cse634

DATA MINING

BASICS of CLUSTER ANALYSIS

Chapter 7, 2nd edition

Chapter 10, 3rd edition

Professor Anita Wasilewska
Computer Science Department
Stony Brook University

Introduction to Cluster Analysis

- Introduction
- Clustering Requirements
- Data Representation
- Partitioning Methods
- K-Means Clustering
- K-Medoids Clustering
- Constrained *K-Means* clustering
- *PAM* and *CLARA*

Introduction to Cluster Analysis

- The process of grouping a set of physical or abstract objects into classes of *similar objects* is called clustering

- A **cluster** is a collection of data objects that are *similar to one another* within the **same cluster** and are *dissimilar to the objects in other* ***clusters***

Formal Definition

- Cluster analysis

Statistical method for grouping a set of data objects into **clusters**

A good clustering method produces high quality clusters with high **intraclass** similarity and low **interclass** similarity

- **Cluster:** Collection of data objects

Intra-class similarity: Objects are similar to objects in same cluster

Inter-class dissimilarity: Objects are dissimilar to objects in other clusters

- Clustering is unsupervised classification

Supervised vs. Unsupervised Learning

- Unsupervised learning - clustering
  - The class labels of training data are unknown
  - Given a set of measurements, observations, etc. establish the existence of clusters in the data
- Supervised learning - classification
  - Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  - New data is classified based on the training set

- Clustering is also called **data segmentation** in some applications because clustering partitions large data sets into groups according to their ***similarity***

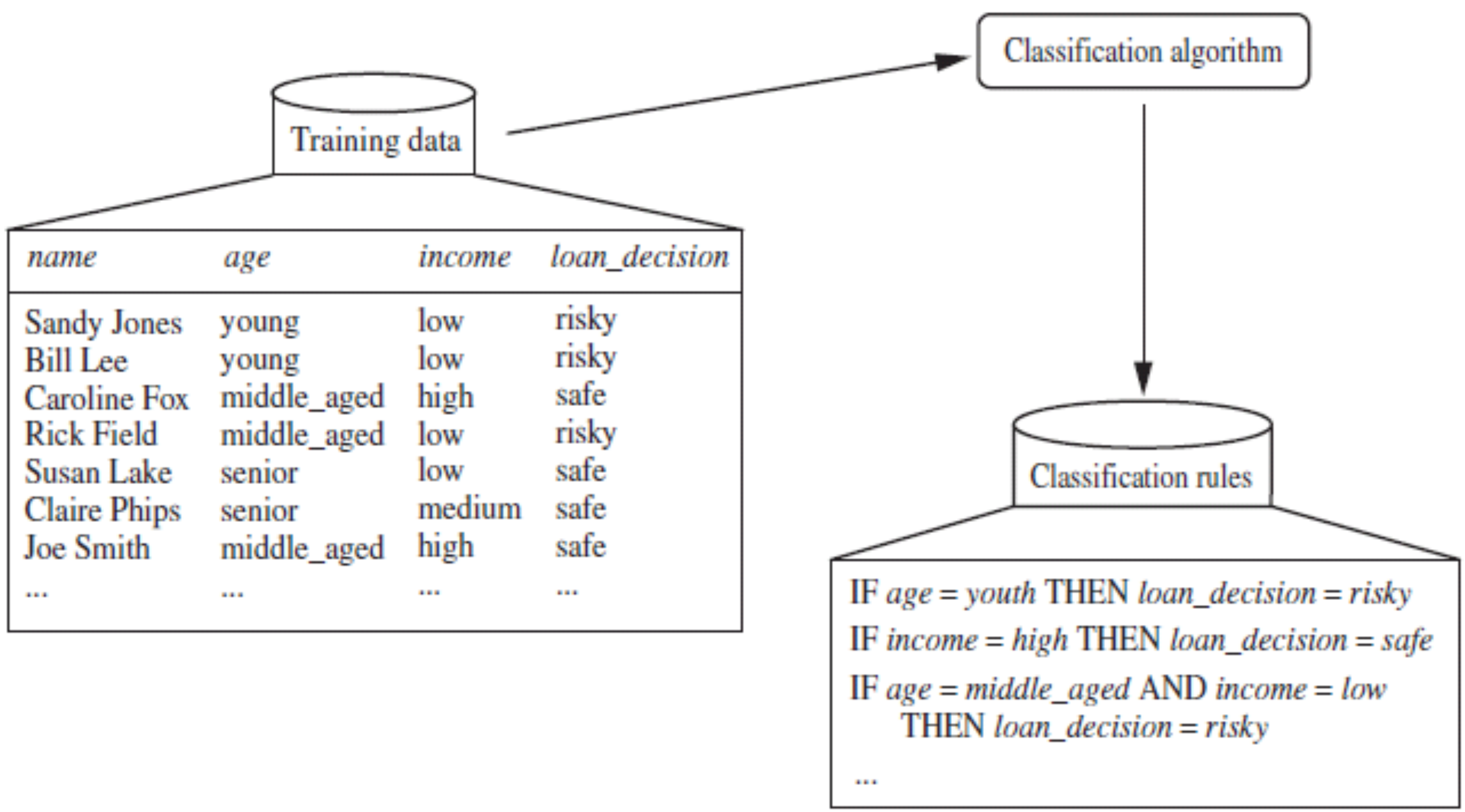
Clustering vs. Classification

- **Clustering - learning by observations**
  - Unsupervised
  - Input
    - Clustering algorithm
    - Similarity measure
    - Number of clusters
  - No specific information for each set of data

- **Classification - learning by examples**
  - Supervised
  - Consists of class labeled training data examples
  - Build a classifier that assigns data objects to one of the classes

Clustering vs. Classification

- Class Label Attribute : *loan_decision*
- *Learning of Classifier is "supervised" → it is told to which class each training tuple (sample) belongs*



Data Mining Concept and Techniques (Chapter 6, Page 287).

Clustering vs. Classification

- **Clustering**
  - class label of training tuple not known
  - number or set of classes to be learned may not be known in advance
  - e.g. if we did not have *loan_decision* data available we use **clustering** and NOT classification to determine "groups of like tuples"
  - These *"groups of like tuples"* may eventually correspond to risk groups within loan application data

Typical Requirements Of Clustering

- **Minimal requirements** for domain knowledge to determine input **parameters**
- Many clustering algorithms require users to input certain **parameters** in cluster analysis (such as the number of desired clusters)
- The clustering results can be quite sensitive to **input parameters**.
- Parameters are often **difficult** to determine, especially for data sets containing high-dimensional **objects**

Typical Requirements Of Clustering

- **Scalability**

  Many clustering algorithms work well on small data sets

  Large database may contain millions of objects

  Clustering on a *sample* of a given large data set may lead to biased results

  Highly scalable clustering algorithms are **needed**

- **Ability** to deal with different types of attributes

  Many algorithms are **designed** to cluster numerical data

  Applications may require clustering other **types** of data: binary, categorical (nominal), and ordinal data, or mixtures of these data types

Typical Requirements Of Clustering

- Ability to deal with noisy data

Some clustering algorithms are sensitive to noisy data and may lead to **clusters** of poor quality

- Incremental clustering and insensitivity to the order of input records
- Constraint-based clustering
- Interpretability and usability

Typical Requirements Of Clustering

- **Discovery** of clusters with **arbitrary shape**
- Many clustering algorithms determine clusters based on Euclidean or Manhattan **distance** measures
- **Algorithms** based on such **distance** measures tend to find spherical clusters with similar size and density
- **A cluster** could be of any shape
- It is important to develop algorithms that can detect clusters of **arbitrary shape**

Examples of Clustering Applications

- **Marketing:**
- Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- **Insurance:**
- Identifying groups of insurance policy holders with a high average claim cost

Examples of Clustering Applications

- **City-planning:**
- Identifying groups of houses according to their house type, value, and geographical location

- **Earth-quake studies:**
- Observe earth quake epicenters clustered along continent faults

- **Fraud detection:**
- Detection of credit card fraud and the monitoring
- of criminal activities in electronic commerce

Data Representation

- Data matrix

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

- n objects with p attributes

- Dissimilarity

  d(i,j) : dissimilarity (similarity)

  distance between records i and j

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix}$$

Data Mining Concept and Techniques (Chapter 7, Page 386-387).

Types of Data in Cluster Analysis

- Interval-Scaled Variables
- (values od attributes)

- Binary Variables (values od attributes)

- Categorical, Ordinal, and Ratio-Scaled
- Variables (values od attributes)

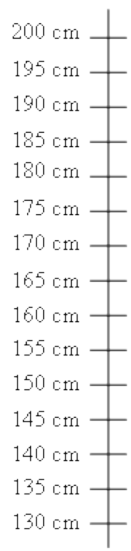
- Variables of Mixed Types

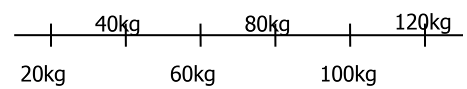
Interval-Scaled Variables

- Continuous measurements of a roughly linear scale
  - E.g. weight, height, temperature, etc.

<table>
<tr><th>Height Scale</th><th>Weight Scale</th></tr>
<tr>
<td>
1. Scale ranges over the metre or foot scale

2. Need to standardize heights as different scale can be used to express same absolute measurement



</td>
<td>


1. Scale ranges over the kilogram or pou nd scale
</td>
</tr>
</table>

Using Interval-Scaled Values

Step 1: Standardize the data

- To ensure they all have equal weight
- To match up different scales into a uniform, single scale
- Not always needed! Sometimes we require unequal weights for an attribute

Step 2:

Compute dissimilarity between records

- Use Euclidean, Manhattan or Minkowski distance

Data Types and Distance Metrics

Distances are normally used to measure the **similarity** or **dissimilarity** between two data objects (records)

- Minkowski **distance:**

$$d(i,j) = \sqrt[q]{(|x_{i_1} - x_{j_1}|^q + |x_{i_2} - x_{j_2}|^q + \ldots + |x_{i_p} - x_{j_p}|^q)}$$

where $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and $j = (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two *p*-dimensional data objects, and q is a positive integer

Data Types and Distance Metrics

- If *q = 1*, Minkowski *d* is Manhattan **distance**

$$d(i,j)=|x_{i_1}-x_{j_1}|+|x_{i_2}-x_{j_2}|+...+|x_{i_p}-x_{j_p}|$$

- *If q = 2*, Minkowski d is Euclidean distance

$$d(i,j)=\sqrt{(|x_{i_1}-x_{j_1}|^2+|x_{i_2}-x_{j_2}|^2+...+|x_{i_p}-x_{j_p}|^2)}$$

Data Types and Distance Metrics

- Distance Properties
  - $d(i,j) \geq 0$
  - $d(i,i) = 0$
  - $d(i,j) = d(j,i)$
  - $d(i,j) \leq d(i,k) + d(k,j)$
- Can also use **weighted distance**, or **other** dissimilarity measures

$$d(i,j) = \sqrt[q]{w_1(|x_{i_1} - x_{j_1}|^q + w_2|x_{i_2} - x_{j_2}|^q + \ldots + w_p|x_{i_p} - x_{j_p}|^q)}$$

Binary Attributes

- A **contingency table** for binary data

		Object j		
		1	0	*sum*
	1	a	b	$a+b$
	0	c	d	$c+d$
Object i	*sum*	$a+c$	$b+d$	p

- **Simple matching** coefficient (applicable only for database with **all symmetric binary** attributes):

$$d(i,j) = \frac{b+c}{a+b+c+d}$$

- **Jaccard coefficient** (applicable only for database with **all asymmetric binary** attributes):

$$d(i,j) = \frac{b+c}{a+b+c} \qquad sim(i,j) = \frac{a}{a+b+c}$$

- For **mixed binary** attributes, please refer Data Mining Concept and Techniques (Chapter 7, Section 7.2.4).

Binary Attributes

- Example:

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

Points to be **considered** (refer Chapter 7 of the book for the above example):

• In this book, gender is assumed as an asymmetric attribute and the **rest** of the attributes are assumed **symmetric**

• The book ignores the gender attribute and **continues** to consider the other attributes

Binary Attributes

- Example:

Name	Gender	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	M	Y	N	P	N	N	N
Mary	F	Y	N	P	N	P	N
Jim	M	Y	P	N	N	N	N

Formulas defined for similarity and dissimilarity are **applicable** only when all attributes under consideration are asymmetric or symmetric

Calculation of similarity and dissimilarity between attributes when a **combination** of asymmetric and symmetric attributes is involved, is explained in section 7.2.4

Dissimilarity between Binary Attributes

- We now consider

Name	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	Y	N	P	N	N	N
Mary	Y	N	P	N	P	N
Jim	Y	P	N	N	N	N

Since the table was a combination of symmetric and asymmetric attributes, we now omit Gender which is a symmetric attribute from our consideration

We are now left with the asymmetric attributes – Fever, Cough, Test-1, Test-2, Test-3, Test-4

Calculating the dissimilarity considering only asymmetric attributes using Jaccard coefficient is as follows

Dissimilarity between Binary Attributes

- Example

Name	Fever	Cough	Test-1	Test-2	Test-3	Test-4
Jack	Y	N	P	N	N	N
Mary	Y	N	P	N	P	N
Jim	Y	P	N	N	N	N

Let the values Y and P be set to 1, and the value N be set to 0

We **calculate** the dissimilarity considering only asymmetric attributes using **Jaccard coefficient** is as follows

$$d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$$

$$d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$$

$$d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$$

Categorical Attributes

- **Categorical attribute** is a generalization of the **binary** attribute in that it can take more than 2 states, e.g., red, yellow, blue, green

- **Method 1:** Simple matching

  m: # of attributes that are the **same** for both records,

  p: total # of attributes

$$d(i,j)=\frac{p-m}{p}$$

- **Method 2:** rewrite the database and **create** a new binary attribute for each of the m states

  For an object with color yellow, the yellow attribute is set to 1, while the remaining attributes are set to 0

Major Clustering Approaches

- **Partitioning approach:**

  Construct various partitions and then **evaluate** them by some criterion, e.g., minimizing the sum of square errors

  Typical methods: **k-means, k-medoids, CLARANS**

- **Hierarchical approach:**

  Create a hierarchical decomposition of the set of data (or objects) using some criterion

  Typical methods: **Diana, Agnes, BIRCH, ROCK, CAMELEON**

Major Clustering Approaches

Density-based approach:

Based on connectivity and density functions

Typical methods: **DBSACN, OPTICS, DenClue**

- **Grid-based approach:**

  based on a multiple-level granularity structure

  Typical methods: **STING, WaveCluster, CLIQUE**

- **Model-based:**

  A model is hypothesized for each of the clusters and tries to find the best fit of that model to each other

  Typical methods: **EM, SOM, COBWEB**

Major Clustering Approaches

- **Frequent pattern-based:**

  Based on the analysis of frequent patterns

  Typical methods: **pCluster**

- **User-guided or constraint-based:**

  Clustering by considering user-specified or application-specific constraints

  Typical methods: **COD** (obstacles), constrained clustering

Methods to Calculate the Distance between Clusters

- **Single link:** smallest distance between an **element** in one cluster and an element in the other,

$$dis(K_i, K_j) = min(t_{ip}, t_{jq})$$

- **Complete link:** largest distance between an **element** in one cluster and an element in the other,

$$dis(K_i, K_j) = max(t_{ip}, t_{jq})$$

- **Average:** avg distance between an **element** in one cluster and an element in the other,

$$dis(K_i, K_j) = avg(t_{ip}, t_{jq})$$

Methods to Calculate the Distance between Clusters

- Centroid:
- distance between the centroids of two clusters,

$$dis(K_i, K_j) = dis(C_i, C_j)$$

- Medoid:
- distance between the medoids of two clusters,

$$dis(K_i, K_j) = dis(M_i, M_j)$$

Medoid is one **chosen**, centrally located object in the **cluster**

Numerical Data: Centroid, Radius, Diameter

- **Centroid:** the "middle" of a cluster for numerical data

$$C_m = \frac{\sum_{i=1}^{N} (t_{ip})}{N}$$

- **Radius:** square root of average distance from any point of the cluster to its centroid

$$R_m = \sqrt{\frac{\sum_{i=1}^{N} (t_{ip} - c_m)^2}{N}}$$

Numerical Data: Centroid, Radius, Diameter

- **Diameter:**
- square root of average mean squared distance between **all pairs** of points in the cluster

$$R_m = \sqrt{\frac{\sum_{i=1}^{N} (t_{ip} - c_m)^2}{N}}$$

Partitioning Algorithms: Basic Concept

- **Partitioning method:**

Construct a partition of a database ***D*** of ***n*** objects (records) into a set of ***k*** clusters

- Given a ***k***, find a partition of ***k*** *clusters* that optimizes the chosen partitioning criterion

  Global optimal method:

  exhaustively enumerate all partitions

Partitioning Algorithms: Basic Concept

- Given a *k*, find a partition of *k clusters* that **optimizes** the chosen partitioning criterion

  Heuristic methods: *k-means* and *k-medoids* algorithms

  k-means (MacQueen' 67):

  Each cluster is represented by the **center** of the cluster

  k-medoids or PAM (Partition around medoids) (Kaufman & Rousseeuw' 87)

    Each cluster is represented by **one** of the objects in the cluster

The *K-Means* Clustering Method

- Given **k**, the ***k-means*** algorithm is **implemented** in four steps:

  1. Partition objects into k nonempty subsets

  2. Compute **seed points** as the **centroids** of the clusters of the current partition (the centroid is the **center**, i.e., *mean point*, of the cluster)

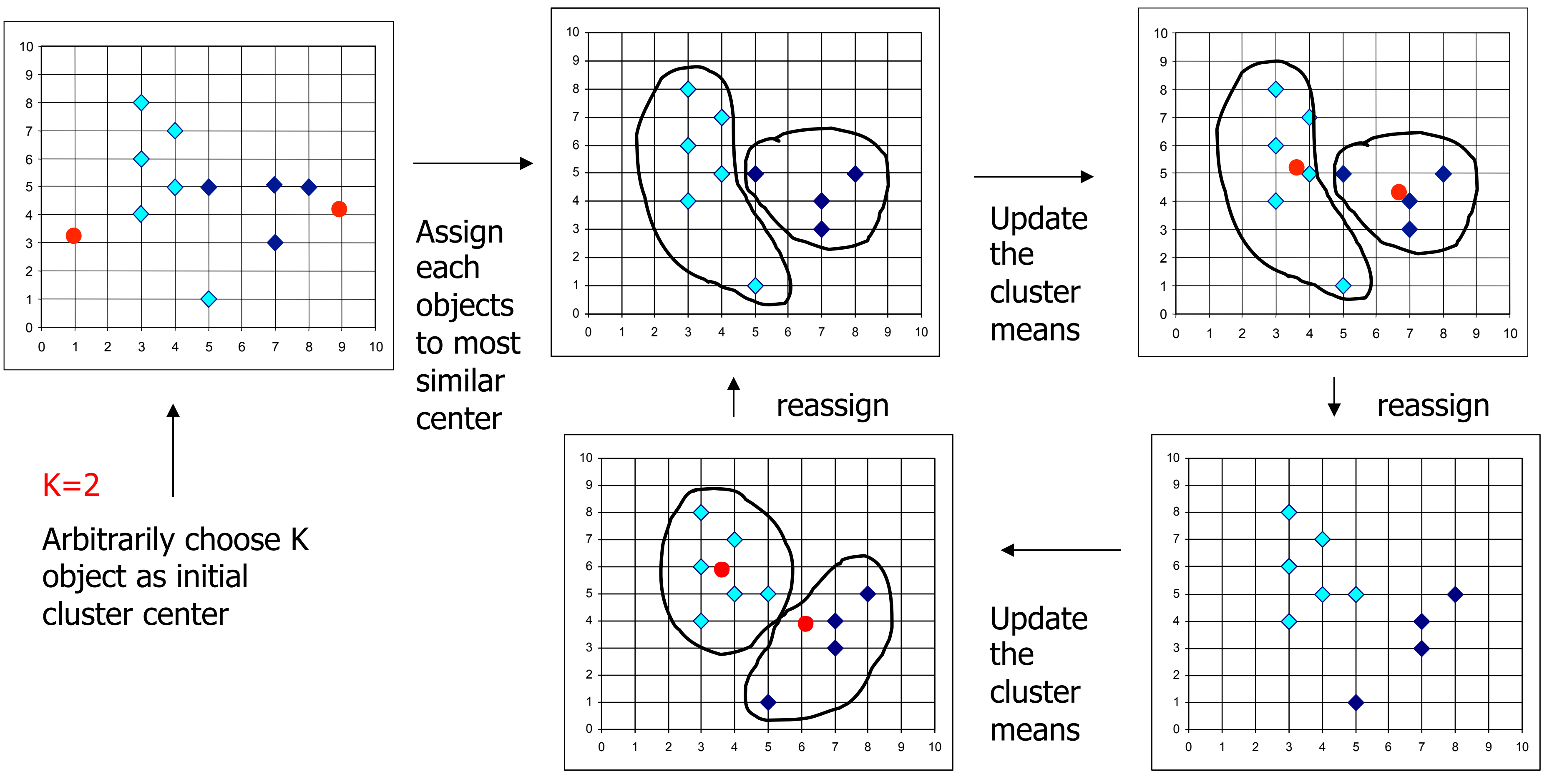
  3. Assign each object to the cluster with the **nearest seed** point

  4. Go back to step **2.**

  STOP when **no more** new assignment

The *K-Means* Clustering Method

Example (Book page 31)



The *k-Means* Algorithm

The basic step of *k-means* clustering is simple:

- Iterate until ***stable,*** i.e. there is no change in the clusters of objects
- Determine the **centroid** coordinate
- Determine the **distance** of each object to the centroids
- Group the object based on minimum **distance**

Comments on the *K-Means* Method

- **Strength:**
- Relatively efficient: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations

  Normally, $k, t << n$

  - Comparing: PAM: $O(k(n-k)^2)$
  - CLARA: $O(ks^2 + k(n-k))$
- **Comment:** Often terminates at a *local optimum*
- The *global optimum* may be **found** using techniques such as: *deterministic annealing* and *genetic algorithms*

Comments on the *K-Means* Method

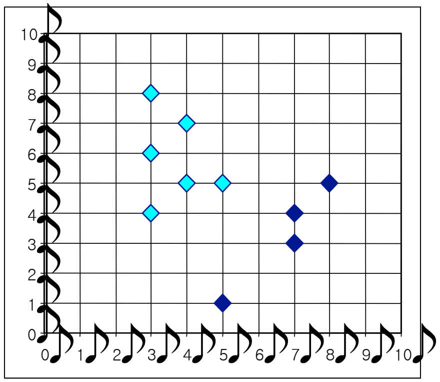
- **Weakness**
  - Applicable only when *mean* is **defined**,
  - then what about categorical data?
  - Need to specify *k*, the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*

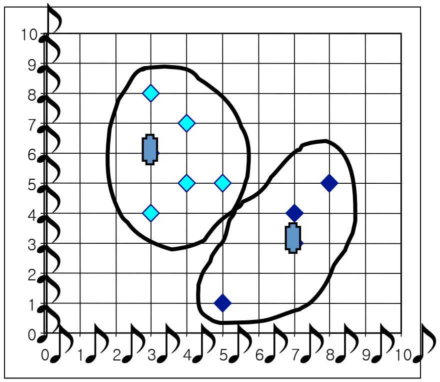
Variations of the *K-Means* Method

- A **few variants** of the *k-means* which **differ** in are
  - Selection of the **initial** *k* means,
  - Dissimilarity calculations
  - Strategies to calculate cluster means

What Is the Problem of the *K-Means* Method?

- The *k-means* algorithm is sensitive to **outliers**
- **K-Medoids:** Instead of taking the **mean** value of the object in a cluster as a reference point, the **medoids**, the **most centrally located** object in a cluster can be used





Variations of the *K-Means* Method

- Handling **categorical** data: ***k-modes*** (Huang' 98)

  Replacing **means** of clusters with **modes**

  Using new dissimilarity **measures** to deal with categorical objects

  Using a **frequency-based** method to update modes of clusters

  A mixture of **categorical** and **numerical** data:

  k-prototype method

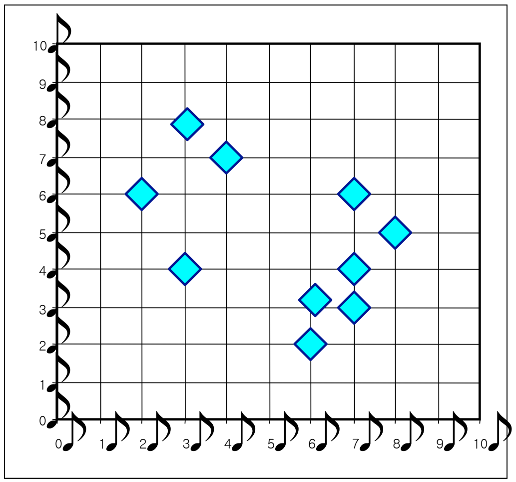
The *K-Medoids* Clustering Method

- **Find** *representative* objects, called medoids, in clusters
- **PAM** (**P**artitioning **A**round **M**edoids, 1987)

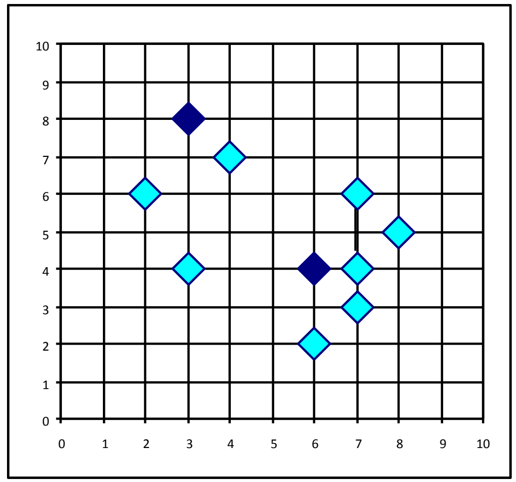
  starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it **improves** the total **distance** of the resulting clustering

  PAM works effectively for small data sets, but **does not** scale well for large data sets
- *CLARA* (Kaufmann & Rousseeuw, 1990), *CLARANS* (Ng & Han, 1994)

A Typical *K-Medoids* Algorithm (PAM)

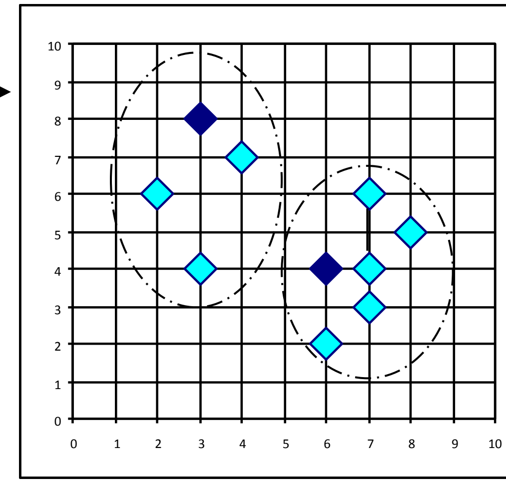


Arbitrary choose k object as initial medoids

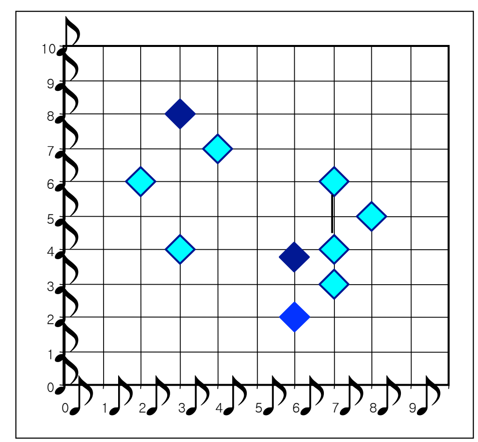


Assign each remaining object to nearest medoids

Total Cost = 20

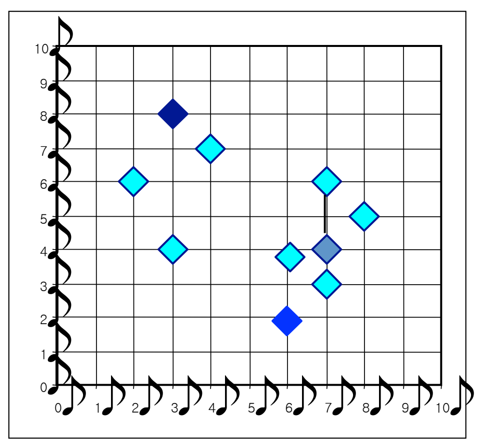


Randomly select a nonmedoid object, O_{ramdom}



Compute total cost of swapping

Total Cost = 26



Swapping O and O_{ramdom}

If quality is improved.

K=2

Do loop

Until no change

Algorithm- K Medoids PAM

Algorithm: ***k-medoids***. PAM, *a k-medoids algorithm for partitioning based on medoid* or central objects.

Input:

k: the number of clusters,

D: a data set containing n objects.

Output:

A set of *k clusters*

Method:

(1) arbitrarily choose *k objects in D as the initial representative objects or seeds;*

(2) **repeat**

(3) assign each remaining object to the cluster with the nearest representative object;

(4) randomly select a non representative object, $\boldsymbol{O_{random}}$***;***

(5) compute the total cost, *S, of swapping representative object,* $\boldsymbol{O_j}$***, with*** $\boldsymbol{O_{random}}$***;***

(6) if *S < 0 then swap* $\boldsymbol{O_j}$ ***with*** $\boldsymbol{O_{random}}$ ***to form the new set of k representative objects;***

(7) **until** no change;

What Is the Problem with PAM?

PAM is more **robus**t than *k-means* in the presence of noise and outliers because a **medoid** is less influenced by **outliers** or other **extreme** values than a *mean*

PAM works *efficiently* for small data sets but does not scale well for large data sets

- $O(k(n-k)^2)$ for each iteration

where n is # of data, k is # of clusters

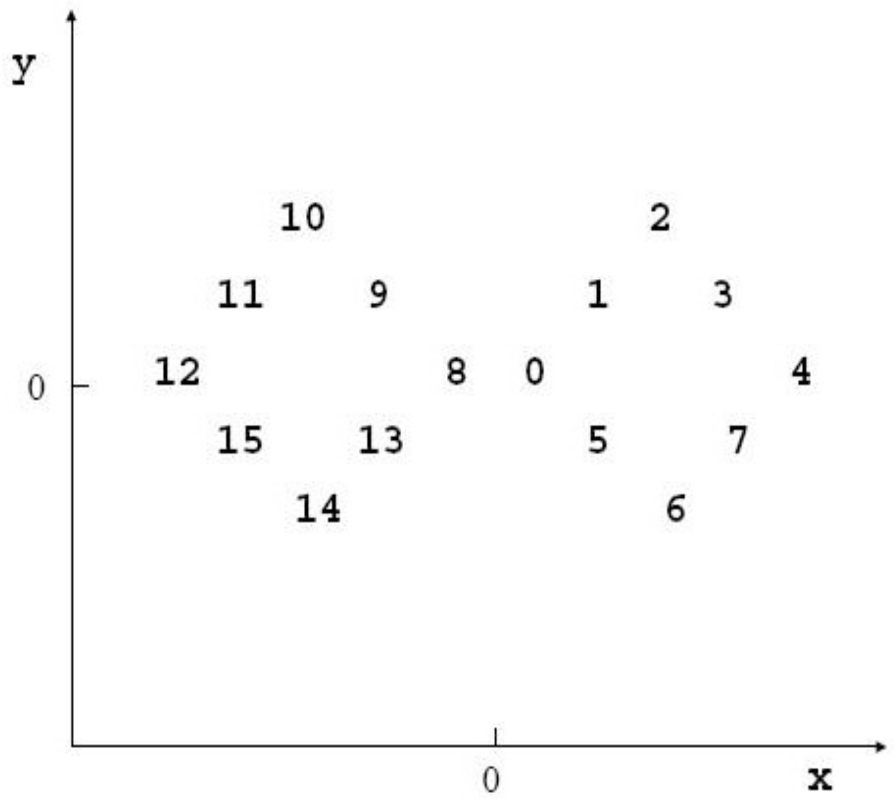
Next Sampling based method:

CLARA (CLustering LARge Applications)

K-Means Clustering Method

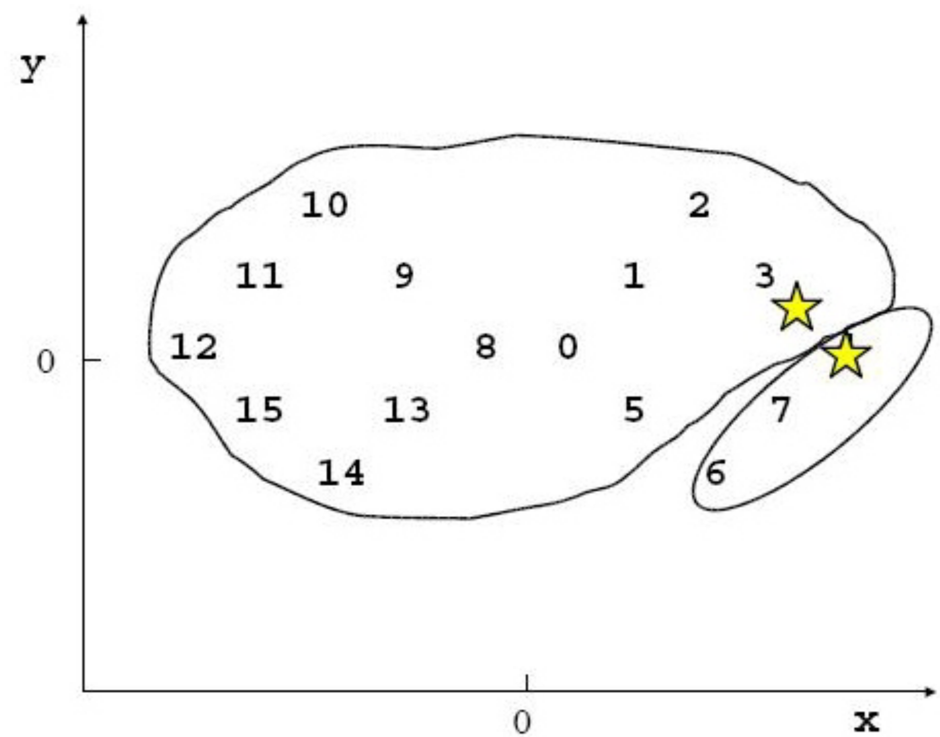
Example (“Maschine Learning and Data Mining” (page 3-11))

Id	x	y
0:	1.0	0.0
1:	3.0	2.0
2:	5.0	4.0
3:	7.0	2.0
4:	9.0	0.0
5:	3.0	-2.0
6:	5.0	-4.0
7:	7.0	-2.0
8:	-1.0	0.0
9:	-3.0	2.0
10:	-5.0	4.0
11:	-7.0	2.0
12:	-9.0	0.0
13:	-3.0	-2.0
14:	-5.0	-4.0
15:	-7.0	-2.0



K-Means Clustering Method

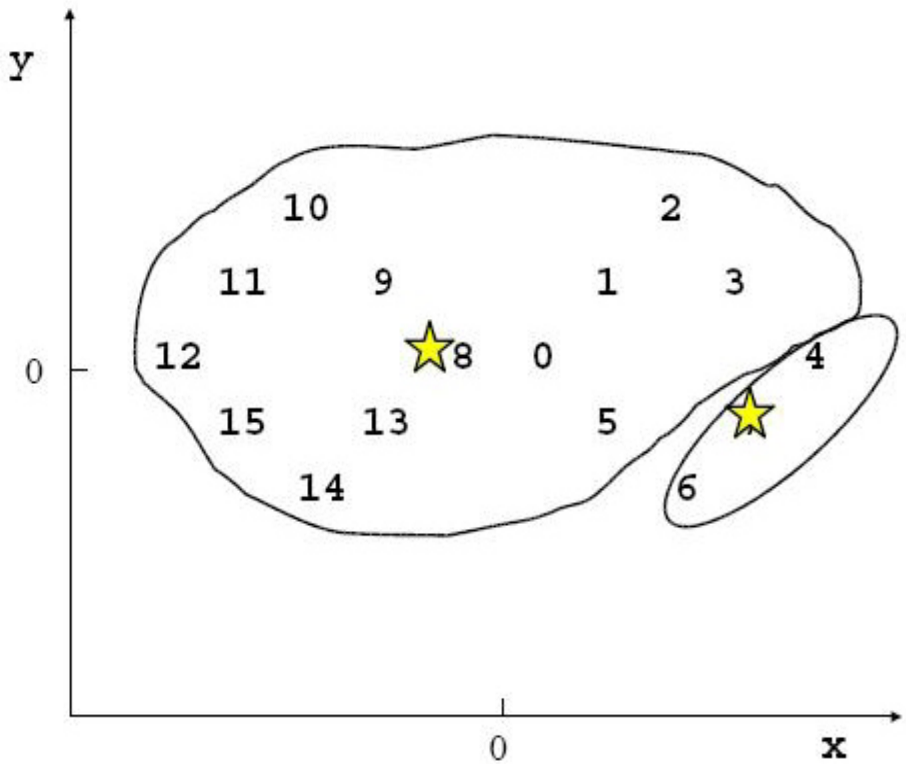
Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887



K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

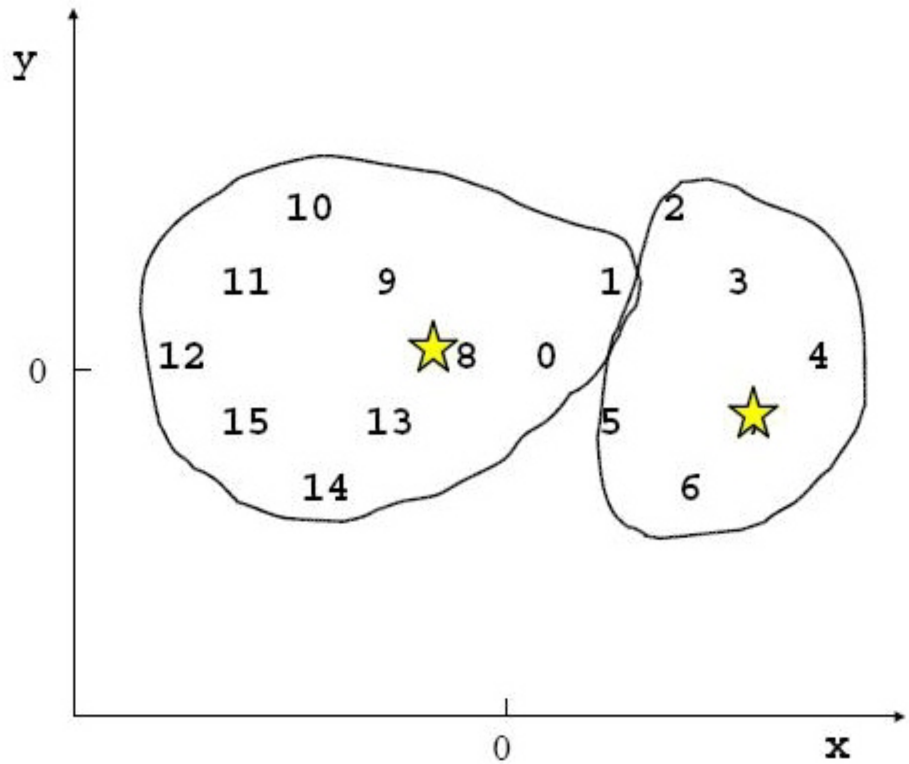
Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)



K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

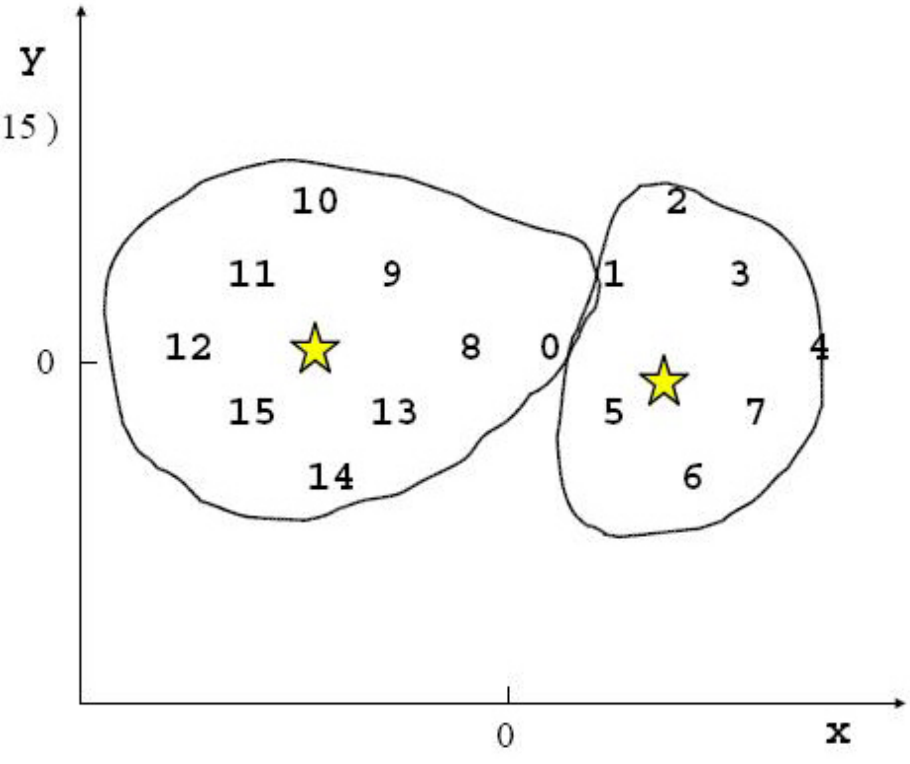


K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)

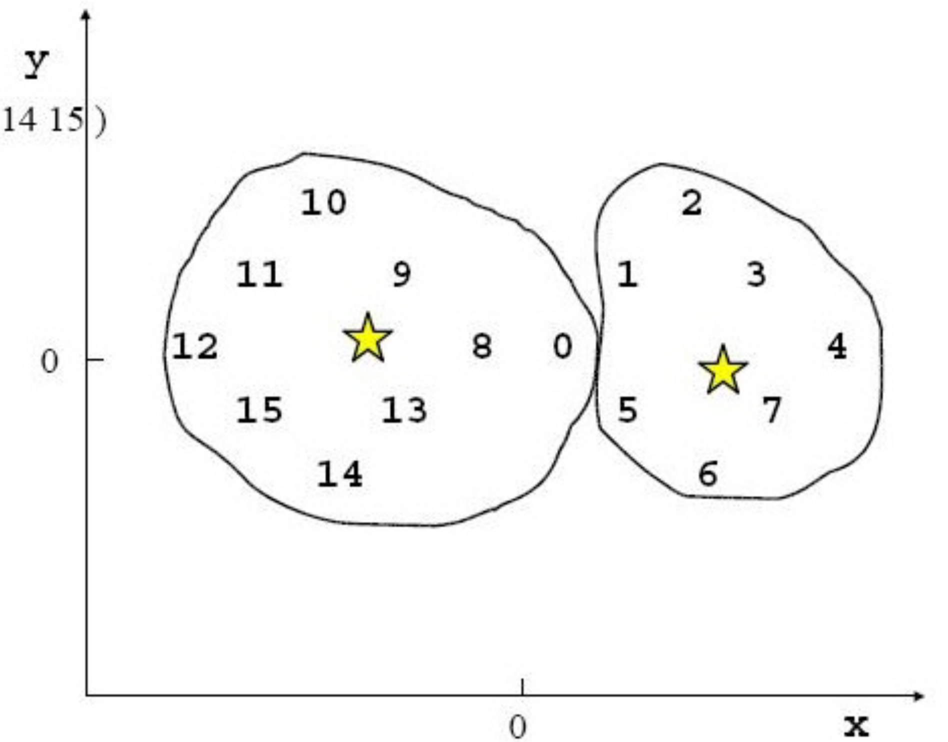


K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)
Cluster Centers: (5.57143 0.0) (-4.33334 0.0)
Average Distance: 3.49115



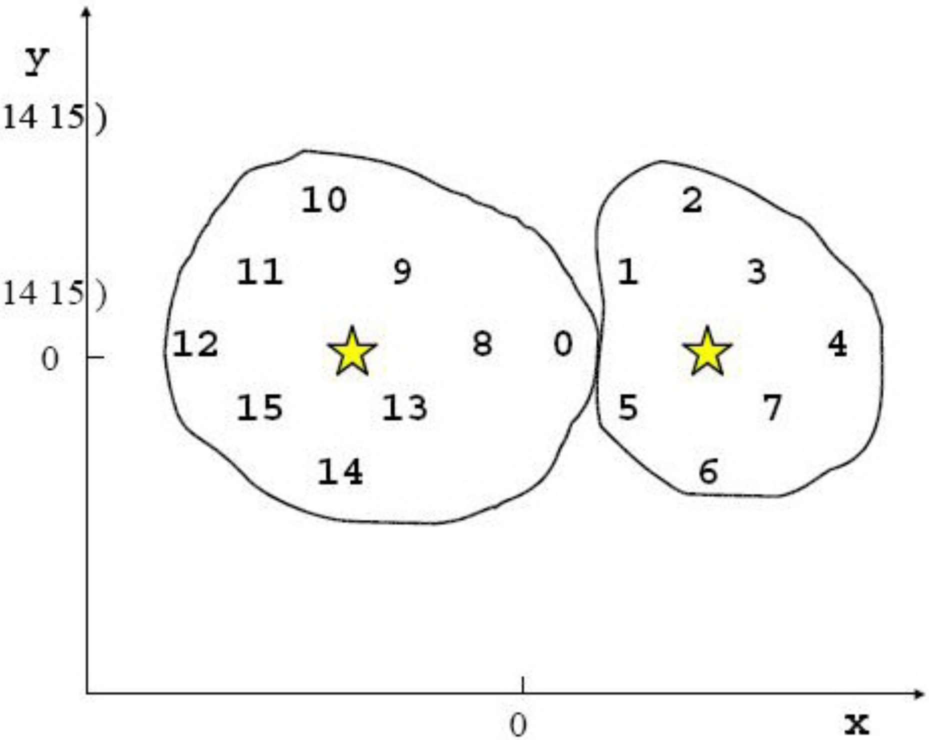
K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)
Cluster Centers: (5.57143 0.0) (-4.33334 0.0)
Average Distance: 3.49115

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)



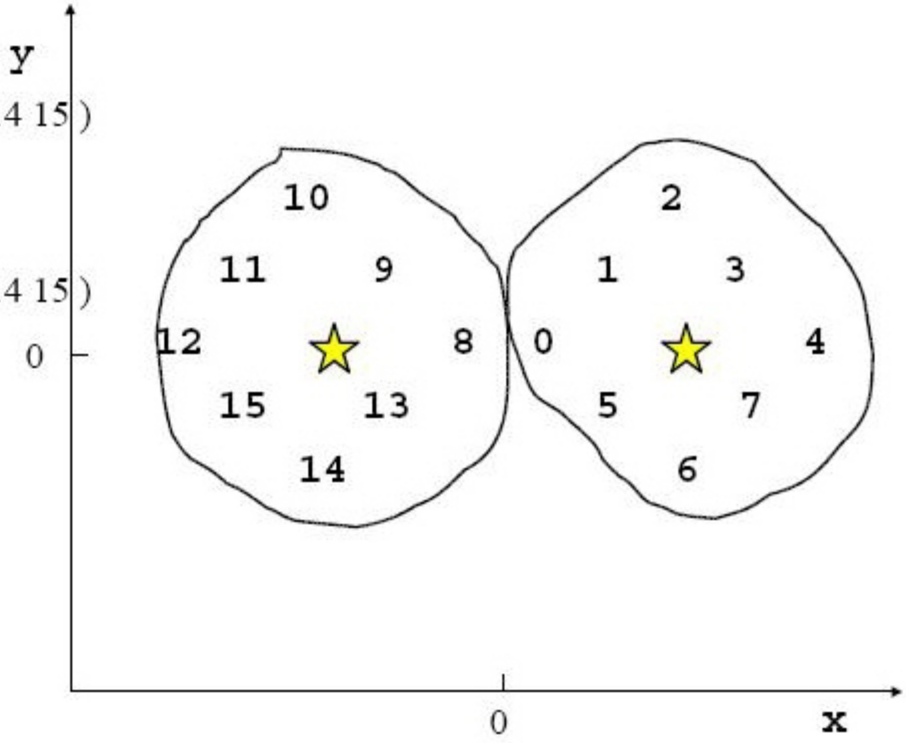
K-Means Clustering Method

Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)
Cluster Centers: (5.57143 0.0) (-4.33334 0.0)
Average Distance: 3.49115

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)
Cluster Centers: (5.0 0.0) (-5.0 0.0)
Average Distance: 3.41421



K-Means Clustering Method

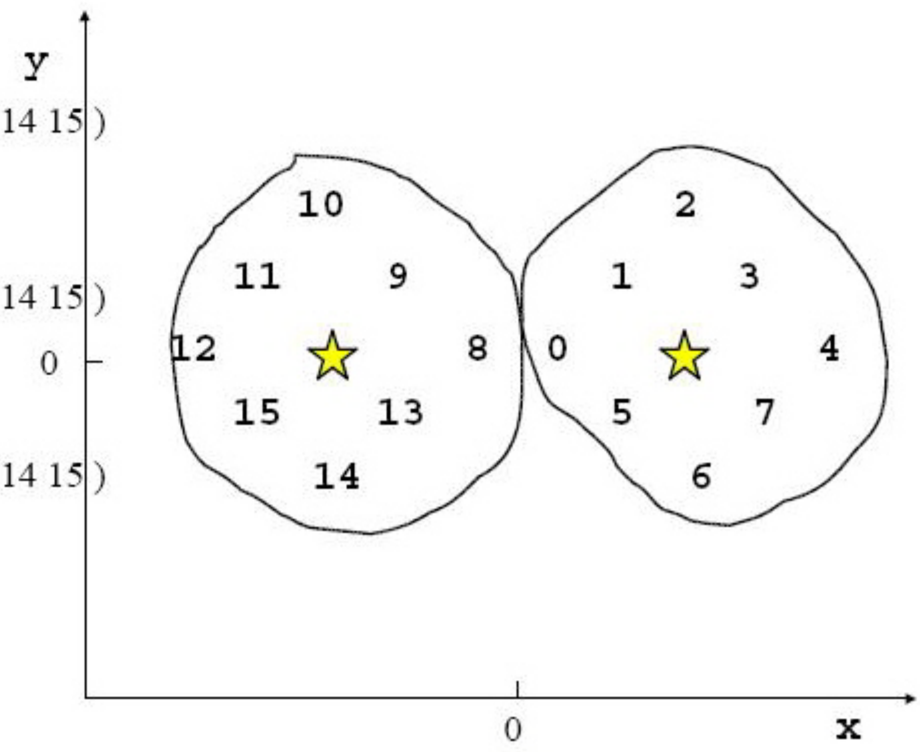
Clustering: (4 6 7) (0 1 2 3 5 8 9 10 11 12 13 14 15)
Cluster Centers: (7.0 -2.0) (-1.61538 0.46153)
Average Distance: 4.35887

Clustering: (2 3 4 5 6 7) (0 1 8 9 10 11 12 13 14 15)
Cluster Centers: (6.0 -0.33334) (-3.6 0.2)
Average Distance: 3.6928

Clustering: (1 2 3 4 5 6 7) (0 8 9 10 11 12 13 14 15)
Cluster Centers: (5.57143 0.0) (-4.33334 0.0)
Average Distance: 3.49115

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)
Cluster Centers: (5.0 0.0) (-5.0 0.0)
Average Distance: 3.41421

Clustering: (0 1 2 3 4 5 6 7) (8 9 10 11 12 13 14 15)
No improvement.



CLARA (Clustering LARge Applications)

CLARA (Kaufmann and Rousseew in 1990)

- Built in statistical analysis packages, such as S+

It **draws** multiple samples of the data set,

applies PAM on each sample, and gives the best clustering as the output

Strength: deals with larger data sets than PAM

Weakness:

- Efficiency depends on the sample size
- A **good** clustering based on samples will not necessarily **represent** a good clustering of the whole data set if the sample is biased