Outline

- Unsupervised learning vs supervised learning
- A categorization of major clustering methods
- Partitioning-based clustering
- Hierarchical-based clustering

Partitioning methods idea

- Construct a partition of a database *D* of *n* objects into a set of *k* clusters
  - Each object belongs to exactly one cluster (*hard* or *crisp* clustering)
  - The number of clusters *k* is given in advance
- The partition should optimize the chosen partitioning criterion
  - e.g., minimize the intra-cluster variance, i.e., the sum of the squared distances from each data point to its cluster center.
  - Possible solutions:
    - Global optimal: exhaustively enumerate all partitions
    - Heuristic methods: *k*-means and *k*-medoids algorithms
    - *k*-means: Each cluster is represented by the center of the cluster
    - *k*-medoids: Each cluster is represented by one of the objects in the cluster .

The k-Means problem

- Given a database D of n points in a d-dimensional space and an integer k

- Task: choose a set of k points $\{c_1, c_2, \ldots, c_k\}$ in the d-dimensional space to form clusters $\{C_1, C_2, \ldots, C_k\}$ such that the clustering cost is minimized:

$$\underbrace{Cost(C) = \sum_{i=1}^{k} \underbrace{\sum_{x \in C_i} (x - c_i)^2}_{\textit{Cluster cost}}}_{\textit{Clustering cost}}$$

- This is an optimization problem, with the objective function to minimize the cost

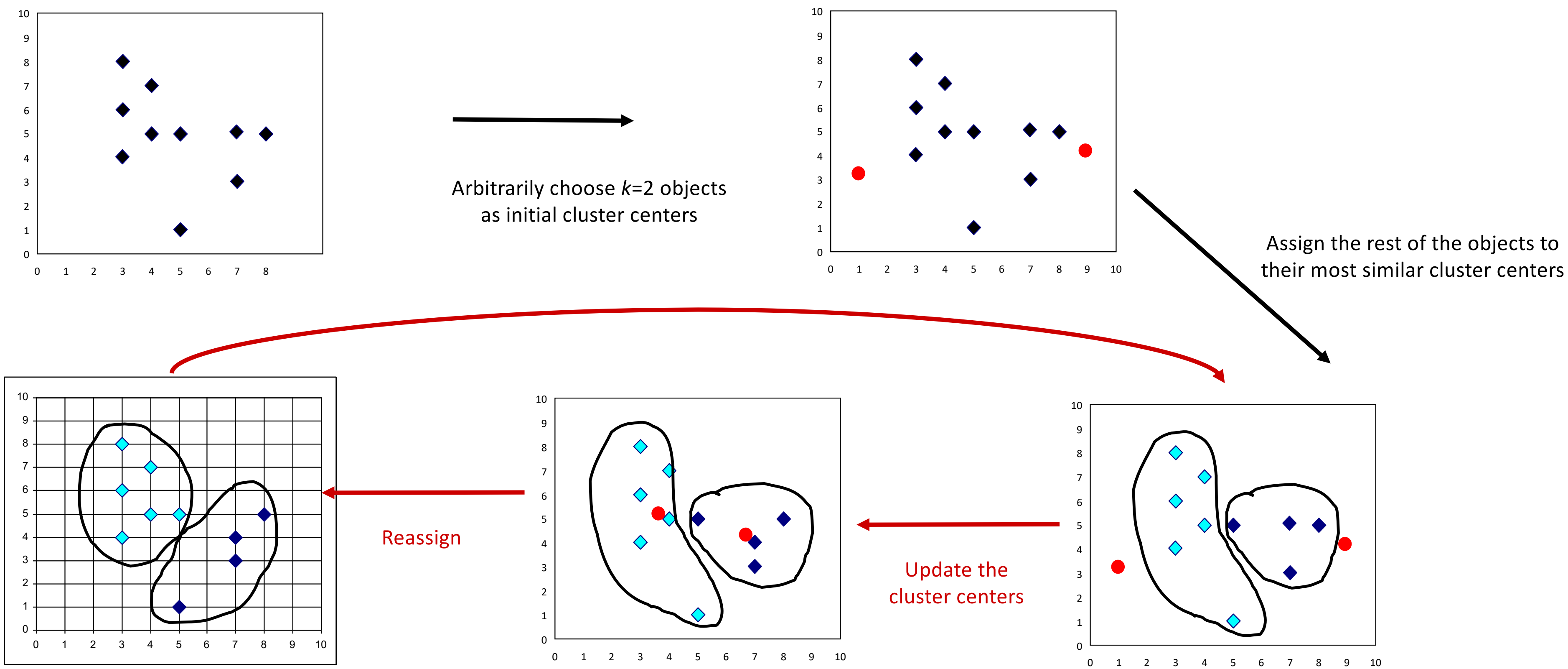
- Enumerating all possible solutions and choosing the global optimum is infeasible.

The *k*-Means algorithm

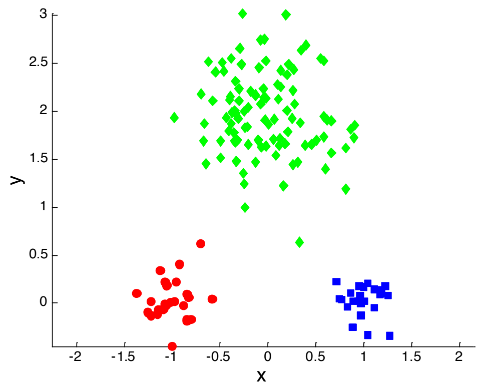
- Given k, the k-Means algorithm is implemented in four steps:
  - Randomly pick k objects as cluster centers $\{c_1, \ldots, c_k\}$.
  - Assign the rest of the points to their closest cluster centers.
  - Update the center of each cluster based on the new point assignments.
  - Repeat until convergence.
    - E.g., cluster centers do not change, cost is not improved significantly, after t iterations, etc.

- Complexity
  - Relatively efficient: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.

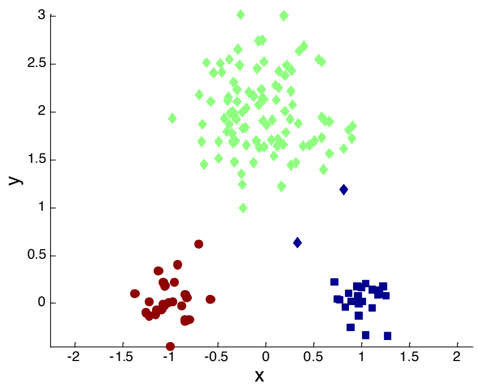
k-Means example



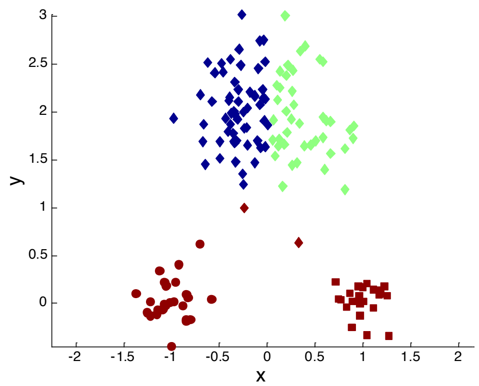
k-Means finds a local optimum



original points

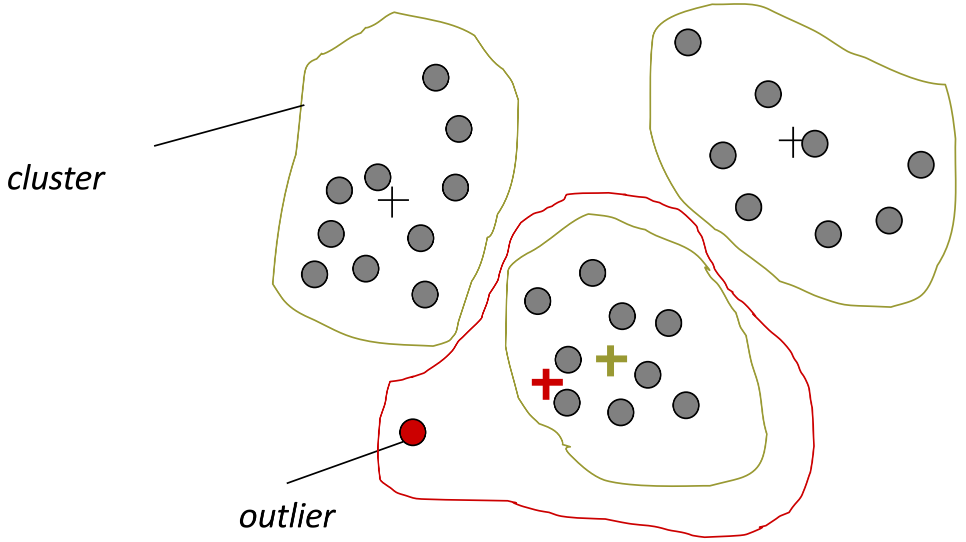


optimal clustering



sub-optimal clustering

k-Means is sensitive to outliers



k-Means variations

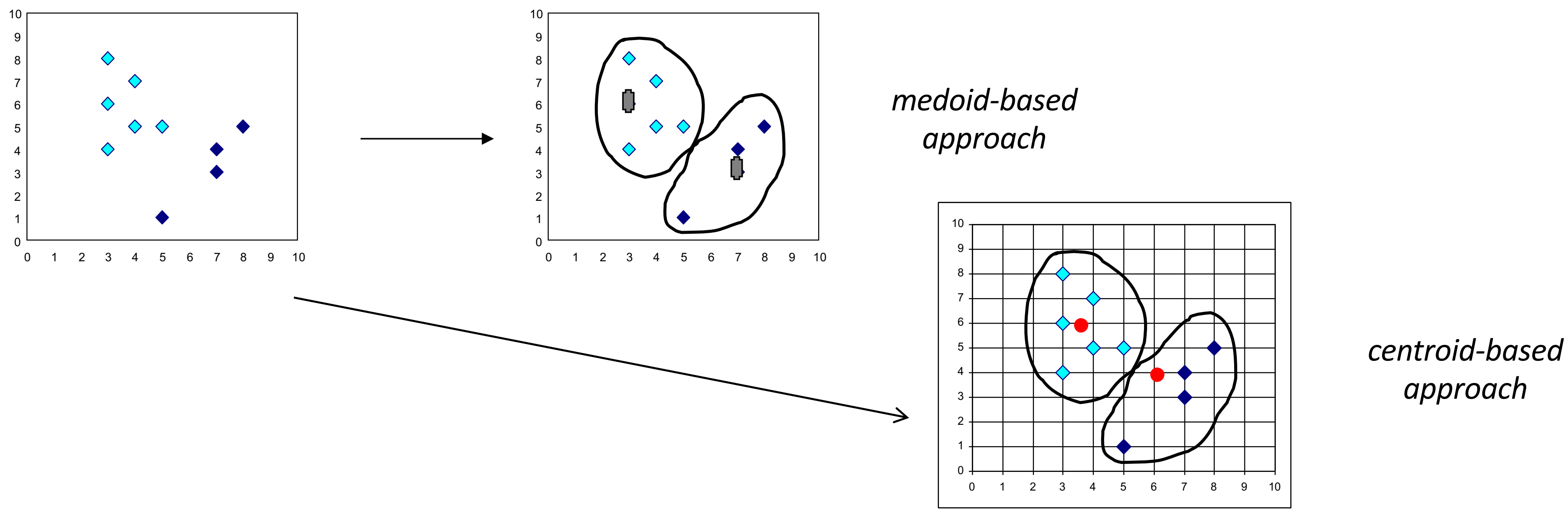
- A few variants of the *k-means* which differ in
  - Selection of the initial *k* means
    - Multiple runs
    - Not random selection of centers. e.g., pick the most distant (from each other) points as cluster centers (*k*Means++ algorithm)
  - Dissimilarity calculations
  - Strategies to calculate cluster means
- Handling categorical data: *k-modes* (Huang'98)
  - Replacing means of clusters with modes (mode = value that occurs more often)
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters

k-Means overview

- Relatively efficient: O(*tkn*), *n*: # objects, *k*: # clusters, *t:* # iterations. Normally, *k*, *t* << *n*.
  - Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$
- Finds a local optimum
- The choice of initial points can have large influence in the result
- Weaknesses
  - 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
  - Applicable only when mean is defined, then what about categorical data?

From *k*-Means to *k*-Medoids

- The *k*-Means algorithm is sensitive to outliers!
  - 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 objects in a cluster as a reference point, medoids can be used, which are the most centrally located object in the clusters.

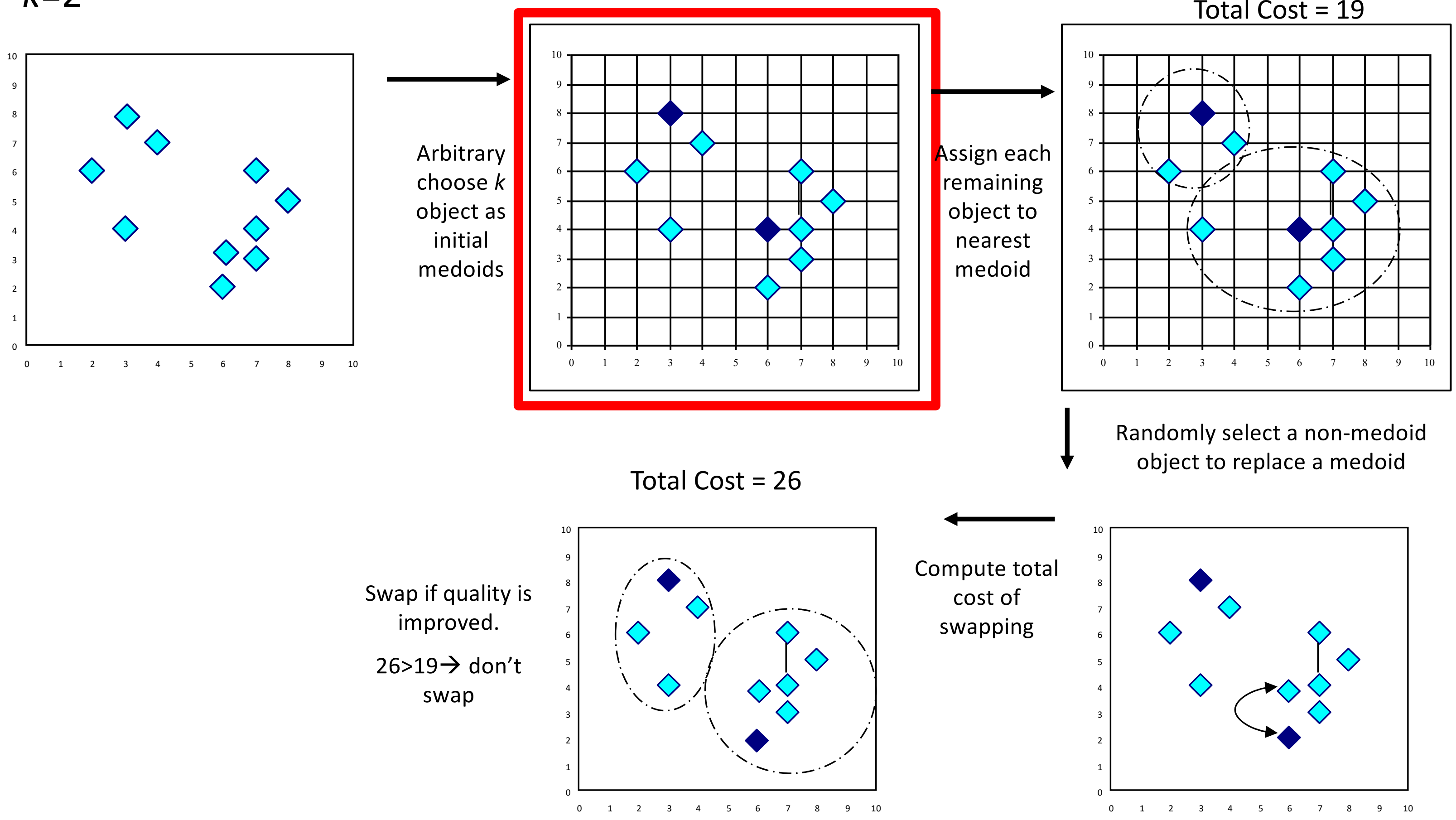


The k-Medoids clustering algorithm

- Clusters are represented by real objects called *medoids*.
- PAM (Partitioning Around Medoids, Kaufman and Rousseeuw, 1987)
  - starts from an initial set of *k* medoids and iteratively replaces one of the medoids by one of the non-medoid poins if such a replacement improves the total clustering cost
- Pseudocode:
  - Select *k* representative objects arbitrarily
  - Assign the rest of the objects to the *k* clusters
  - Representative replacement:
    - For each medoid *m* and each non-medoid object *o* do, check whether *o* could replace *m*
    - Replacement is possible if the clustering cost is improved.
  - Repeat until no improvements can be achieved by any replacement

PAM example:

k=2

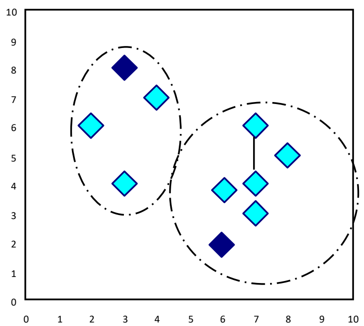


Cost computed using Manhattan distance (L1)

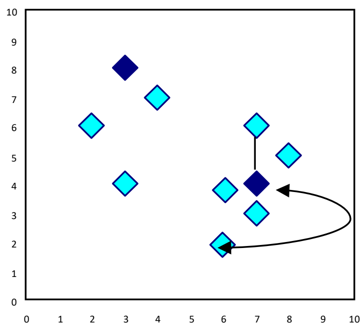
PAM example: swap case

k=2

Total Cost = 26

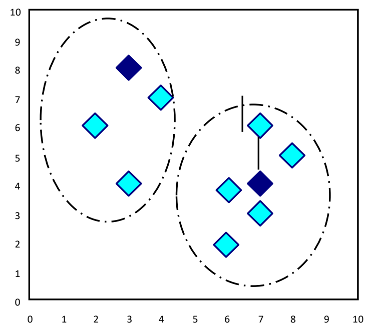


Randomly select a non-medoid object to replace a medoid



Assign each remaining object to nearest medoid

Total Cost = 18



Cost computed using Manhattan distance (L1)

Swap if quality is improved

18<26 → swap

Do loop

Until no change

PAM overview

- Very similar to *k*-Means
- PAM is more robust to outliers comparing to *k*-Means because a medoid is less influenced by outliers or other extreme values than a centroid.
- 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

- Sampling based method:
  - CLARA(Clustering LARge Applications)
  - CLARANS ("Randomized" CLARA)

CLARA (Clustering Large Applications)

- CLARA (Kaufmann and Rousseeuw, 1990)
- It draws multiple samples of the dataset, applies PAM on each sample, and gives the best clustering as the output.
- Strength: deals with larger datasets 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 dataset if the sample is biased

What is the right number of clusters 1/2

- The number of clusters *k* is required as input by the partitioning algorithms. Choosing the right *k* is challenging.
- Silhouette coefficient (Kaufman & Rousseeuw 1990)
  - Let *a(o)* the distance of an object *o* to the representative of its cluster and *b(o)* the distance to the representatives of its "second best" cluster
  - Silhouette *s*(*o*) of an object *o:*

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

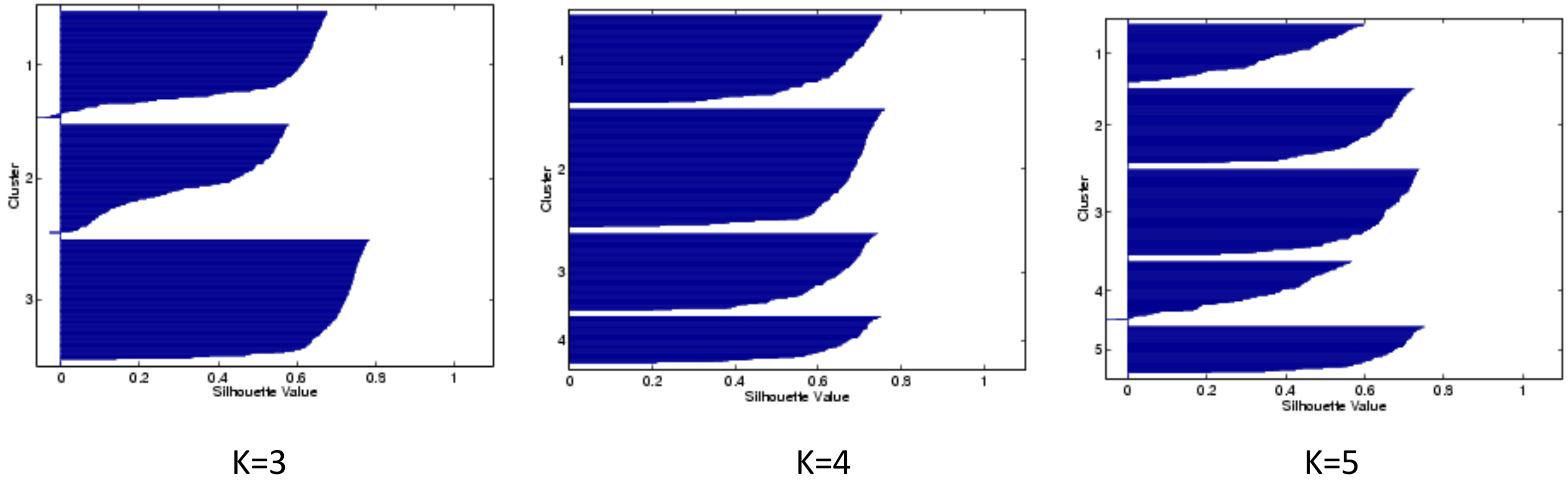
$-1 \leq s(o) \leq +1$

s(o) ~ -1 / 0 / +1 : bad / indifferent / good assignment

  - s(o)~1→ *a*(*o*) < < *b*(*o*). Small *a*(*o*) means it is well matched to its own cluster. Large b(o) means is badly matched to its neighbouring cluster.
  - s(o)~-1→ the neighbor cluster seems more appropriate
  - s(o)~0→ in the border between two natural clusters

What is the right number of clusters 2/2

- The Silhouette coefficient of a cluster is the avg silhouette of all its objects
  - Is a measure of how tightly grouped all the data in the cluster are.
  - > 0,7: strong structure, > 0,5: usable structure ….
- The Silhouette coefficient of a clustering is the avg silhouette of all objects
  - is a measure of how appropriately the dataset has been clustered



Outline

- Unsupervised learning vs supervised learning
- A categorization of major clustering methods
- Partitioning-based clustering
- Hierarchical-based clustering