



Clustering: Partitioning Methods

Data Mining and Text Mining (UIC 583 @ Politecnico di Milano)

- ❑ Partition-based clustering

- ❑ K-means
 - ▶ The algorithm
 - ▶ Selection of initial Centroids
 - ▶ Limitation

- ❑ K-medoids

- ❑ Construct a partition of a data set containing **n objects** into a set of **k clusters**, so to minimize a criterion (e.g., sum of squared distance)
- ❑ The goal is, given a k , find a partition of k clusters that optimizes the chosen partitioning criterion
- ❑ Global optimal: exhaustively enumerate all partitions
- ❑ Heuristic methods: k-means, k-medoids, etc.

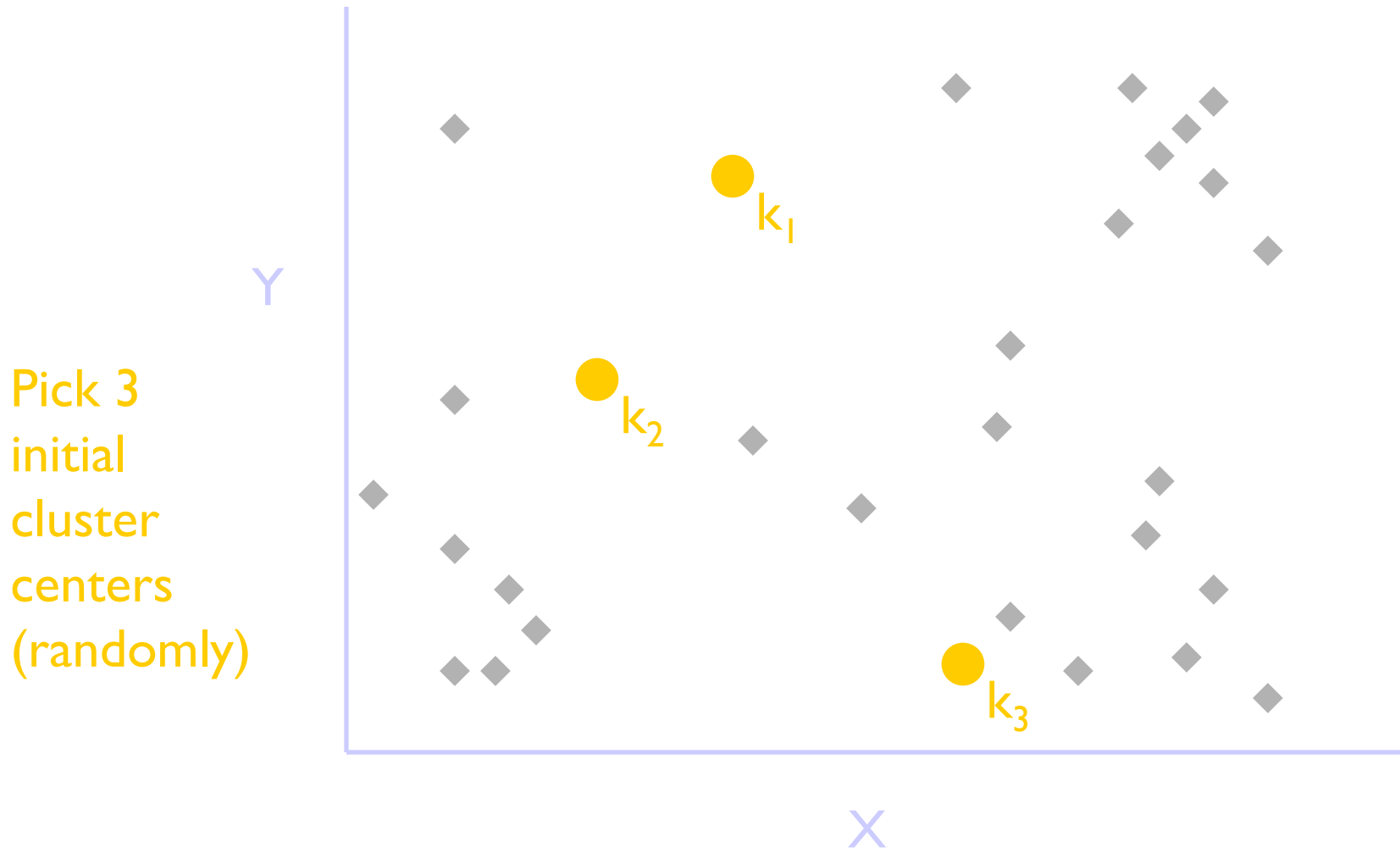
K-means

1. Pick a number (K) of cluster centers (at random)
2. Assign every item to its nearest cluster center (e.g. using Euclidean distance)
3. Move each cluster center to the mean of its assigned items
4. Repeat steps 2,3 until convergence (change in cluster assignments less than a threshold)

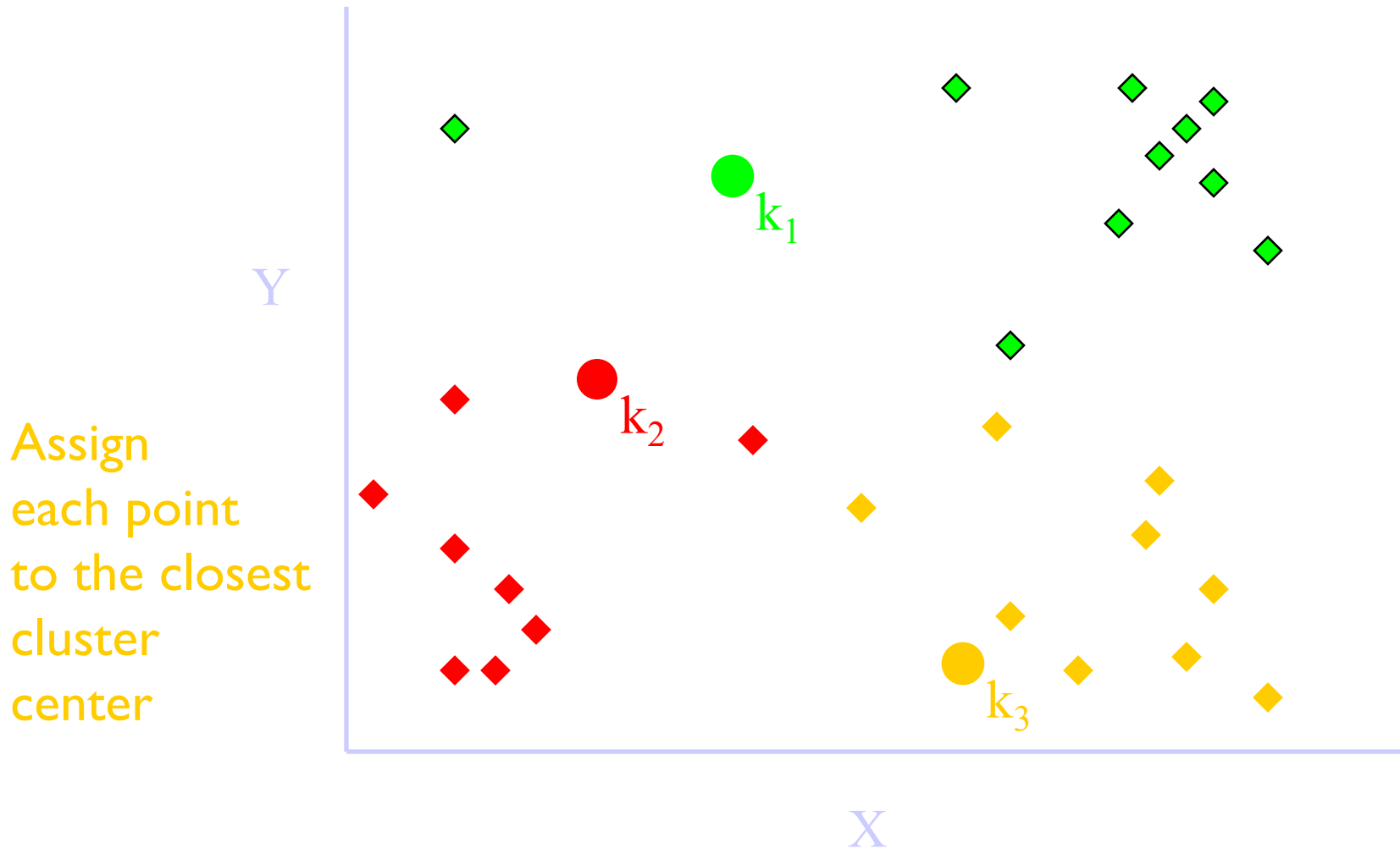
- ❑ Partitional clustering approach
- ❑ Each cluster is associated with a **centroid** (center point)
- ❑ Each point is assigned to the cluster with the closest centroid
- ❑ Number of clusters, K , must be specified
- ❑ The basic algorithm is very simple

-
- 1: Select K points as the initial centroids.
 - 2: **repeat**
 - 3: Form K clusters by assigning all points to the closest centroid.
 - 4: Recompute the centroid of each cluster.
 - 5: **until** The centroids don't change
-

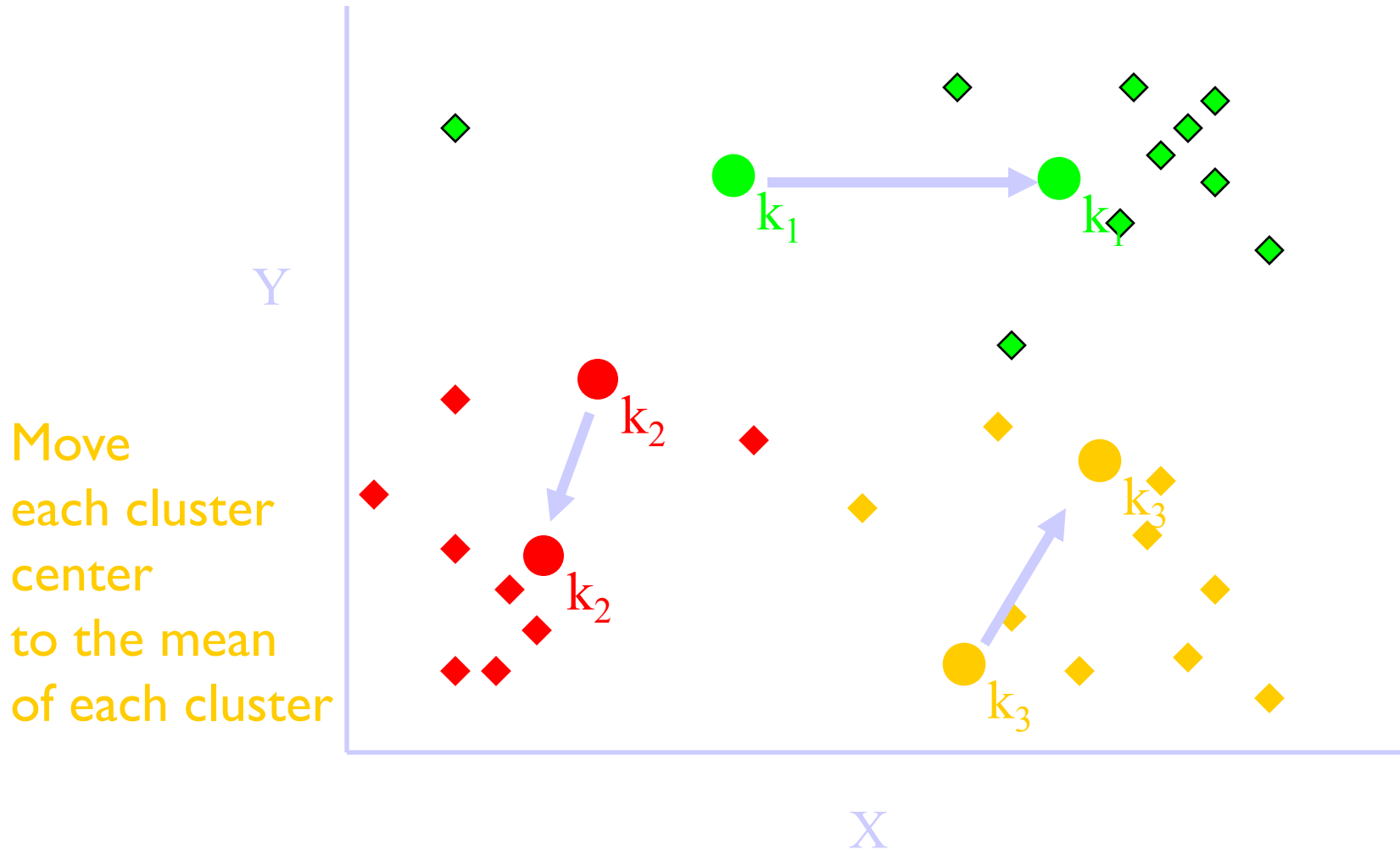
K-means example (step 1)

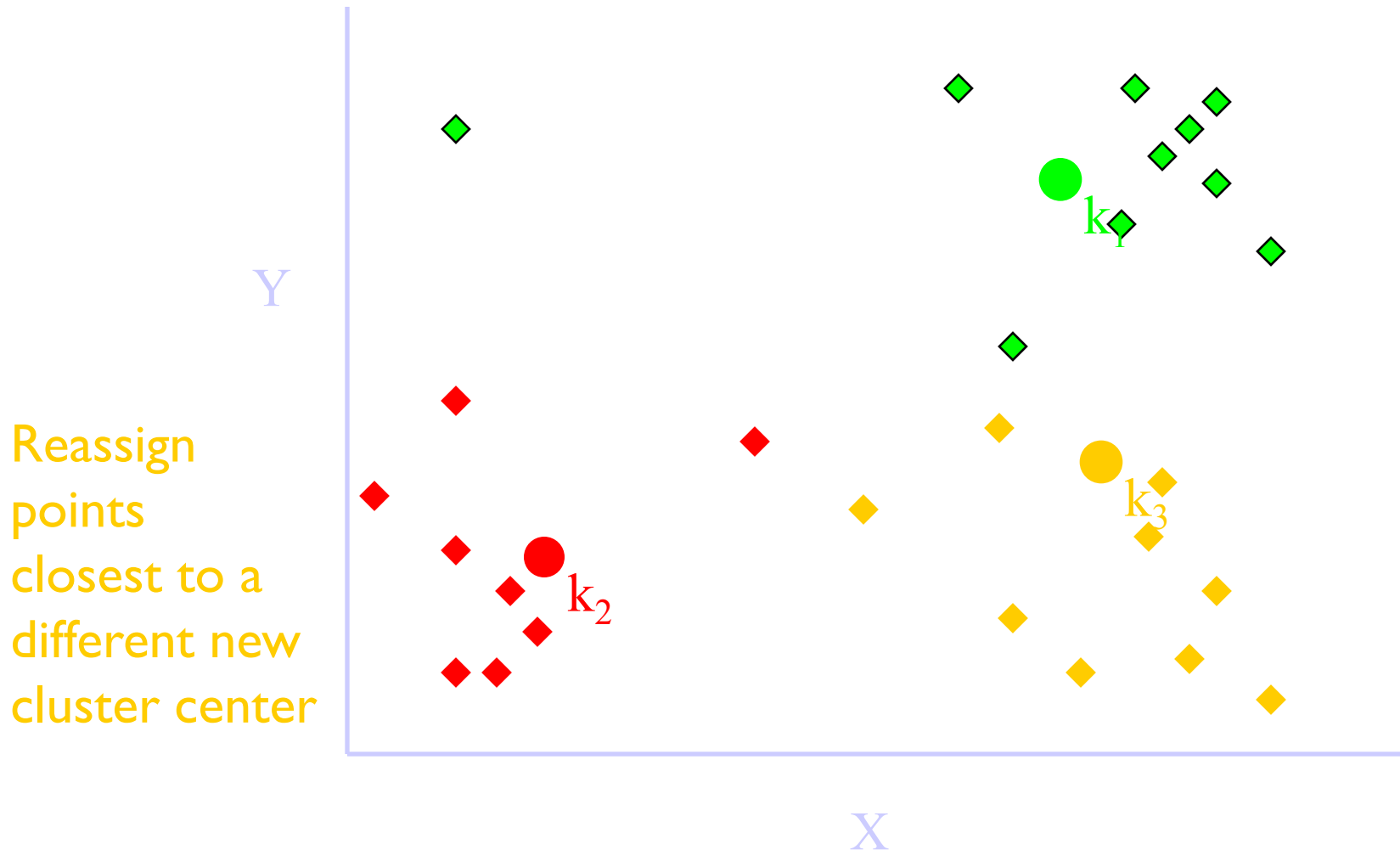


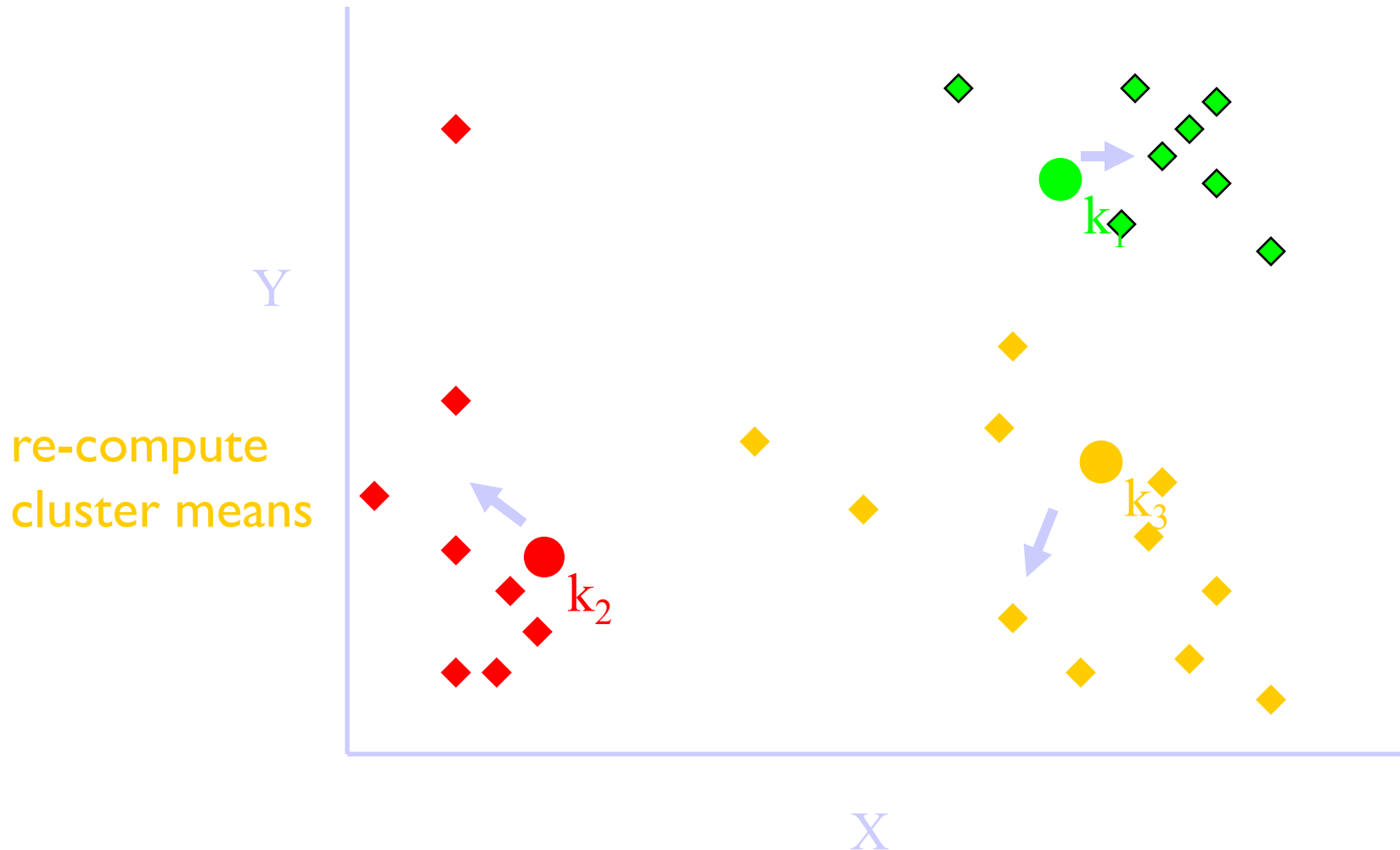
K-means example (step 2)

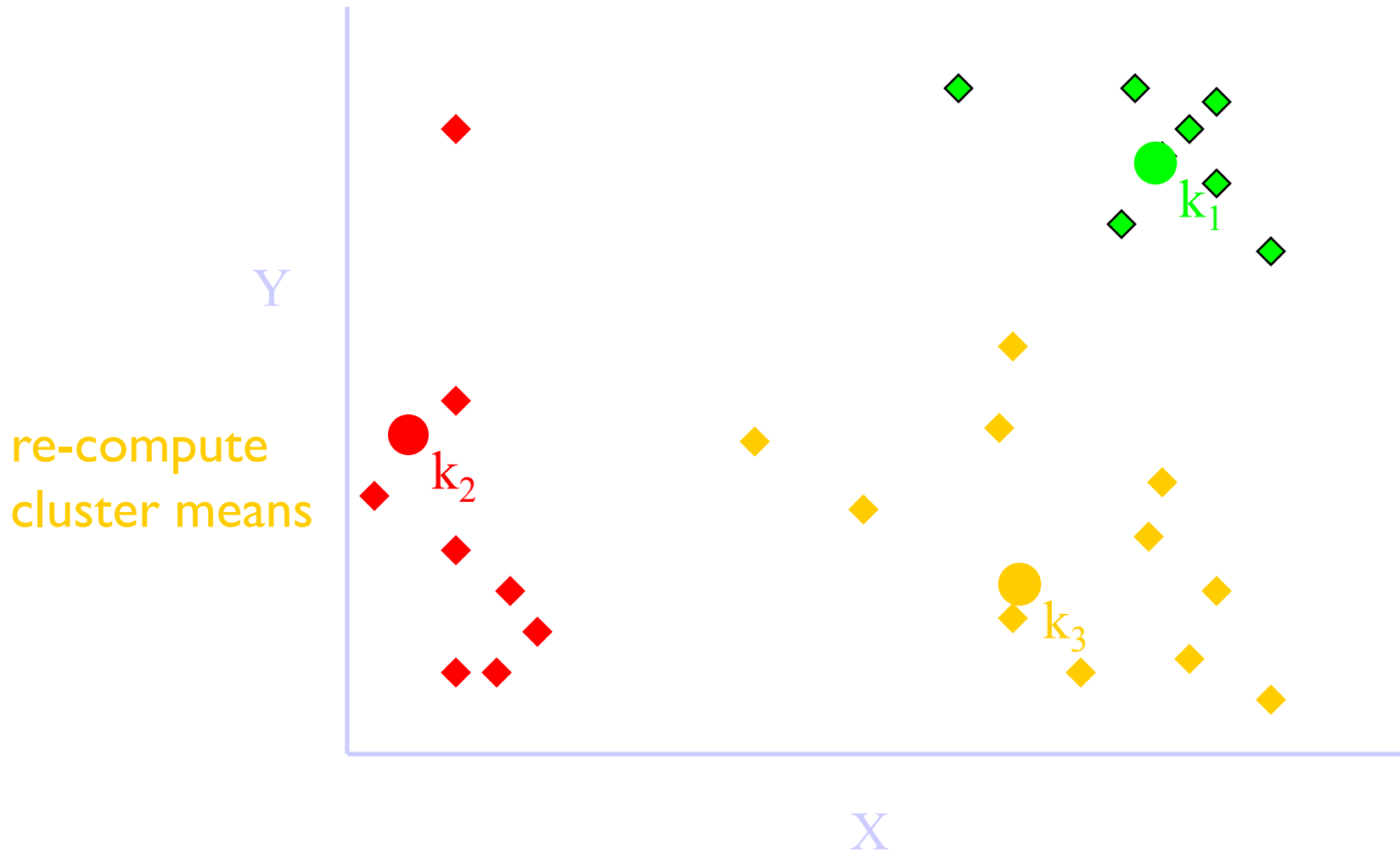


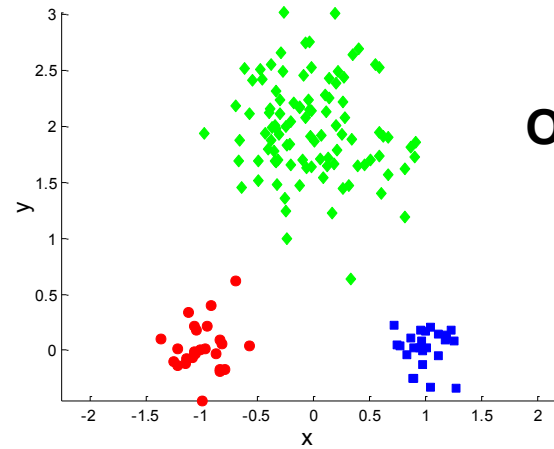
K-means example (step 3)



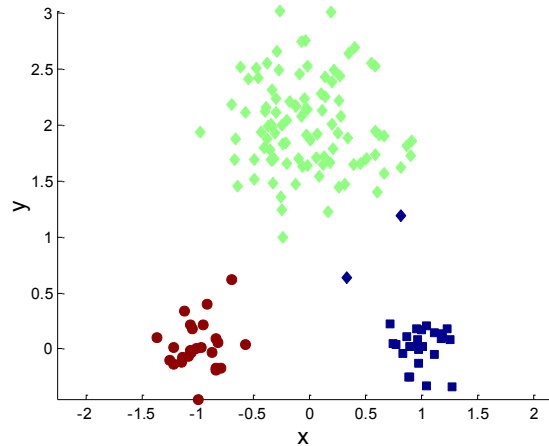




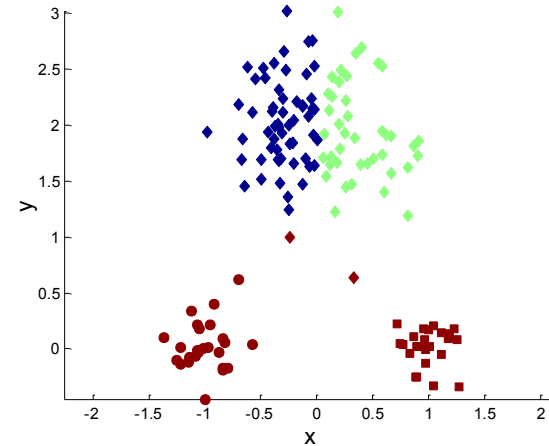




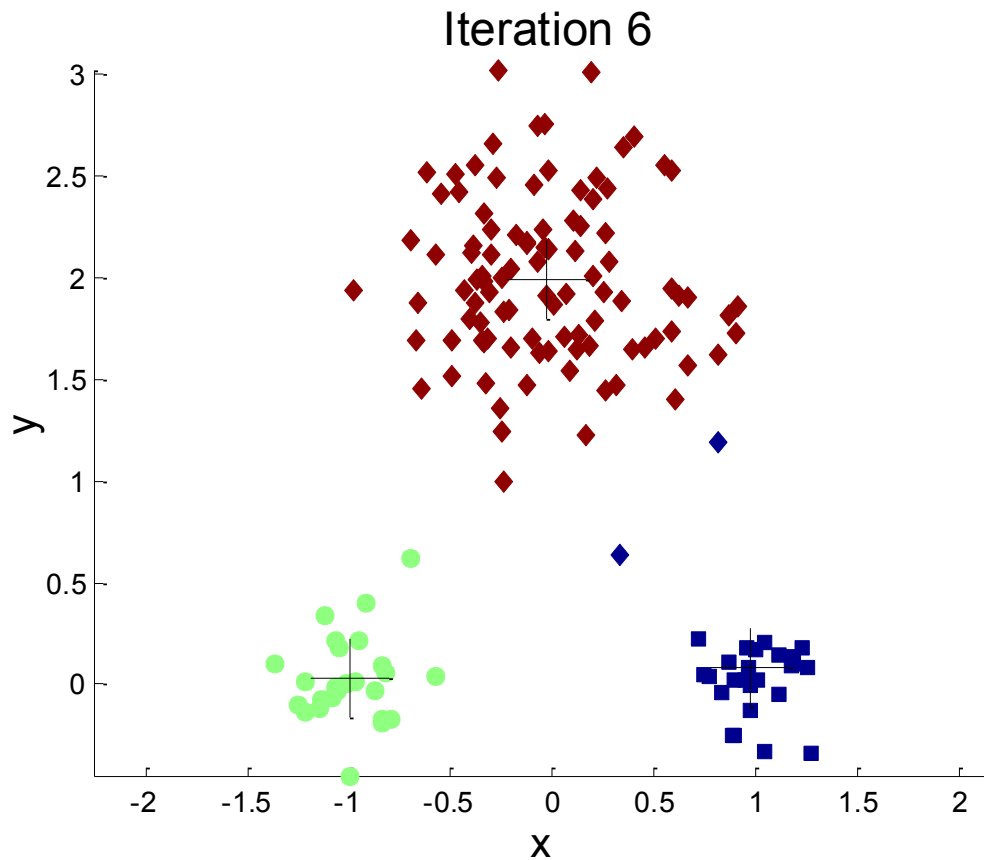
Original Points

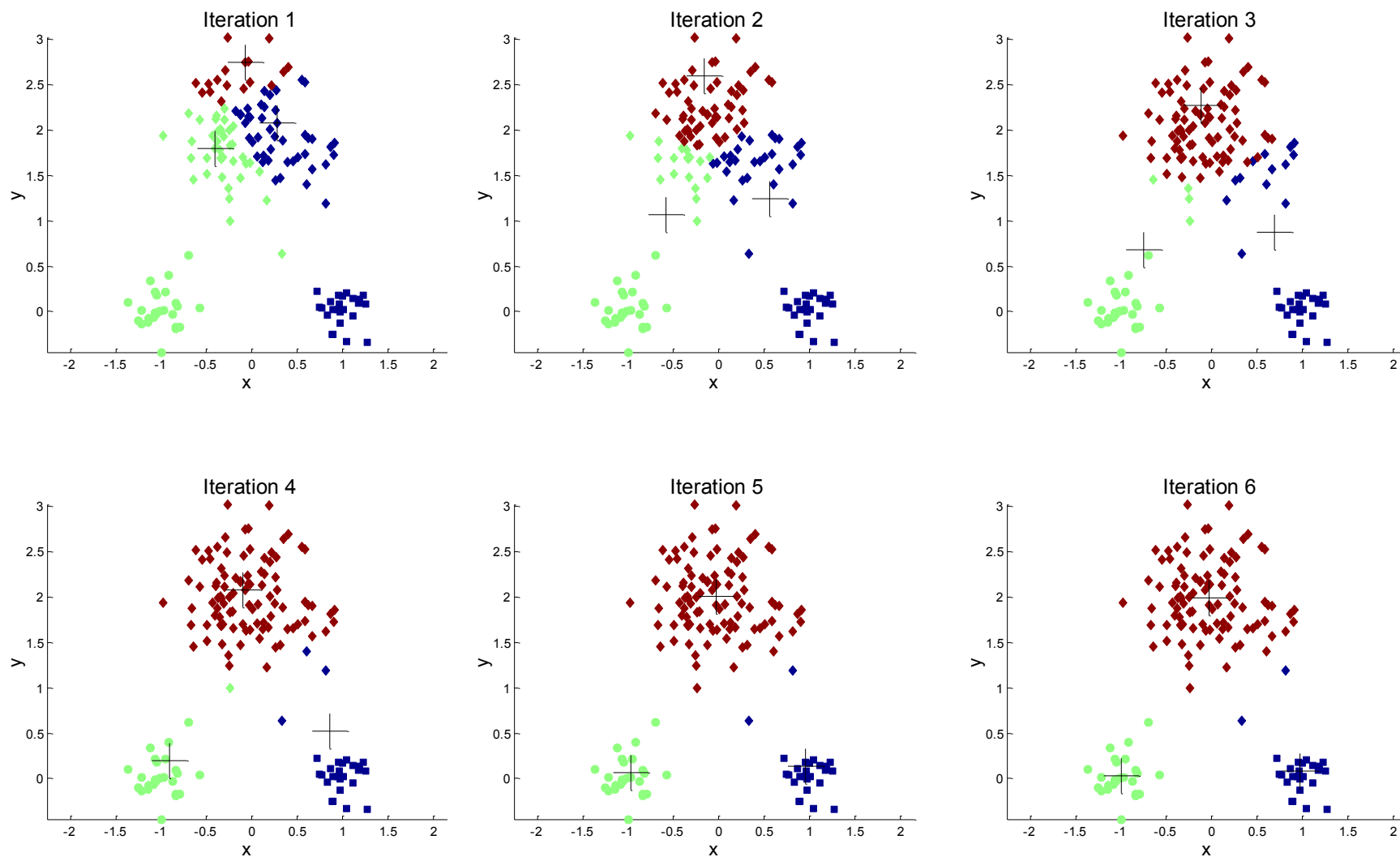


Optimal Clustering



Sub-optimal Clustering

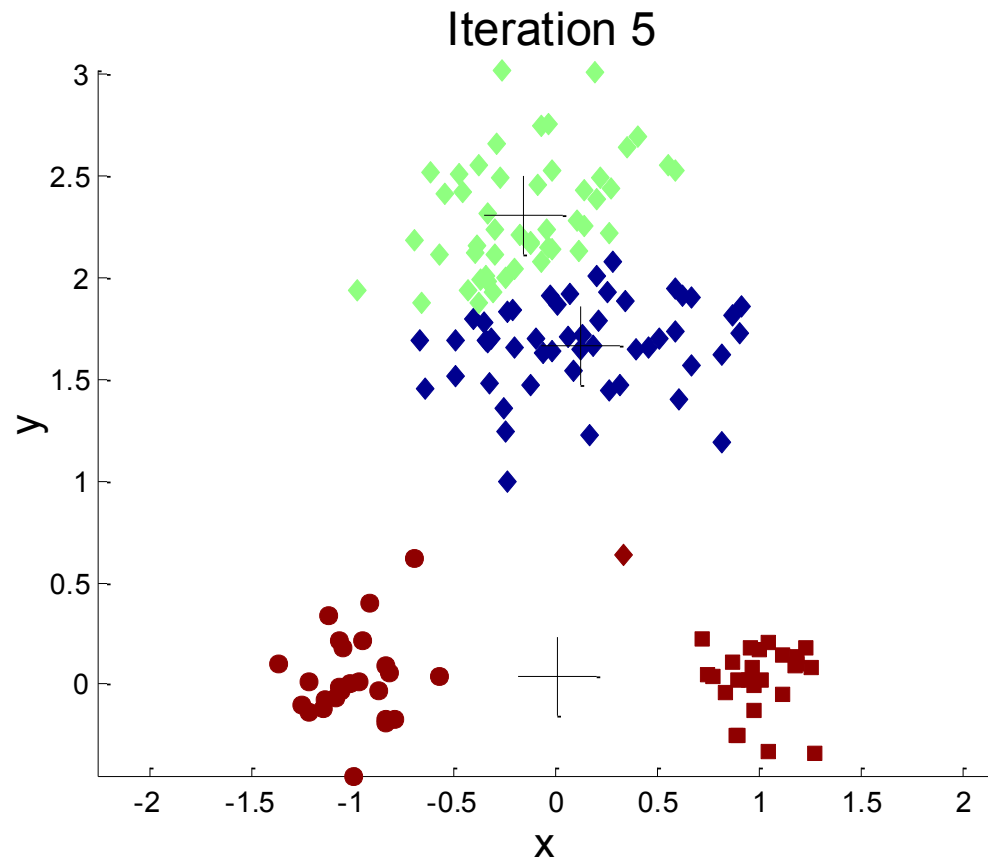


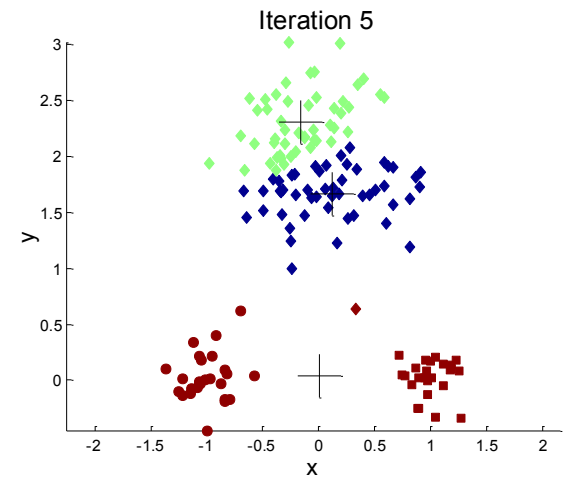
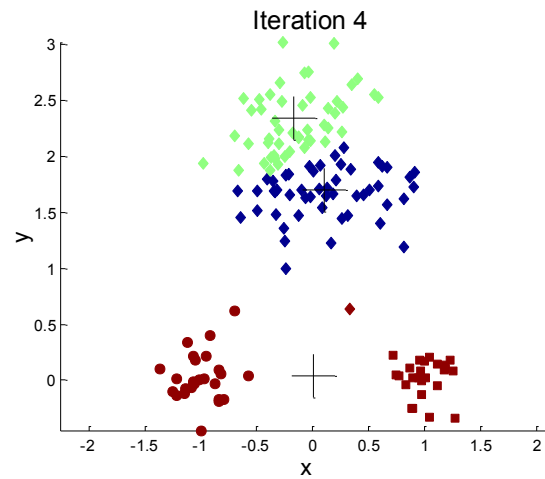
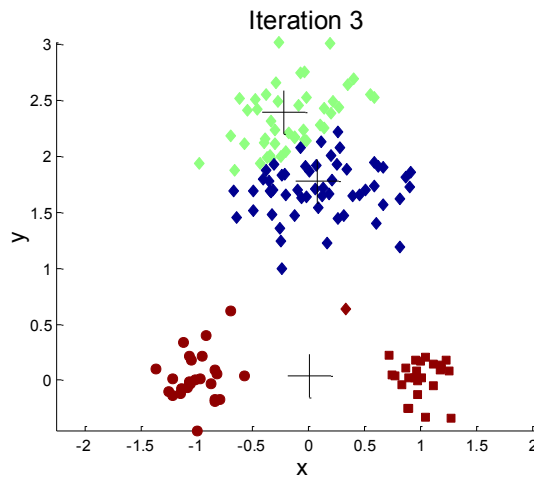
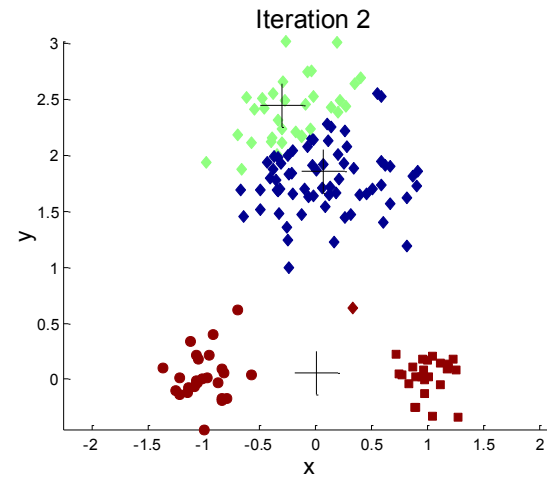
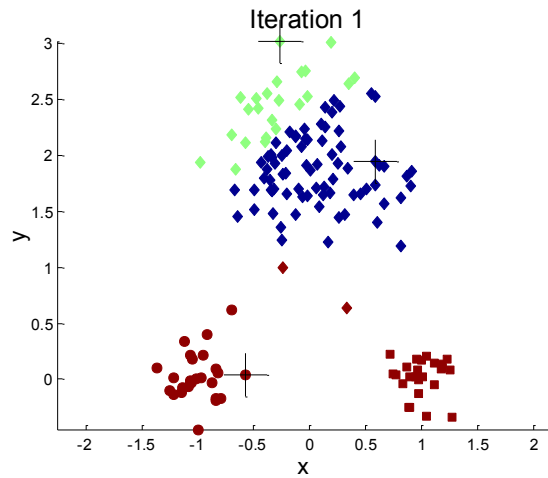


- ❑ Most common measure is the sum of squared error (SSE)
- ❑ For each point, the error is the distance to the nearest cluster
- ❑ To get SSE, we square these errors and sum them.

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

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

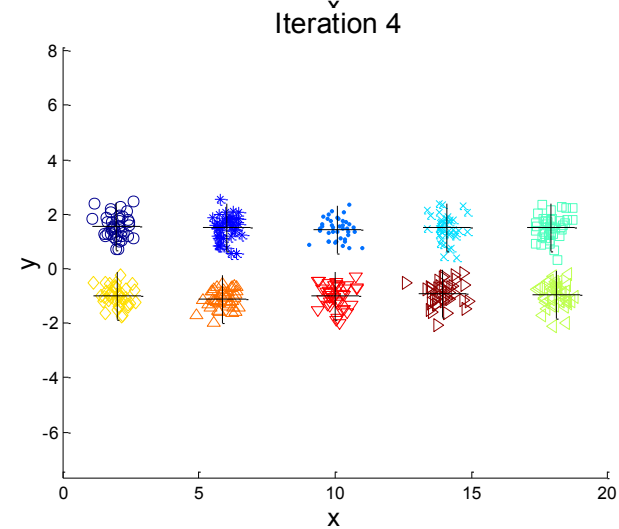
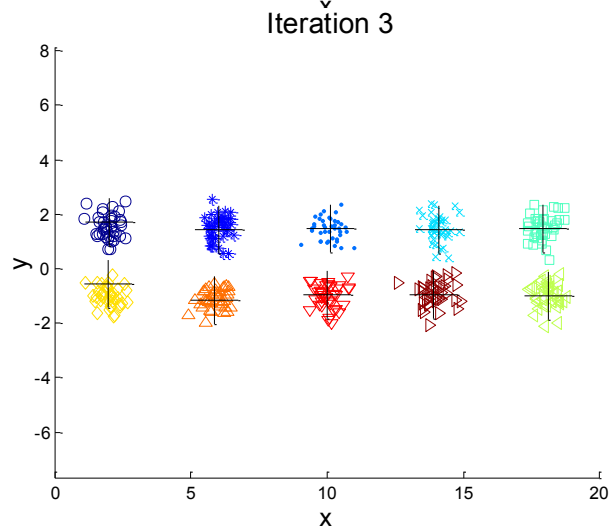
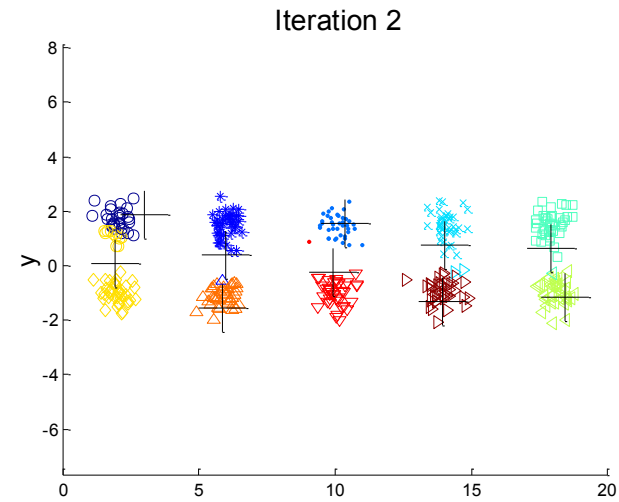
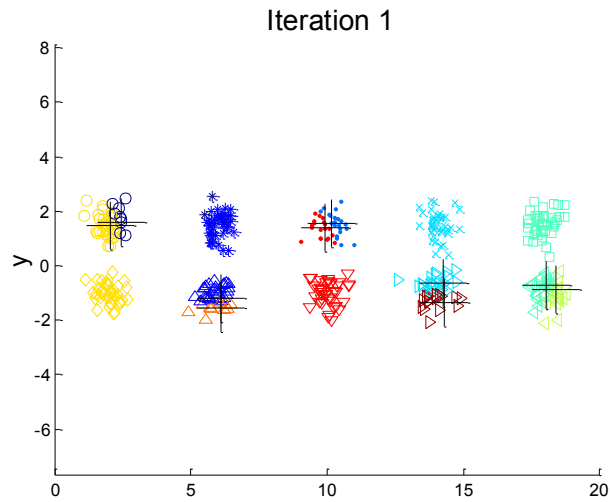




- ❑ If there are K 'real' clusters then the chance of selecting one centroid from each cluster is small.
- ❑ Chance is relatively small when K is large
- ❑ If clusters are the same size, n , then

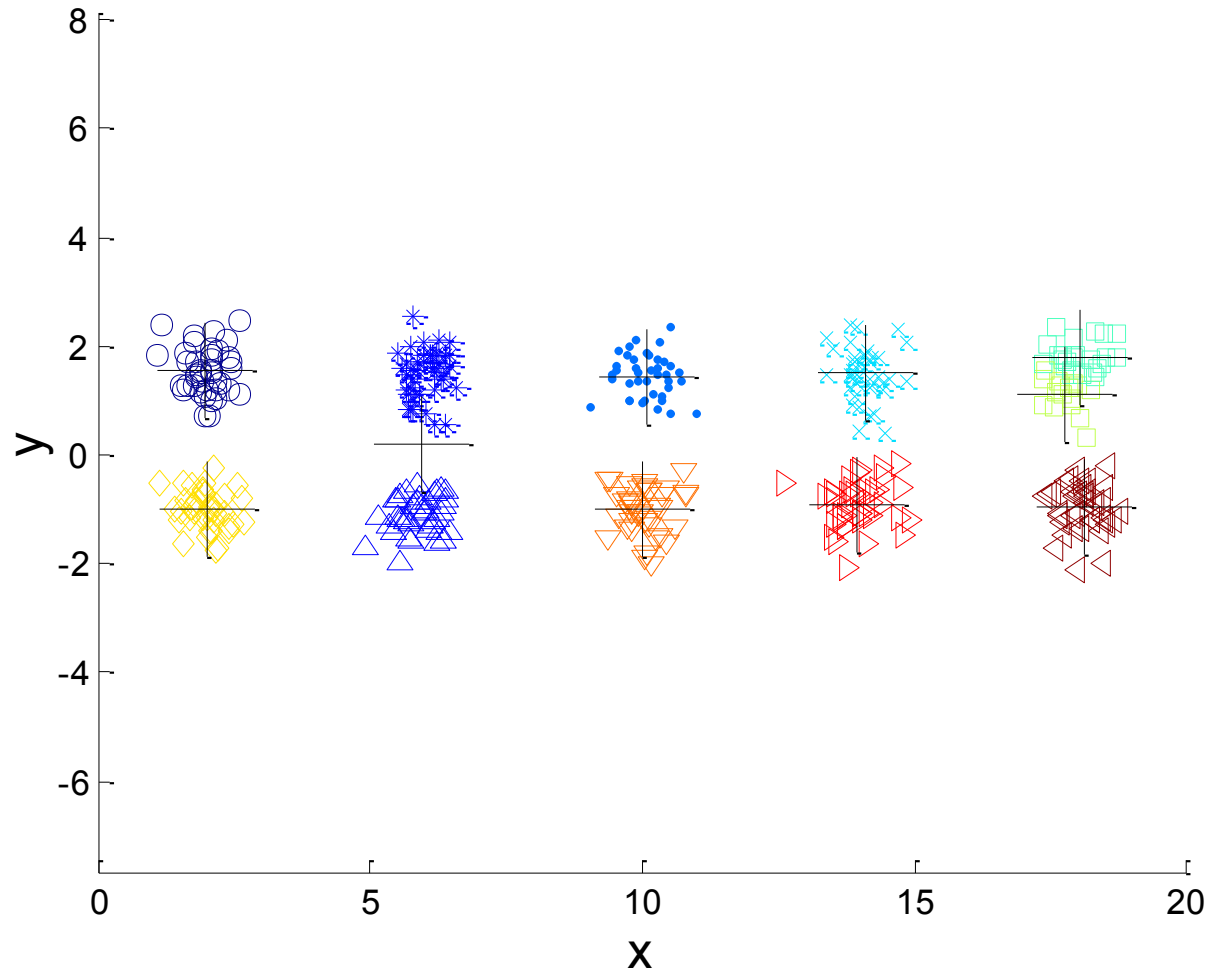
$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

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

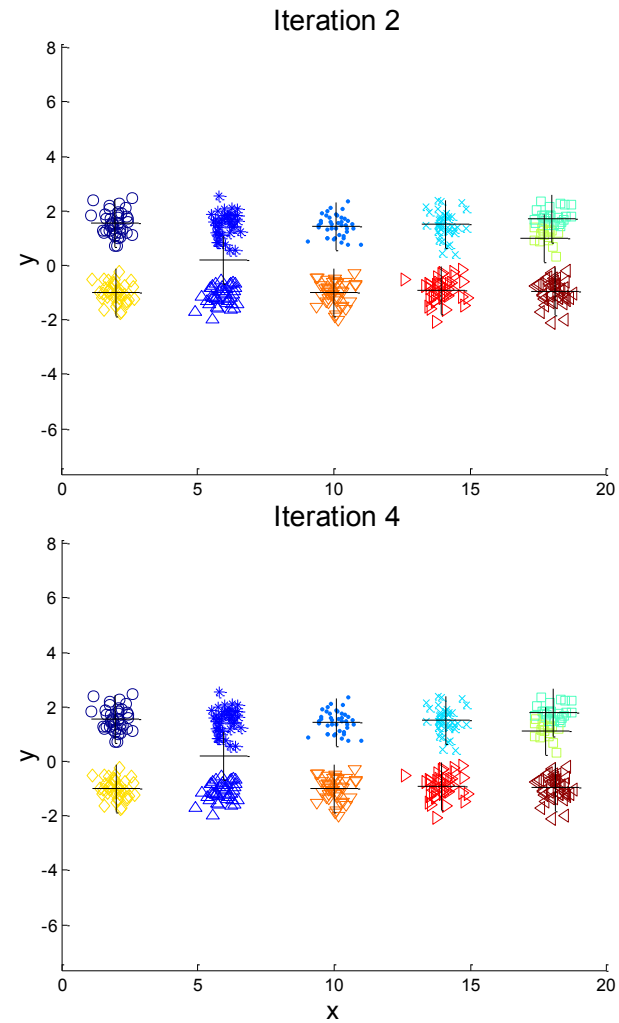
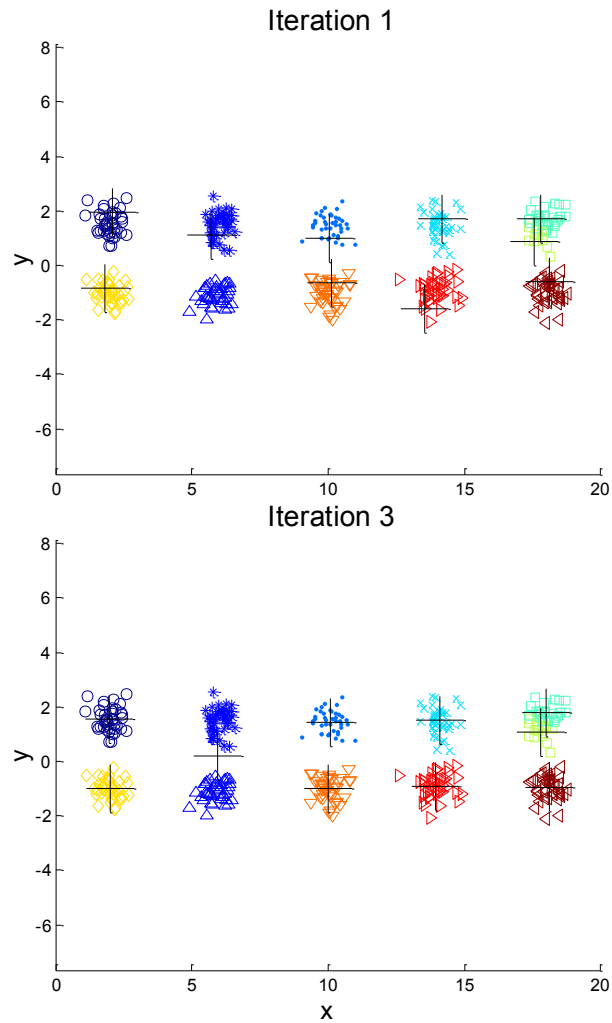


Starting with two initial centroids in one cluster of each pair of clusters

Iteration 4

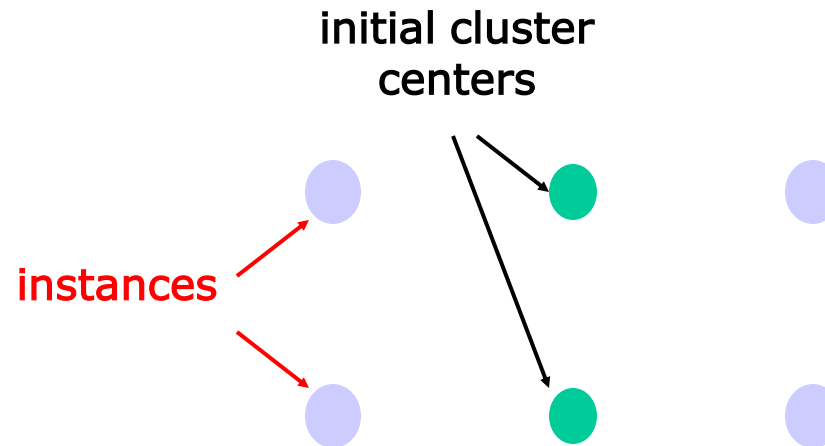


Starting with some pairs of clusters having three initial centroids, while other have only one.



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

- ❑ Result can vary significantly depending on initial choice of seeds
- ❑ Can get trapped in local minimum



- ❑ To increase chance of finding global optimum: restart with different random seeds

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

- ❑ Multiple runs
 - ▶ Helps, but probability is not on your side
- ❑ Sample and use hierarchical clustering to determine initial centroids
- ❑ Select more than k initial centroids and then select among these initial centroids
 - ▶ Select most widely separated
- ❑ Postprocessing
- ❑ Bisecting K-means
 - ▶ Not as susceptible to initialization issues

- ❑ Basic K-means algorithm can yield empty clusters

- ❑ Several strategies
 - ▶ Choose the point that contributes most to SSE
 - ▶ Choose a point from the cluster with the highest SSE
 - ▶ If there are several empty clusters, the above can be repeated several times.

- ❑ In the basic K-means algorithm, centroids are updated after all points are assigned to a centroid

- ❑ An alternative is to update the centroids after each assignment (incremental approach)
 - ▶ Each assignment updates zero or two centroids
 - ▶ More expensive
 - ▶ Introduces an order dependency
 - ▶ Never get an empty cluster
 - ▶ Can use “weights” to change the impact

□ Pre-processing

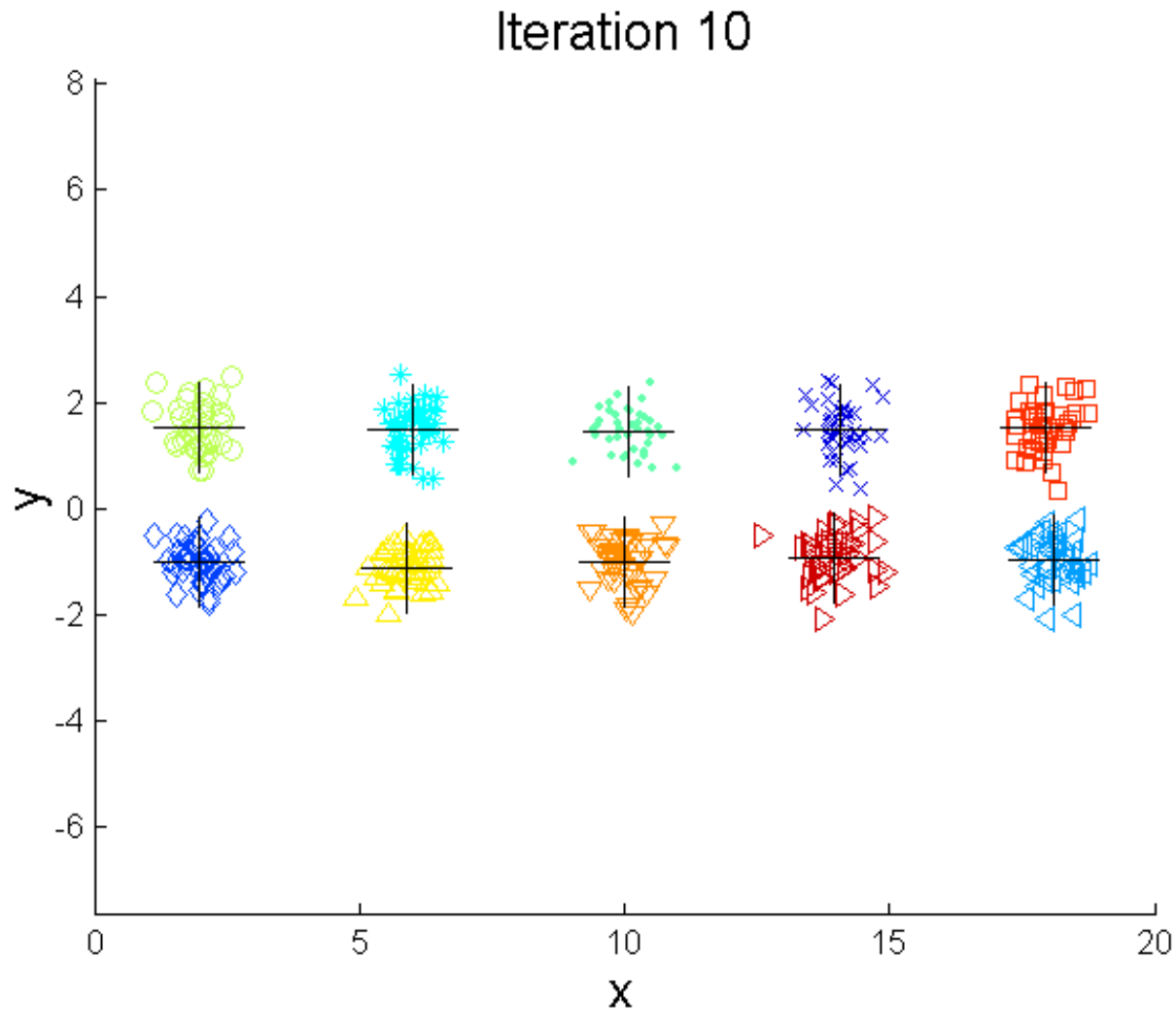
- ▶ Normalize the data
- ▶ Eliminate outliers

□ Post-processing

- ▶ Eliminate small clusters that may represent outliers
- ▶ Split 'loose' clusters, i.e., clusters with relatively high SSE
- ▶ Merge clusters that are 'close' and that have relatively low SSE
- ▶ These steps can be used during the clustering process

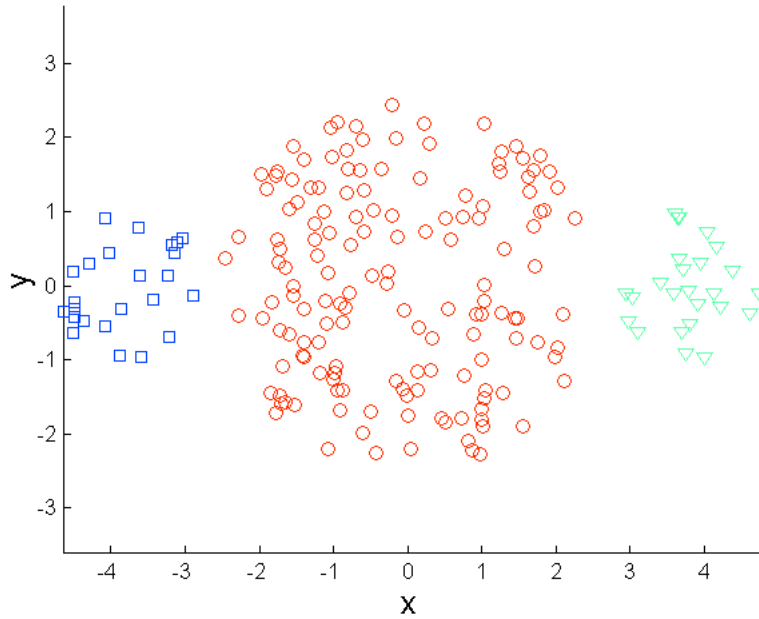
Variant of K-means that can produce a partitional or a hierarchical clustering

-
- 1: Initialize the list of clusters to contain the cluster containing all points.
 - 2: **repeat**
 - 3: Select a cluster from the list of clusters
 - 4: **for** $i = 1$ to *number_of_iterations* **do**
 - 5: Bisect the selected cluster using basic K-means
 - 6: **end for**
 - 7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
 - 8: **until** Until the list of clusters contains K clusters
-

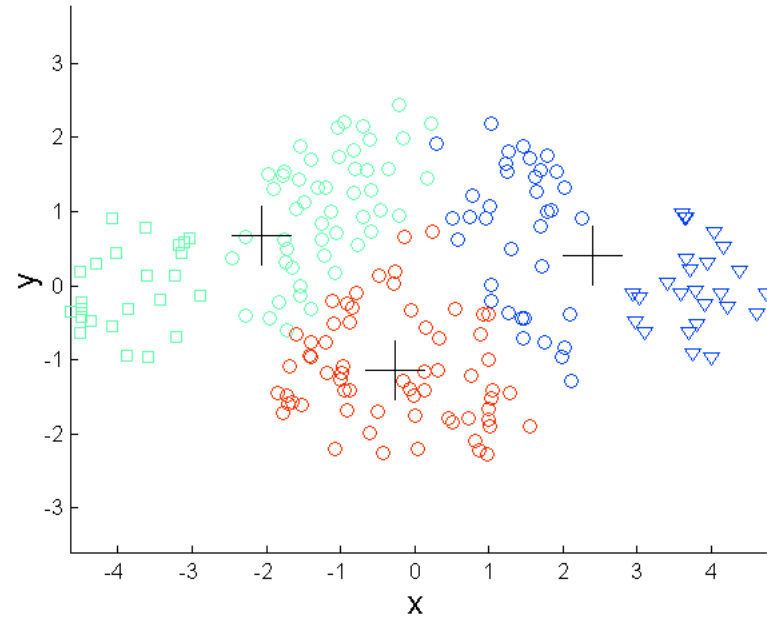


- ❑ K-means has problems when clusters are of differing
 - ▶ Sizes
 - ▶ Densities
 - ▶ Non-globular shapes

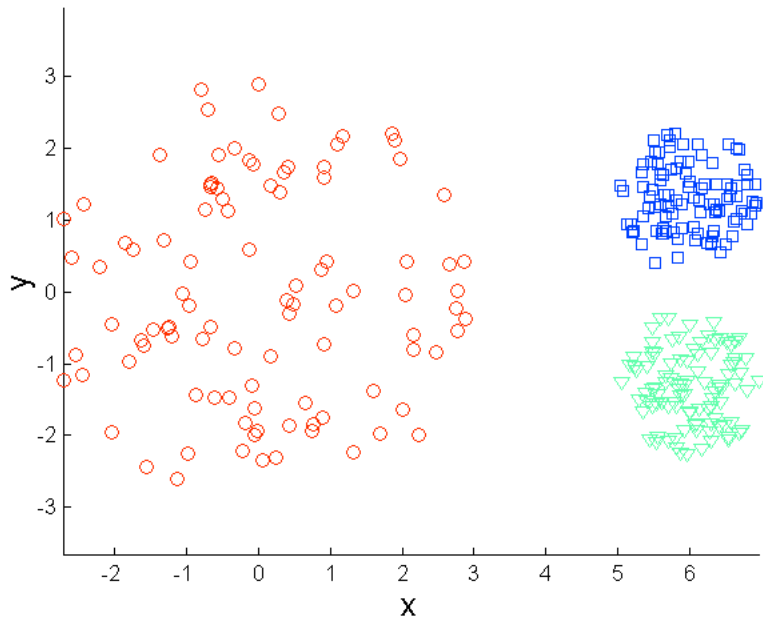
- ❑ K-means has problems when the data contains outliers.



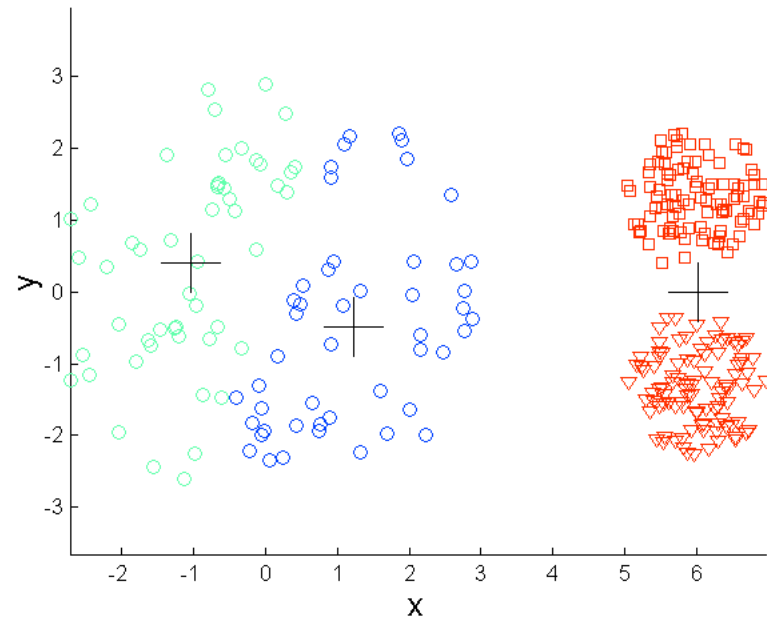
Original Points



K-means (3 Clusters)

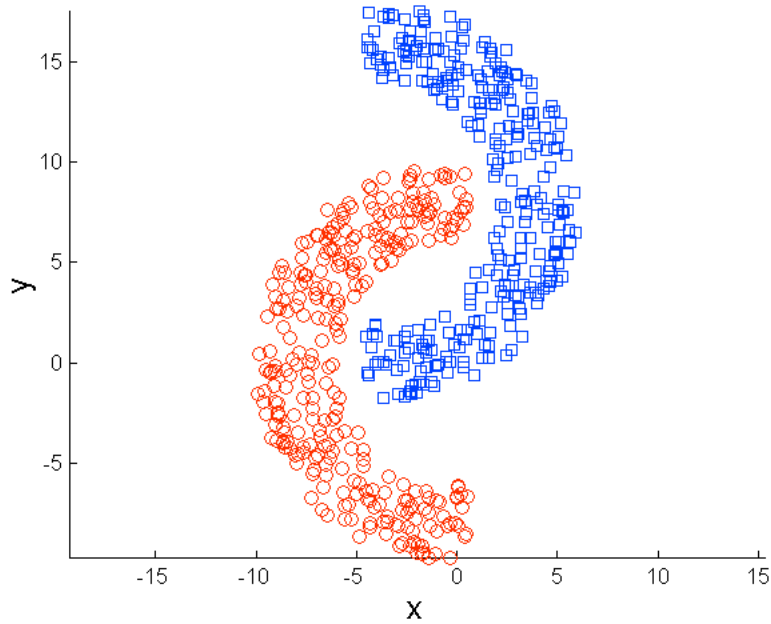


Original Points

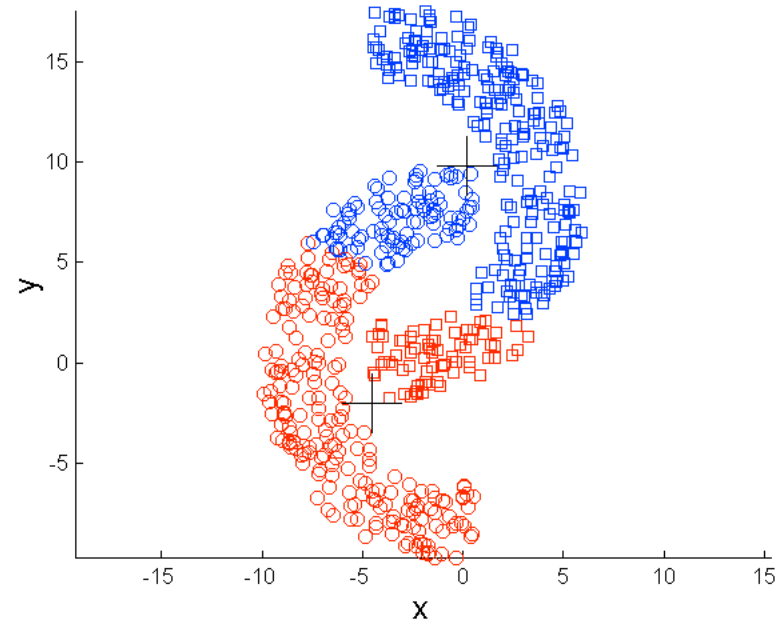


K-means (3 Clusters)

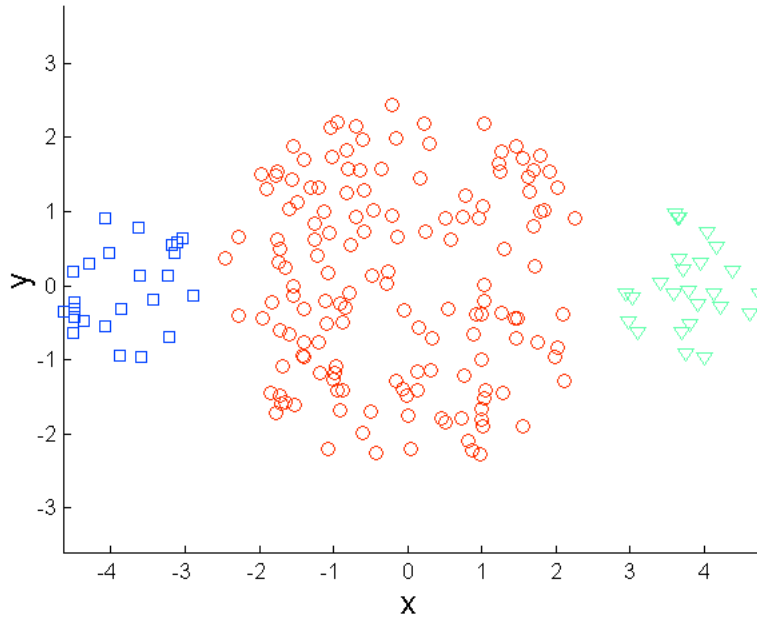
Limitations of K-means: Non-globular Shapes



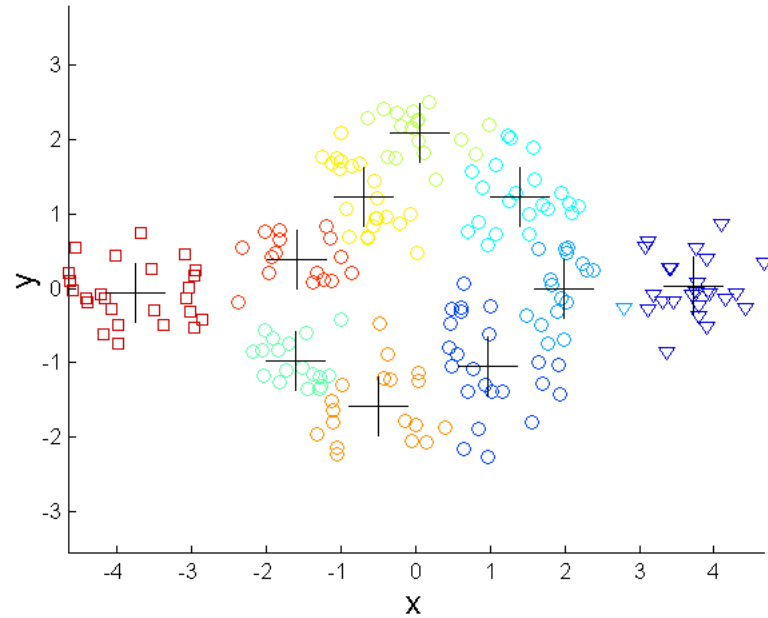
Original Points



K-means (2 Clusters)

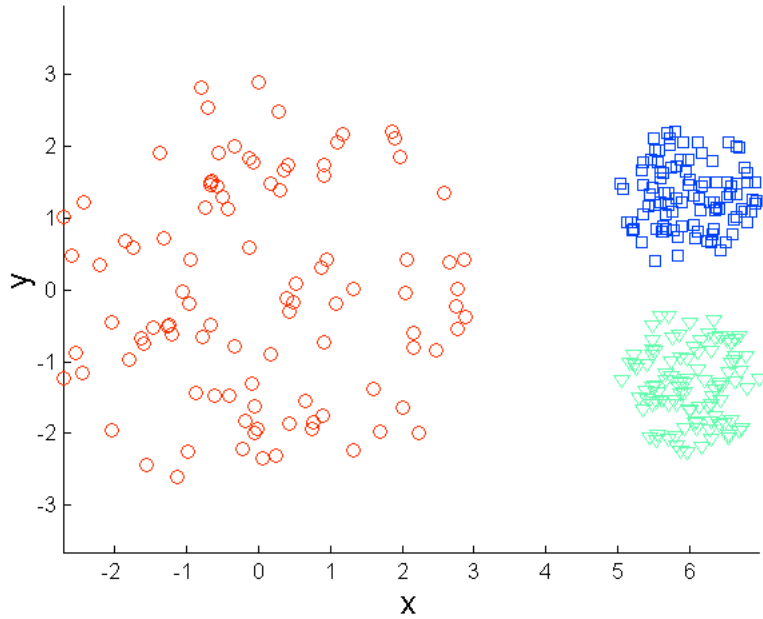


Original Points

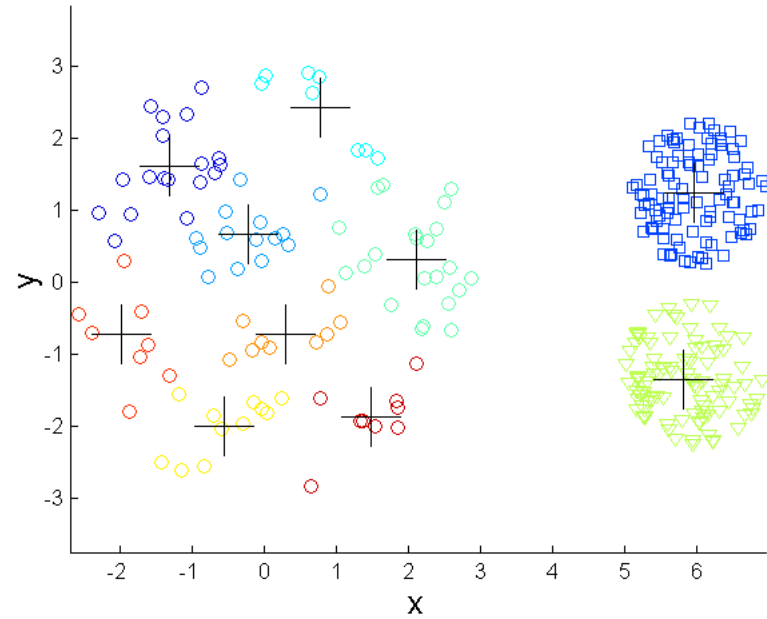


K-means Clusters

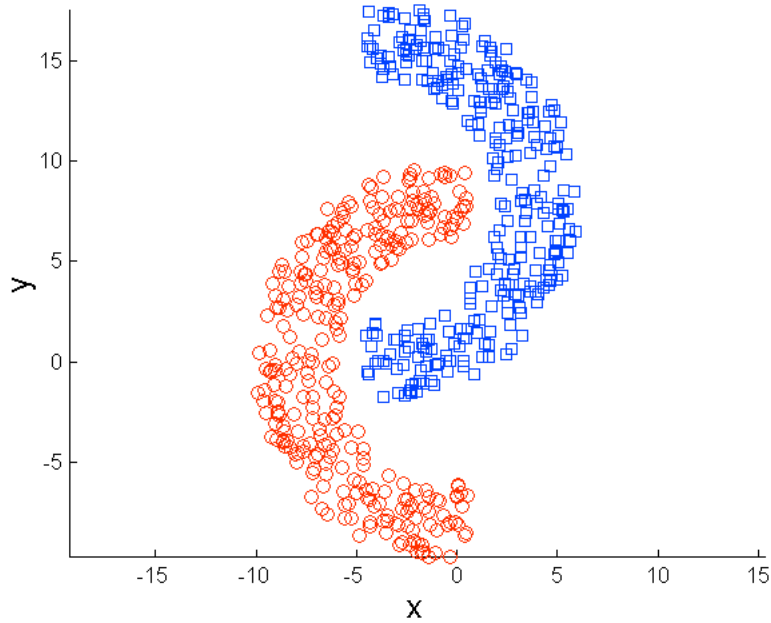
One solution is to use many clusters.
Find parts of clusters, but need to put together.



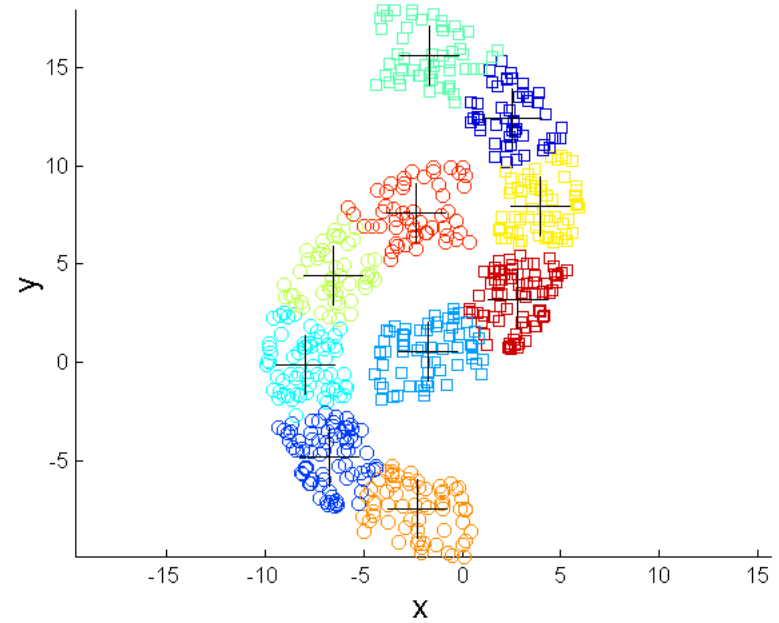
Original Points



K-means Clusters



Original Points



K-means Clusters

Summary of k-means

- ❑ Advantages
 - ▶ Simple, understandable
 - ▶ Items automatically assigned to clusters
- ❑ Disadvantages
 - ▶ Must pick number of clusters before hand
 - ▶ All items forced into a cluster
 - ▶ Too sensitive to outliers

- ❑ Strength: Relatively efficient
 - ▶ Complexity is $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
 - ▶ Comparing: PAM: $O(k(n-k)^2)$, CLARA: $O(ks^2 + k(n-k))$

- ❑ 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

- ❑ A few variants of the k-means which differ in
 - ▶ Selection of the initial k means
 - ▶ Dissimilarity calculations
 - ▶ Strategies to calculate cluster means

- ❑ Handling categorical data: k-modes
 - ▶ 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

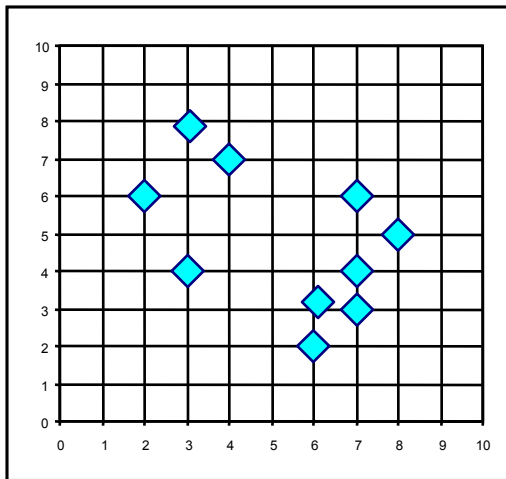
k-medoids

- ❑ Instead of mean, use medians of each cluster
 - ▶ Mean of 1, 3, 5, 7, 9 is 5
 - ▶ Mean of 1, 3, 5, 7, 1009 is 205
 - ▶ Median of 1, 3, 5, 7, 1009 is 5
 - ▶ Median is not affected by extreme values
- ❑ For large databases, use sampling

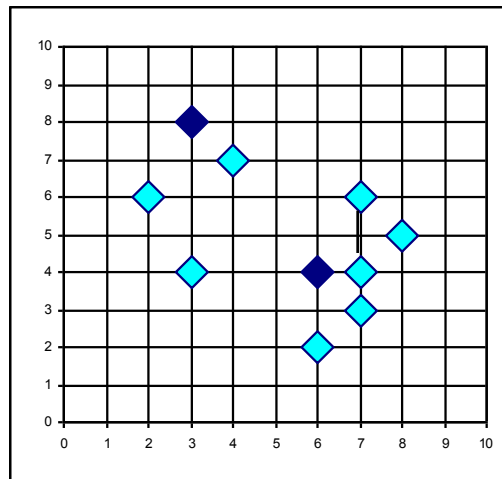
- ❑ Find representative objects, called medoids, in clusters
- ❑ PAM (Partitioning Around Medoids, 1987)
 - ▶ Starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
 - ▶ Works effectively for small data sets, but does not scale well for large data sets

- ❑ References
 - ▶ CLARA (Kaufmann & Rousseeuw, 1990)
 - ▶ CLARANS (Ng & Han, 1994): Randomized sampling
 - ▶ Focusing + spatial data structure (Ester et al., 1995)

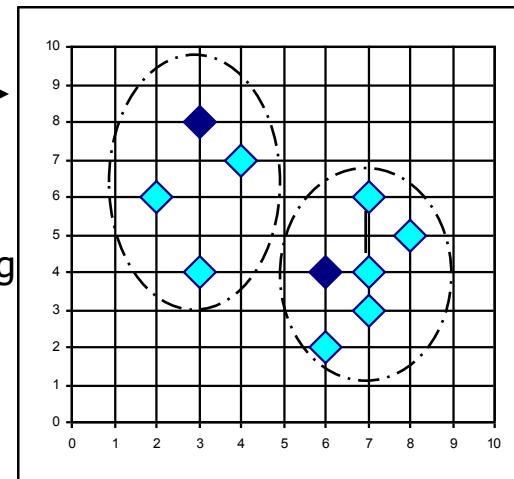
A Typical K-Medoids Algorithm (PAM)



Arbitrary
choose k
object as
initial
medoids



Assign
each
remaining
object to
nearest
medoids



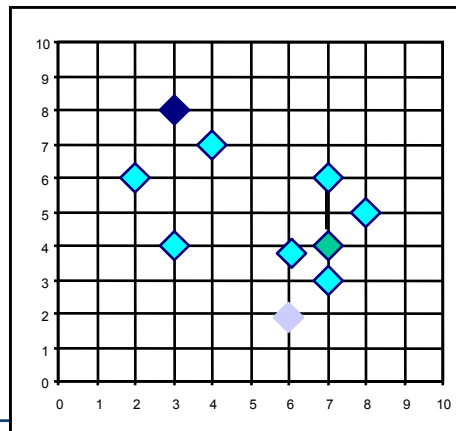
Total Cost = 20

K=2

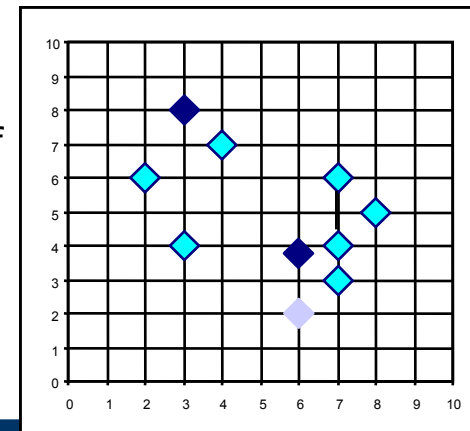
**Do loop
Until no
change**

Swapping O
and O_{random}
If quality is
improved.

Total Cost = 26



Compute
total cost of
swapping



- ❑ Use real object to represent the cluster
- ❑ Select k representative objects arbitrarily
- ❑ For each pair of non-selected object h and selected object i , calculate the total swapping cost TC_{ih}
- ❑ For each pair of i and h ,
 - ▶ If $TC_{ih} < 0$, i is replaced by h
 - ▶ Then assign each non-selected object to the most similar representative object
- ❑ repeat steps 2-3 until there is no change

- ❑ 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 data, k is # of clusters
- ❑ Sampling based method,
CLARA(Clustering LARge Applications)

- ❑ Developed by Kaufmann and Rousseeuw in 1990
- ❑ 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

- ❑ CLARANS stands for Clustering Algorithm based on RANdomized Search
- ❑ CLARANS draws sample of neighbors dynamically
- ❑ The clustering process can be presented as searching a graph where every node is a potential solution, that is, a set of k medoids
- ❑ If the local optimum is found, CLARANS starts with new randomly selected node in search for a new local optimum
- ❑ It is more efficient and scalable than both PAM and CLARA
- ❑ Focusing techniques and spatial access structures may further improve its performance (Ester et al.'95)

Summary

- ❑ Construct a partition of a data set containing n objects into a set of k clusters, so to minimize a criterion

- ❑ The k-means, given the number of clusters k , works as follow
 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

- ❑ What issues? Selection of the initial clusters, cluster sizes, densities, non-globular shapes, etc.

- ❑ K-medoids approaches use medians instead of means