(c_j)$



Soft Clustering with Gaussian Mixture Model

- Every object i is assigned to one cluster j with a probability
  - $P(z_i = j) \in [0,1]$ and $\Sigma_j P(z_i = j) = 1$
  - Where z_i is a hidden variable of which cluster x_i belongs to.

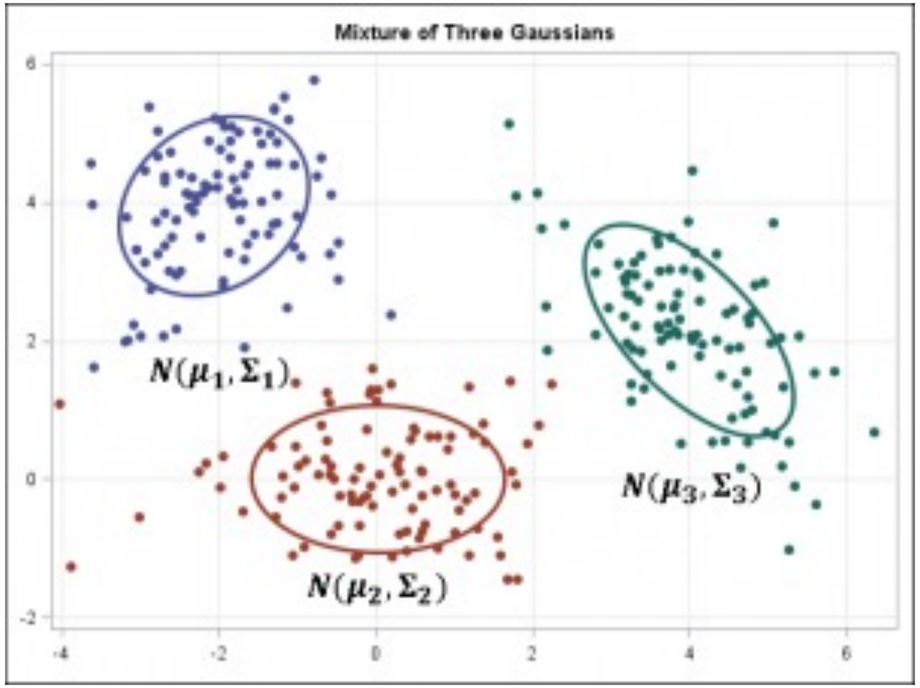
Assume the parameters of the GMM have been learned

- The probability of x_i belonging to cluster c_j :
  - $P(z_i = c_j | x_i) \propto P(x_i, z_i = c_j)$

$$= w_j P(x_i | z_i = c_j)$$

Cluster prior probabilities

Probability density function of each cluster



The E-M(Expectation Maximization) Algorithm

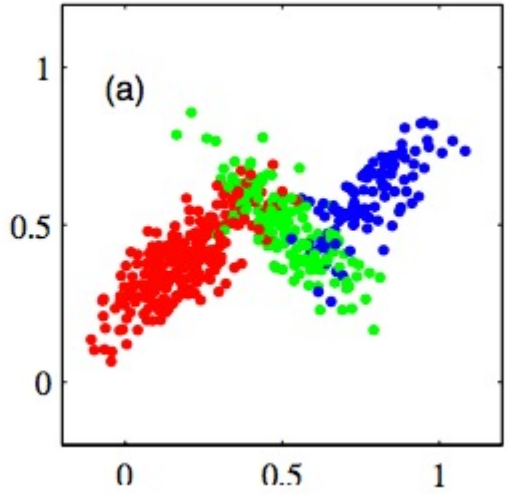
- A framework to approach maximum likelihood or maximum a posteriori estimates of parameters in statistical models.
- Expectation Step:
  - Assigns objects to clusters according to the current soft clustering or parameters of probabilistic clusters
  - $w_{ij}^{t+1} = P\left(z_i = j \middle| x_i, \theta_j^t\right) \propto w_j P\left(x_i \middle| z_i = j, \theta_j^t\right)$ ⇦ conditional probability of x_i given its cluster c_j
- Maximization Step:
  - Finds the new parameters of each cluster that maximize the expected likelihood
  - $\theta_{t+1} = argmax_\theta \Sigma_i \Sigma_j w_{ij}^{t+1} logL(x_i, z_j | \theta)$

Example: Applying E-M algorithm to 1-D GMM

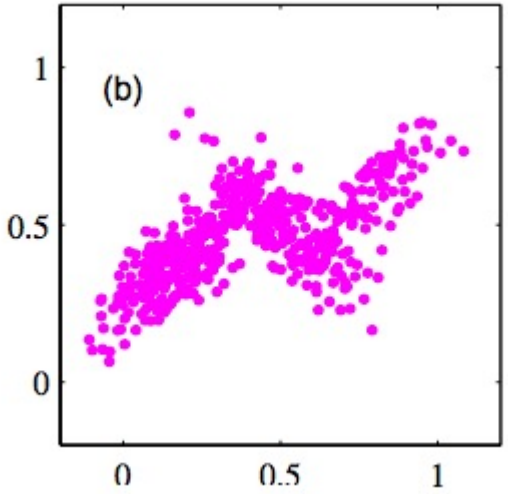
- Iteratively do the following two steps
  - E-Step: Evaluate the soft clustering probability according to $\mu_j^t, \sigma_j^t, w_j^t$
    - $w_{ij}^{t+1} = \frac{w_j^t P\left(x_i \middle| \mu_j^t, \sigma_j^t\right)}{\Sigma_k w_k^t P\left(x_i \middle| \mu_k^t, \sigma_k^t\right)}$
  - M-Step: Find the new parameters μ_j^t, σ_j^t that maximize log likelihood. In Gaussian distribution, this is equivalent to doing parameter estimation when each data point has a weight.
    - $\mu_j^{t+1} = \frac{\Sigma_i w_{ij}^{t+1} x_i}{\Sigma_i w_{ij}^{t+1}}, \left(\sigma_j^2\right)^{t+1} = \frac{\Sigma_i w_{ij}^{t+1} (x_i - \mu_j^{t+1})^2}{\Sigma_i w_{ij}^{t+1}}$ ⇦ Weighted average means and variance
    - $w_j^{t+1} = \frac{\Sigma_i w_{ij}^{t+1}}{n}$

Gaussian Mixture Model

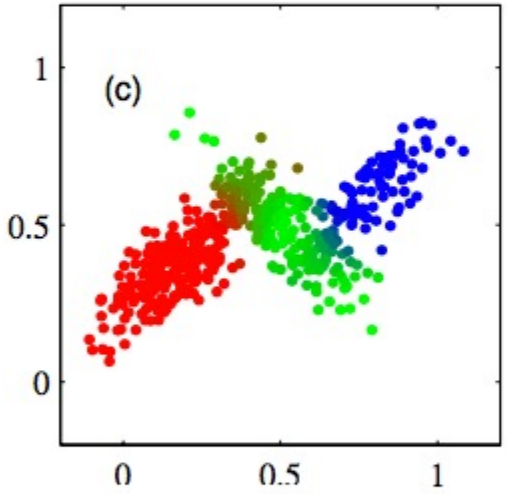
- Example of applying Gaussian Mixture Model



The data points belong to three classes. Each class follows a Gaussian distribution.



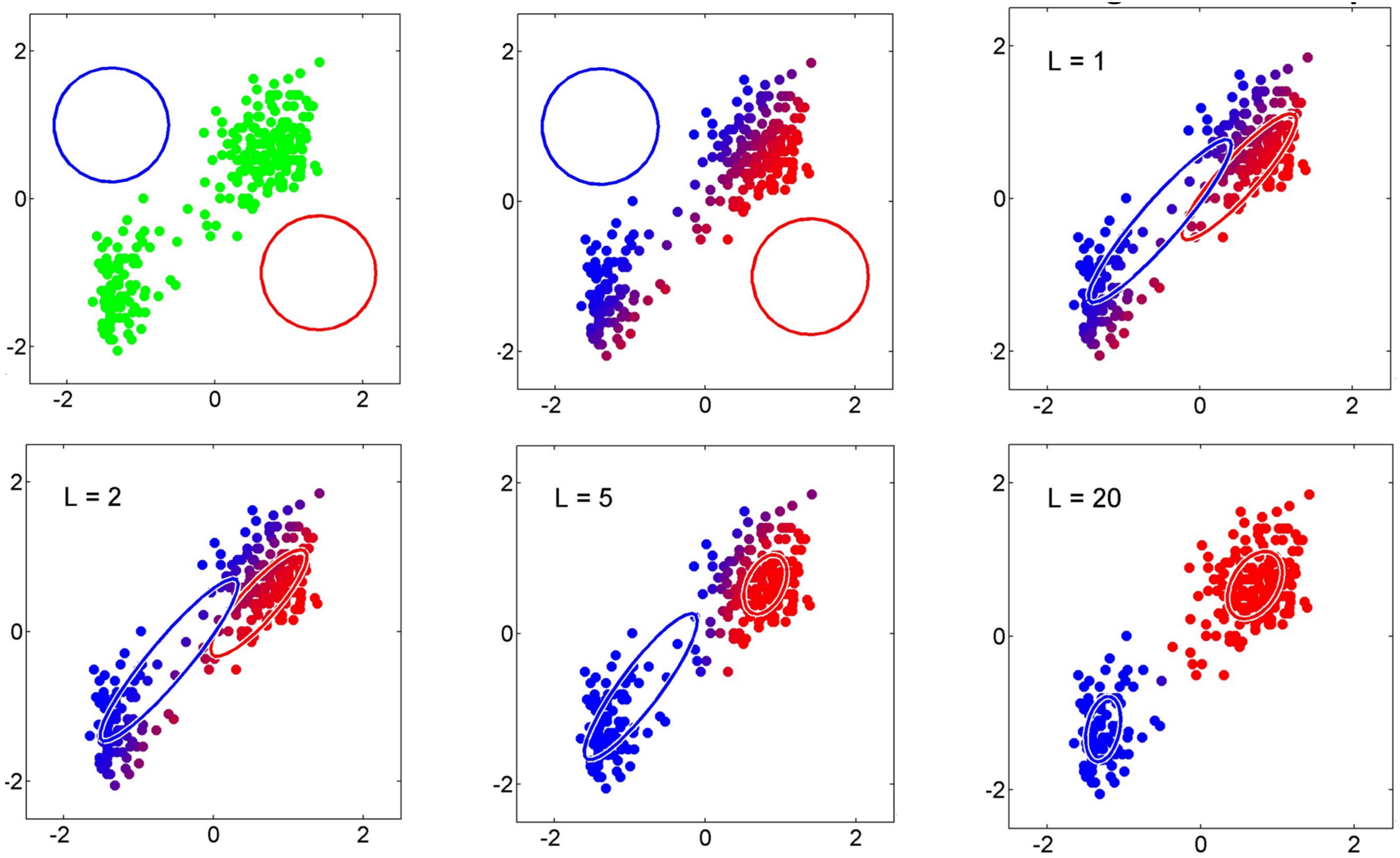
We hide the class information.



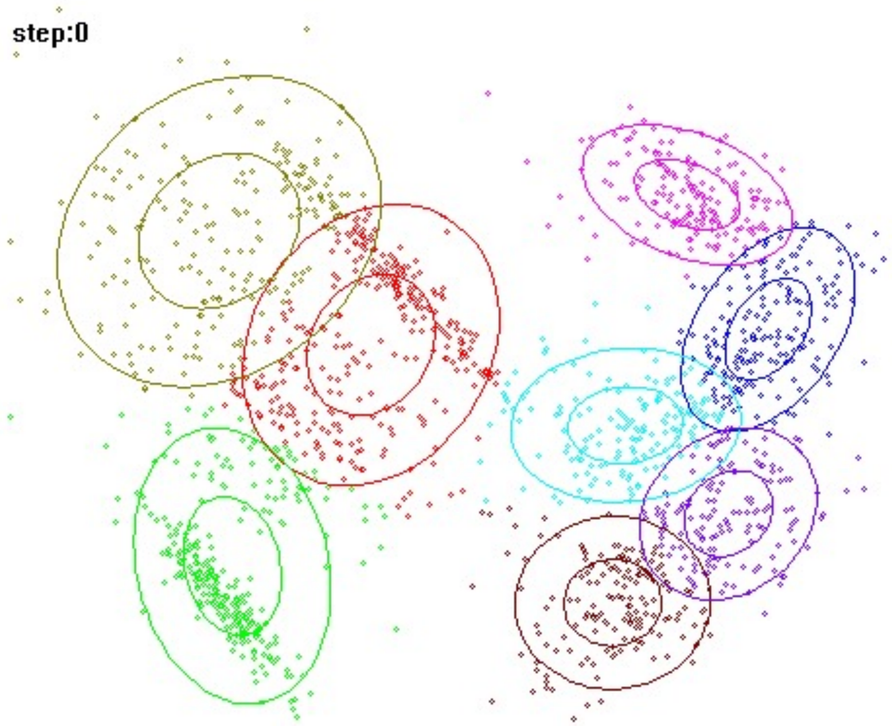
Class information inferred by GMM.

- We can use E-M algorithm to learn the parameters.

Example: Applying E-M algorithm to 2-D GMM



EM for Learning 2D Gaussian Mixture Model



Gaussian Mixture Model – Strength and Weakness

- Advantages
  - Mixture models are more general than partitioning: different densities and sizes of clusters
  - Clusters can be characterized by a small number of parameters
  - The results satisfy the statistical assumptions of generative models
- Disadvantages
  - Converges to local optimal ⇦ Overcome it by running multi-times w. random initialization
  - Computationally more expensive
  - Hard to estimate the number of clusters
  - Can only deal with spherical clusters

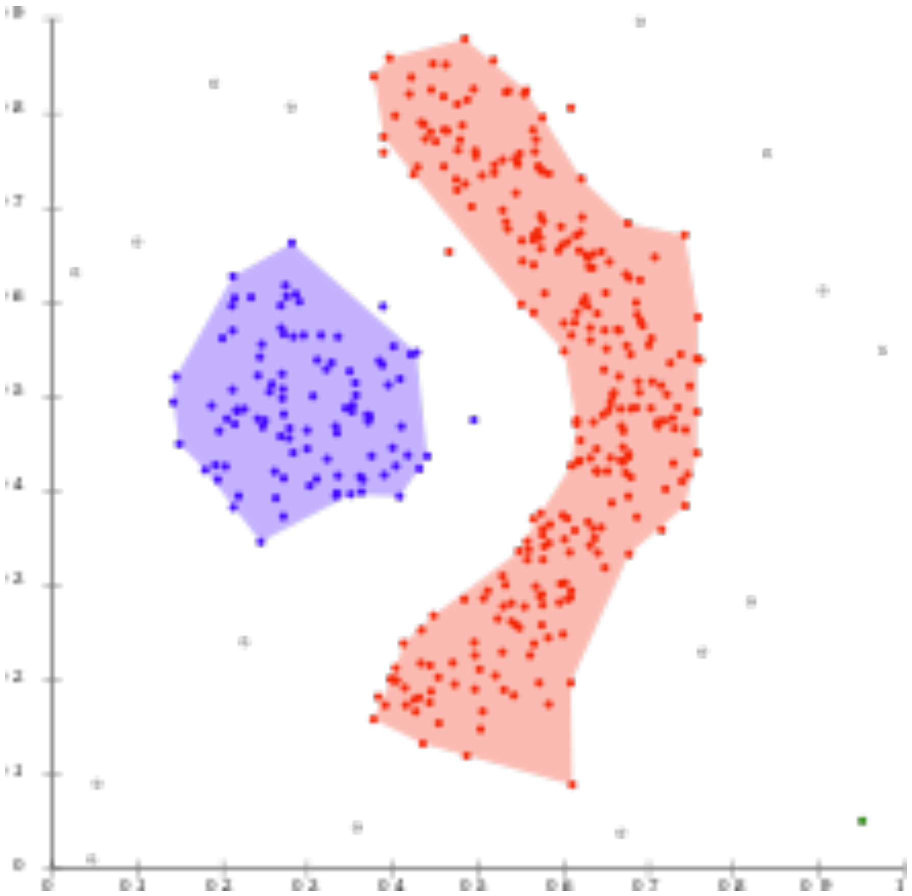
Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Evaluation of Clustering
- Summary

Density-Based Clustering

- Clustering based on density (a local criterion), such as densely-connected points

- Main Advantages
  - Discover clusters of arbitrary shape
  - Handle noise



Representative Density-Based Clustering Methods

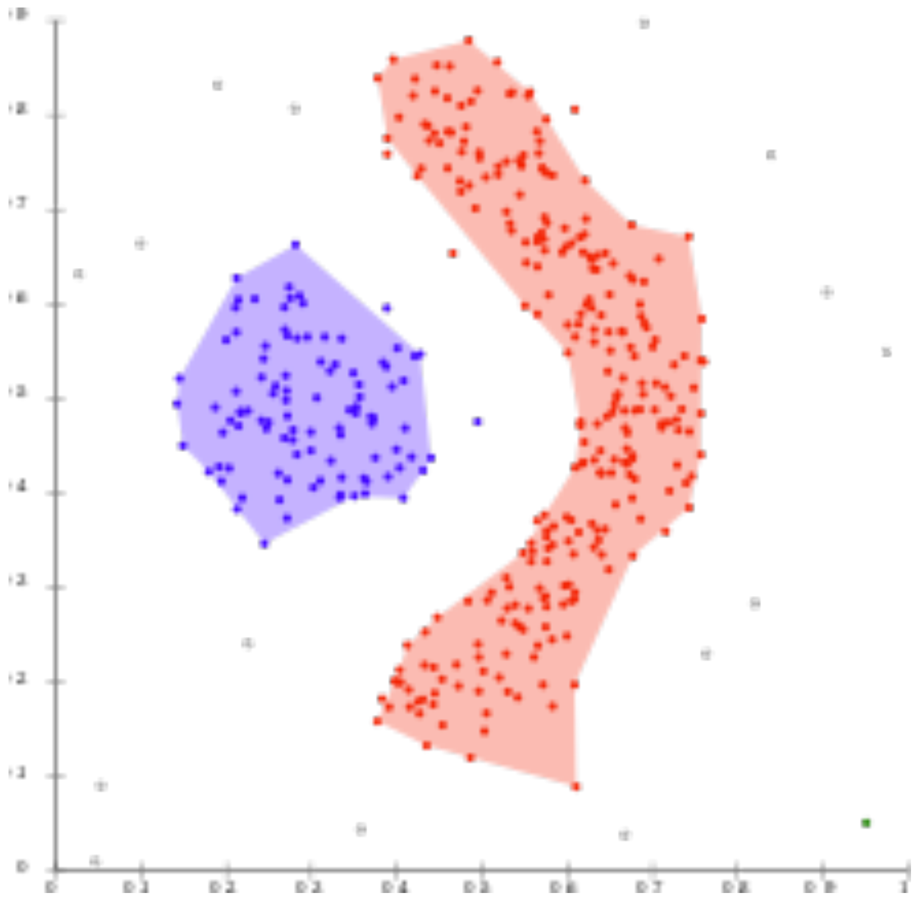
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99)
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98) (also, grid-based)

To be covered in this lecture

DBSCAN: High-Level Idea

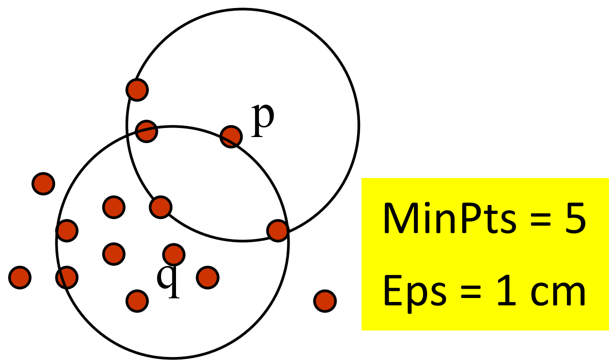
- DBSCAN
  - Discovers clusters of arbitrary shape: Density-Based Spatial Clustering of Applications with Noise

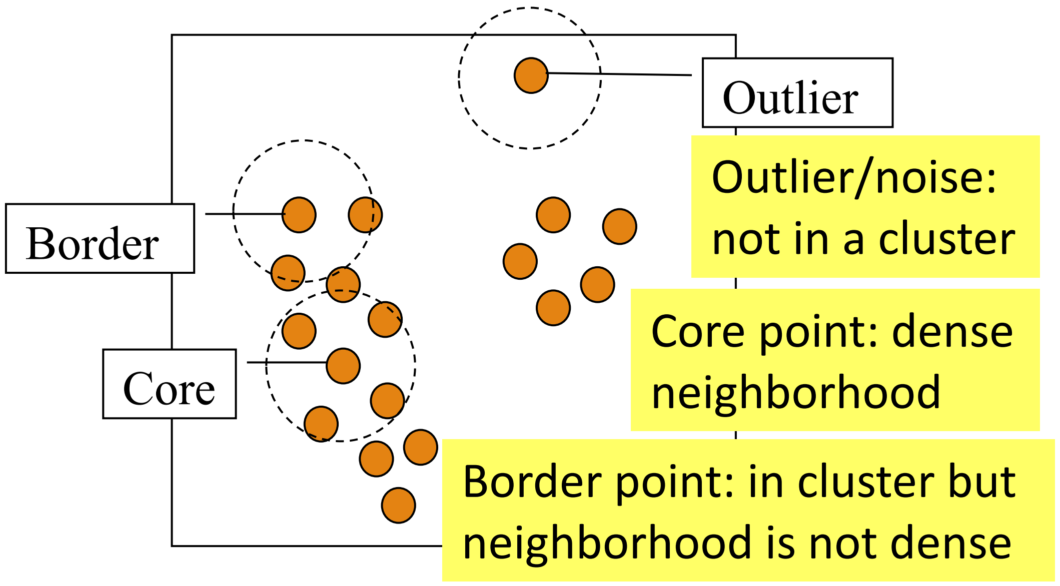
- A *density-based* notion of cluster
  - **A *cluster* is defined as a maximal set of density-connected points**



DBSCAN: Core Concepts

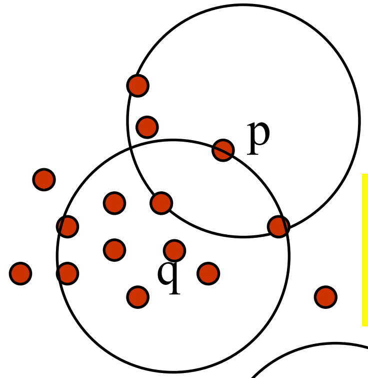
- DBSCAN: A *cluster* is defined as a maximal set of density-connected points
- Two parameters*:*
  - *Eps* (*ε*): Maximum radius of the neighborhood
  - *MinPts*: Minimum number of points in the Eps-neighborhood of a point
- The Eps(*ε*)-neighborhood of a point *q*:
  - $N_{Eps}(q)$: {p belongs to D | dist(p, q) ≤ Eps}





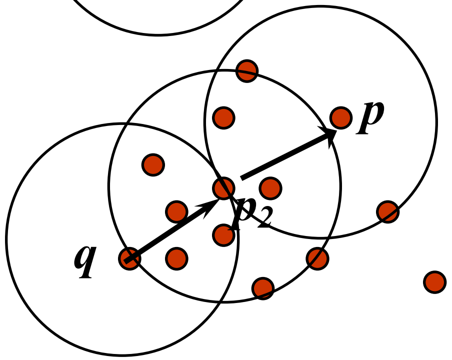
DBSCAN: Density-Reachable and Density-Connected

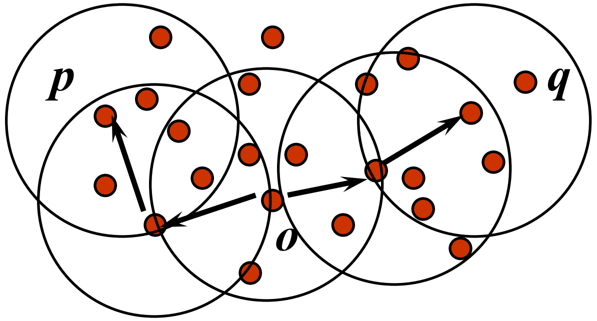
- **Directly density-reachable**:
  - A point p is directly density-reachable from a point q w.r.t. *Eps* (ε), *MinPts* if
    - p belongs to $N_{Eps}(q)$
    - **core point** condition: $|N_{Eps}(q)| \geq MinPts$
- **Density-reachable**: (asymmetric)
  - A point p is density-reachable from a point q w.r.t. *Eps*, *MinPts* if there is a chain of points $p_1, \ldots, p_n$, $p_1 = q$, $p_n = p$ such that p_{i+1} is directly density-reachable from p_i
- **Density-connected:** (symmetric)
  - A point p is density-connected to a point q w.r.t. *Eps*, *MinPts* if there is a point o such that both p and q are density-reachable from o w.r.t. *Eps* and *MinPts*



MinPts = 5

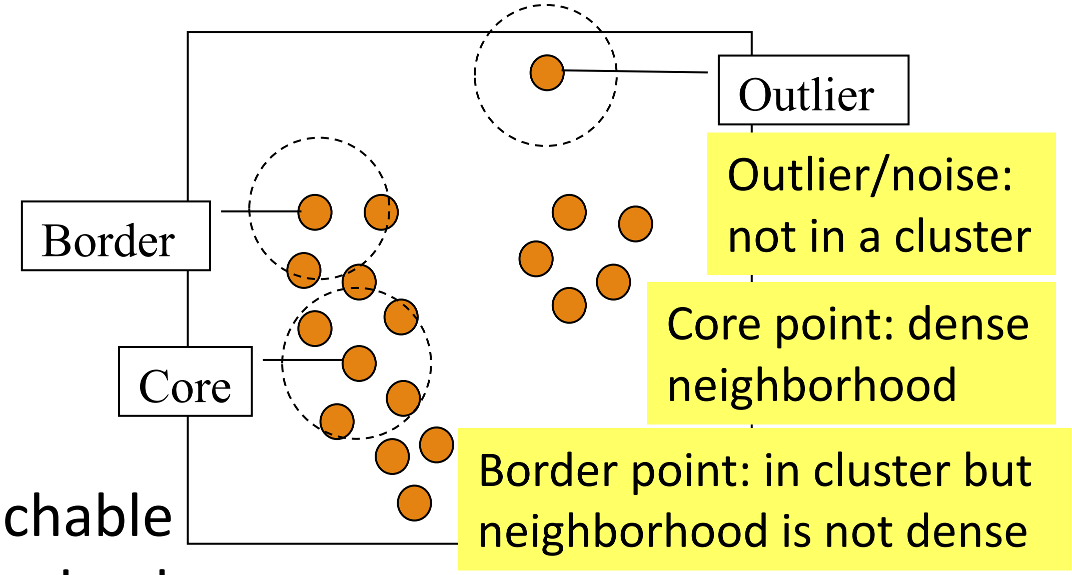
Eps = 1 cm





DBSCAN: The Algorithm

- **Algorithm**
  - Arbitrarily select a point *p*
  - Retrieve all points density-reachable from *p* w.r.t. *Eps* and *MinPts*
    - If *p* is a core point, a cluster is formed
    - If *p* is a border point, no points are density-reachable from *p,* and DBSCAN visits the next point of the database
  - Continue the process until all of the points have been processed



- **Computational complexity**
  - If a spatial index is used, the computational complexity of DBSCAN is O(nlogn), where n is the number of database objects
  - Otherwise, the complexity is $O(n^2)$

DBSCAN Is Sensitive to the Setting of Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

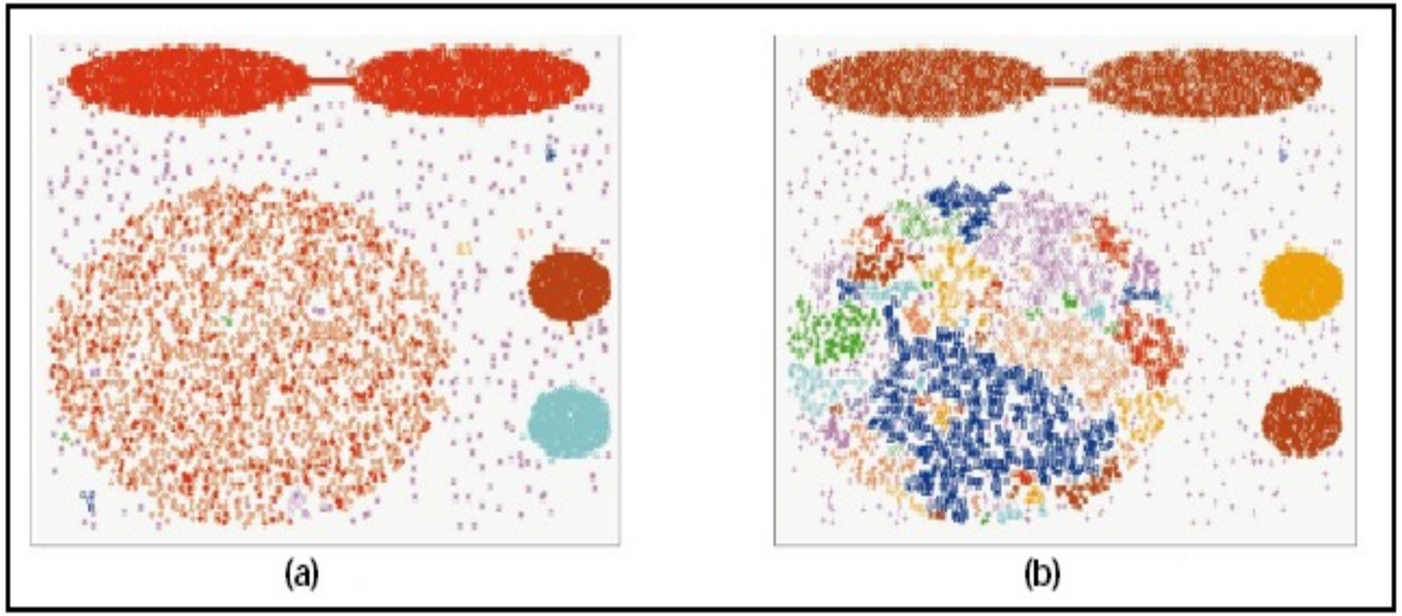
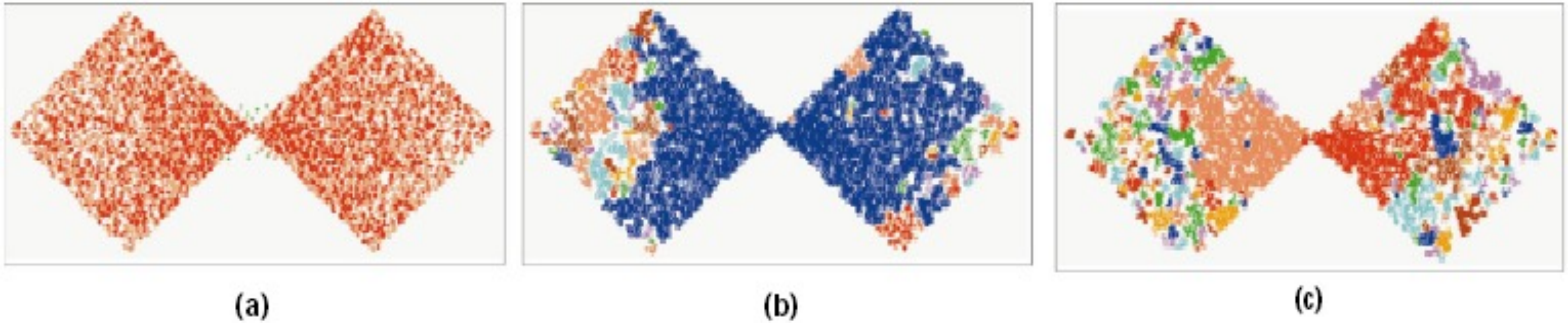


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



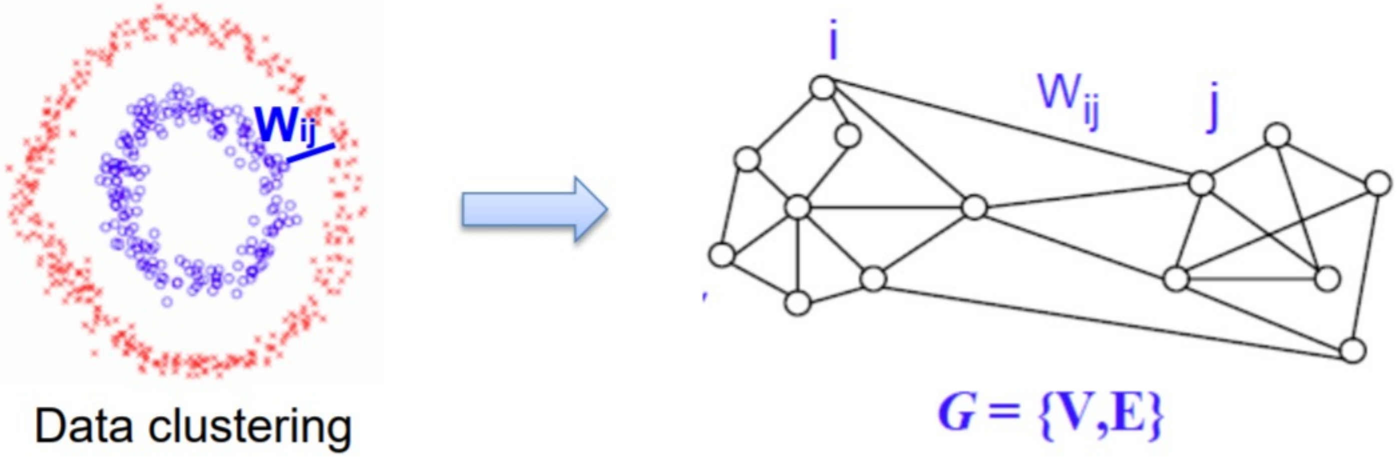
Ack. Figures from G. Karypis, E.-H. Han, and V. Kumar, *COMPUTER*, 32(8), 1999

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Spectral Clustering
- Evaluation of Clustering
- Summary

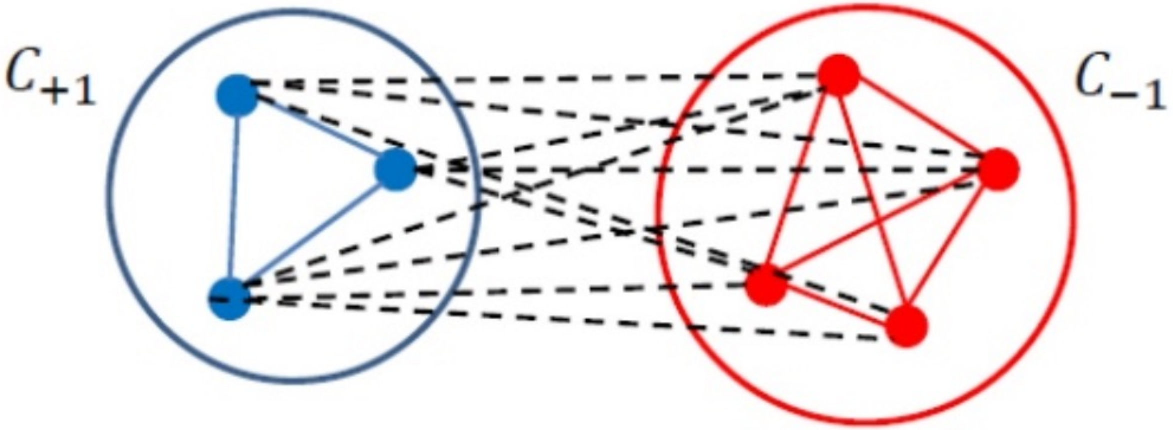
Spectral Clustering: Key Idea

- Similarity Graphs: Model local neighborhood relations between data points
  - A fully connected graph
  - K-nearest neighbor graph (each node is only connected to its K-nearest neighbors)
    - Union
    - Intersection (mutual)
  - ε-neighborhood graph



Spectral Clustering: Key Idea

- Partitioning a graph into two clusters
- Minimum cut



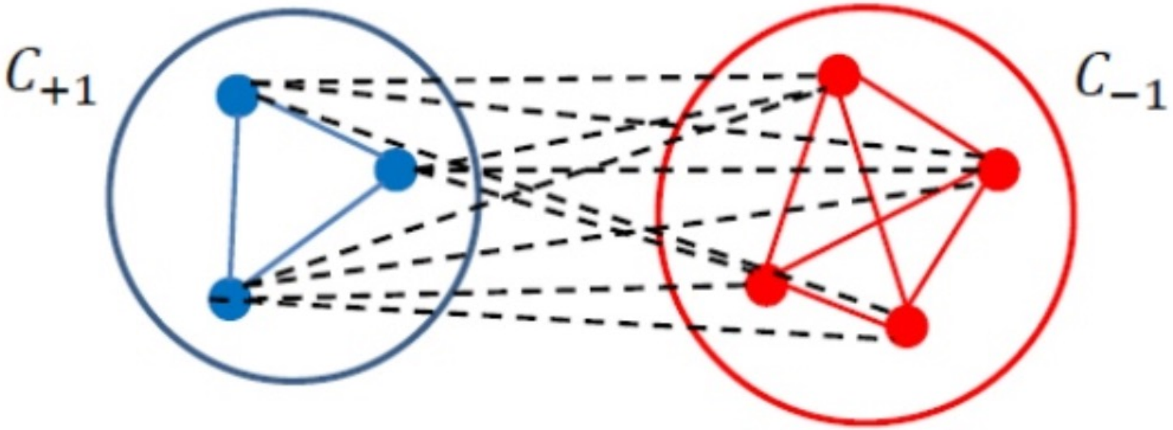
$$\min \sum_{i \in C^+, j \in C^-} w_{ij} = \frac{1}{4} \sum_{i,j} w_{ij} \left(y_i - y_j\right)^2$$

$$y_i \in \{+1, -1\}$$

Cut at outlier

Spectral Clustering: Key Idea

- Partitioning a graph into two clusters
- Normalized Minimum cut



$s(C^+, C^-) = \sum_{i \in C^+, j \in C^-} w_{ij}$

$$\min \frac{s(C^+, C^-)}{s(C^+, C^+) + s(C^+, C^-)} + \frac{s(C^+, C^-)}{s(C^-, C^-) + s(C^+, C^-)}$$

NP-hard

Spectral Clustering: graph Laplacian

$$\Leftrightarrow \frac{1}{4}\sum_{i,j} w_{ij}\left(z_i - z_j\right)^2 = \frac{1}{4}\sum_{i,j} w_{ij}\left(z_i^2 - 2z_i z_j + z_j^2\right)$$

$$= \frac{1}{2}\sum_{i}\left(\sum_{j} w_{ij}\right) z_i^2 - \frac{1}{2}\sum_{i,j} w_{ij}\, z_i z_j = \frac{1}{2} z^T (D - W) z$$

where D is a diagonal matrix and $D_{ii} = \sum_j w_{ij}$

$L = D - W$ is known as the graph Laplacian

$$\min \frac{1}{2} z^T (D - W) z$$
$$s.t.\ z^T D z = 1, z^T D\mathbf{1} = 0$$

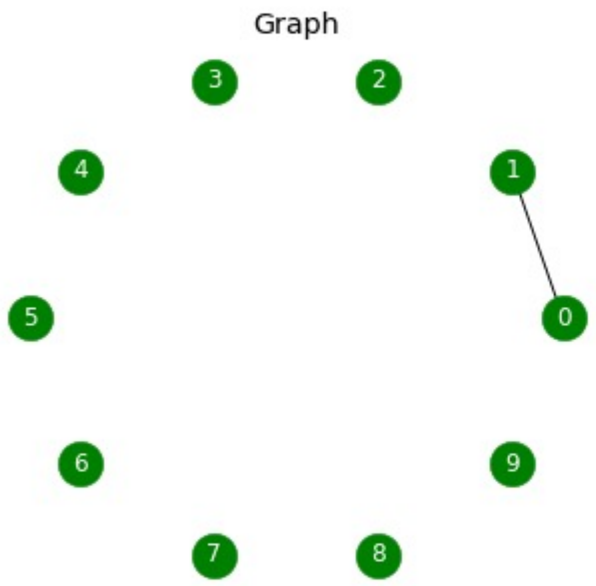
Balanced cut

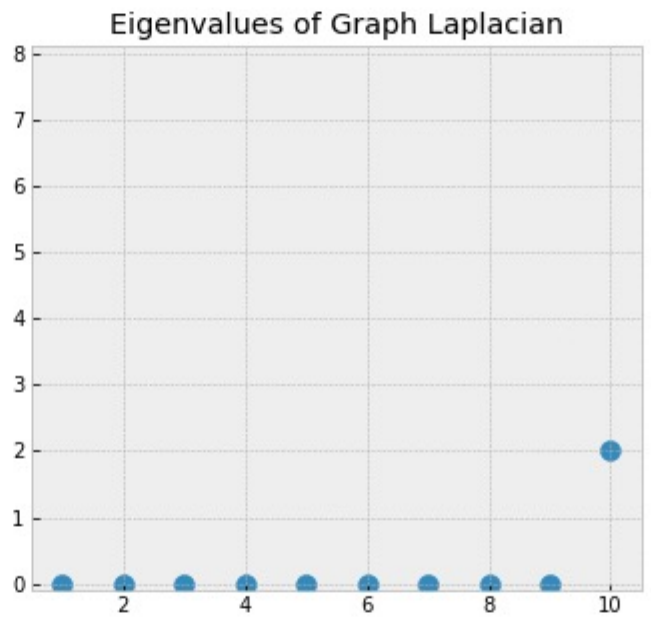
Spectral Clustering: eigenvalue and eigenvector

$$\min \frac{1}{2} z^T (D - W) z$$
$$s.t.\ z^T D z = 1, z^T D \mathbf{1} = 0$$

- Lagrange multiplier (first ignore $z^T D \mathbf{1} = 0$)
- $\boldsymbol{Ax} = \lambda \boldsymbol{x}$
- $(D - W)z = \lambda D z \Leftrightarrow (I - D^{-1} W)z = \lambda z$
- normalized graph Laplacians: $I - D^{-1} W$
- The smallest eigenvalue leads to trivial solution: all data points belong to one cluster.
- **Second smallest eigenvector**
- Final label: $y_i = sign(z_{2i})$

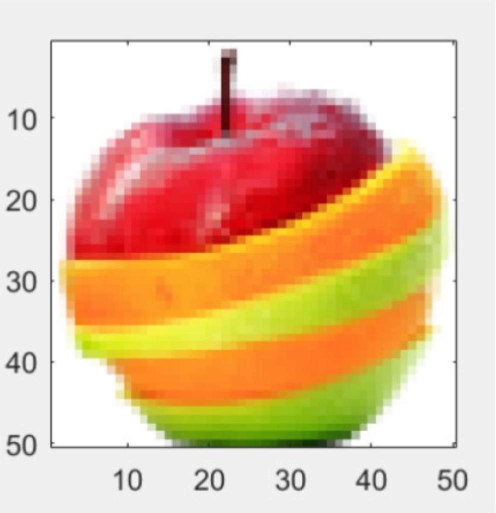
Spectral Clustering: eigenvalue and eigenvector

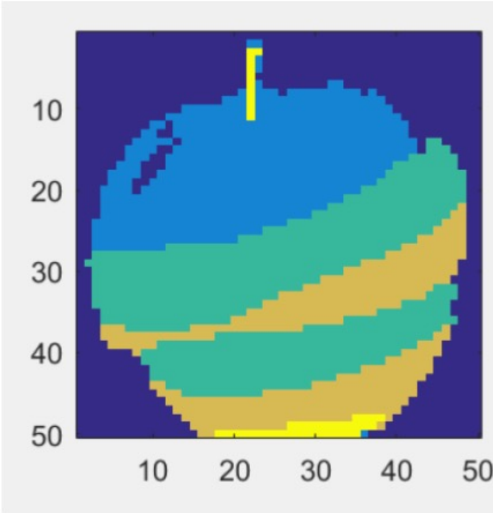




Spectral Clustering

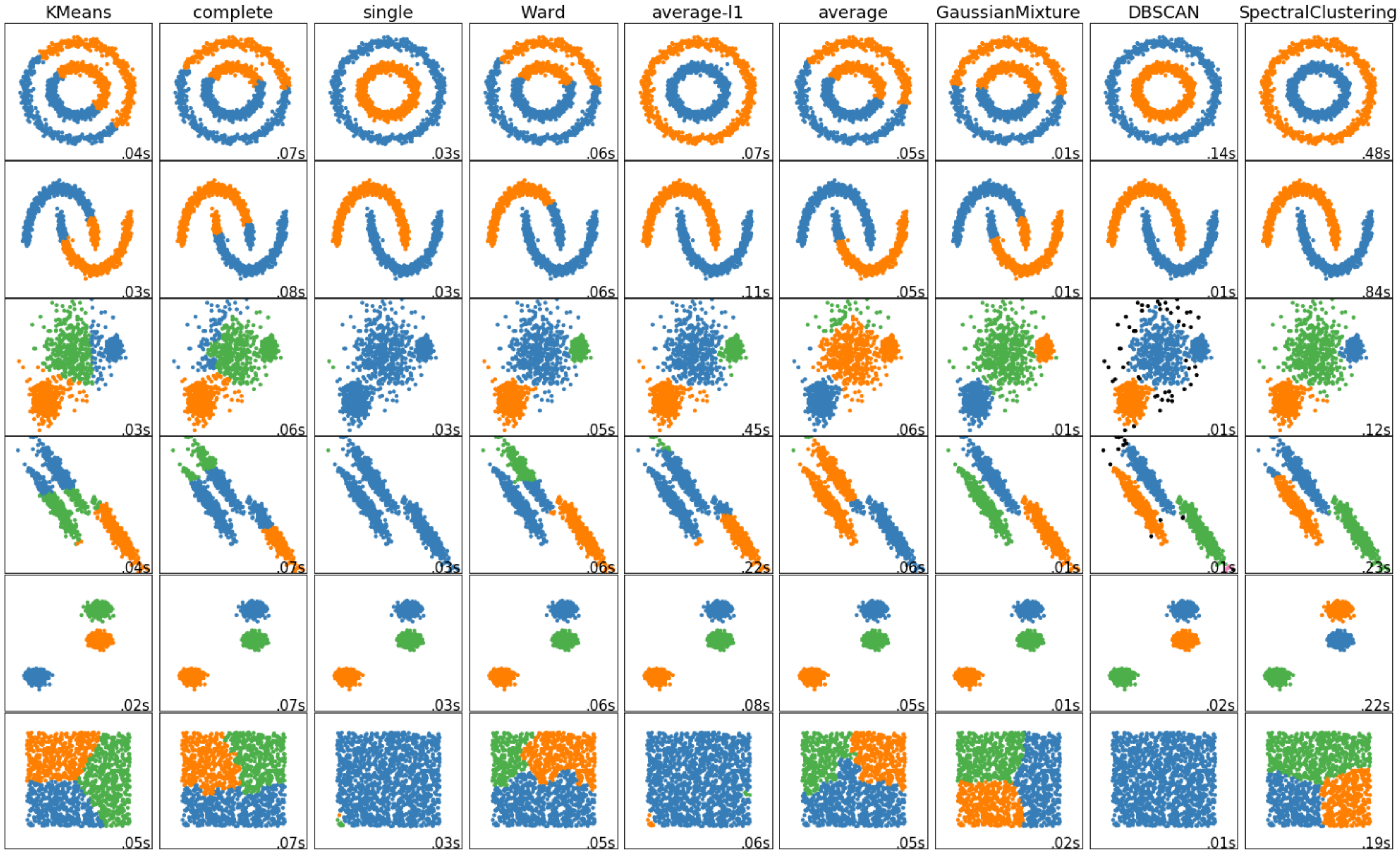
- Solving a standard eigenvalue problem for all eigenvectors takes $O(n^3)$
- No need to get all eigenvector
- Good for arbitrary shape
- For image segmentation





Original image (left) and segmented image using spectral clustering (right)

Some Comparison



Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density-Based Methods
- Evaluation of Clustering
- Summary

Clustering Validation

- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

Clustering Validation and Assessment

- Major issues on clustering validation and assessment
  - **Clustering evaluation**
    - Evaluating the goodness of the clustering
  - **Clustering stability**
    - To understand the sensitivity of the clustering result to various algorithm parameters, e.g., # of clusters
  - **Clustering tendency**
    - Assess the suitability of clustering, i.e., whether the data has any inherent grouping structure

Clustering Validation

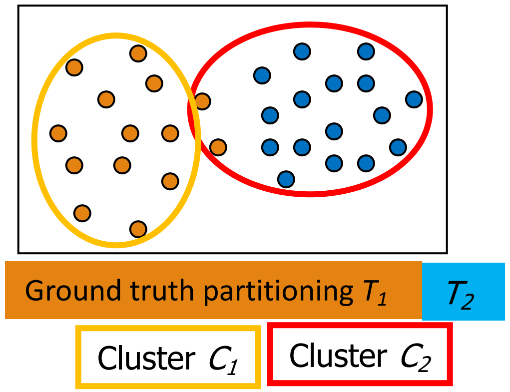
- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

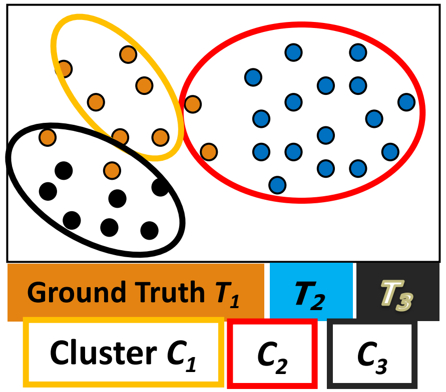
Measuring Clustering Quality

- **Clustering Evaluation**: Evaluating the goodness of clustering results
  - No commonly recognized best suitable measure in practice
- **Three categorization of measures**: External, internal, and relative
  - **External**: Supervised, employ criteria not inherent to the dataset
    - Compare a clustering against prior or expert-specified knowledge (i.e., the ground truth) using certain clustering quality measure
  - **Internal**: Unsupervised, criteria derived from data itself
    - Evaluate the goodness of a clustering by considering how well the clusters are separated and how compact the clusters are, e.g., silhouette coefficient
  - **Relative**: Directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm

Commonly Used External Measures

- **Matching-based measures**
  - Purity, maximum matching, F-measure
- **Entropy-Based Measures**
  - Conditional entropy
  - Normalized mutual information (NMI)
- **Pairwise measures**
  - Four possibilities: True positive (TP), FN, FP, TN
  - Jaccard coefficient, Rand statistic, Fowlkes-Mallow measure





Matching-Based Measures (I): Purity vs. Maximum Matching

C\T	T_1	T_2	T_3	Sum
C_1	0	20	30	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	40	35	100

C\T	T_1	T_2	T_3	Sum
C_1	0	30	20	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	50	25	100

- **Purity**: Quantifies the extent that cluster C_i contains points only from one (ground truth) partition: $$purity_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\}$$
  - Total purity of clustering C: $$purity = \sum_{i=1}^{r} \frac{n_i}{n} purity_i = \frac{1}{n} \sum_{i=1}^{r} \max_{j=1}^{k} \{n_{ij}\}$$
  - Perfect clustering if purity = 1 and $r = k$ (the number of clusters obtained is the same as that in the ground truth)
  - Ex. 1 (green or orange): $purity_1$ = 30/50; $purity_2$ = 20/25; $purity_3$ = 25/25; $purity$ = (30 + 20 + 25)/100 = 0.75
  - Two clusters may share the same majority partition

Problem?
High purity is easy to achieve when the number of clusters is large - in particular, purity is 1 if each document gets its own cluster.

Matching-Based Measures (I): Purity vs. Maximum Matching

$C \backslash T$	T_1	T_2	T_3	Sum
C_1	0	20	30	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	40	35	100

$C \backslash T$	T_1	T_2	T_3	Sum
C_1	0	30	20	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	50	25	100

- **Maximum matching**: Only one cluster can match one partition
  - Match: Pairwise matching, weight $w(e_{ij}) = n_{ij}$ $$\mathrm{w(M)} = \sum_{e \in M} w(e)$$
  - Maximum weight matching: $$match = \arg\max_{M} \{\frac{w(M)}{n}\}$$
  - Ex2. (green) *match* = *purity* = 0.75; (orange) *match* = 0.65 > 0.6

Matching-Based Measures (II): F-Measure

- **Precision**: The fraction of points in C_i from the majority partition T_{j_i} (i.e., the same as purity), where j_i is the partition that contains the maximum # of points from C_i

$$prec_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\} = \frac{n_{ij_i}}{n_i}$$

  - Ex. For the green table
    - $prec_1$ = 30/50; $prec_2$ = 20/25; $prec_3$ = 25/25
- **Recall**: The fraction of point in partition T_{j_i} shared in common with cluster C_i, where $m_{j_i} = |T_{j_i}|$

$$recall_i = \frac{n_{ij_i}}{|T_{j_i}|} = \frac{n_{ij_i}}{m_{j_i}}$$

  - Ex. For the green table
    - $recall_1$ = 30/35; $recall_2$ = 20/40; $recall_3$ = 25/25
- **F-measure** for C_i: The harmonic means of $prec_i$ and $recall_i$: $F_i = \dfrac{2n_{ij_i}}{n_i + m_{j_i}}$
- F-measure for clustering C: average of all clusters: $F = \dfrac{1}{r}\sum_{i=1}^{r} F_i$
  - Ex. For the green table
    - F_1 = 60/85; F_2 = 40/65; F_3 = 1; F = 0.774

C\T	T_1	T_2	T_3	Sum
C_1	0	20	30	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	40	35	100

Matching-Based Measures (II): F-Measure

- **Precision**: The fraction of points in C_i from the majority partition T_{j_i} (i.e., the same as purity), where j_i is the partition that contains the maximum # of points from C_i

$$prec_i = \frac{1}{n_i} \max_{j=1}^{k} \{n_{ij}\} = \frac{n_{ij_i}}{n_i}$$

  - Ex. For the orange table
    - $prec_1$ = 30/50; $prec_2$ = 20/25; $prec_3$ = 25/25
- **Recall**: The fraction of point in partition T_{j_i} shared in common with cluster C_i, where $m_{j_i} = |T_{j_i}|$

$$recall_i = \frac{n_{ij_i}}{|T_{j_i}|} = \frac{n_{ij_i}}{m_{j_i}}$$

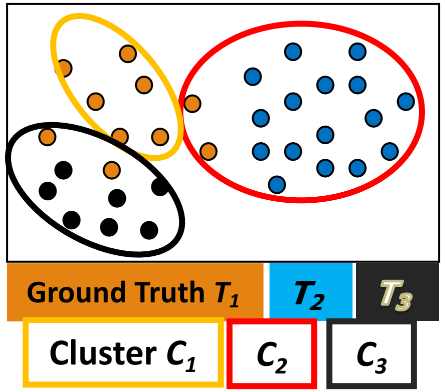
  - Ex. For the orange table
    - $recall_1$ = 30/50; $recall_2$ = 20/50; $recall_3$ = 25/25

C\T	T_1	T_2	T_3	Sum
C_1	0	30	20	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	50	25	100

- **F-measure** for C_i: The harmonic means of $prec_i$ and $recall_i$: $F_i = \frac{2n_{ij_i}}{n_i + m_{j_i}}$
- F-measure for clustering C: average of all clusters: $F = \frac{1}{r}\sum_{i=1}^{r} F_i$
  - Ex. For the green table
    - F_1 = 60/100; F_2 = 40/75; F_3 = 1; F=0.711

Entropy-Based Measures (I): Conditional Entropy

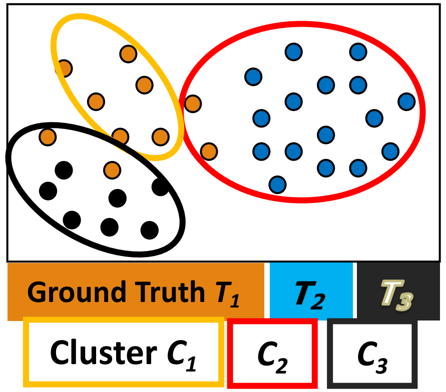
- Entropy of clustering C: $$H(\mathcal{C}) = -\sum_{i=1}^{r} p_{C_i} \log p_{C_i}$$ $p_{C_i} = \frac{n_i}{n}$ (i.e., the probability of cluster C_i)
- Entropy of partitioning T: $$H(\mathcal{T}) = -\sum_{j=1}^{k} p_{T_j} \log p_{T_j}$$
- Entropy of T with respect to cluster C_i: $$H(\mathcal{T}|C_i) = -\sum_{j=1}^{k} \left(\frac{n_{ij}}{n_i}\right) \log\left(\frac{n_{ij}}{n_i}\right)$$
- Conditional entropy of T with respect to clustering C: $$H(\mathcal{T}|\mathcal{C}) = -\sum_{i=1}^{r} \left(\frac{n_i}{n}\right) H(\mathcal{T}|C_i) = -\sum_{i=1}^{r}\sum_{j=1}^{k} p_{ij} \log\left(\frac{p_{ij}}{p_{C_i}}\right)$$
  - The more a cluster's members are split into different partitions, the higher the conditional entropy
  - For a perfect clustering, the conditional entropy value is 0



$$H(\mathcal{T}|\mathcal{C}) = -\sum_{i=1}^{r}\sum_{j=1}^{k} p_{ij}(\log p_{ij} - \log p_{C_i}) = -\sum_{i=1}^{r}\sum_{j=1}^{k} p_{ij}\log p_{ij} + \sum_{i=1}^{r}(\log p_{C_i} \sum_{j=1}^{k} p_{ij})$$
$$= -\sum_{i=1}^{r}\sum_{j=1}^{k} p_{ij}\log p_{ij} + \sum_{i=1}^{r}(p_{C_i}\log p_{C_i}) = H(\mathcal{C},\mathcal{T}) - H(\mathcal{C})$$

Entropy-Based Measures (II): Normalized Mutual Information (NMI)

- **Mutual information**:
  - Quantifies the amount of shared info between the clustering C and partitioning T $$I(C,T) = \sum_{i=1}^{r}\sum_{j=1}^{k} p_{ij} \log\left(\frac{p_{ij}}{p_{C_i} \cdot p_{T_j}}\right)$$
  - Measures the dependency between the observed joint probability p_{ij} of C and T, and the expected joint probability $p_{Ci} \cdot p_{Tj}$ under the independence assumption
  - When C and T are independent, $p_{ij} = p_{Ci} \cdot p_{Tj}$, $I(C, T) = 0$.



- **Normalized mutual information** (NMI)

$$NMI(C,T) = \sqrt{\frac{I(C,T)}{H(C)} \cdot \frac{I(C,T)}{H(T)}} = \frac{I(C,T)}{\sqrt{H(C) \cdot H(T)}}$$

  - Value range of NMI: [0,1]. Value close to 1 indicates a good clustering

Pairwise Measures

Data points	Output clustering	Ground truth (class)
A	1	2
B	1	2
C	2	2
D	2	1

- # pairs of data points: 6
  - (a, b): same class, same cluster
  - (a, c): same class, different cluster
  - (a, d): different class, different cluster
  - (b, c): same class, different cluster
  - (b, d): different class, different cluster
  - (c, d): different class, same cluster

TP = 1
FP = 1
FN = 2
TN = 2

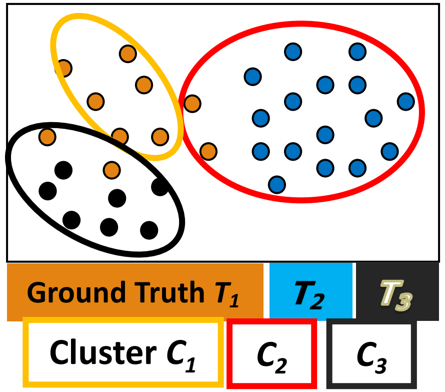
RI = 0.5
Precision= ½, Recall= 1/3
F = 0.4

Pairwise Measures: Four Possibilities for Truth Assignment

- **Four possibilities** based on the agreement between cluster label and partition label
  - *TP*: true positive—Two points $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same partition T, and they also in the same cluster C

$$TP = |\{(\mathbf{x}_i, \mathbf{x}_j) : y_i = y_j \text{ and } \hat{y}_i = \hat{y}_j\}|$$

where y_i: the true partition label , and $\hat{y}_i$: the cluster label for point $\mathbf{x}_i$



  - *FN*: false negative: $FN = |\{(\mathbf{x}_i, \mathbf{x}_j) : y_i = y_j \text{ and } \hat{y}_i \neq \hat{y}_j\}|$
  - *FP: false positive* $FP = |\{(\mathbf{x}_i, \mathbf{x}_j) : y_i \neq y_j \text{ and } \hat{y}_i = \hat{y}_j\}|$
  - *TN*: true negative $TN = |\{(\mathbf{x}_i, \mathbf{x}_j) : y_i \neq y_j \text{ and } \hat{y}_i \neq \hat{y}_j\}|$
- Calculate the four measures:

$$N = \binom{n}{2}$$

Total # of pairs of points

$$TP = \sum_{i=1}^{r}\sum_{j=1}^{k}\binom{n_{ij}}{2} = \frac{1}{2}\left(\left(\sum_{i=1}^{r}\sum_{j=1}^{k} n_{ij}^{2}\right) - n\right) \quad FN = \sum_{j=1}^{k}\binom{m_j}{2} - TP$$

$$FP = \sum_{i=1}^{r}\binom{n_i}{2} - TP \quad TN = N - (TP + FN + FP) = \frac{1}{2}\left(n^2 - \sum_{i=1}^{r} n_i^{2} - \sum_{j=1}^{k} m_j^{2} + \sum_{i=1}^{r}\sum_{j=1}^{k} n_{ij}^{2}\right)$$

Pairwise Measures: Jaccard Coefficient and Rand Statistic

- **Jaccard coefficient:** Fraction of true positive point pairs, but after ignoring the true negatives (thus asymmetric)
  - *Jaccard* = *TP*/(*TP* + *FN* + *FP*) [i.e., denominator ignores *TN*]
  - Perfect clustering: *Jaccard* = 1
- **Rand Statistic**:
  - *Rand* = (*TP* + *TN*)/*N*
  - Symmetric; perfect clustering: *Rand* = 1
- **Fowlkes-Mallow Measure**:
  - Geometric mean of precision and recall

$$FM = \sqrt{prec \times recall} = \frac{TP}{\sqrt{(TP+FN)(TP+FP)}}$$

- Using the above formulas, one can calculate all the measures for the green and orange table (leave as an exercise)

C\T	T_1	T_2	T_3	Sum
C_1	0	30	20	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	50	25	100

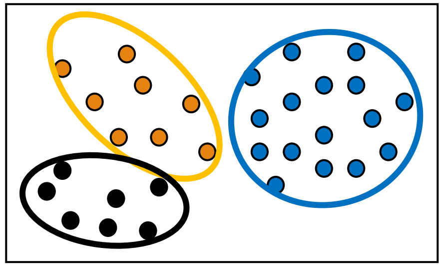
C\T	T_1	T_2	T_3	Sum
C_1	0	20	30	50
C_2	0	20	5	25
C_3	25	0	0	25
m_j	25	40	35	100

Clustering Validation

- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

Internal Measures (I): BetaCV Measure

- A trade-off in maximizing intra-cluster compactness and inter-cluster separation
- Given a clustering $C = \{C_1, . . ., C_k\}$ with k clusters, cluster C_i containing $n_i = |C_i|$ points
  - Let $W(S, R)$ be sum of weights on all edges with one vertex in S and the other in R
  - The sum of all the intra-cluster weights over all clusters: $W_{in} = \frac{1}{2}\sum_{i=1}^{k} W(C_i, C_i)$
  - The sum of all the inter-cluster weights: $W_{out} = \frac{1}{2}\sum_{i=1}^{k} W(C_i, \overline{C_i}) = \sum_{i=1}^{k-1}\sum_{j>i} W(C_i, C_j)$
  - The number of distinct intra-cluster edges: $N_{in} = \sum_{i=1}^{k} \binom{n_i}{2}$
  - The number of distinct inter-cluster edges: $N_{out} = \sum_{i=1}^{k-1}\sum_{j=i+1}^{k} n_i n_j$



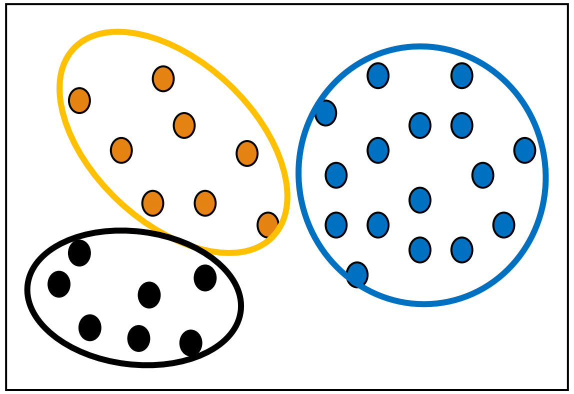
- **Beta-CV measure**: $BetaCV = \dfrac{W_{in} / N_{in}}{W_{out} / N_{out}}$
  - The ratio of the mean intra-cluster distance to the mean inter-cluster distance
  - The smaller, the better the clustering

Internal Measures (II): Normalized Cut

- **Normalized cut:** $NC = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i})}{vol(C_i)} = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i})}{W(C_i, \mathrm{V})} = \sum_{i=1}^{k} \frac{W(C_i, \overline{C_i})}{W(C_i, C_i) + W(C_i, \overline{C_i})} = \sum_{i=1}^{k} \frac{1}{\frac{W(C_i, C_i)}{W(C_i, \overline{C_i})} + 1}$

  where $vol(C_i) = W(C_i, V)$ is the volume of cluster C_i

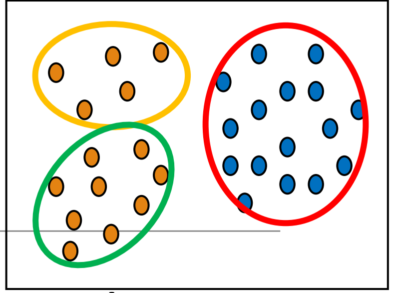
  - The lower normalized cut value, the better the clustering



Clustering Validation

- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

Relative Measure



- Relative measure: Directly compare different clusterings, usually those obtained via different parameter settings for the same algorithm
- **Silhouette coefficient** as an **internal measure**: Check cluster cohesion and separation
  - For each point $\boldsymbol{x}_i$, its silhouette coefficient s_i is: $s_i = \dfrac{\mu_{out}^{\min}(\mathbf{x}_i) - \mu_{in}(\mathbf{x}_i)}{\max\{\mu_{out}^{\min}(\mathbf{x}_i), \mu_{in}(\mathbf{x}_i)\}}$

    where $\mu_{in}(\mathbf{x}_i)$ is the mean distance from $\boldsymbol{x}_i$ to points in its own cluster

    $\mu_{out}^{\min}(\mathbf{x}_i)$ is the mean distance from $\boldsymbol{x}_i$ to points in its closest cluster
  - Silhouette coefficient (SC) is the mean values of s_i across all the points: $SC = \dfrac{1}{n}\sum_{i=1}^{n} s_i$
  - SC close to +1 implies good clustering
    - Points are close to their own clusters but far from other clusters
- **Silhouette coefficient** as a **relative measure**: Estimate the # of clusters in the data

$$SC_i = \frac{1}{n_i}\sum_{x_j \in C_i} s_j$$

Pick the k value that yields the best clustering, i.e., yielding high values for SC and SC_i ($1 \le i \le k$)

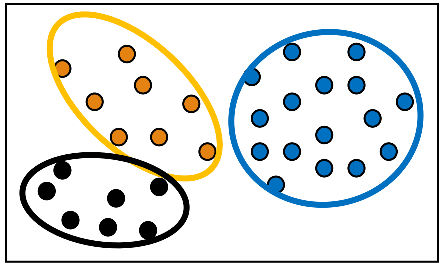
Silhouette Coefficient

- Advantages
  - The score is bounded between -1 for incorrect clustering and +1 for highly dense clustering. Scores around zero indicate overlapping clusters.
  - The score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster.
- Drawbacks
  - The Silhouette Coefficient is generally higher for convex clusters than other concepts of clusters, such as density based clusters like those obtained through DBSCAN.

Clustering Validation

- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

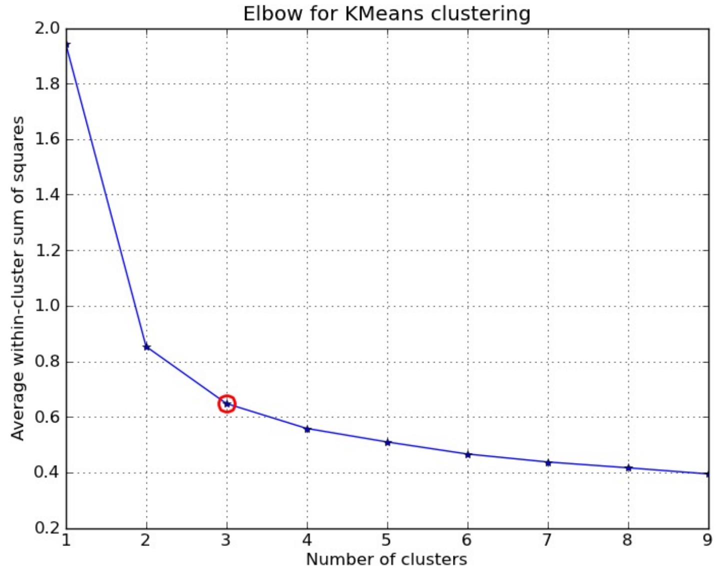
Cluster Stability



- Clusters obtained from several datasets sampled from the same underlying distribution as ***D*** should be similar or "stable"
- Typical approach:
  - Find good parameter values for a given clustering algorithm
- Example: Find a good value of k, the correct number of clusters
- A **bootstrapping approach** to find the best value of k (judged on stability)
  - Generate t samples of size n by sampling from ***D*** with replacement
  - For each sample $\boldsymbol{D}_i$, run the same clustering algorithm with k values from 2 to k_{max}
  - Compare the distance between all pairs of clusterings $C_k(\boldsymbol{D}_i)$ and $C_k(\boldsymbol{D}_j)$ via some distance function
    - Compute the expected pairwise distance for each value of k
  - The value k^* that exhibits the least deviation between the clusters obtained from the resampled datasets is the best choice for k since it exhibits the most stability

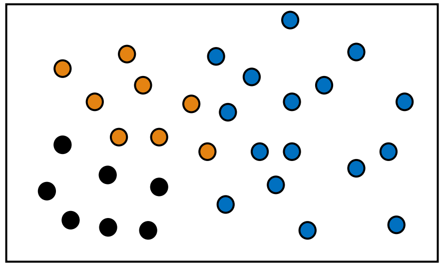
Other Methods for Finding K, the Number of Clusters

- **Empirical method**
  - # of clusters: $k \approx \sqrt{n/2}$ for a dataset of n points (e.g., n = 200, k = 10)
- **Elbow method**: Use the turning point in the curve of the sum of within cluster variance with respect to the # of clusters



- **Cross validation method**
  - Divide a given data set into m parts
  - Use m – 1 parts to obtain a clustering model
  - Use the remaining part to test the quality of the clustering
    - For example, for each point in the test set, find the closest centroid, and use the sum of squared distance between all points in the test set and the closest centroids to measure how well the model fits the test set
  - For any k > 0, repeat it m times, compare the overall quality measure w.r.t. different k's, and find # of clusters that fits the data the best

Clustering Tendency: Whether the Data Contains Inherent Grouping Structure



- Assessing the **suitability of clustering**
  - (i.e., whether the data has any inherent grouping structure)
- Determining ***clustering tendency*** or ***clusterability***
  - **A hard task** because there are so many different definitions of clusters
    - E.g., partitioning, hierarchical, density-based, graph-based, etc.
  - Even fixing cluster type, still hard to define an appropriate null model for a data set
- Still, there are some **clusterability assessment methods**, such as
  - **Spatial histogram**: Contrast the histogram of the data with that generated from random samples

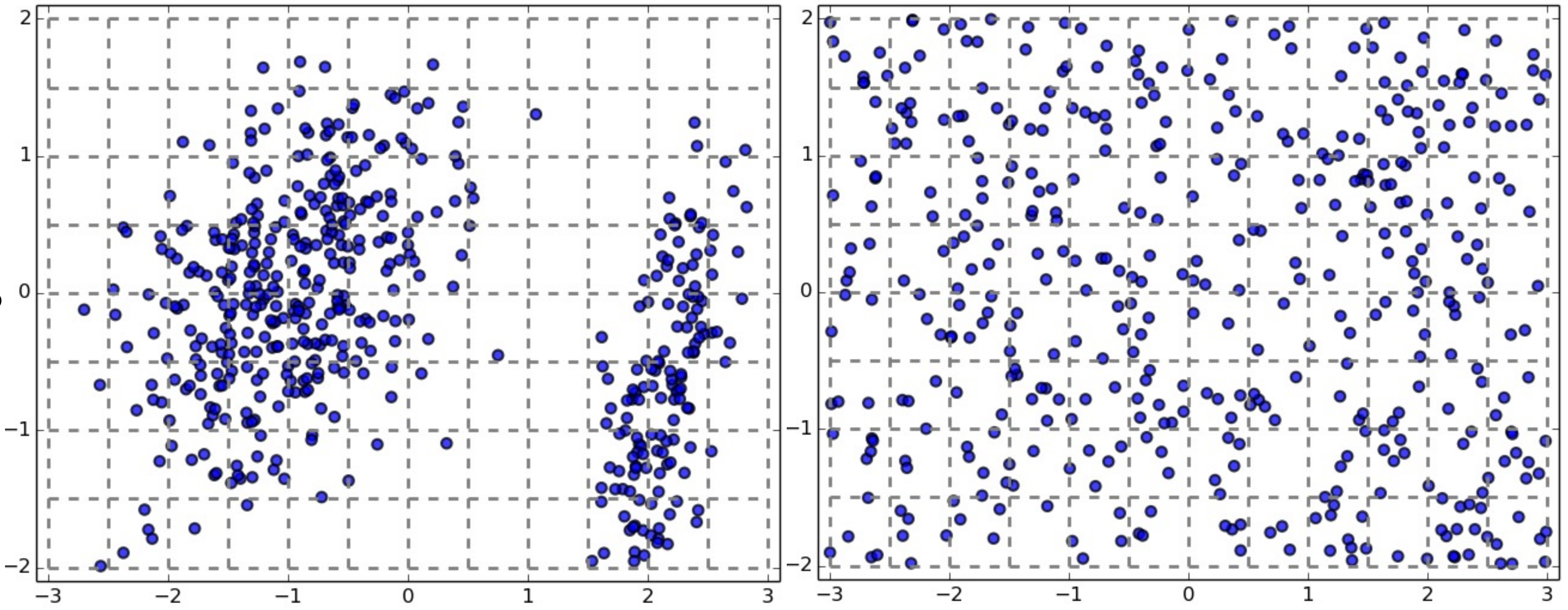
    To be covered here
  - **Distance distribution**: Compare the pairwise point distance from the data with those from the randomly generated samples
  - **Hopkins Statistic**: A sparse sampling test for spatial randomness

Clustering Validation

- Clustering Validation: Basic Concepts
- External Measures for Clustering Validation
  - I: Matching-Based Measures
  - II: Entropy-Based Measures
  - III: Pairwise Measures
- Internal Measures for Clustering Validation
- Relative Measures
- Cluster Stability
- Clustering Tendency

Testing Clustering Tendency: A Spatial Histogram Approach

- **Spatial Histogram Approach:** Contrast the *d*-dimensional histogram of the input dataset ***D*** with the histogram generated from random samples
  - Dataset D is clusterable if the distributions of two histograms are rather different
- Method outline
  - Divide each dimension into equi-width bins, count how many points lie in each cells, and obtain the empirical joint probability mass function (EPMF)



  - Do the same for the randomly sampled data
  - Compute how much they differ using the *Kullback-Leibler* (*KL*) *divergence* value

Chapter 10. Cluster Analysis: Basic Concepts and Methods

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Gaussian Mixture Models and E-M algorithm
- Density- and Grid-Based Methods
- Evaluation of Clustering
- Summary

Summary

- Cluster Analysis: An Introduction
- Partitioning Methods
- Hierarchical Methods
- Density- and Grid-Based Methods
- Evaluation of Clustering

References: (I) Cluster Analysis: An Introduction

- Jiawei Han, Micheline Kamber, and Jian Pei. Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd ed. , 2011 (Chapters 10 & 11)
- Charu Aggarwal and Chandran K. Reddy (eds.). Data Clustering: Algorithms and Applications. CRC Press, 2014
- Mohammed J. Zaki and Wagner Meira, Jr.. Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, 2014
- L. Kaufman and P. J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley & Sons, 1990
- Charu Aggarwal. An Introduction to Clustering Analysis. *in* Aggarwal and Reddy (eds.). Data Clustering: Algorithms and Applications (Chapter 1). CRC Press, 2014

References: (II) Partitioning Methods

- J. MacQueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkeley Symp. on Mathematical Statistics and Probability*, 1967
- S. Lloyd. Least Squares Quantization in PCM. *IEEE Trans. on Information Theory*, 28(2), 1982
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988
- R. Ng and J. Han. Efficient and Effective Clustering Method for Spatial Data Mining. VLDB'94
- B. Schölkopf, A. Smola, and K. R. Müller. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural computation*, 10(5):1299–1319, 1998
- I. S. Dhillon, Y. Guan, and B. Kulis. Kernel K-Means: Spectral Clustering and Normalized Cuts. *KDD*’04
- D. Arthur and S. Vassilvitskii. K-means++: The Advantages of Careful Seeding. SODA’07
- C. K. Reddy and B. Vinzamuri. A Survey of Partitional and Hierarchical Clustering Algorithms, in (Chap. 4) Aggarwal and Reddy (eds.), Data Clustering: Algorithms and Applications. CRC Press, 2014

References: (III) Hierarchical Methods

- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988
- L. Kaufman and P. J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley & Sons, 1990
- T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. SIGMOD'96
- S. Guha, R. Rastogi, and K. Shim. Cure: An Efficient Clustering Algorithm for Large Databases. SIGMOD’98
- G. Karypis, E.-H. Han, and V. Kumar. CHAMELEON: A Hierarchical Clustering Algorithm Using Dynamic Modeling. *COMPUTER*, 32(8): 68-75, 1999.
- C. K. Reddy and B. Vinzamuri. A Survey of Partitional and Hierarchical Clustering Algorithms, in (Chap. 4) Aggarwal and Reddy (eds.), Data Clustering: Algorithms and Applications. CRC Press, 2014

References: (IV) Density- and Grid-Based Methods

- M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases. KDD'96
- W. Wang, J. Yang, R. Muntz, STING: A Statistical Information Grid Approach to Spatial Data Mining, VLDB’97
- R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. SIGMOD’98
- A. Hinneburg and D. A. Keim. An Efficient Approach to Clustering in Large Multimedia Databases with Noise. KDD’98
- M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering Points to Identify the Clustering Structure. SIGMOD’99
- M. Ester. Density-Based Clustering. In (Chapter 5) Aggarwal and Reddy (eds.), Data Clustering: Algorithms and Applications . CRC Press. 2014
- W. Cheng, W. Wang, and S. Batista. Grid-based Clustering. In (Chapter 6) Aggarwal and Reddy (eds.), Data Clustering: Algorithms and Applications. CRC Press. 2014

References: (IV) Evaluation of Clustering

- M. J. Zaki and W. Meira, Jr.. Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, 2014
- L. Hubert and P. Arabie. Comparing Partitions. *Journal of Classification*, 2:193–218, 1985
- A. K. Jain and R. C. Dubes. Algorithms for Clustering Data. Printice Hall, 1988
- M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On Clustering Validation Techniques. *Journal of Intelligent Info. Systems*, 17(2-3):107–145, 2001
- J. Han, M. Kamber, and J. Pei. Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd ed. , 2011
- H. Xiong and Z. Li. Clustering Validation Measures. in (Chapter 23) C. Aggarwal and C. K. Reddy (eds.), Data Clustering: Algorithms and Applications. CRC Press, 2014