Data Mining
Cluster Analysis: Basic Concepts and Algorithms

Lecture Notes for Chapter 7

Introduction to Data Mining, 2nd Edition

by

Tan, Steinbach, Karpatne, Kumar

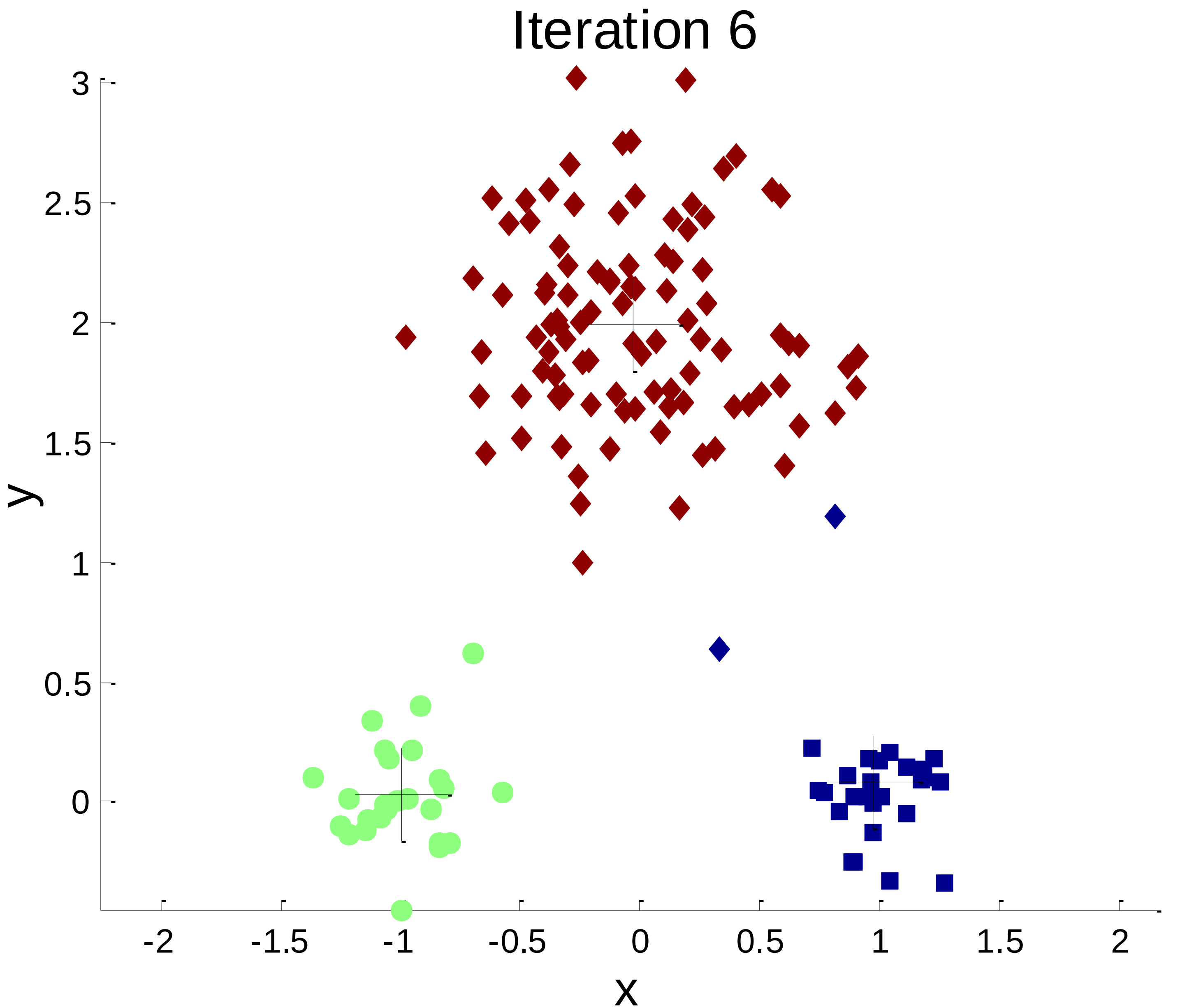
K-means

K-means Clustering

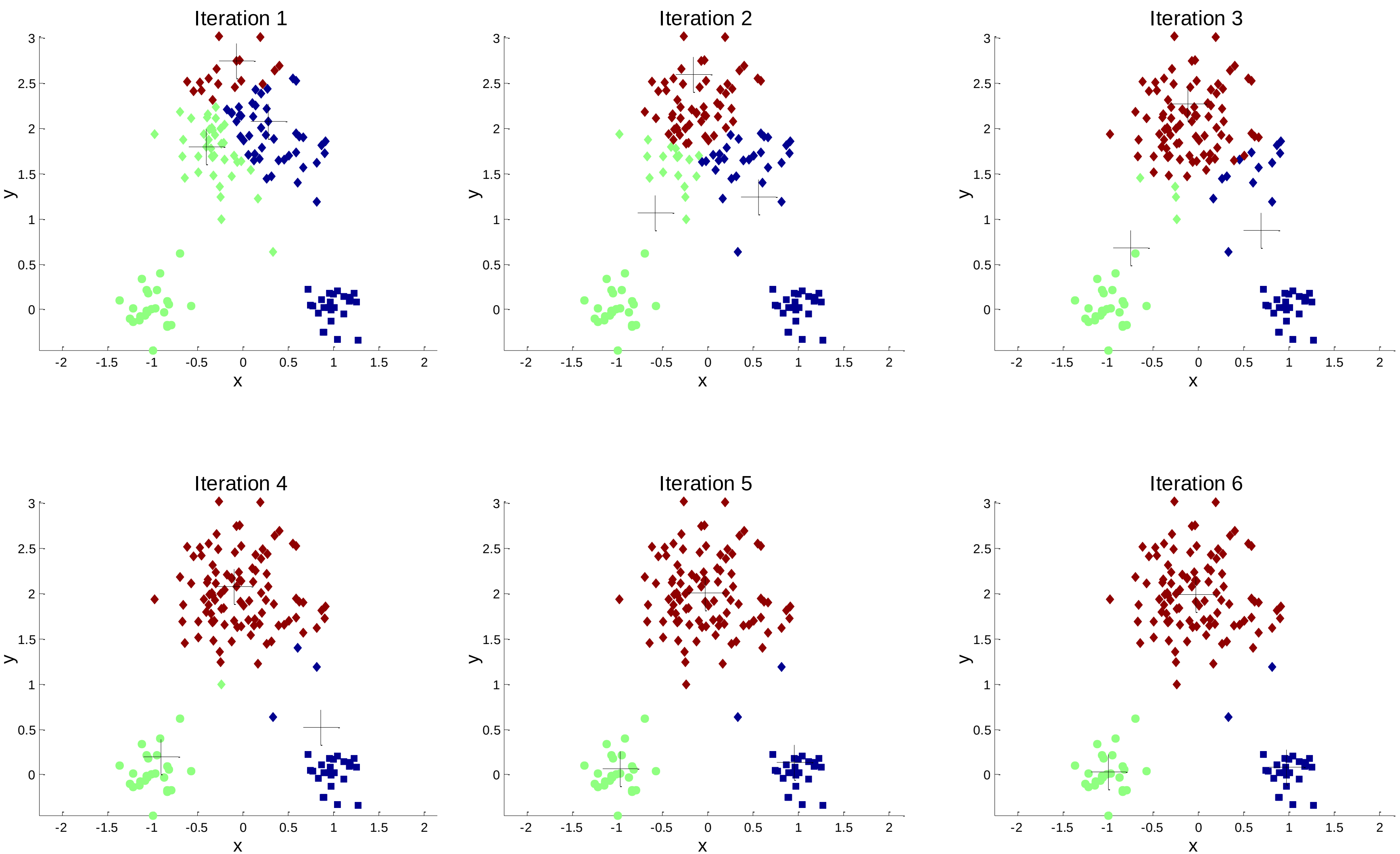
- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the **closest centroid**
- The basic algorithm is very simple

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

Example of K-means Clustering



Example of K-means Clustering



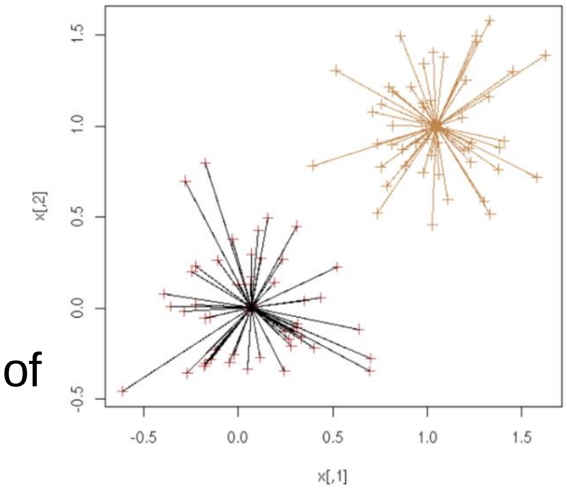
K-means Clustering – Details

- 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

Evaluating K-means Clusters

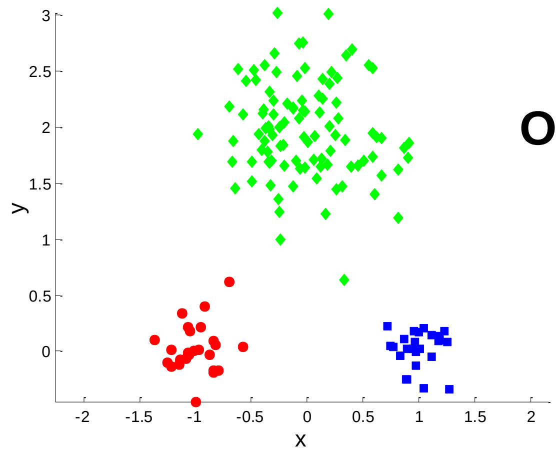
- Most common measure is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster
  - To get SSE, we square these errors and sum them.

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

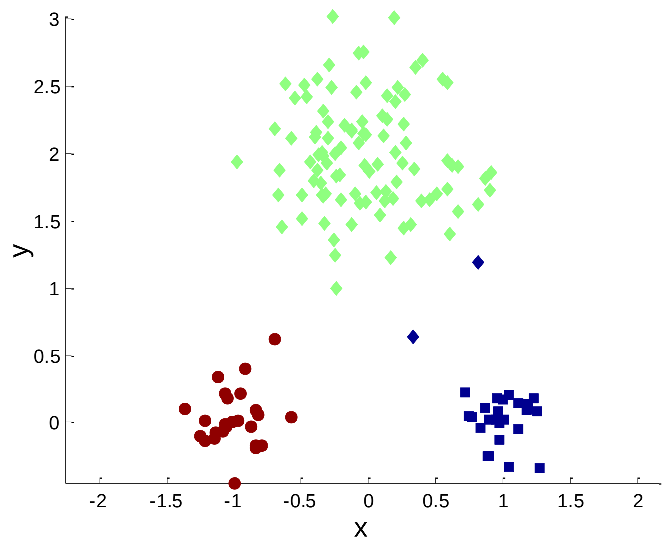


- x is a data point in cluster C_i and m_i is the representative point for cluster Ci
  - can show that mi corresponds to the center (mean) of the cluster
- Given two sets of clusters, we prefer the one with the smallest error
- One easy way to reduce SSE is to increase K, the number of clusters
- A good clustering with smaller K can have a lower SSE than a poor clustering with higher K

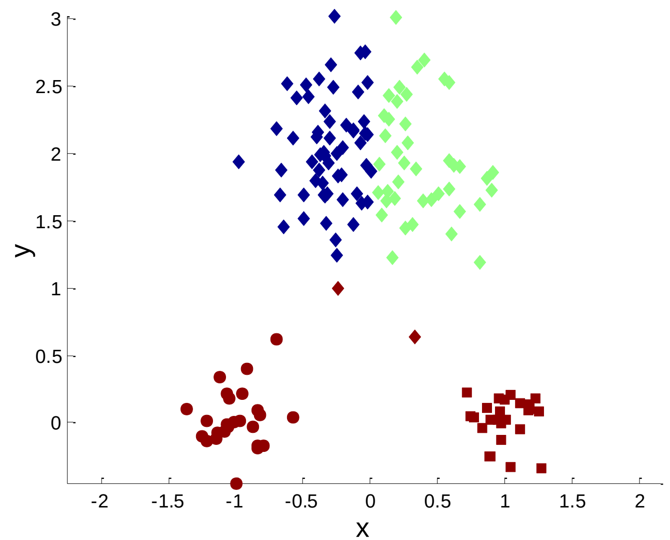
Two different K-means Clusterings



Original Points



Optimal Clustering



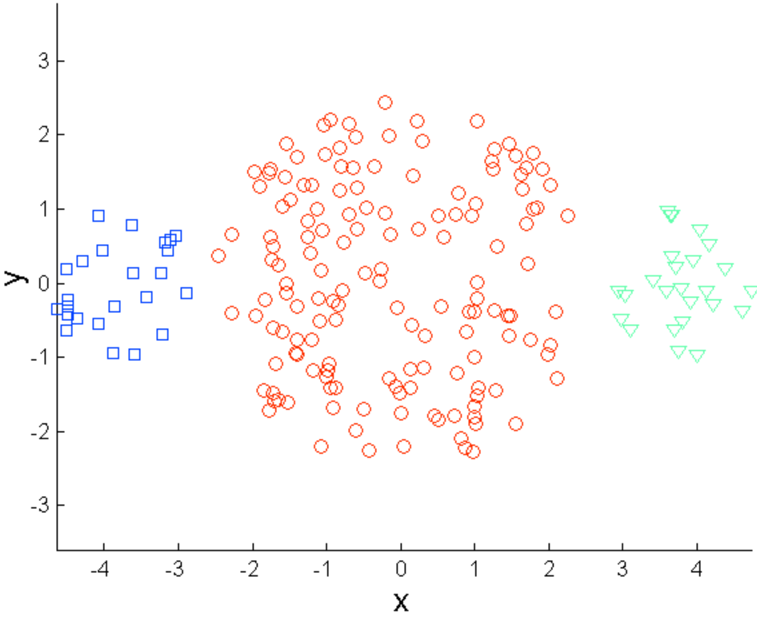
Sub-optimal Clustering

Limitations of K-means

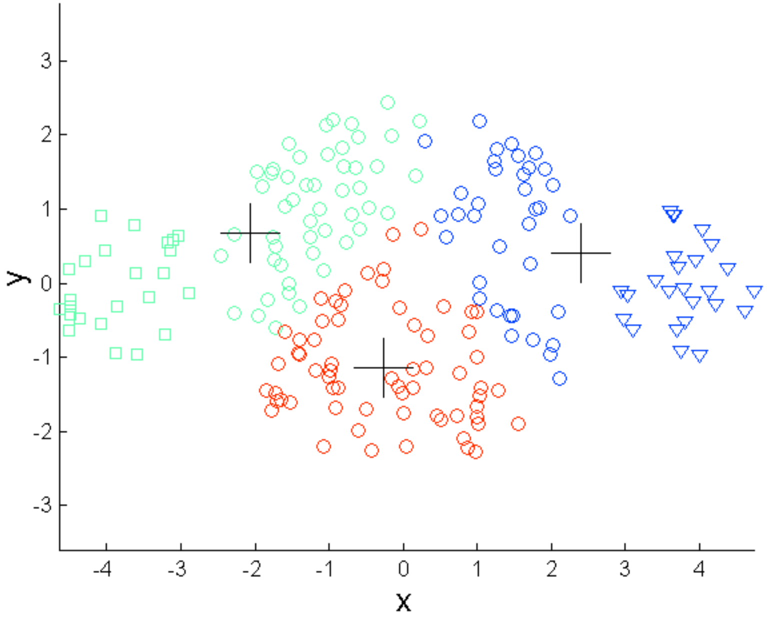
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

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

Limitations of K-means: Differing Sizes

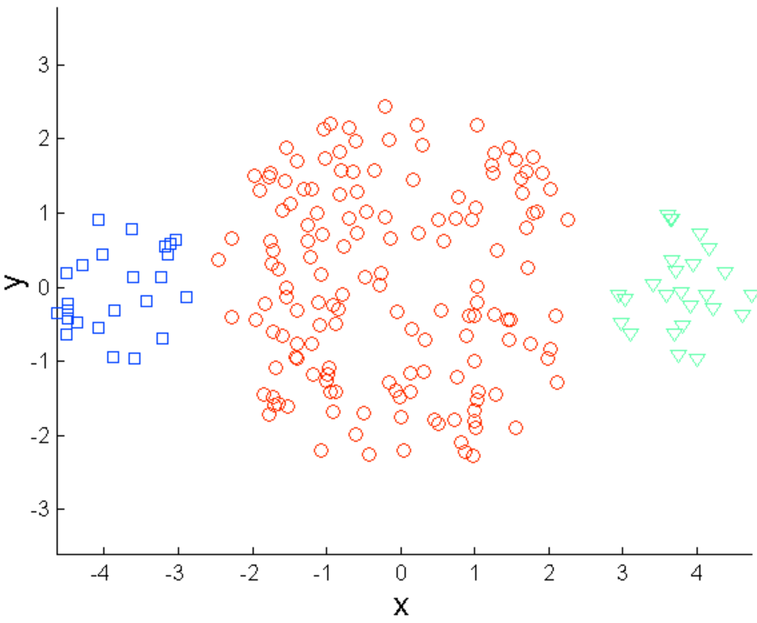


Original Points

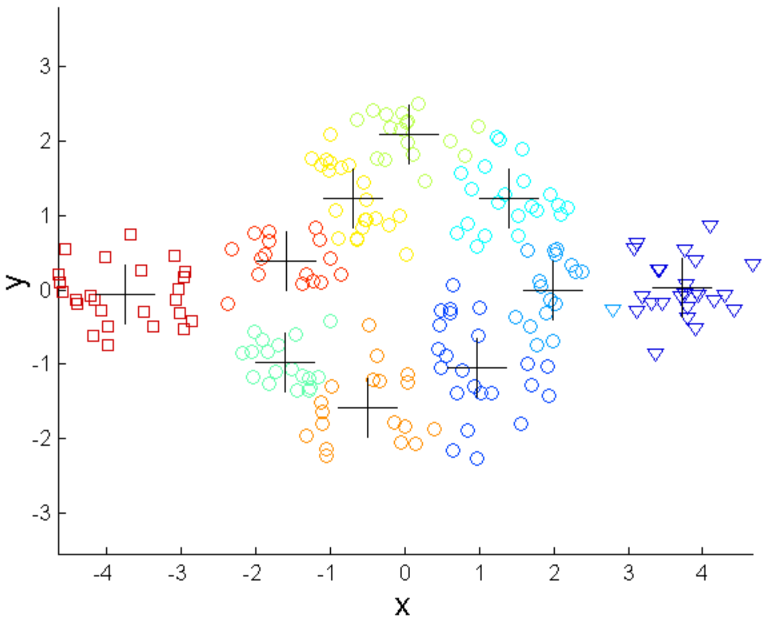


K-means (3 Clusters)

Overcoming K-means Limitations



Original Points

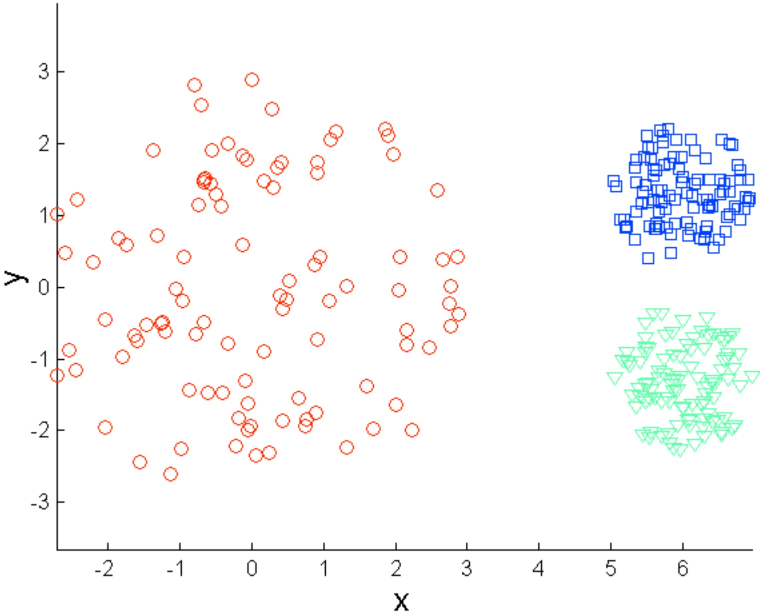


K-means Clusters

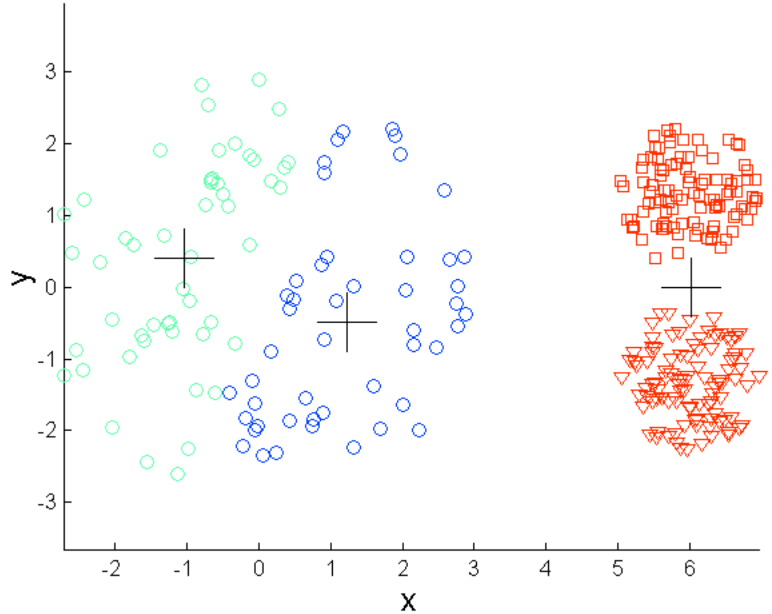
One solution is to use many clusters.

Find parts of clusters, but need to put together.

Limitations of K-means: Differing Density

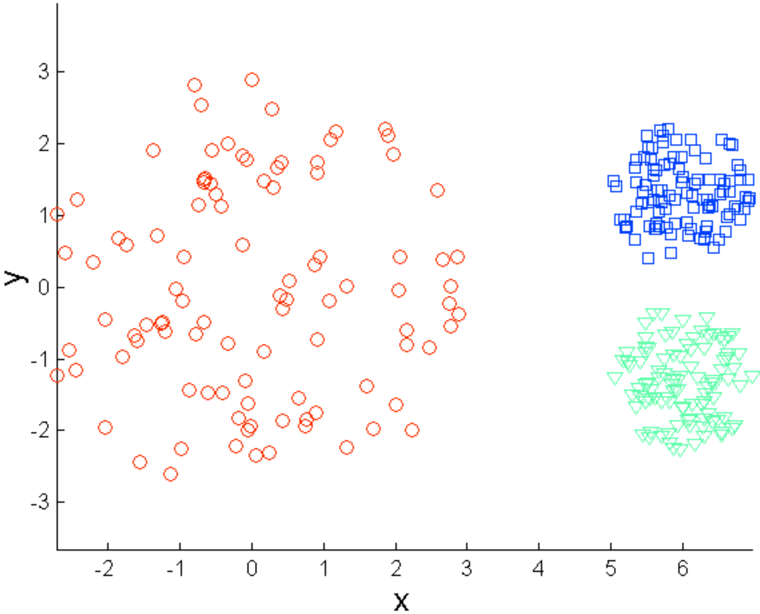


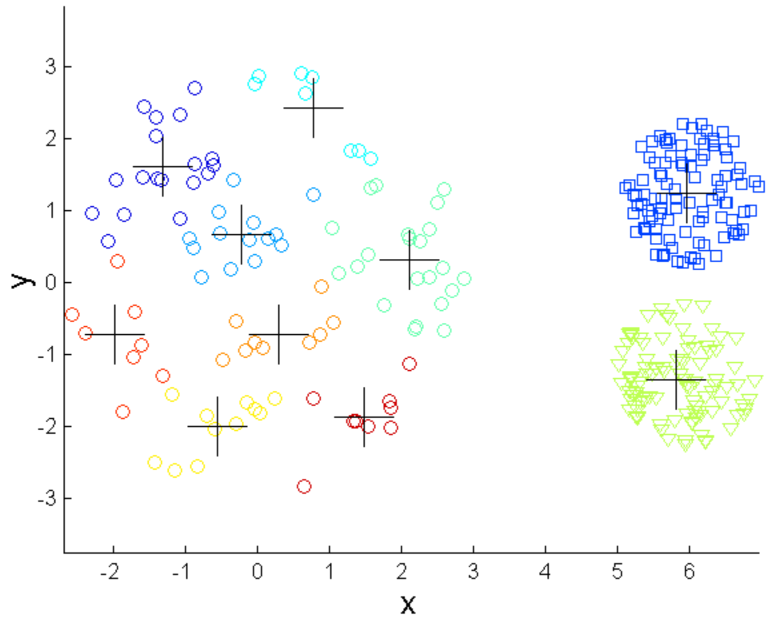
Original Points



K-means (3 Clusters)

Overcoming K-means Limitations

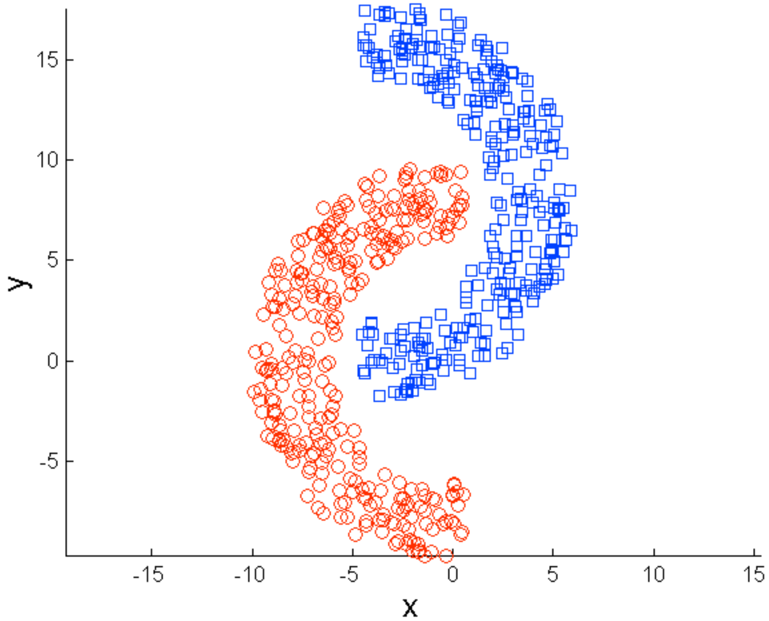




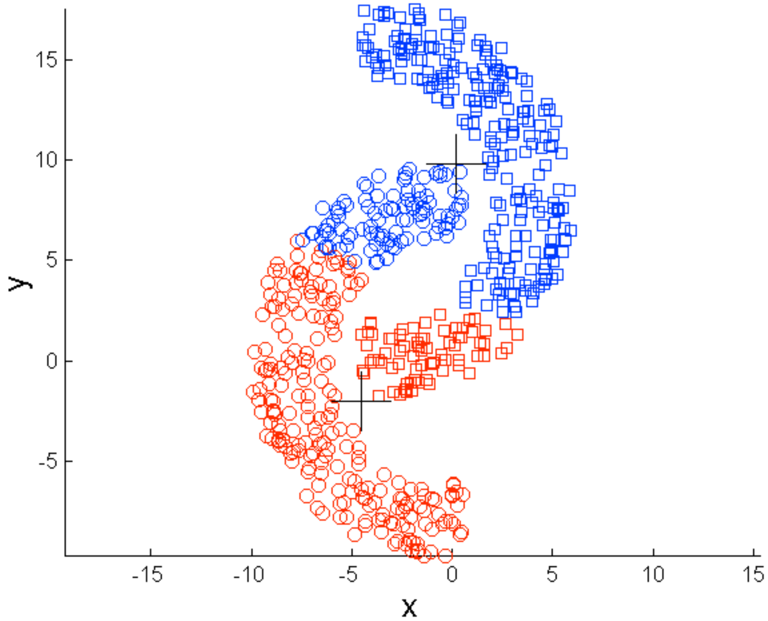
Original Points

K-means Clusters

Limitations of K-means: Non-globular Shapes

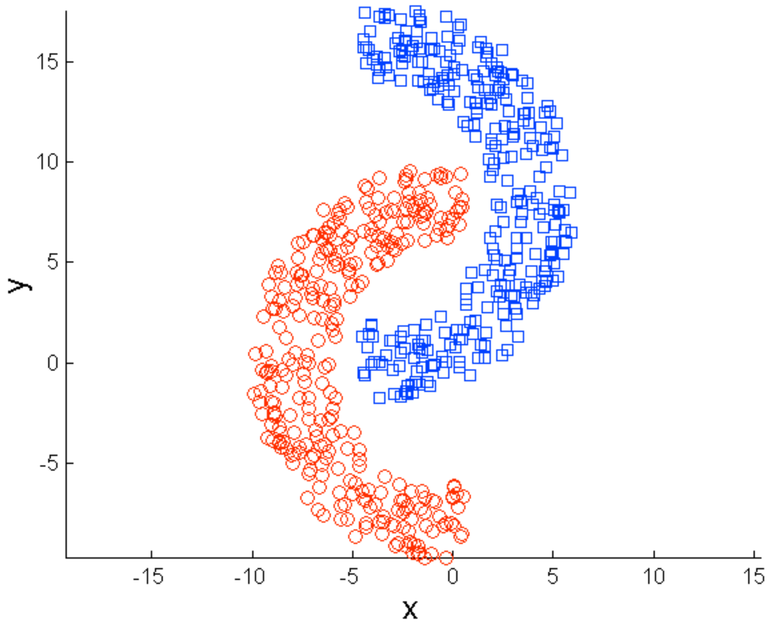


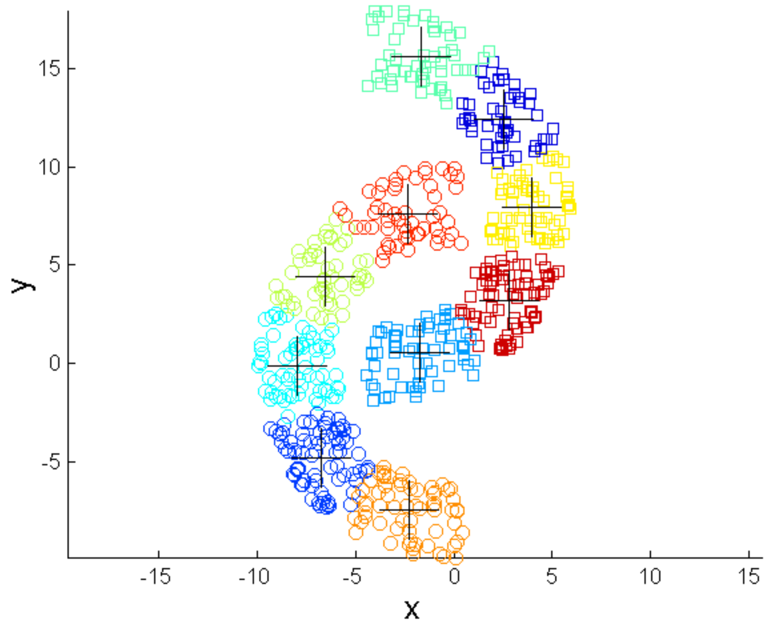
Original Points



K-means (2 Clusters)

Overcoming K-means Limitations

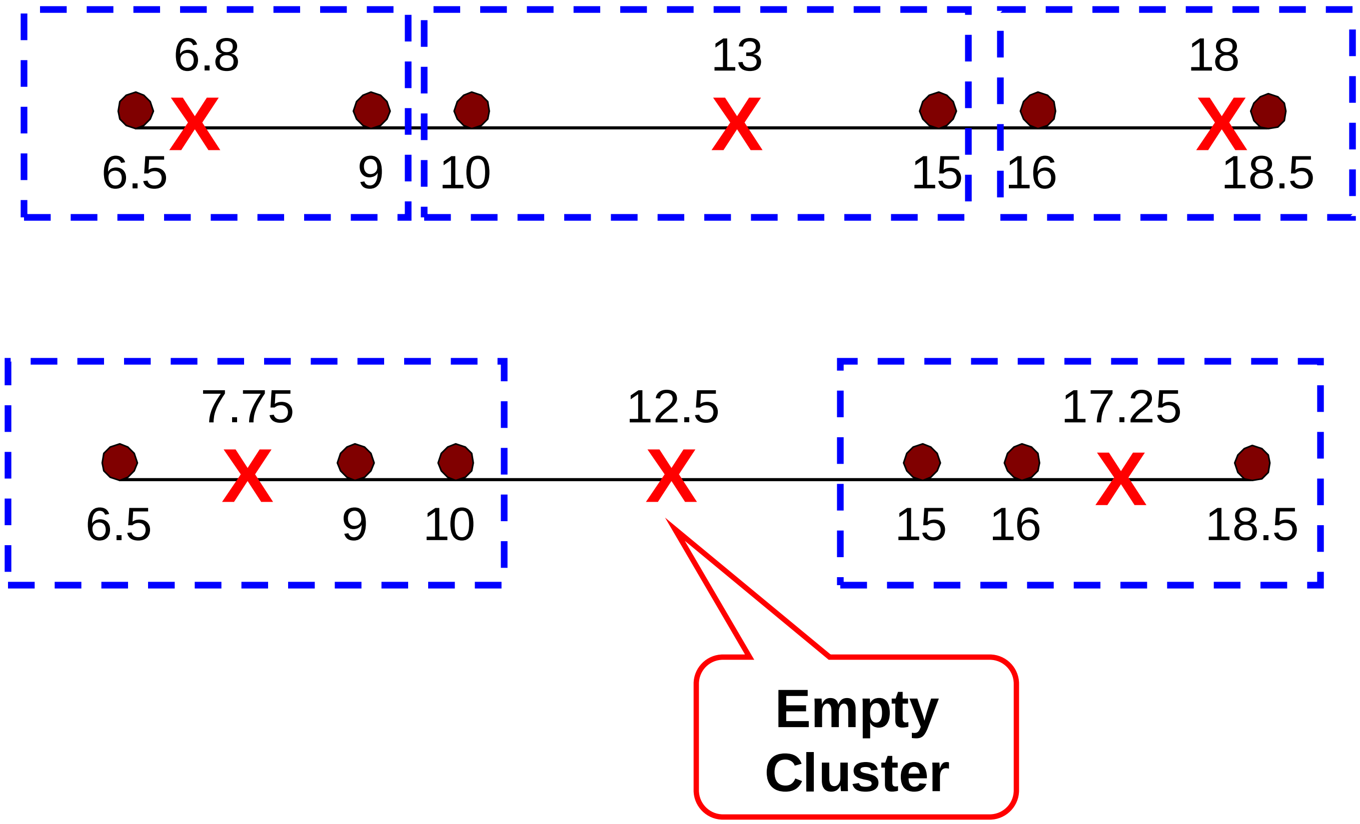




Original Points **K-means Clusters**

Empty Clusters

- K-means can yield empty clusters



Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters

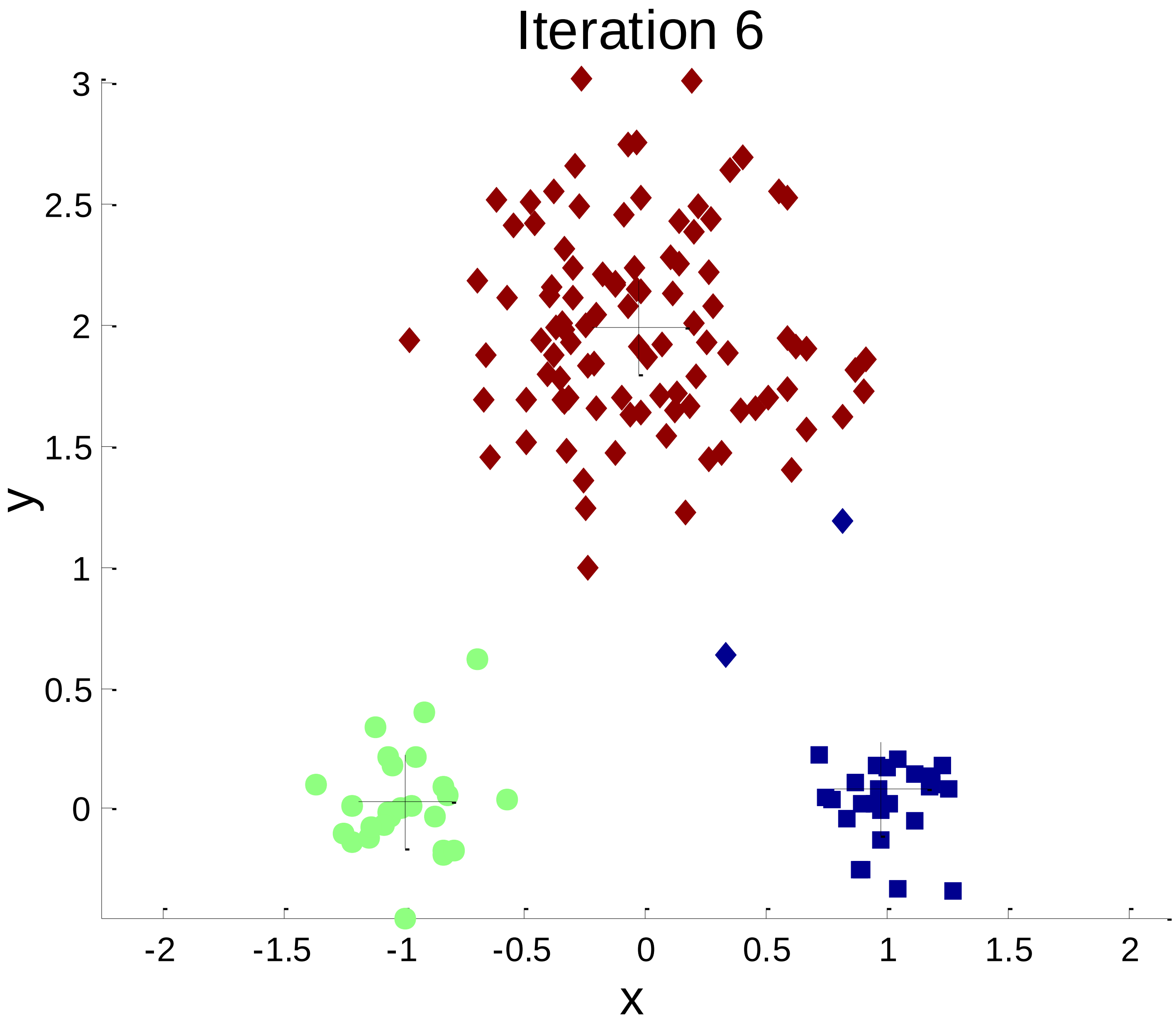
- Several strategies
  - Choose a point and assign it to the cluster
    - Choose the point that contributes most to SSE
    - Choose a point from the cluster with the highest SSE

- If there are several empty clusters, the above can be repeated several times.

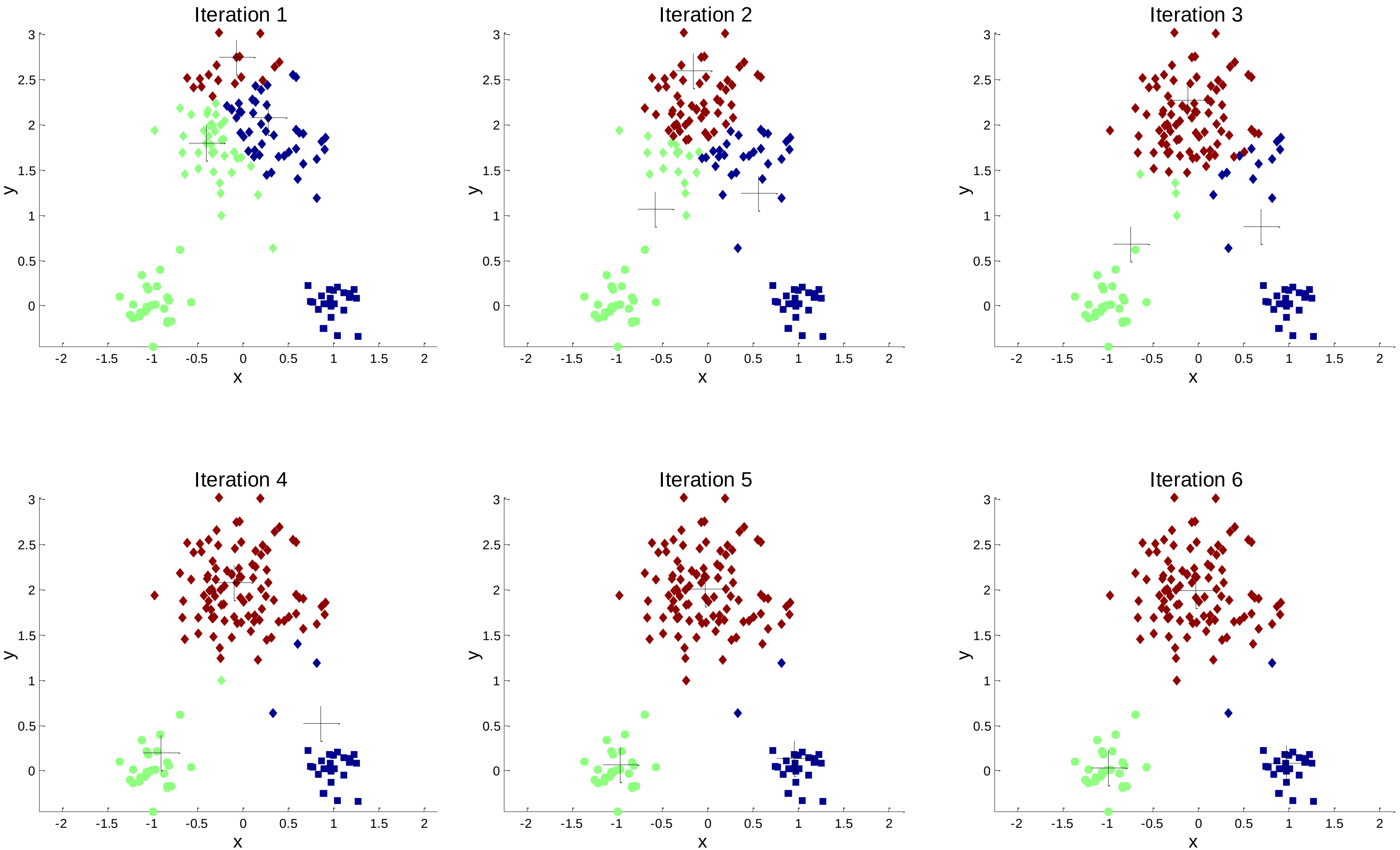
Pre-processing and Post-processing

- 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
  - Can use these steps during the clustering process
    - ISODATA

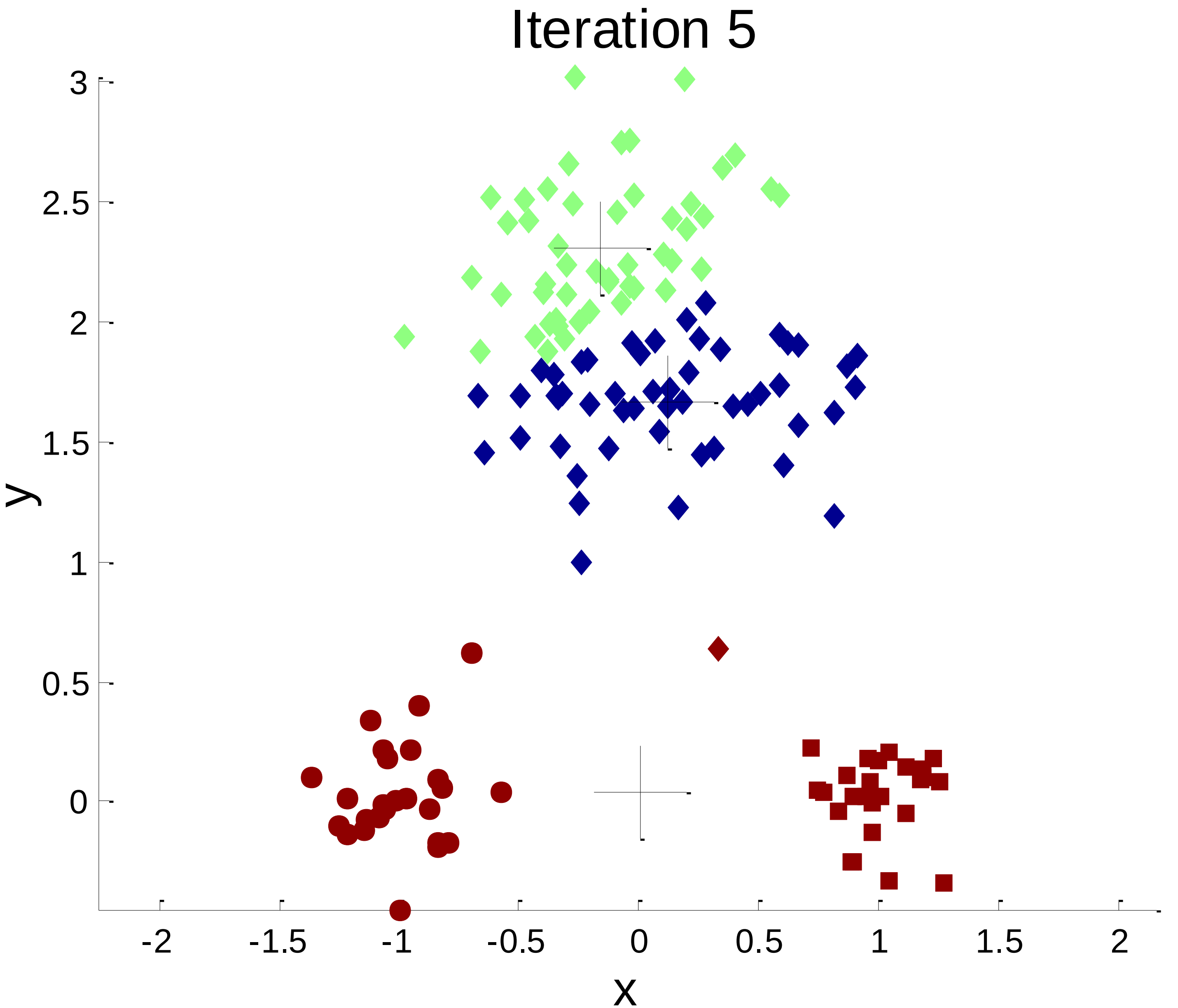
Importance of Choosing Initial Centroids



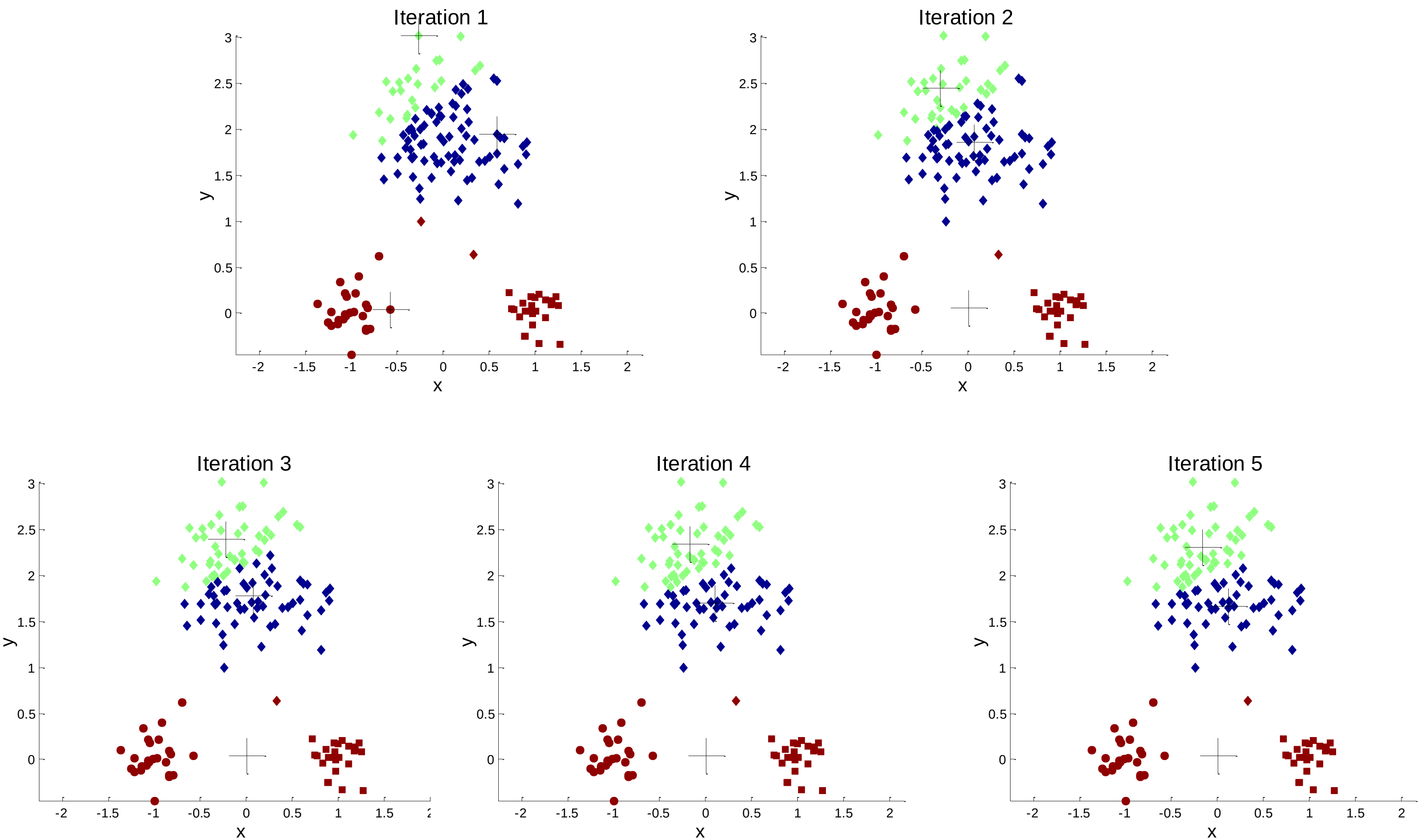
Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids …



Importance of Choosing Initial Centroids …



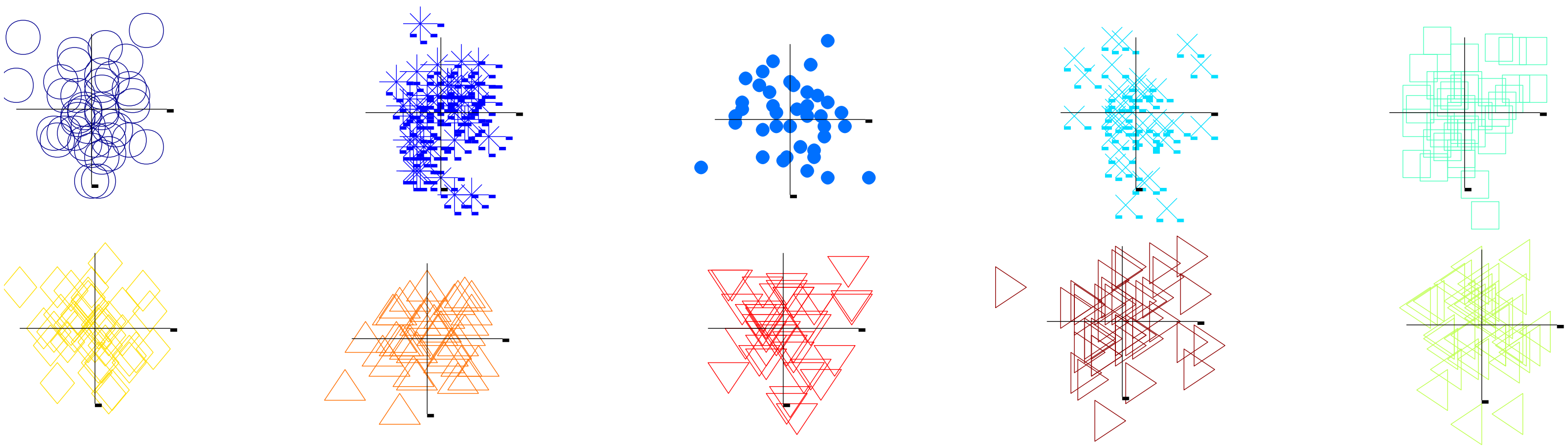
Problems with Selecting Initial Points

- 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

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

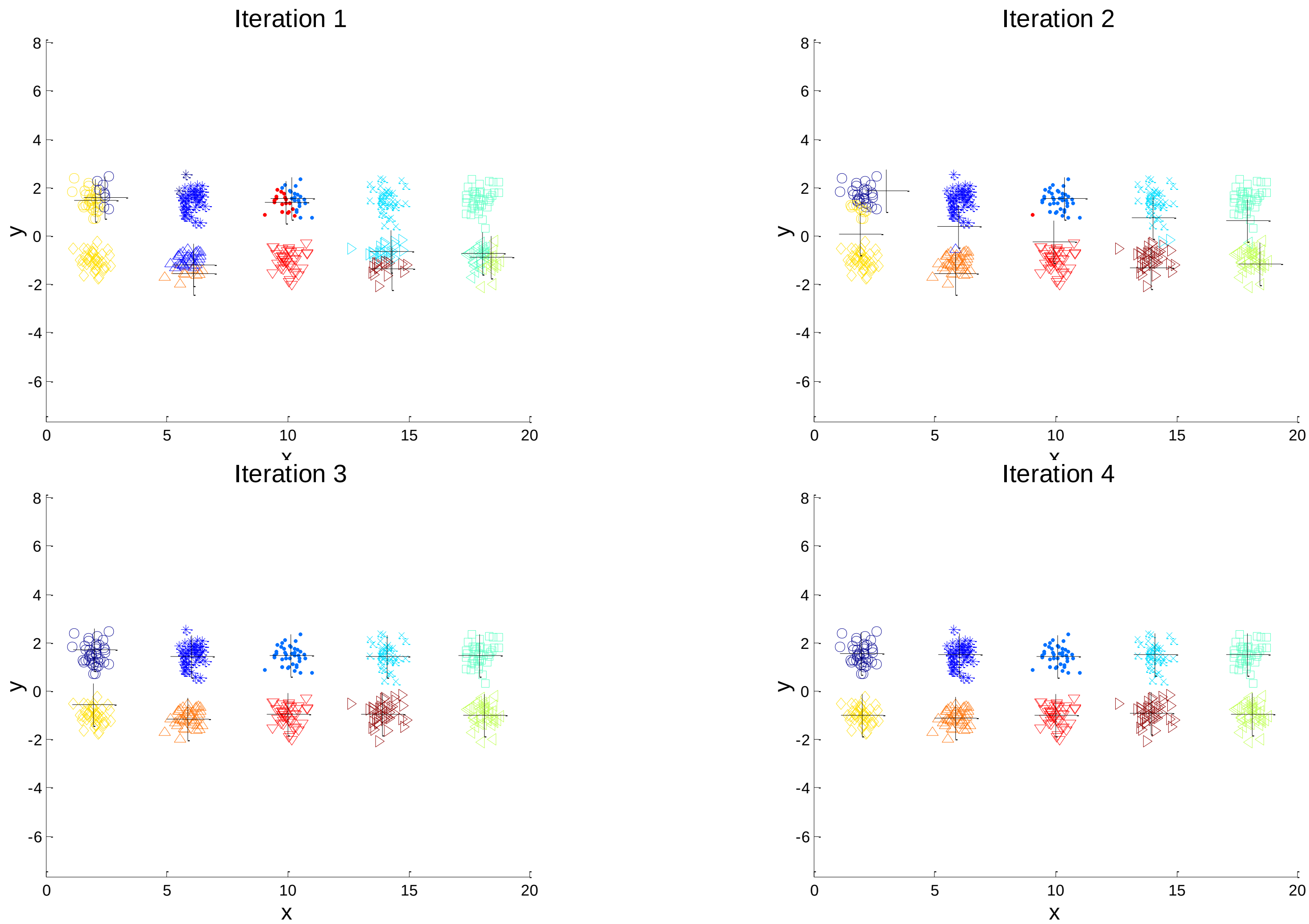
  - For example, if K = 10, then probability = $10!/10^{10} = 0.00036$
  - Sometimes the initial centroids will readjust themselves in ‘right’ way, and sometimes they don’t
  - Consider an example of five pairs of clusters

10 Clusters Example



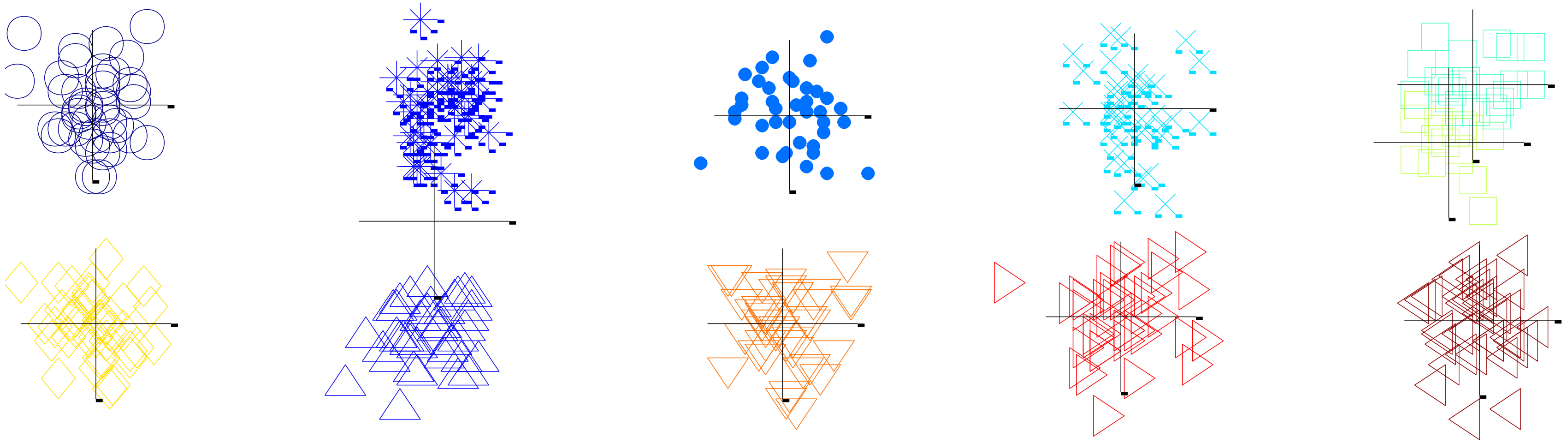
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



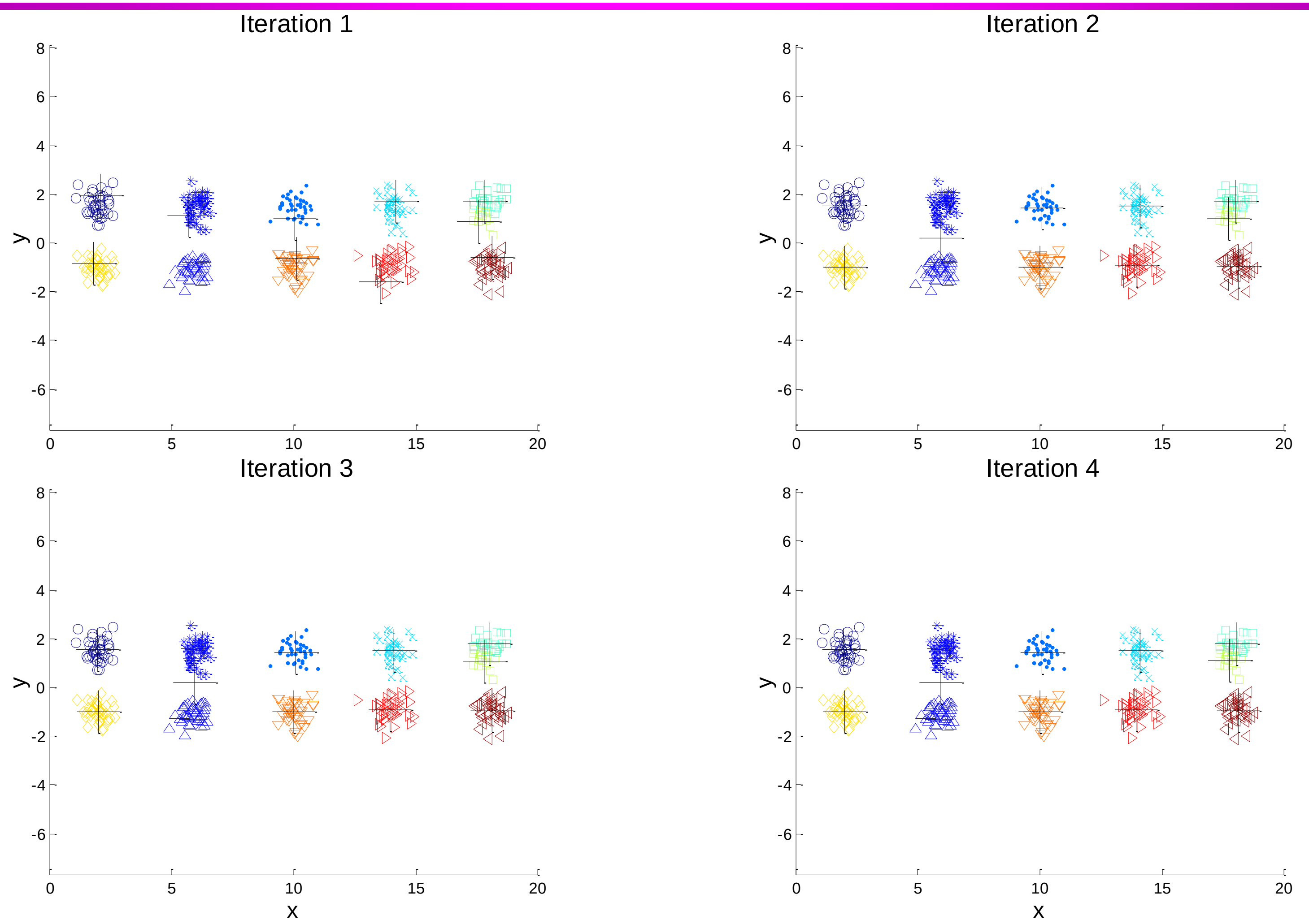
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



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

10 Clusters Example



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

Solutions to Initial Centroids Problem

- 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
- Generate a larger number of clusters and then perform a hierarchical clustering
- Bisecting K-means
  - Not as susceptible to initialization issues

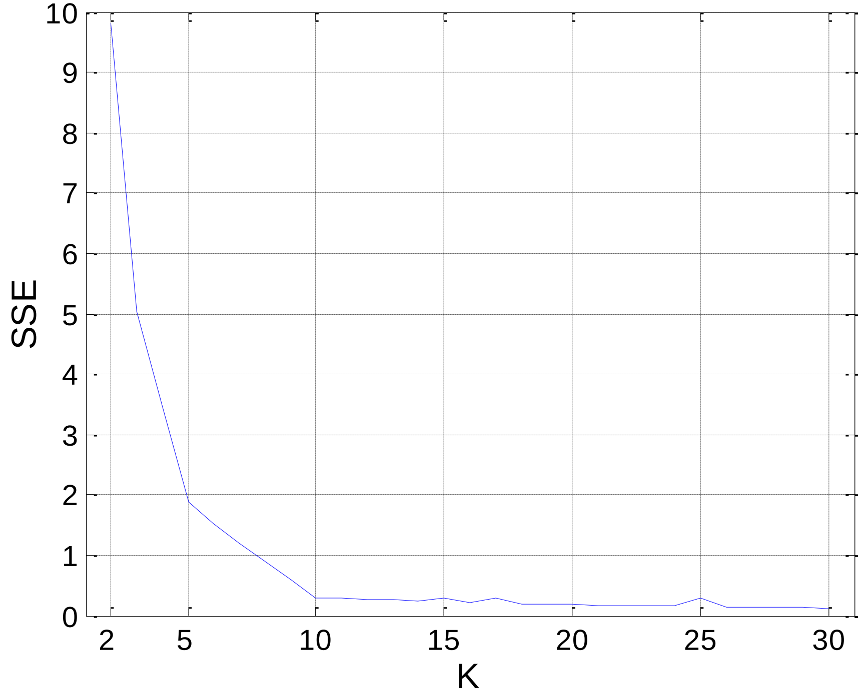
Updating Centers Incrementally

- 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

Finding the best number of clusters

- In k-means the number of clusters *K* is given
  - Partition *n* objects into predetermined number of clusters
  - **Finding the "right" number of clusters is part of the problem**



Convergence of *K*-Means

- Define goodness measure of cluster c as sum of squared distances from cluster centroid:
  - $SSE_c(c,s) = \Sigma_i (d_i - s_c)^2$ (sum over all d_i in cluster c)
  - $G(C,s) = \Sigma_c SSE_c(c,s)$
- Re-assignment monotonically decreases G
  - It is a coordinate descent algorithm (**opt one component at a time**)
- At any step we have some value for G(C,s)

1) Fix s, optimize C → assign d to the closest centroid → G(C',s) <= G(C,s)

2) Fix C', optimize s → take the new centroids → G(C',s') <= G(C',s) <= G(C,s)

The new cost is smaller than the original one → local minimum