COMP7/8118 M50

Data Mining

K-Means Clustering

Xiaofei Zhang

Slides compiled from Jiawei Han and Raymond C.W. Wong's work

THE UNIVERSITY OF MEMPHIS

Outline

- K-means Clustering
  - Original K-means Clustering
  - Sequential K-means Clustering
  - Forgetful Sequential K-means Clustering

K-means Clustering

- 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 objective is to minimize the sum of distances of the points to their respective centroid

K-means Clustering

- **Problem:** Given a set X of n points in a d-dimensional space and an integer K group the points into K clusters $C= \{C_1, C_2,\ldots,C_k\}$ such that the following objective function is minimized, where c_i is the centroid of the points in cluster C_i

$$Cost(C) = \sum_{i=1}^{k} \sum_{x \in C_i} distance(x, c_i)$$

K-means Clustering

- Most common definition is with Euclidean distance, minimizing the Sum of Squares Error (SSE) function

$$Cost(C) = \sum_{i=1}^{k} \sum_{x \in C_i} (x - c_i)^2$$

Sum of Squares Error (SSE)

Complexity of the k-means problem

- NP-hard if the dimensionality of the data is at least 2 ($d \geq 2$)
  - Finding the best solution in polynomial time is infeasible

- For $d=1$ the problem is solvable in polynomial time

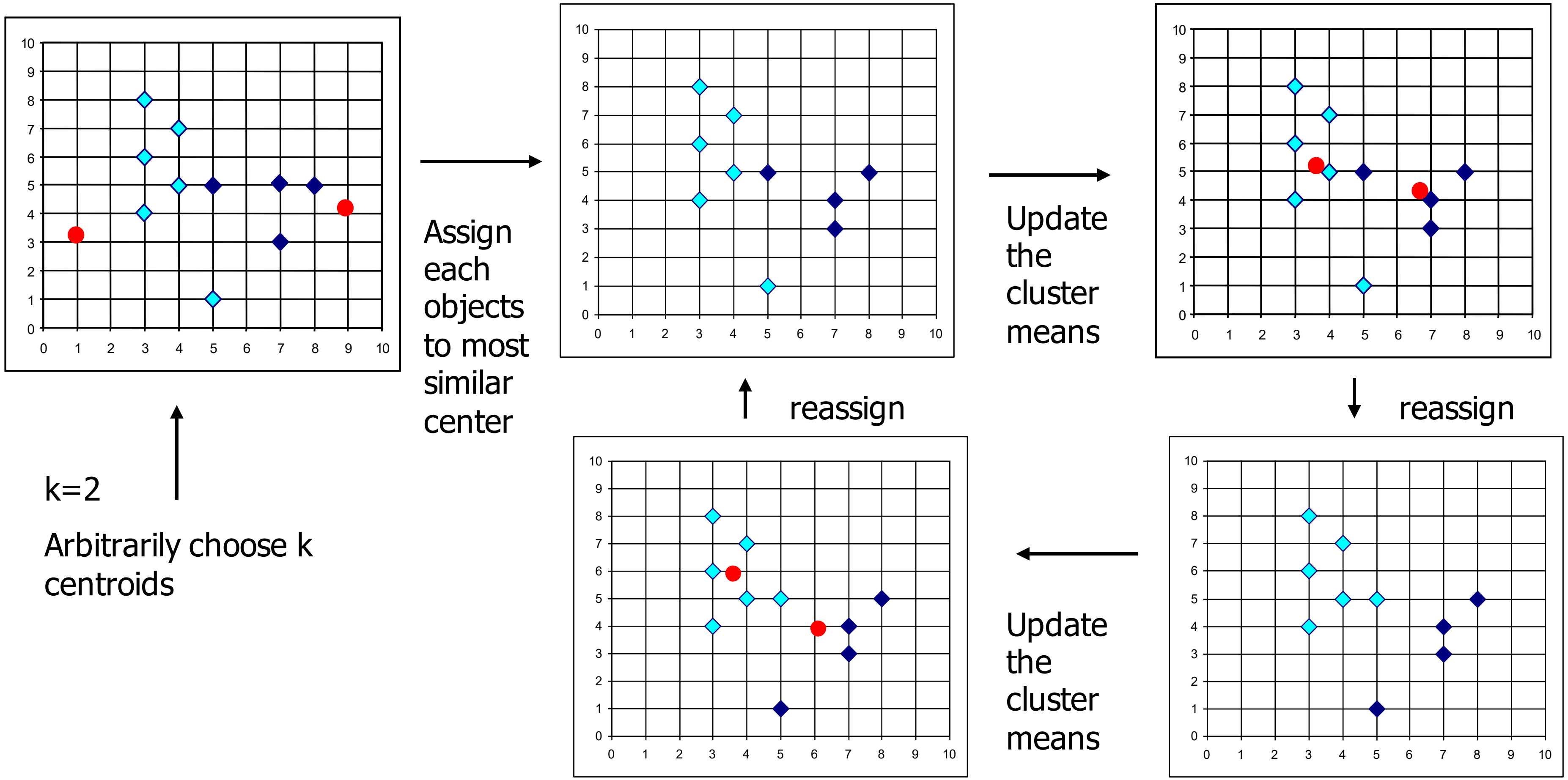
- A simple iterative algorithm works quite well in practice

K-means Algorithm

- Also known as Lloyd's algorithm.
- K-means is sometimes synonymous with this algorithm

```
Select K points as the initial centroids.
repeat
    Form K clusters by assigning all points to the closest centroid.
    Recompute the centroid of each cluster.
until The centroids don't change
```

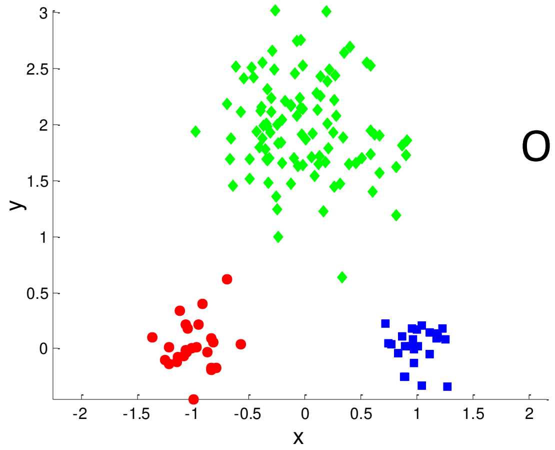
Procedure for finding k-means



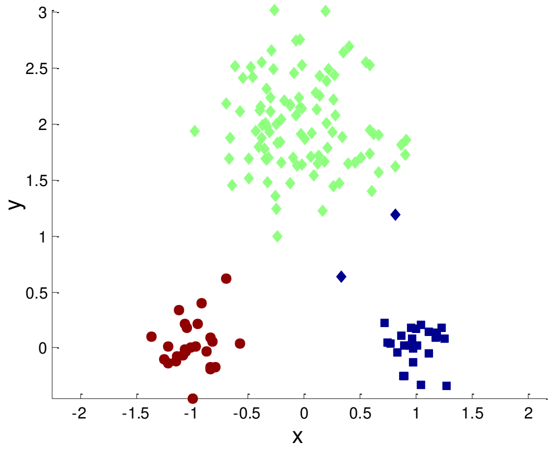
K-means Algorithm – Initialization

- Initial centroids are often chosen randomly.
    - Clusters produced vary from one run to another.

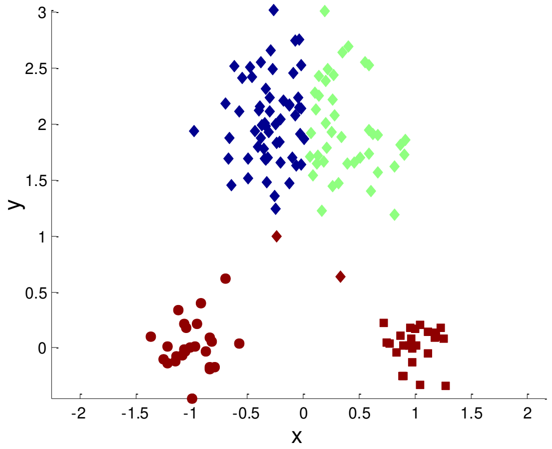
Two different K-means Clusterings



Original Points

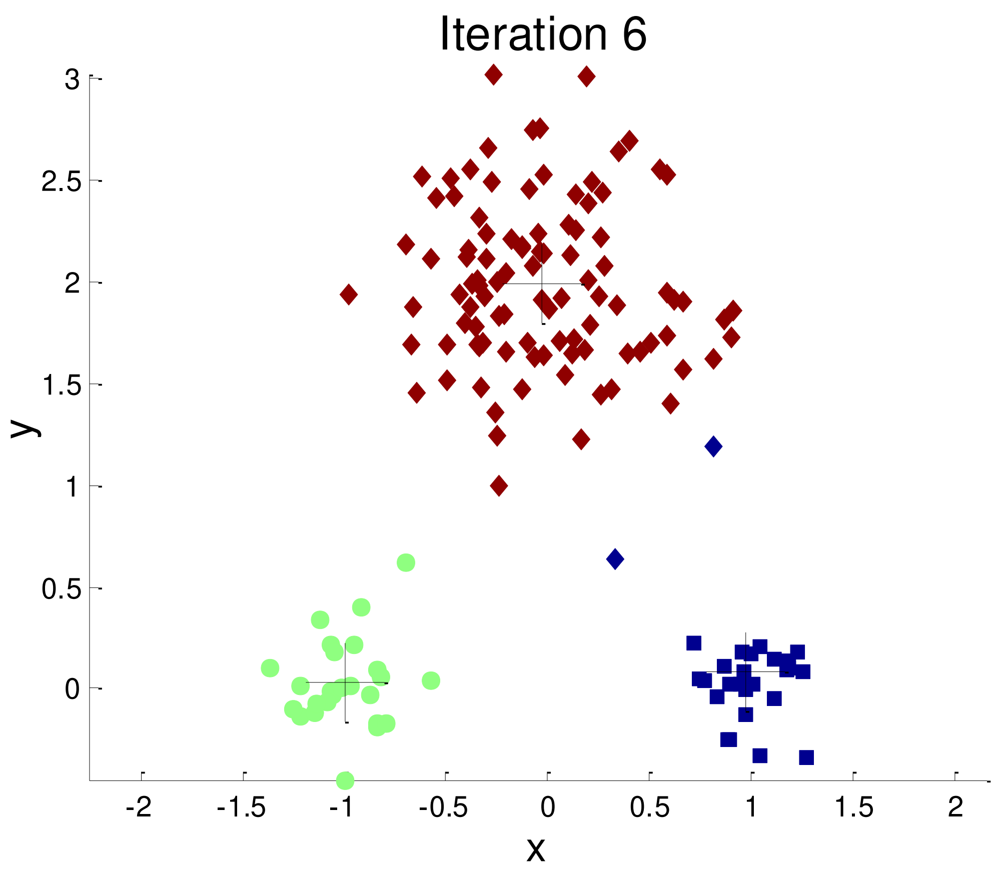


Optimal Clustering

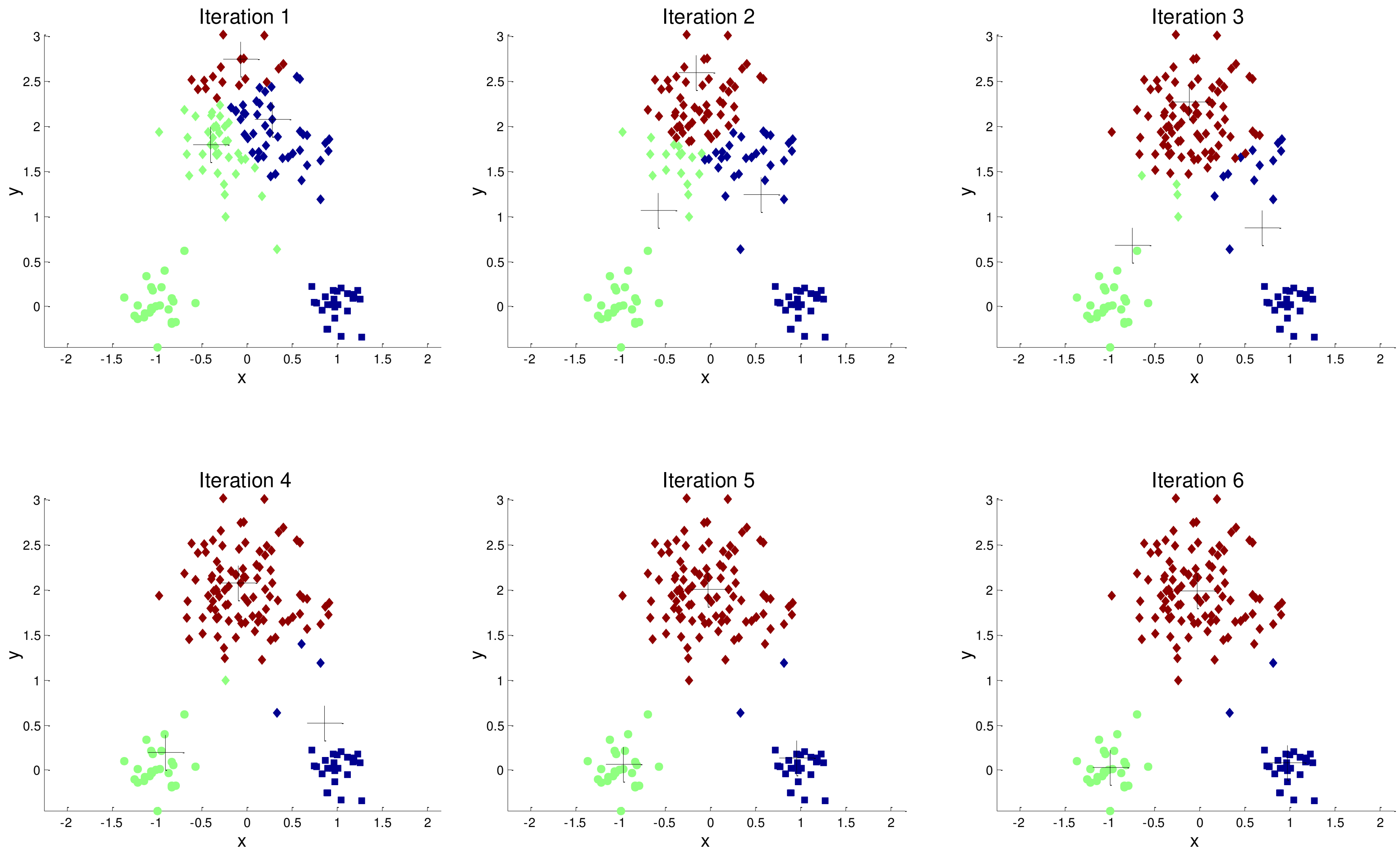


Sub-optimal Clustering

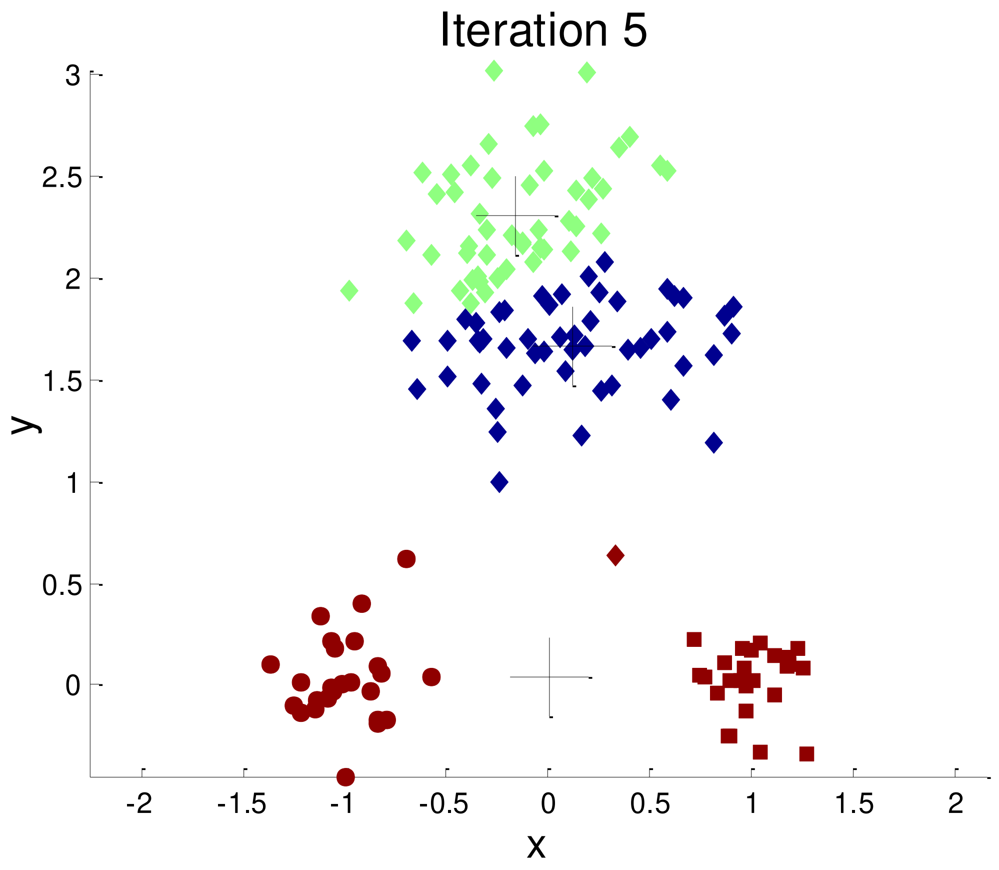
Importance of Choosing Initial Centroids



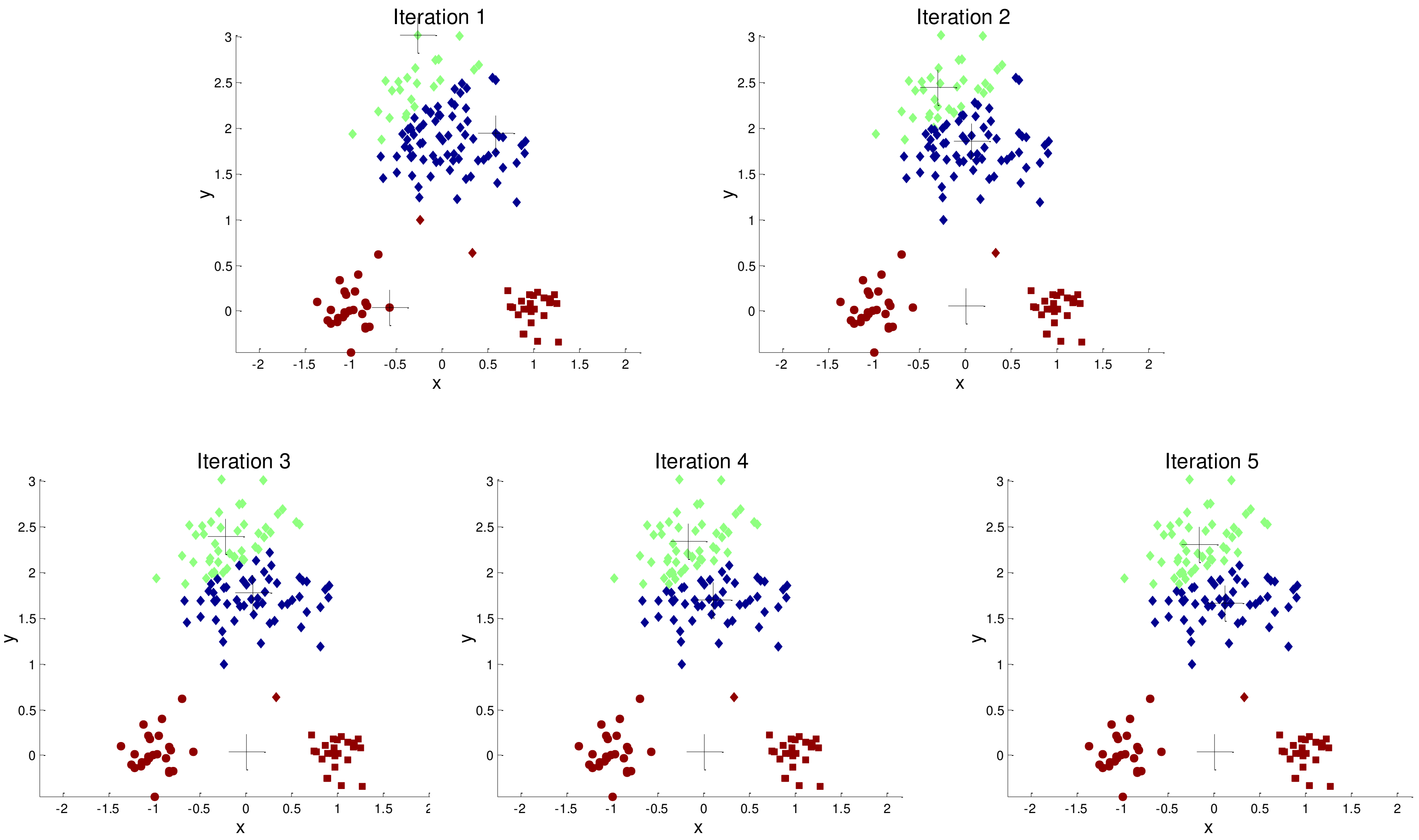
Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids …



Dealing with Initialization

- Do multiple runs and select the clustering with the smallest error

- Select original set of points by methods other than random . E.g., pick the most distant (from each other) points as cluster centers (K-means++ algorithm)

K-means Algorithm – Centroids

- The centroid depends on the distance function
  - The minimizer for the distance function
- 'Closeness' is measured by Euclidean distance (SSE), cosine similarity, correlation, etc.
- Centroid:
  - The mean of the points in the cluster for SSE, and cosine similarity
  - The median for Manhattan distance.

- Finding the centroid is not always easy
  - It can be an NP-hard problem for some distance functions
    - E.g., median form multiple dimensions

K-means Algorithm – Convergence

- 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 = dimensionality

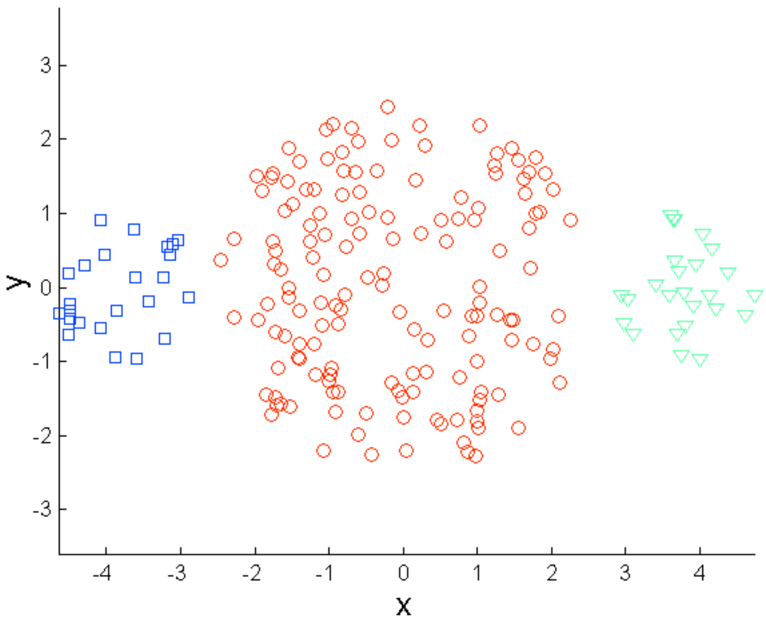
Limitations of K-means

- K-means has problems when clusters are of different
  - Sizes
  - Densities
  - Non-globular shapes

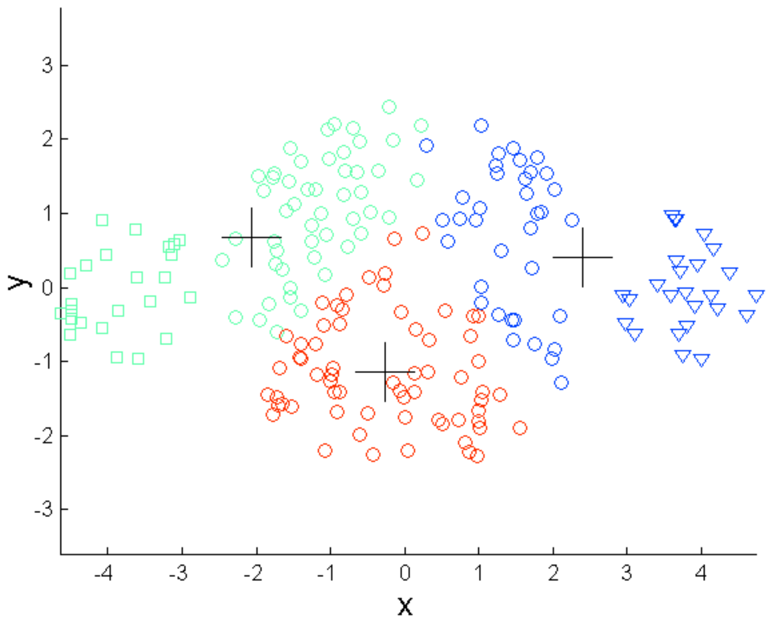
- K-means has problems when the data contains outliers

- Determining K is not user-friendly

Limitations of K-means: Differing Sizes

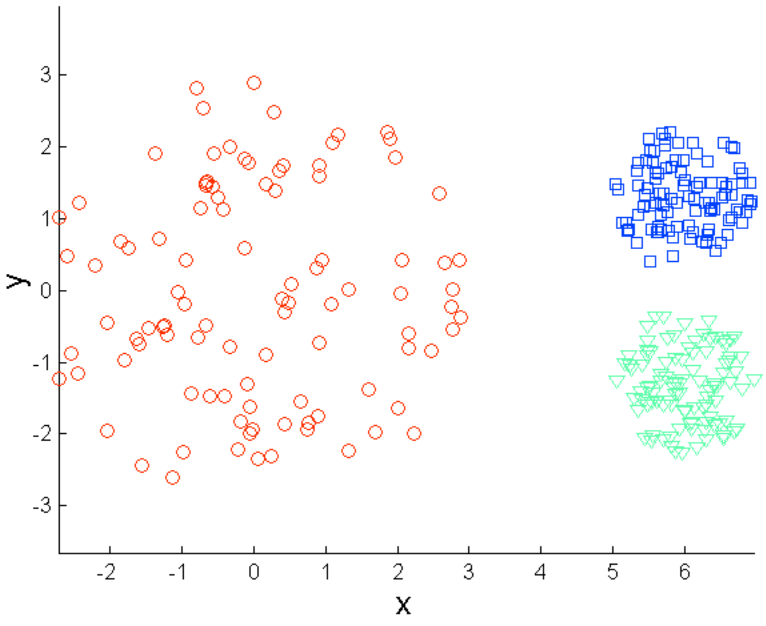


Original Points

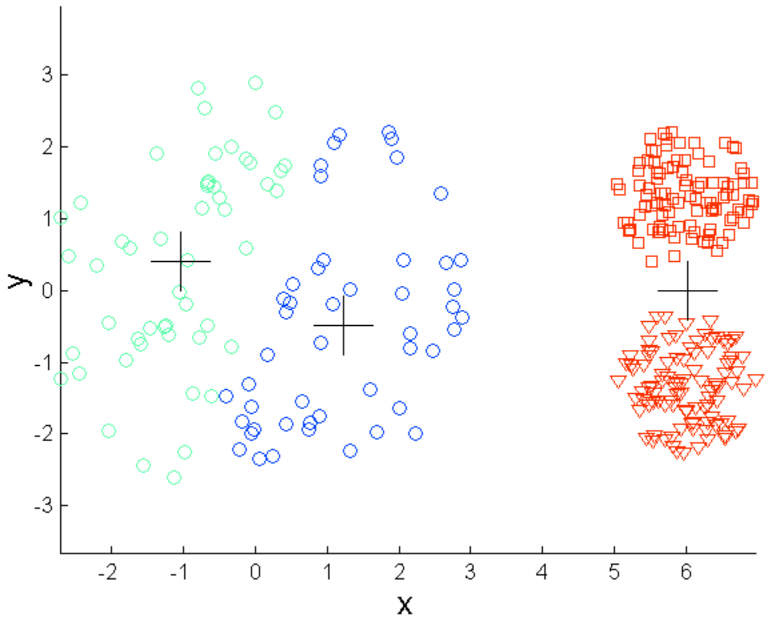


K-means (3 Clusters)

Limitations of K-means: Differing Density

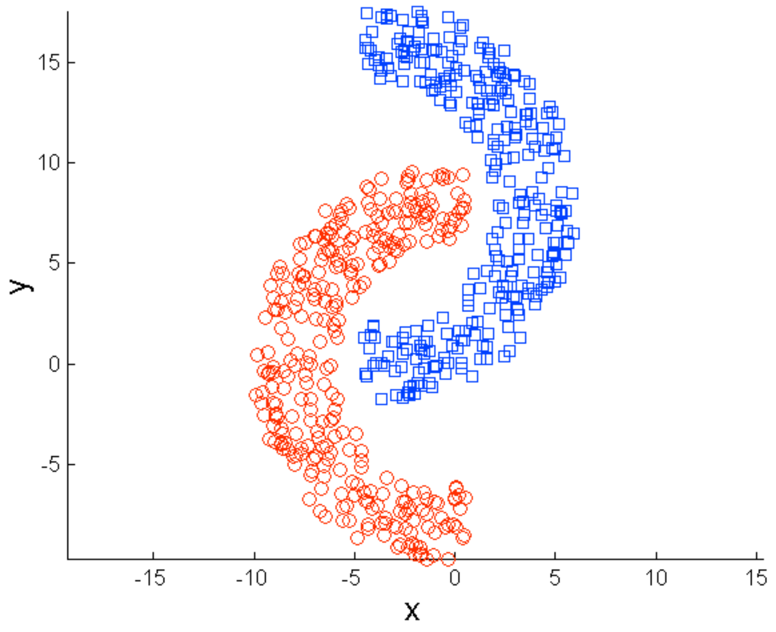


Original Points

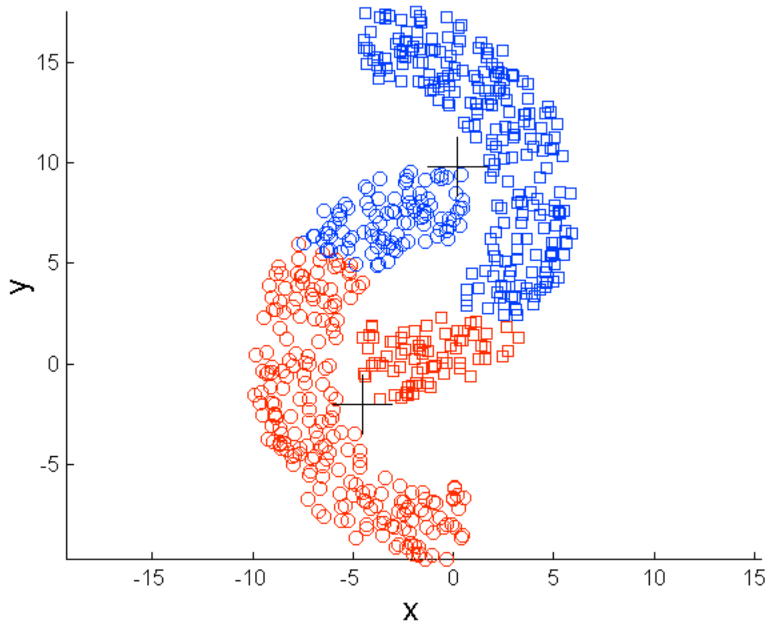


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

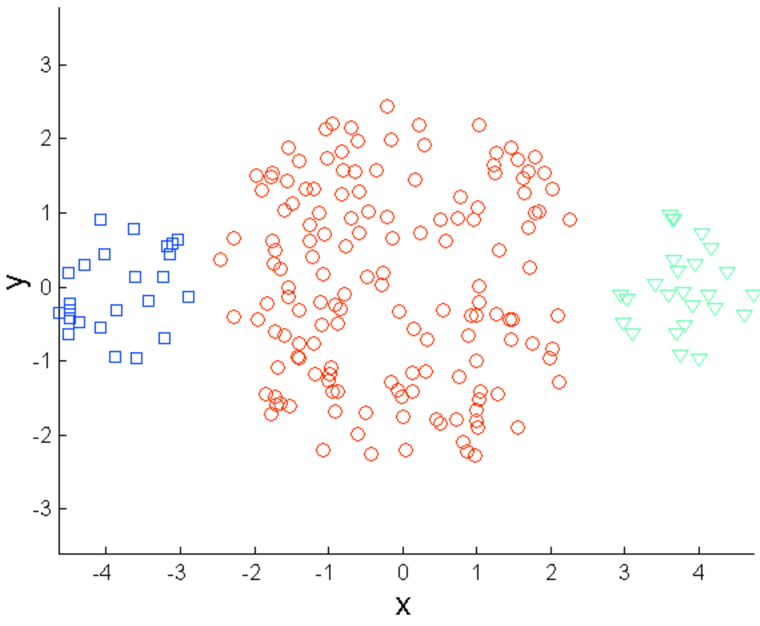


Original Points

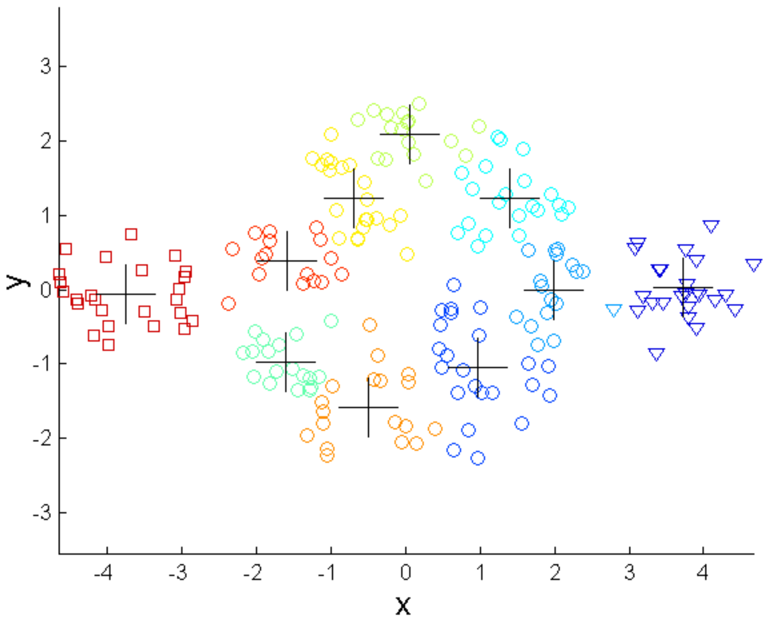


K-means (2 Clusters)

Overcoming K-means Limitations



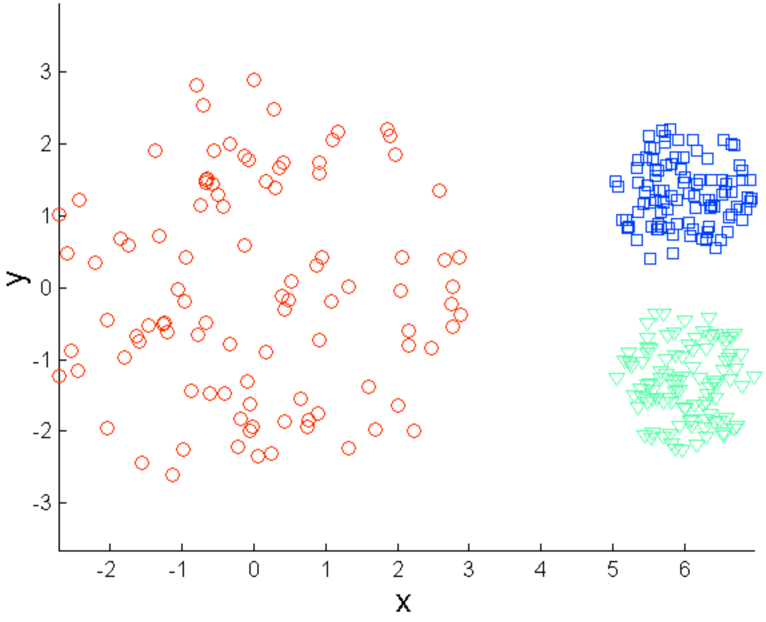
Original Points



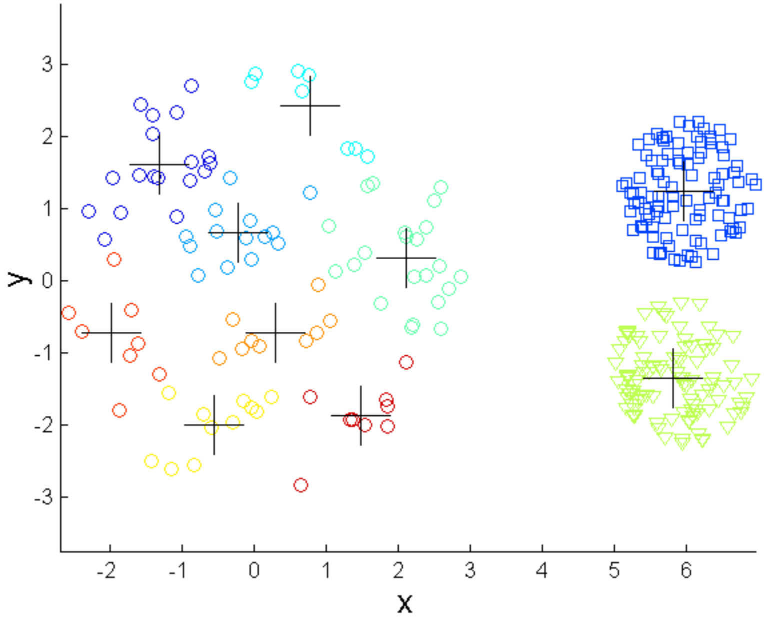
K-means Clusters

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

Overcoming K-means Limitations

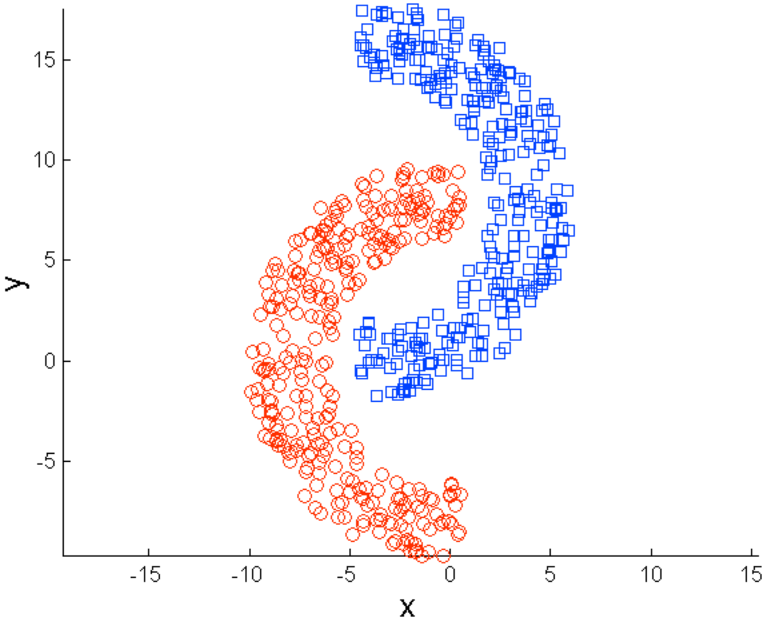


Original Points

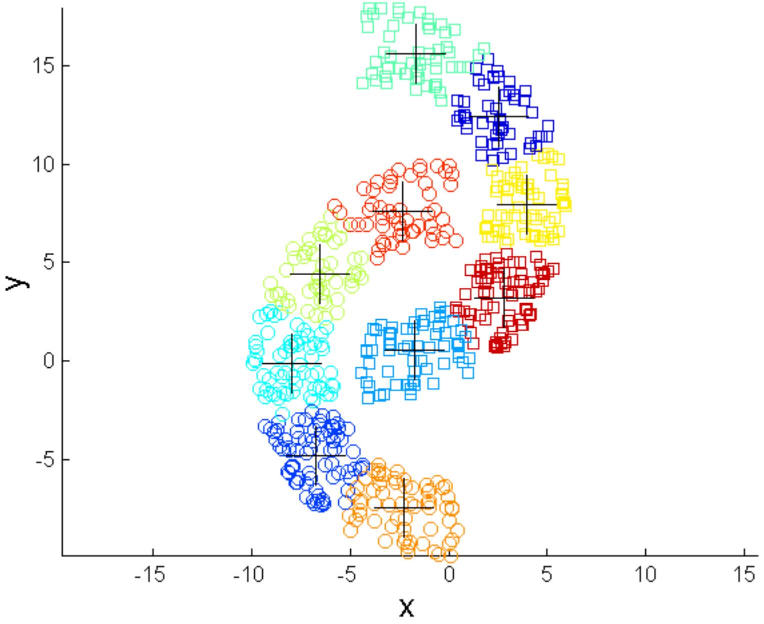


K-means Clusters

Overcoming K-means Limitations



Original Points



K-means Clusters

Variations

- K-medoids: Similar problem definition as in K-means, but the centroid of the cluster is defined to be one of the points in the cluster (the medoid).

- K-centers: Similar problem definition as in K-means, but the goal now is to minimize the maximum diameter of the clusters (diameter of a cluster is maximum distance between any two points in the cluster).

Clustering Methods

- K-means Clustering
  - Original k-means Clustering
  - Sequential K-means Clustering
  - Forgetful Sequential K-means Clustering

Sequential k-Means Clustering

- Another way to modify the k-means procedure is to update the means one example at a time, rather than all at once.
- This is particularly attractive when we acquire the examples over a period of time, and we want to start clustering before we have seen all of the examples
- Here is a modification of the k-means procedure that operates sequentially

Sequential k-Means Clustering

- Make initial guesses for the means $m_1, m_2, \ldots, m_k$
- Set the counts $n_1, n_2, .., n_k$ to zero
- Until interrupted
  - Acquire the next example, x
  - If m_i is closest to x
    - Increment n_i
    - Replace m_i by $m_i + (1/n_i) \cdot (x - m_i)$

Clustering Methods

- K-means Clustering
  - Original k-means Clustering
  - Sequential K-means Clustering
  - Forgetful Sequential K-means Clustering

Forgetful Sequential k-means

- This also suggests another alternative in which we replace the counts by constants. In particular, suppose that *a* is a constant between 0 and 1, and consider the following variation:

- Make initial guesses for the means m_1, m_2, …, m_k
- Until interrupted
  - Acquire the next example x
  - If m_i is closest to x, replace m_i by $m_i+a(x-m_i)$

Forgetful Sequential k-means

- The result is called the "forgetful" sequential k-means procedure.
- It is not hard to show that m_i is a weighted average of the examples that were closest to m_i, where the weight decreases exponentially with the "age" to the example.
- That is, if m_0 is the initial value of the mean vector and if x_j is the j-th example out of n examples that were used to form m_i, then it is not hard to show that

$$m_n = (1-a)^n \, m_0 + a \; \sum_{k=1}^{n} (1-a)^{n-k} x_k$$

Forgetful Sequential k-means

- Thus, the initial value m_0 is eventually forgotten, and recent examples receive more weight than ancient examples.
- This variation of k-means is particularly simple to implement, and it is attractive when the nature of the problem changes over time and the cluster centers “drift”.