THE UNIVERSITY OF WESTERN AUSTRALIA

Data Warehousing

Lecture 10 Density and Partition based Data Clustering

CITS3401
CITS5504

Zeyi Wen

Computer Science and Software Engineering

School of Maths, Physics and Computing

Acknowledgement: The lecture slides are based on online sources.

Project 2: Common Questions

- The data have been modified by the data set provider (e.g. to hide some privacy).
  - It is like performing normalisation.
  - We can normalise any attribute to the range of [0,1], where 0 just represents the smallest value of that attribute.
- You can decide how to deal with those “nonsensible” attributes.
  - Remove them, use them as they are, transform them, etc.
- Do I need to discretise the attributes?
  - Better to do that for association rule mining
  - May or may not need discretisation in other tasks.

Lecture Outline

- Introduction to Clustering
  - Computing Distance: Review of Lecture 7
- Density based Clustering
  - DBScan
- Partition based Clustering
  - K-Means
  - K-Medoids

What is Cluster Analysis?

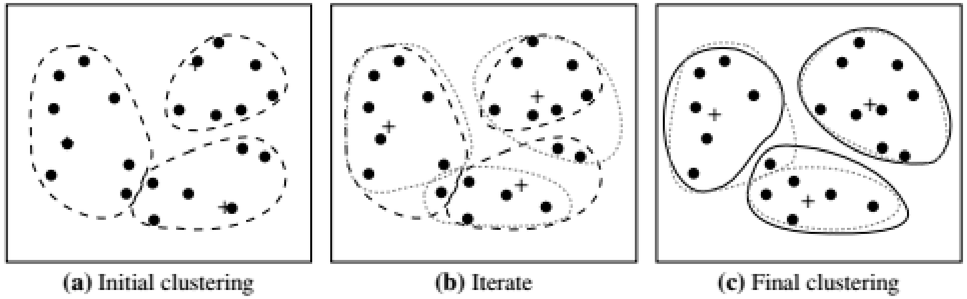
- **Cluster:**
  - instances similar to one another are within the same cluster
  - Instances dissimilar are in different clusters
- **Cluster analysis: Finding characteristics for similar instances**
- **Unsupervised learning: no predefined classes**
- **Typical applications**
  - As a stand-alone tool to get insight into data distribution
  - As a preprocessing step for other algorithm
- **Rich Applications**
  - Document classification
  - Market research
  - DNA analysis
  - Create thematic maps in GIS

What is good clustering?

- Good clustering methods produce high quality clusters with
  - high intra-class similarity
  - low inter-class similarity
  - The definitions of similarity, measured as a distance functions may be different for numeric, boolean, and categorical attributes. Often is highly problem dependent.
- The quality of a clustering method is also measured by its ability to discover some or all of the hidden patterns.

Major Types of Clustering Algorithms

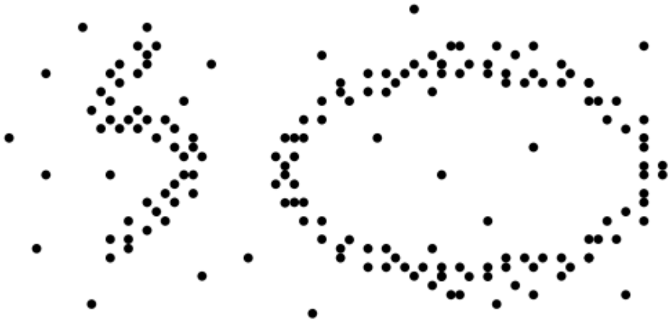
- **Partition-based Clustering**
  - K-Means Algorithm
  - K-Medoids Algorithm
- **Density-Based Clustering**
  - DBScan



(a) Initial clustering

(b) Iterate

(c) Final clustering



Major Clustering Approaches

- **Partitioning approach: *k-means, k-medoids*, CLARANS**
  - Construct k-partitions for the given n-instances (k ≤ n). Each group contains at least one instance. Each instance must belong to exactly one group.
- **Hierarchical approach: Diana, Agnes, BIRCH, ROCK, CAMELEON**
  - Create a hierarchical decomposition of the set of objects using some criterion (linkage function)
  - Agglomerative Approach: bottom-up merging
  - Divisive Approach: top-down splitting
- **Density-based approach: *DBSACN*, OPTICS, DenClue**
  - Based on connectivity and density functions. i.e. for each data point within a given cluster, the radius of a given cluster has to contain at least a minimum number of points.

Lecture Outline

- Introduction to Clustering
  - Computing Distance: Review of Lecture 7
- Density based Clustering
  - DBScan
- Partition based Clustering
  - K-Means
  - K-Medoids

Distance Measures for Numeric Attributes

- Distances are normally used to measure the similarity or dissimilarity between two data objects
- Some popular ones include: Minkowski distance:
  - $\boldsymbol{d(i,j) = \sqrt[q]{|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \cdots + |x_{ip} - x_{jp}|^q}}$
  - where i $= (x_{i1}, x_{i2}, \ldots, x_{ip})$ and j $= (x_{j1}, x_{j2}, \ldots, x_{jp})$ are two p-dimensional instances, and q is a positive integer.
  - If q =1, d is **Manhattan** Distance
    - $d(i,j) = |\boldsymbol{x_{i1} - x_{j1}}| + |\boldsymbol{x_{i2} - x_{j2}}| + \cdots + |\boldsymbol{x_{ip} - x_{jp}}|$
  - If q=2, d is **Euclidean** Distance
    - $\boldsymbol{d(i,j)} = \sqrt{|\boldsymbol{x_{i1} - x_{j1}}|^2 + |\boldsymbol{x_{i2} - x_{j2}}|^2 + \cdots + |\boldsymbol{x_{ip} - x_{jp}}|^2}$
    - Properties:
      - $\boldsymbol{d(i,j) \geq 0}$; $\boldsymbol{d(i,j) = 0}$; $\boldsymbol{d(i,j) = d(j,i)}$ and
      - $\boldsymbol{d(i,j) \leq d(i,k) + d(j,k)}$; **(Triangle inequality)**
- Also, one can use weighted distance, parametric Pearson product moment correlation, or other dissimilarity measures

Compute Dissimilarity for Binary Attributes

- **A contingency table for binary data**
  - Symmetric: both matches are equally important.
  - Asymmetric: the match of “0” is not important.
- **Distance Measure for**
  - Symmetric binary attributes:
    - $d(i,j) = \frac{b+c}{a+b+c+d} = \frac{b+c}{p}$
  - Asymmetric binary attributes:
    - $d(i,j) = \frac{b+c}{a+b+c}$

		Instance j		
		1	**0**	**total**
Instance i	**1**	a	b	$a+b$
	0	c	d	$c+d$
	total	$a+c$	$b+d$	p

- **Jaccard Coefficient (similarity measure for asymmetric binary attributes):**
  - $sim_{Jaccard}(i,j) = \frac{a}{a+b+c}$
- **Convert binary attribute into numerical attribute**

Dissimilarity between Asymmetric Binary Attributes

- **Given the following 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

		Mary		
		1	**0**	Σ_{row}
Jack	1	2	0	2
	0	1	3	4
	Σ_{col}	3	3	6

  - Gender is a symmetric attribute (**not counted in**)
  - The remaining attributes are asymmetric binary
  - Let the values of Y and P to be 1, and value N to be 0.

		Jim		
		1	**0**	Σ_{row}
Jack	1	1	1	2
	0	1	3	4
	Σ_{col}	2	4	6

- **We have**
  - $d(jack, mary) = \frac{0+1}{2+0+1} = 0.33$
  - $d(jack, jim) = \frac{1+1}{1+1+1} = 0.67$
  - $d(jim, mary) = \frac{1+2}{1+1+2} = 0.75$

		Mary		
		1	**0**	Σ_{row}
Jim	1	1	1	2
	0	2	2	4
	Σ_{col}	3	3	6

Nominal/Categorical Attributes

- A generalisation of the binary variable in that it can take more than 2 states, e.g. red, yellow, blue, green
- Method 1: Simple matching
  - *m*: # of matches, *p*: total # of attributes
    - $d(i,j) = \frac{p-m}{p}$
- Method 2: convert to a number of binary attributes
  - creating a new binary attribute for each of the *M* possible states

Ordinal Attributes

- An ordinal attribute is often discrete.
- Order is important, e.g. rank (e.g. freshman, sophomore)
- Can be treated as numeric attributes
  - replace x_{if} by their rank $r_{if} \in \{1, \dots, Mf\}$
  - map the range of each attribute onto [0, 1] by replacing *i*-th object in the *f*-th attribute by
    - $z_{if} = (r_{if} - 1)/(M_{if} - 1)$
  - example: freshman: 0; sophomore: 1/3; junior: 2/3; senior 1
    - distance: d(freshman, senior) = 1, d(junior, senior) = 1/3
  - compute the dissimilarity using methods for numeric attributes

Attributes of Mixed Types

- A database may contain **different types** of attributes
  - symmetric binary, asymmetric binary, nominal, ordinal, and numeric attributes
- One may use a weighted formula to combine their effects

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{f} d_{ij}^{f}}{\sum_{f=1}^{p} \delta_{ij}^{f}}$$

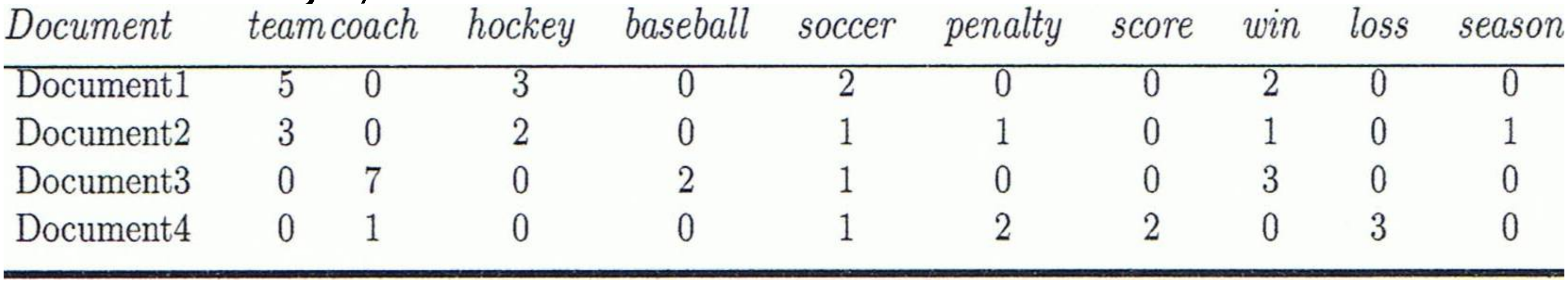
  - *f* is binary or nominal:

$$d_{ij}^{f} = 0, \qquad if\ x_{if} = x_{jf} \quad or$$
$$d_{ij}^{f} = 1, \qquad otherwise$$

  - *f* is numeric: use Euclidean distance
  - *f* is ordinal
    - compute ranks z_{if} where $z_{if} = (r_{if} - 1)/(M_{if} - 1)$
    - and treat z_{if} as numeric attribute

Similarity of Two Vectors

- Vector objects: keywords in documents, gene features in micro-arrays, etc.



Document	team	coach	hockey	baseball	soccer	penalty	score	win	loss	season
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

- Broad applications: information retrieval, natural language understanding, etc.
- Cosine measure
  - $s(x, y) = \frac{d_1{}^T \cdot d_2}{\|d_1\| * \|d_2\|}$
- A variant: Tanimoto coefficient-used in information retrieval
  - $s(x, y) = \frac{d_1{}^T \cdot d_2}{d_1{}^T \cdot d_1 + d_2{}^T \cdot d_2 - d_1{}^T \cdot d_2}$

Lecture Outline

- Introduction to Clustering
  - Computing Distance: Review of Lecture 7
- **Density based Clustering**
  - DBScan
- Partition based Clustering
  - K-Means
  - K-Medoids

Density-based Clustering Algorithms

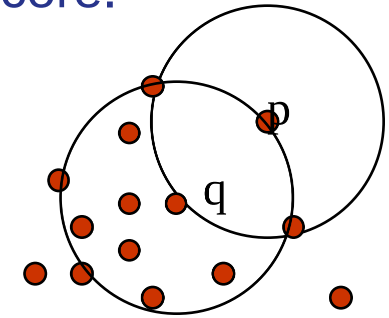
- **Clustering based on density (local cluster criterion), such as density-connected points**

- **Major features:**
  - Discover clusters of arbitrary shape
  - Handle noise
  - **One scan**
  - Need density parameters as termination condition

- **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) (more grid-based)

Density-based Clustering: Basic Concepts

- Two parameters:
  - Eps: Maximum radius of the neighbourhood
  - $MinPts$: Minimum number of points in an Eps-neighbourhood of that point
- $N_{Eps}(p): \{q \in D \mid dist(p,q) \leq Eps\}$
- Directly density-reachable: A point *p* is directly density-reachable from a point *q* w.r.t. *Eps*, *MinPts* if
  - p belongs to $\boldsymbol{N_{Eps}(q)}$ & q is a core.
  - Core point condition:
    - $|N_{Eps}(q)| \geq MinPts|$

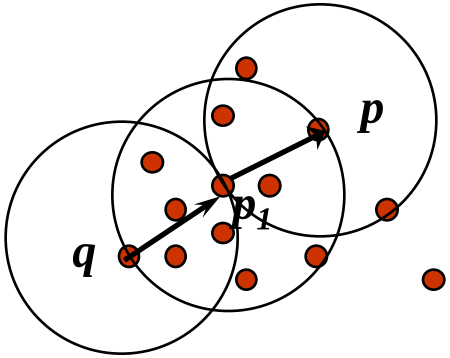


MinPts = 5

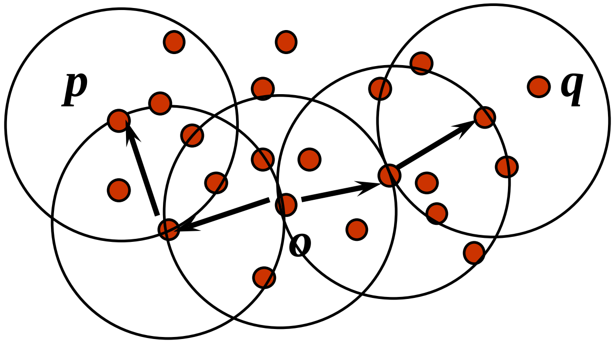
Eps = 1 cm

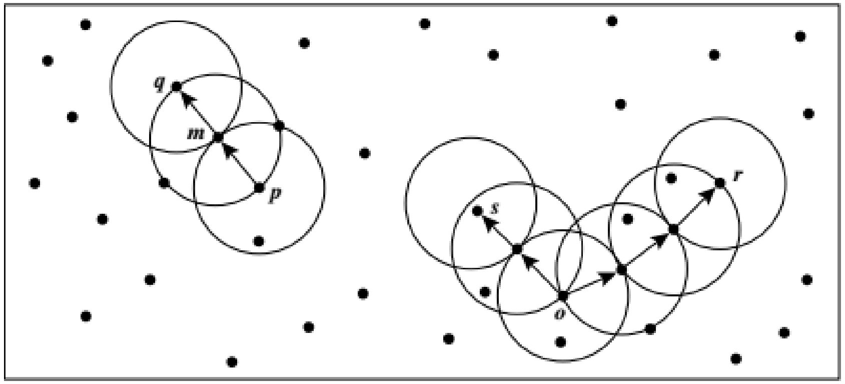
Density-Reachable and Density-Connected

- **Density-reachable:**
  - A point p is density-reachable from a point q w.r.t. Eps, $MinPts$ if there is a chain of points $p_1, \dots, p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i



- **Density-connected**
  - 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 $MinPts$

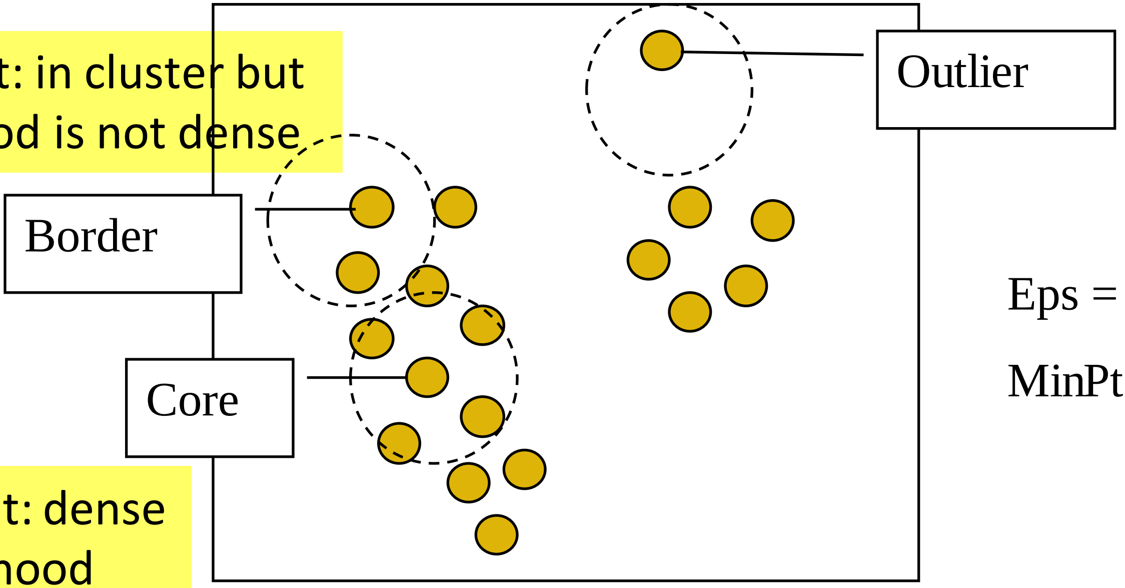




DBSCAN: Density Based Spatial Clustering of Applications with Noise

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

- Discovers clusters of arbitrary shape in spatial databases with noise



Border point: in cluster but neighborhood is not dense

Outlier/noise: not in a cluster

Eps = 1cm

MinPts = 5

Core point: dense neighborhood

DBSCAN: The Algorithm

- Arbitrary 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.

DBSCAN: Sensitive to Parameters

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

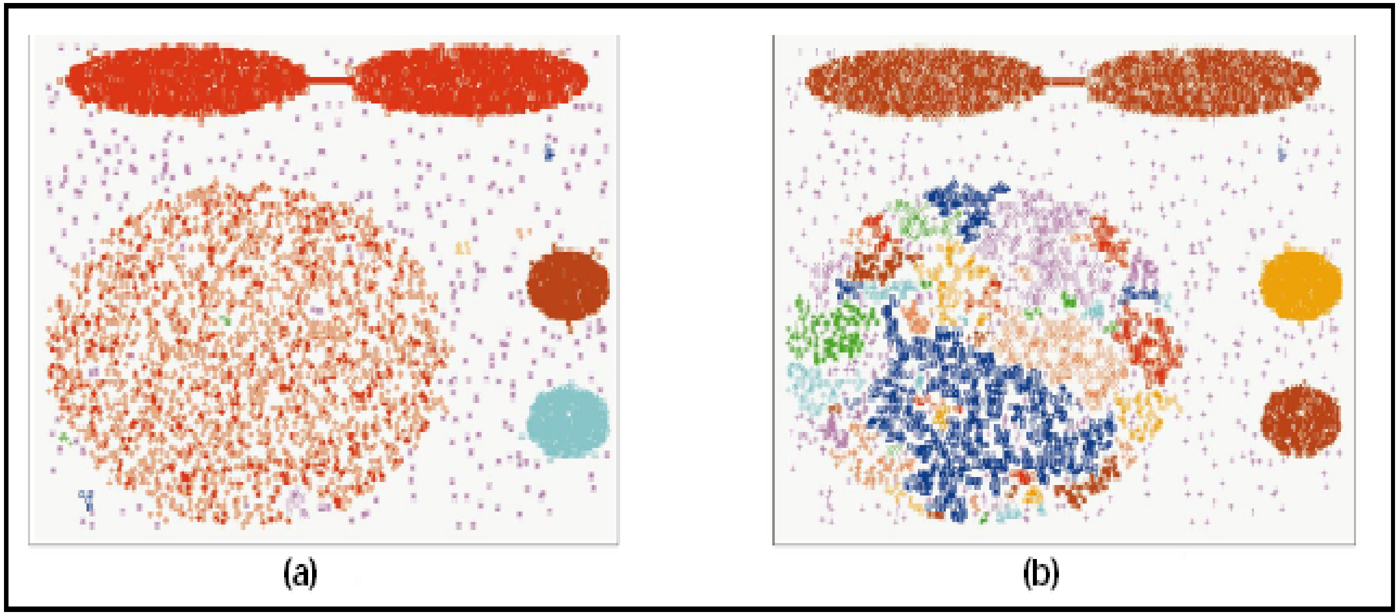
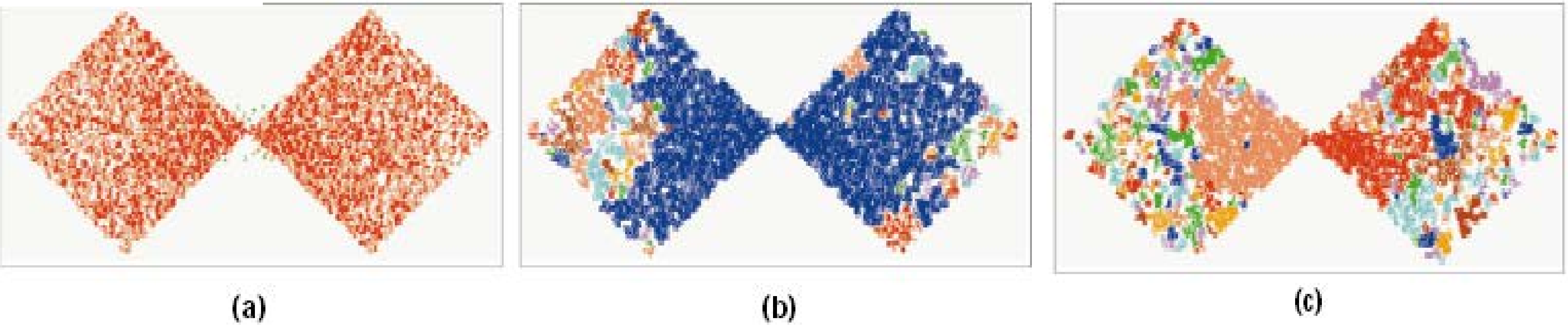


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.



Lecture Outline

- Introduction to Clustering
  - Computing Distance: Review of Lecture 7
- Density based Clustering
  - DBScan
- **Partition based Clustering**
  - K-Means
  - K-Medoids

Partitioning Algorithms: Basic Concept

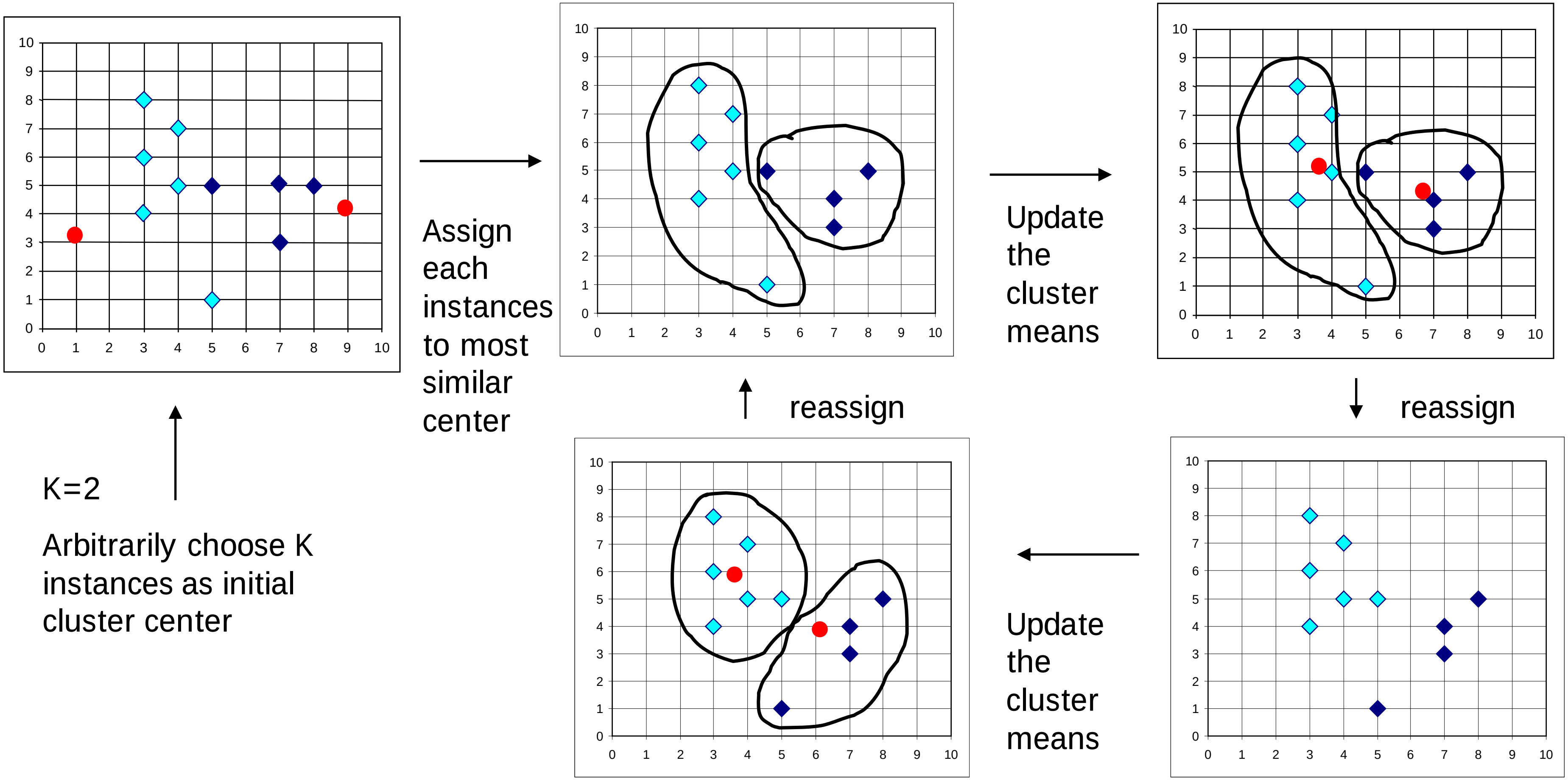
- Partitioning method: Construct a partition of a database *D* of *n* instances into a set of *k* clusters, s.t., minimise the sum of squared distance (within cluster variance)
  - $E = \sum_{i=1}^{k} \sum_{p \in C_i} (p - m_i)^2$
- Given an integer *k*, find a partition of *k clusters* that optimises the chosen partitioning criterion
  - Global optimal: exhaustively enumerate all partitions
  - 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 instances in the cluster

k-Means: A centroid-based partitioning algorithm

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

1. Partition instances 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

Example: the k-Means Clustering Method



K=2

Arbitrarily choose K instances as initial cluster center

Assign each instances to most similar center

Update the cluster means

reassign

Update the cluster means

reassign

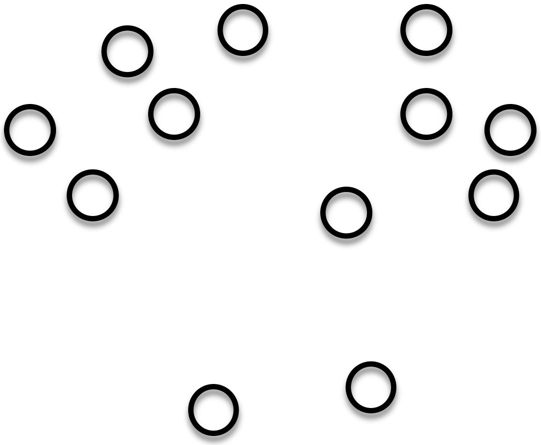
K-means clustering algorithm

- **Input**: *K*, a set of points $x_1, x_2, \ldots, x_n$ where $x_i \in R^d$
- Initialise *K* centroids $c_1, c_2, \ldots, c_K$ at random locations
- Repeat until convergence
  - → for each point x_i: Assignment step
    - ✳ find the nearest centroid c_j
    - ✳ assign x_i to cluster *j*

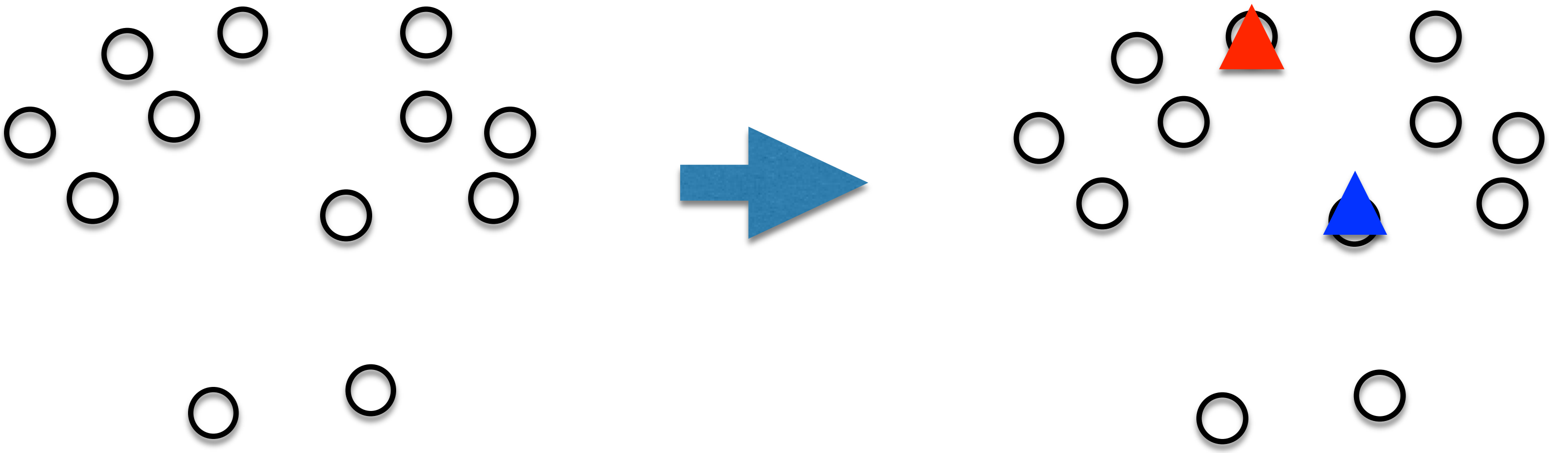
$$\mathbf{c}_{j,a} = \frac{1}{n_j} \sum_{\mathbf{x}_i \to \mathbf{c}_j} \mathbf{x}_{i,a} \quad \text{for } \; a \in \{1, 2, ..., d\}$$

  - → for each centroid c_j: Update step
    - ✳ update c_j to the mean of all the points assigned to cluster *j*
- "Convergence" means none of the points changes its cluster.

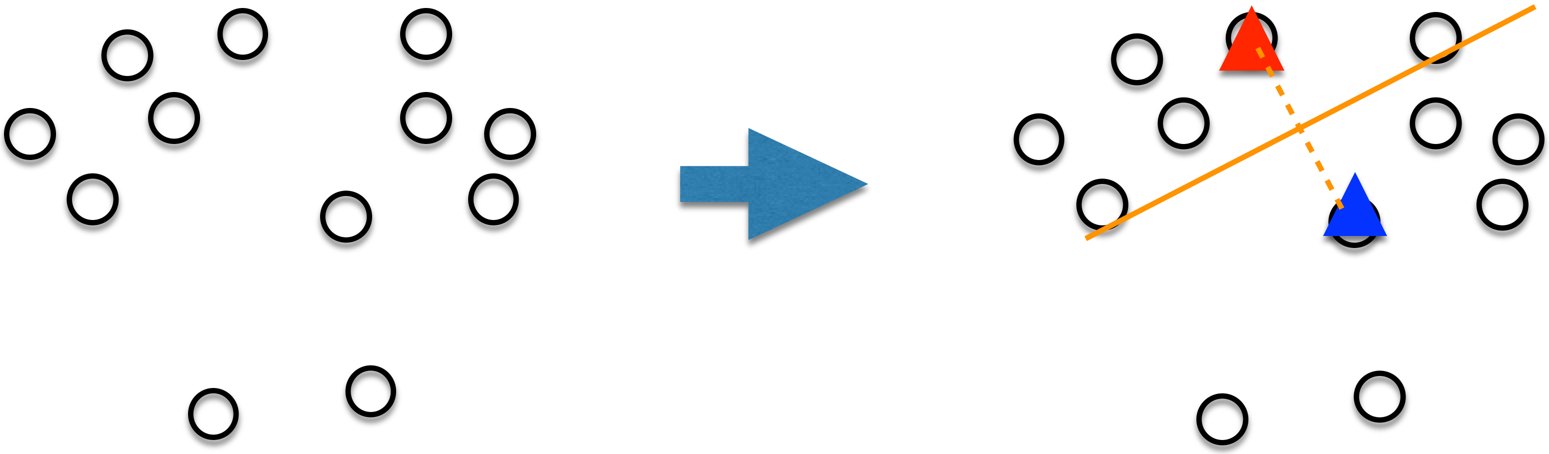
K-means clustering example



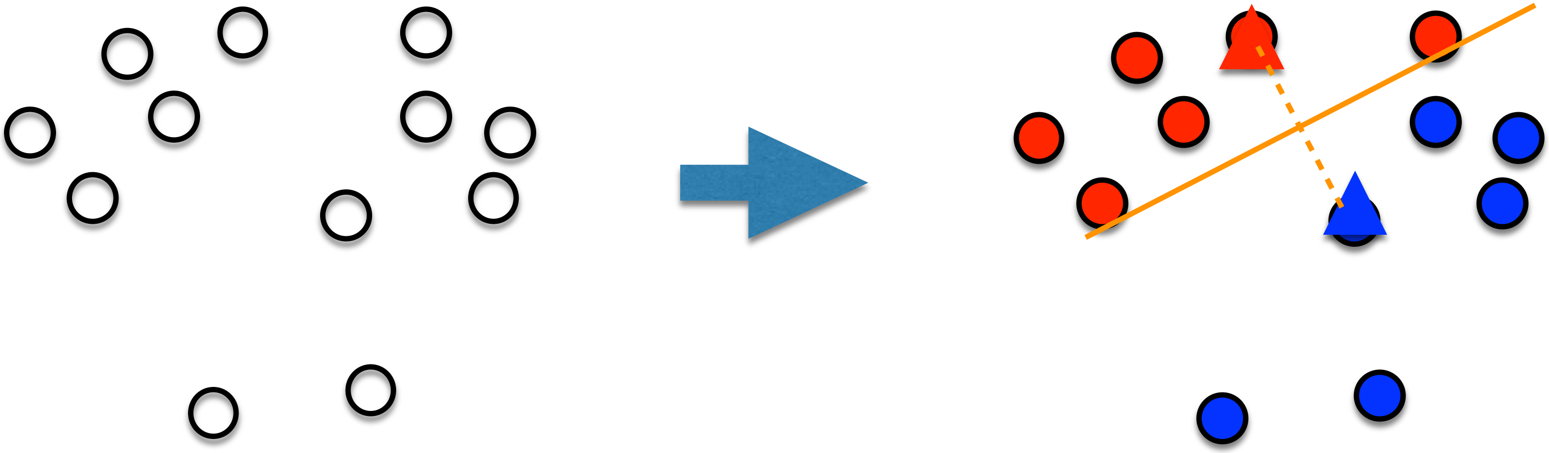
K-means clustering example



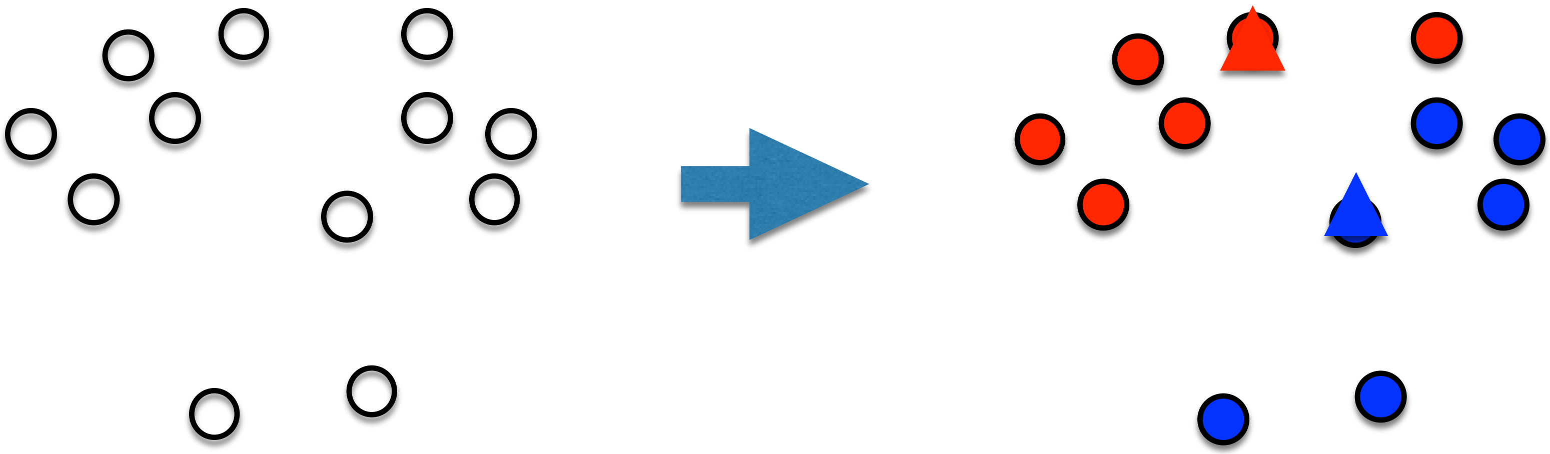
K-means clustering example



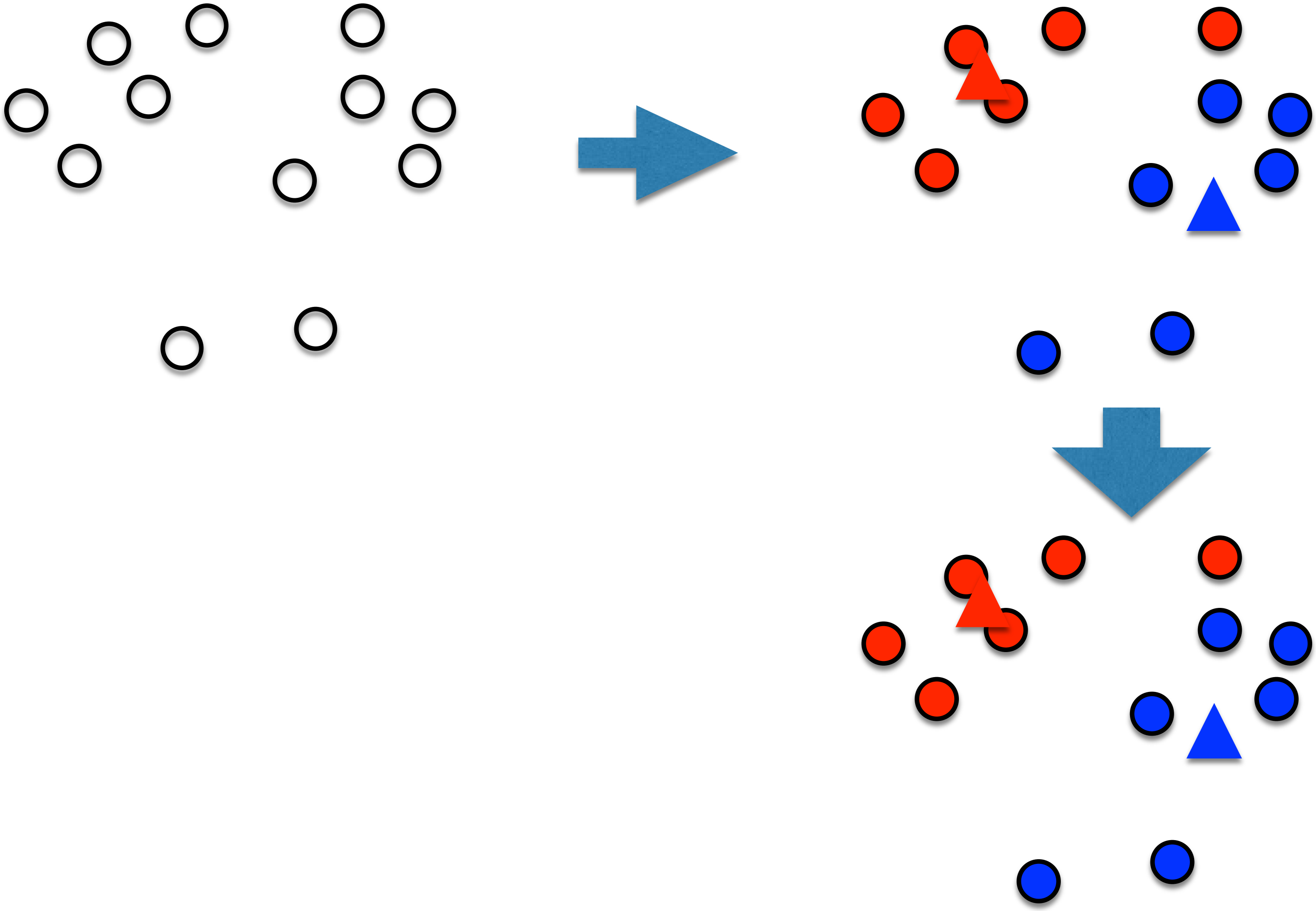
K-means clustering example



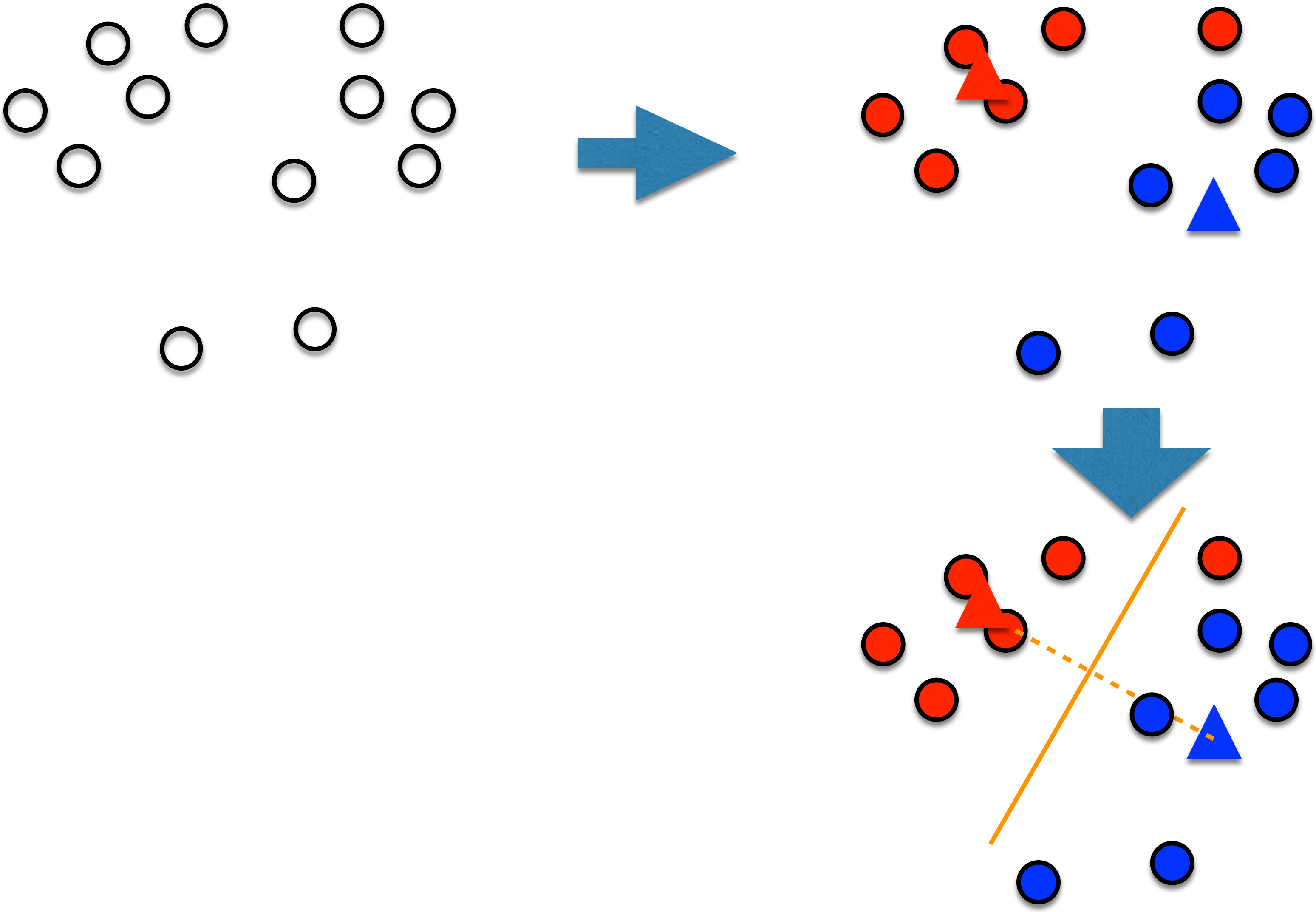
K-means clustering example



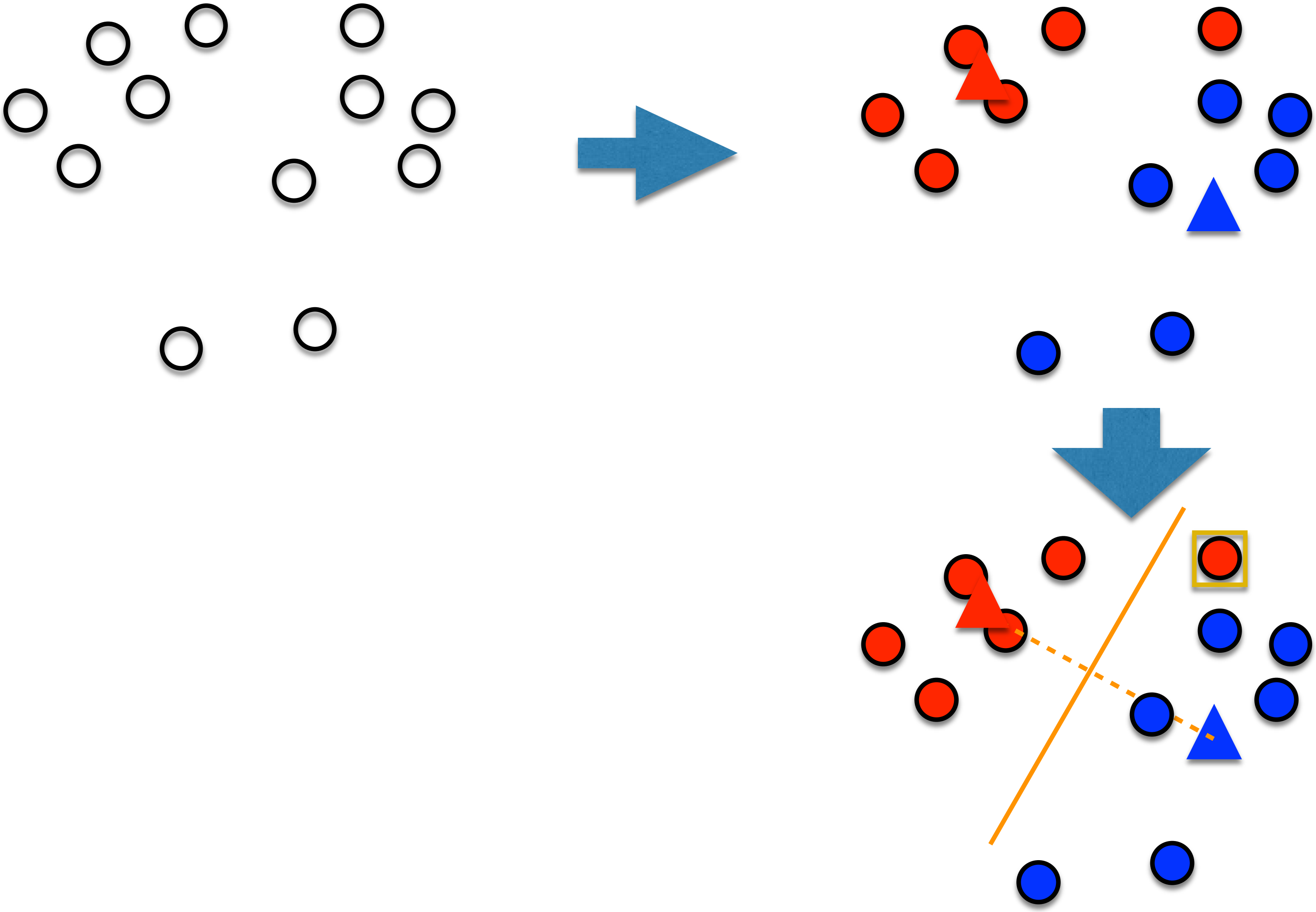
K-means clustering example



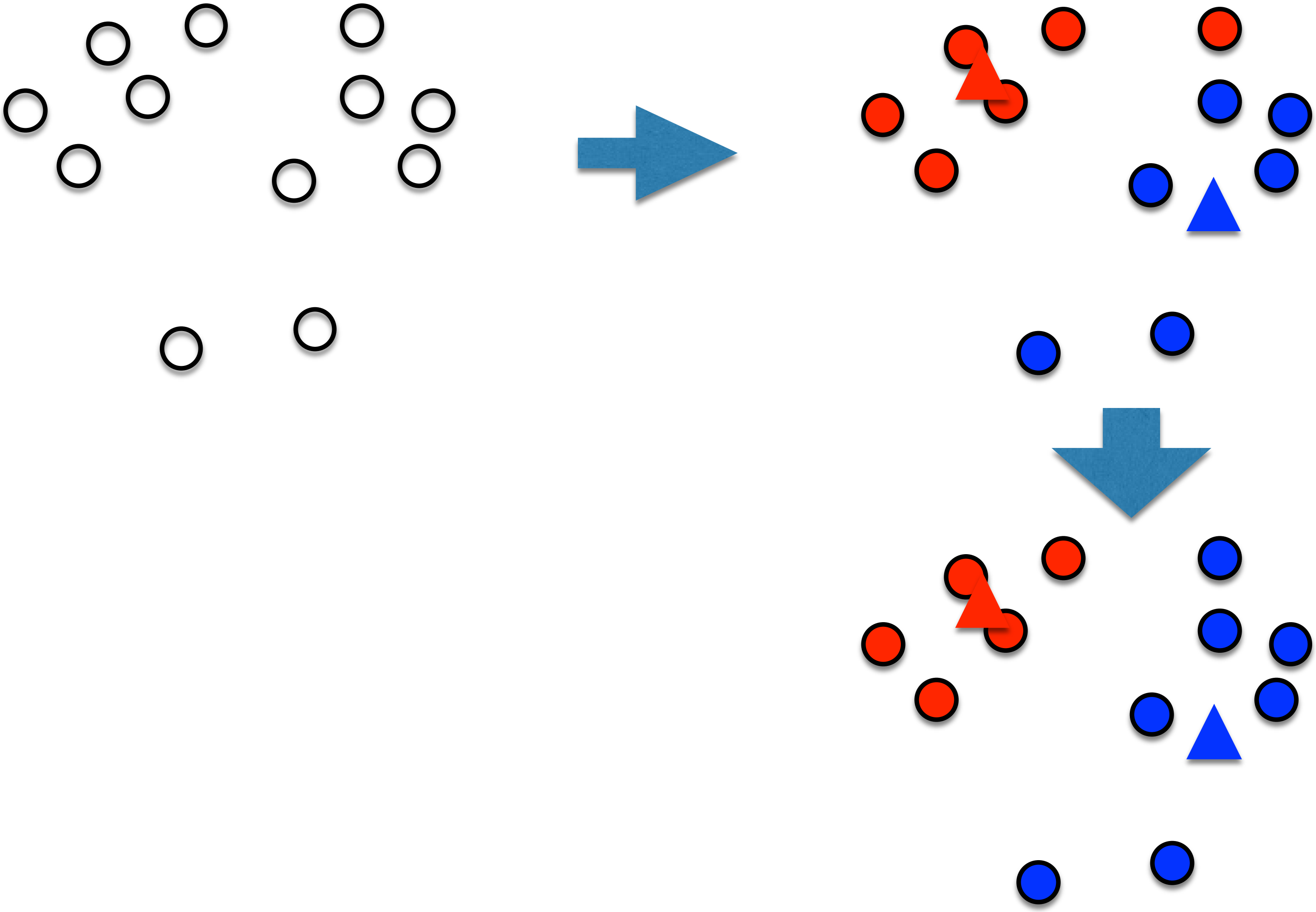
K-means clustering example



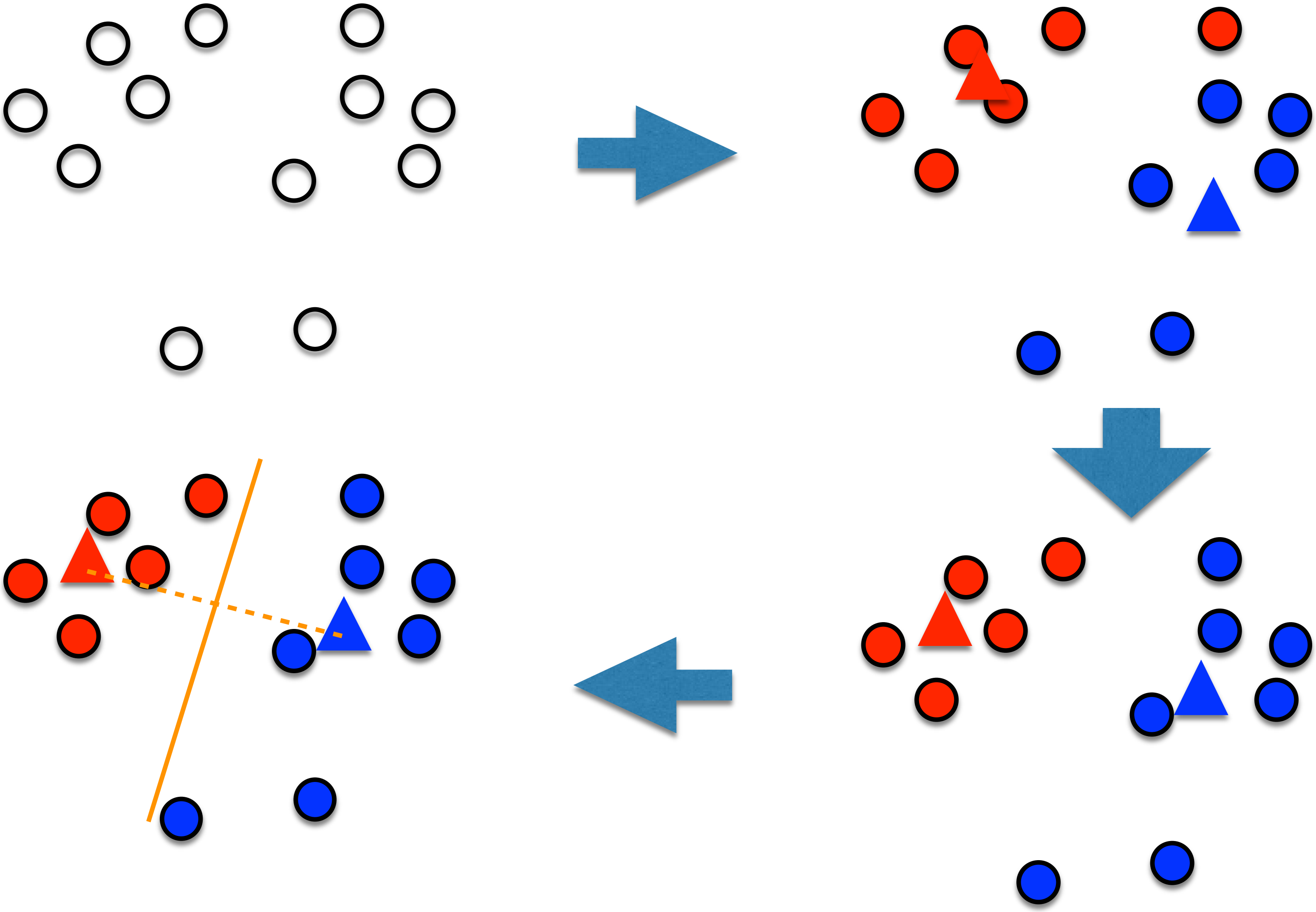
K-means clustering example



K-means clustering example

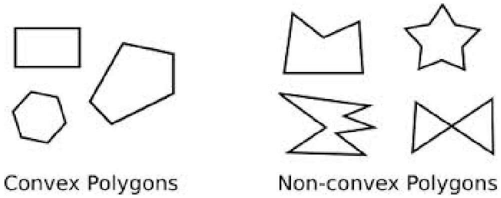


K-means clustering example



Comments on the k-Means Method

- Strength: Relatively efficient: $O(tkn)$, where n is # instances, k is # clusters, and t is # iterations.
  - Normally, $k, t \ll n$.
- Often terminates at a local optimum. The global optimum may be found using techniques such as:
  - deterministic annealing and genetic algorithms
- **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 k-means

- **A few variants of the *k-means* which differ in**
  - Choosing better initial *centroids*
    - *k-means++, Intelligent k-means, Genetic k-means*
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- **Handling categorical data: *k-modes* (Huang'98, aside)**
  - Using new dissimilarity measures to deal with categorical attributes
  - A mixture of categorical and numerical data: *k-prototype* method
  - Replacing means of clusters with modes
  - Using a frequency-based method to update modes of clusters

Centroid, Radius and Diameter of a Cluster (for numerical data sets)

- Centroid: the “middle” of a cluster
  - $C_m = \frac{\sum_{i=1}^{N} t_{ip}}{N}$
- Radius: square root of average distance from any point of the cluster to its centroid
  - $R_m = \sqrt{\frac{\sum_{i=1}^{N}(t_{ip}-C_m)^2}{N}}$
- Diameter: square root of average mean squared distance between all pairs of points in the cluster
  - $D_m = \sqrt{\frac{\sum_{i=1}^{N}\sum_{i=1}^{N}(t_{ip}-t_{iq})^2}{N(N-1)}}$

Calculate the distance between Clusters

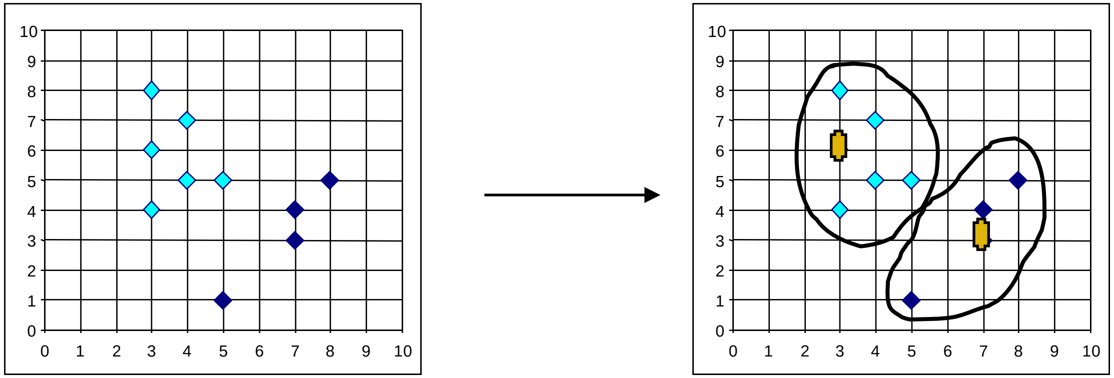
- **Single link**
  - smallest distance between an element in one cluster and an element in the other, i.e. $dis(K_i, K_j) = \min(t_{ip}, t_{jp})$
- **Complete link**
  - largest distance between an element in one cluster and an element in the other, i.e. $dis(K_i, K_j) = max(t_{ip}, t_{jp})$
- **Average**
  - average distance between an element in one cluster and an element in the other, i.e. $dis(K_i, K_j) = avg(t_{ip}, t_{jp})$
- **Centroid**
  - distance between the centroids of two clusters, i.e. $dis(K_i, K_j) = dis(C_i, C_j)$
- **Medoid:**
  - medoid is the most centrally located object in a cluster
  - distance between the medoids of two clusters, i.e. $dis(K_i, K_j) = dis(M_i, M_j)$

Lecture Outline

- Introduction to Clustering
  - Computing Distance: Review of Lecture 7
- Density based Clustering
  - DBScan
- Partition based Clustering
  - K-Means
  - **K-Medoids**

The problem of k-means method

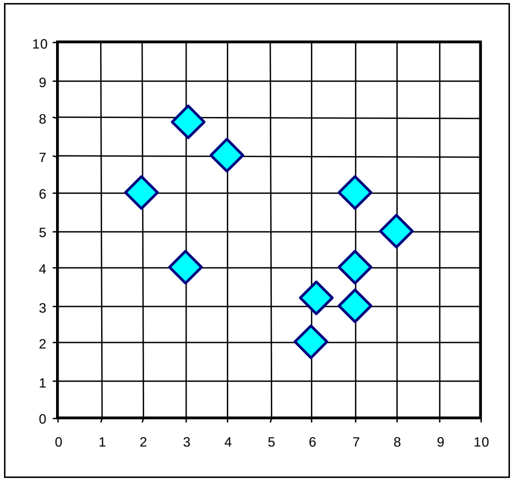
- **The k-means algorithm is sensitive to outliers!**
  - Since an instance with an extremely large value may substantially distort the distribution of the data.
- **K-Medoids:**
  - Instead of taking the mean value of the instance in a cluster as a reference point, medoids can be used, which is the most centrally located instance in a cluster.



The *K-Medoids* Clustering Method

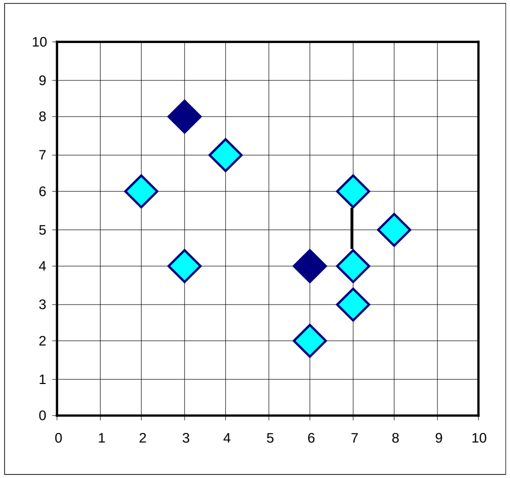
- **Find *representative* instances, called medoids, in clusters**
- ***PAM* (Partitioning Around Medoids, 1987)**
  - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
  - *PAM* works effectively for small data sets, but does not scale well for large data sets
- ***CLARA* (Kaufmann & Rousseeuw, 1990)**
- ***CLARANS* (Ng & Han, 1994): Randomised sampling**
- **Focusing + spatial data structure (Ester et al., 1995)**

A Typical K-Medoids Algorithm (PAM)

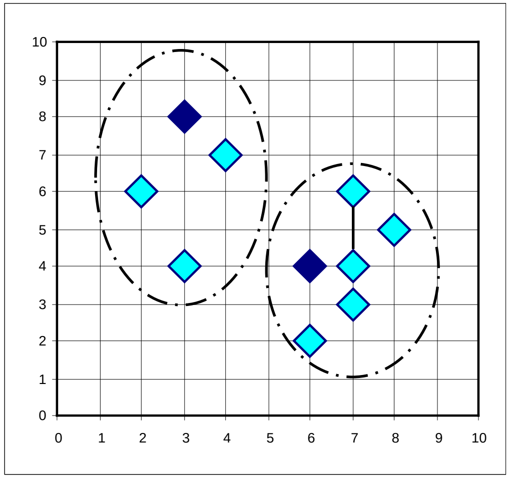


K=2

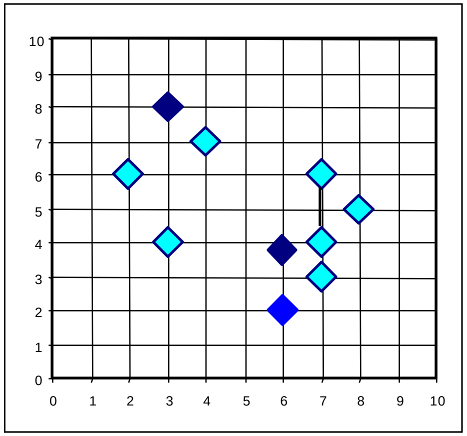
Arbitrary choose k object as initial medoids



Assign each remaining object to nearest medoids

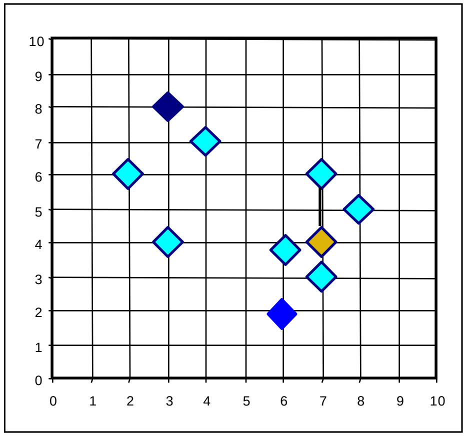


Randomly select a nonmedoid object,O$_{ramdom}$



Compute total cost of swapping

Total Cost = 26



Swapping O and O$_{ramdom}$

If quality is improved.

Do loop

Until no change

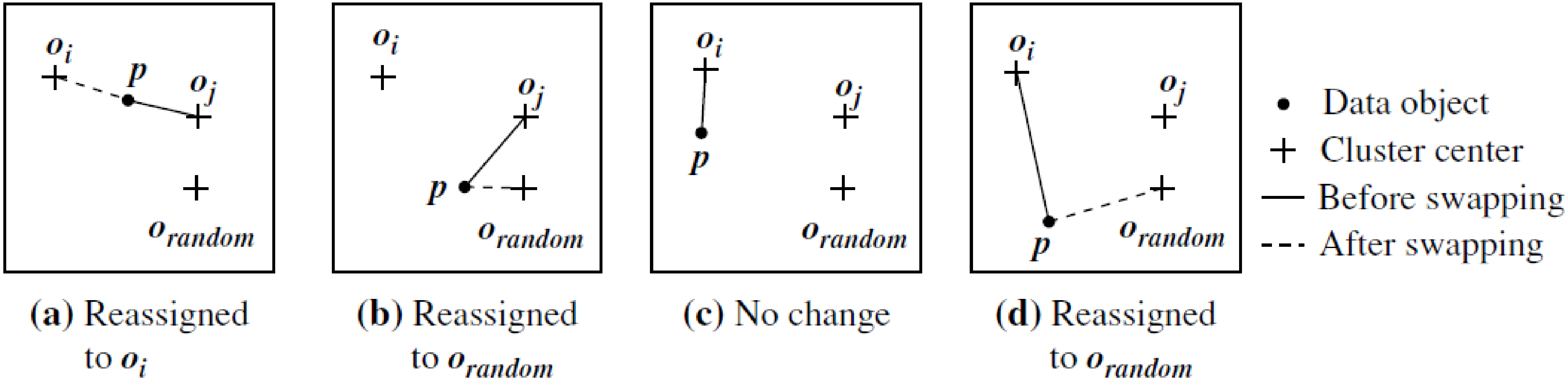
PAM Algorithm

- **Use a real instance/object to represent the cluster**

1. Select $\boldsymbol{k}$ representative instances arbitrarily
2. For each pair of non-selected instance $\boldsymbol{h}$ and selected instance $\boldsymbol{i}$, calculate the total swapping cost TC_{ih}
3. For each pair of $\boldsymbol{i}$ and $\boldsymbol{h}$,
   - If $TC_{ih} < 0$, $\boldsymbol{i}$ is replaced by $\boldsymbol{h}$
   - Then assign each non-selected instance to the most similar representative instance
4. repeat steps 2-3 until there is no change

Determining o_{random} a good replacement of o_j

- **We calculate the distance from every object p to the closest object in the set $\{o_1, \ldots, o_{j-1}, o_{random}, o_{j+1}, \ldots, ok\}$,**
- **Then use the distance to update the cost function.**
  - Suppose object p is currently assigned to a cluster represented by medoid o_j (Figure a or b). Do we need to reassign p to a different cluster (represented by medoid o_i) if o_j is being replaced by o_{random}?
  - What if object p is currently assigned to a different cluster represented by medoid o_i? (Figure c or d)



(a) Reassigned to o_i

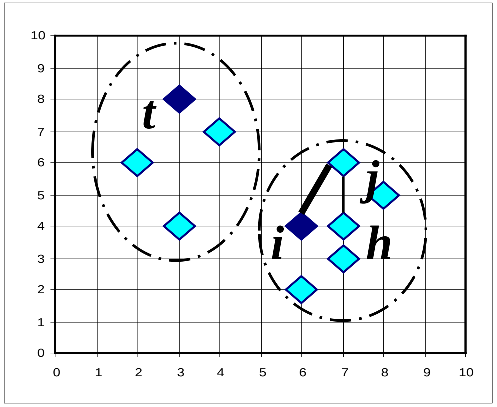
(b) Reassigned to o_{random}

(c) No change

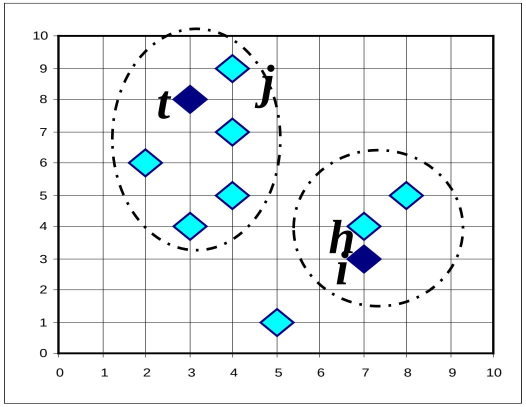
(d) Reassigned to o_{random}

PAM Clustering: Total Swapping Cost: $TC_{ih} = \sum_j C_{jih}$

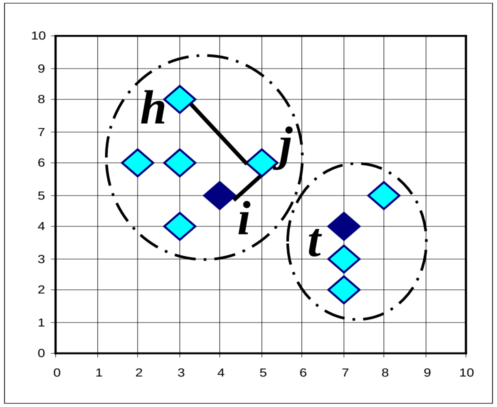
Replace *i* by *h*, the following scenarios are the distance change on *j to its nearest medoid*



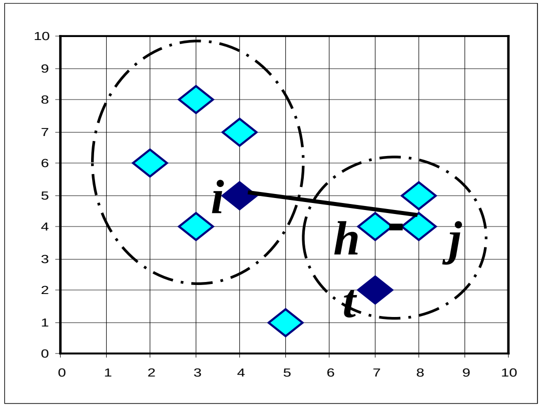
$$C_{jih} = d(j, h) - d(j, i)$$



$$C_{jih} = 0$$



$$C_{jih} = d(j, t) - d(j, i)$$



$$C_{jih} = d(j, h) - d(j, t)$$

What is the problem with PAM

- PAM is more robust 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 instances, *k* is # of clusters

➔Sampling based method,

CLARA (Clustering LARge Applications)

CLARA (Clustering Large Applications) (1990)

- CLARA (Kaufmann and Rousseeuw 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

Overview of Clustering Methods

	General Characteristics
Partitioning methods	– Find mutually exclusive clusters of **spherical** shape – Distance-based – May use **mean or medoid** to represent cluster center – Effective for small- to medium-size data sets
Hierarchical Methods (aside)	– Clustering is a hierarchical decomposition (i.e., multiple levels) – Cannot correct erroneous merges or splits – May incorporate other techniques like microclustering or consider object “linkages”
Density-based methods	– Can find **arbitrarily** shaped clusters – Clusters are dense regions of objects in space that are separated by low-density regions – Cluster density: Each point must have a minimum number of points within its “neighbourhood” – May **filter out outliers**
Grid-based Methods (aside)	– Use a multiresolution grid data structure – Fast processing time (typically independent of the number of data objects, yet dependent on grid size)

Summary

- In clustering, data are
  - similar to one another within the same cluster, and
  - dissimilar to the data in other clusters.
- Cluster analysis can be used as
  - a stand-alone tool to gain insight into the data distribution or
  - can serve as a pre-processing step for other data mining tasks.
- Partitioning method:
  - iterative relocation technique: k-means and k-medoids.
  - K-medoid is efficient in presence of noise and outliers.
- Desnsity based method
  - Discover clusters of arbitrary shape,
  - good at handling noise,
  - requires only one scan
  - sensitive to density parameters (needed to be set manually)

Reference

- Han et al.'s book
  - Chapter 2.1, 2.4 and 10
- Readings
  - An Introduction to k-means
  - K-means++