K-means Clustering

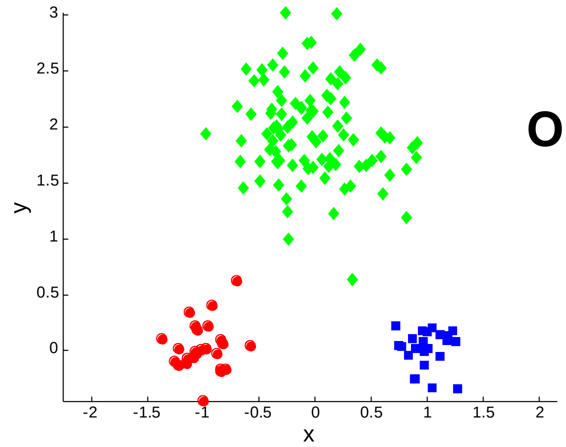
- Partitional clustering approach
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- Number of clusters, K, must be specified
- The basic algorithm is very simple

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

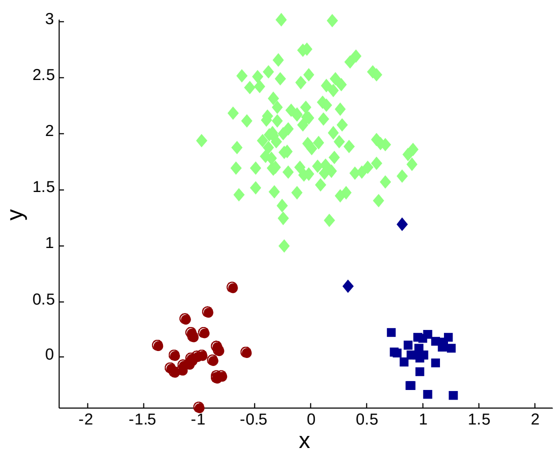
K-means Clustering – Details

- 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

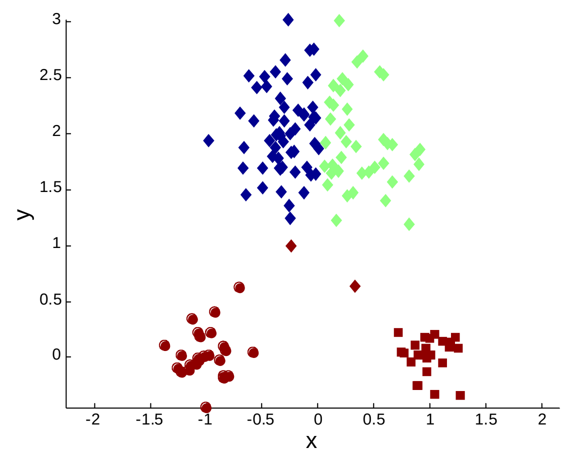
Two different K-means Clusterings



Original Points

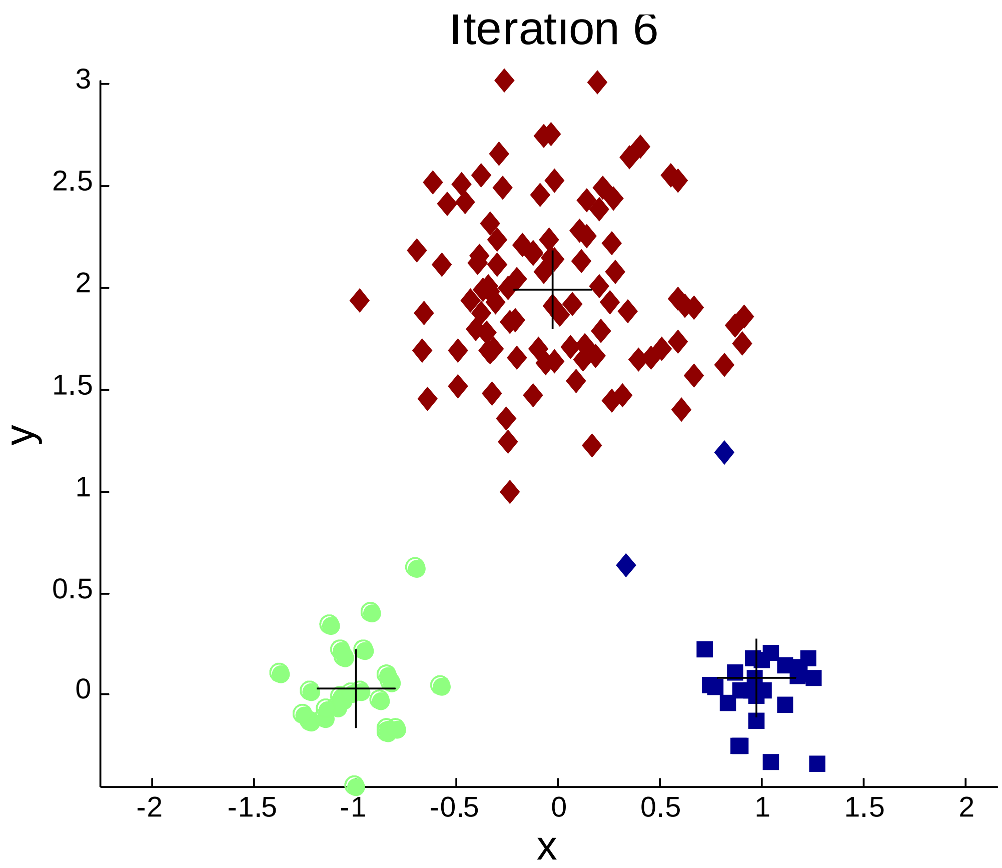


Optimal Clustering

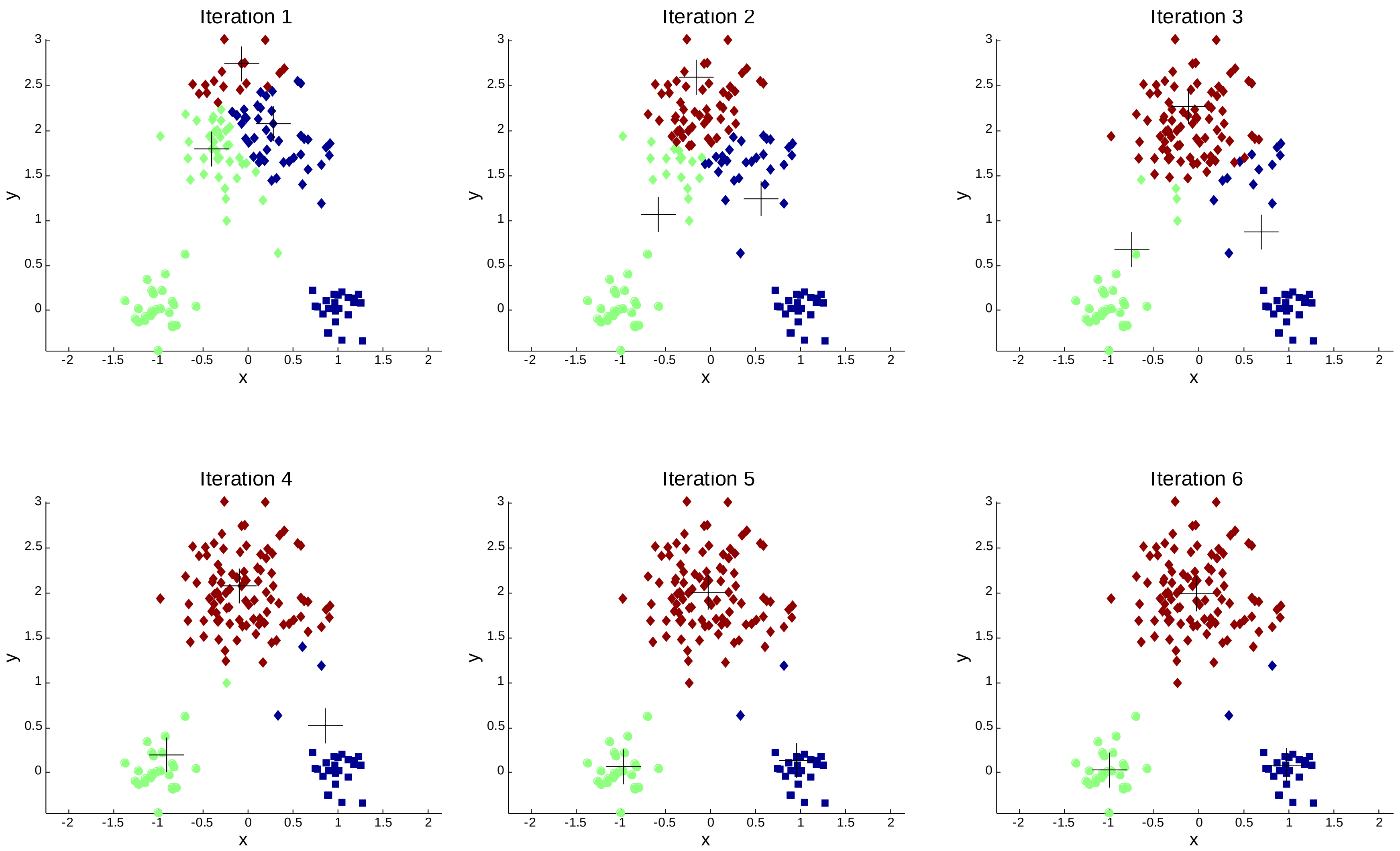


Sub-optimal Clustering

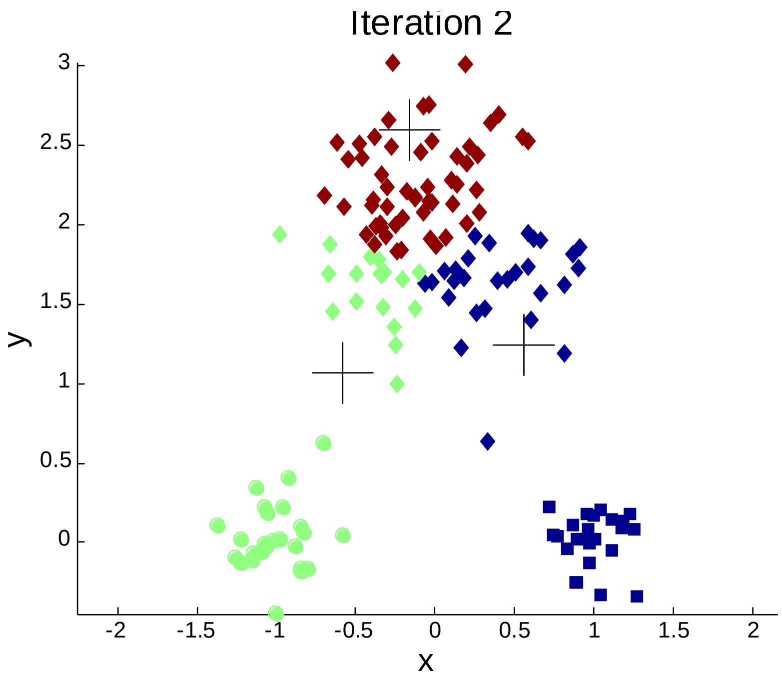
Importance of Choosing Initial Centroids



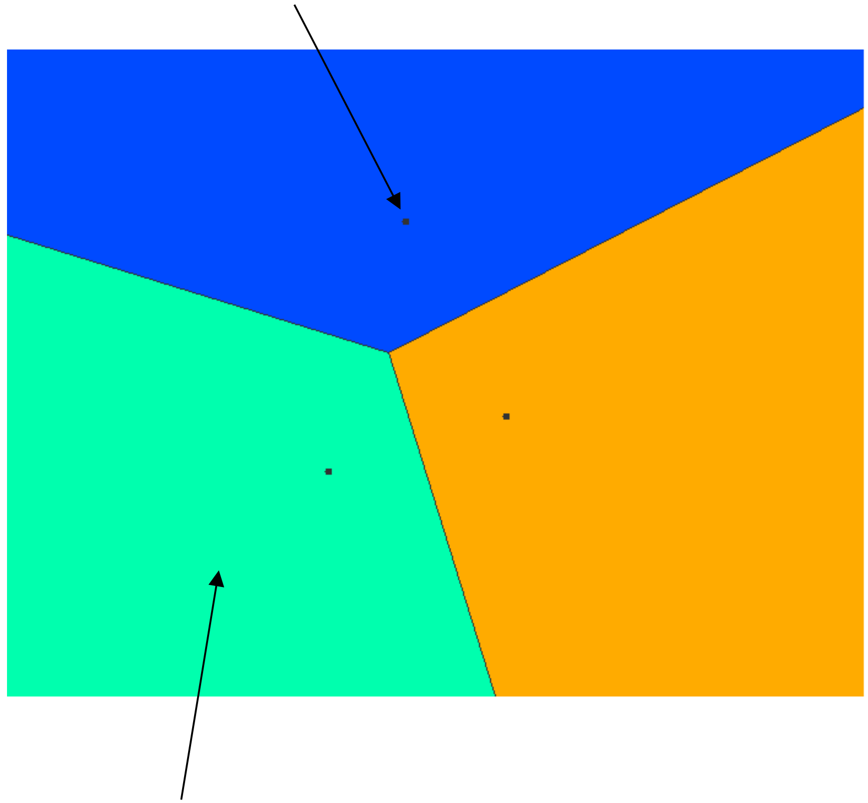
Importance of Choosing Initial Centroids



Clusters vs. Voronoi diagrams



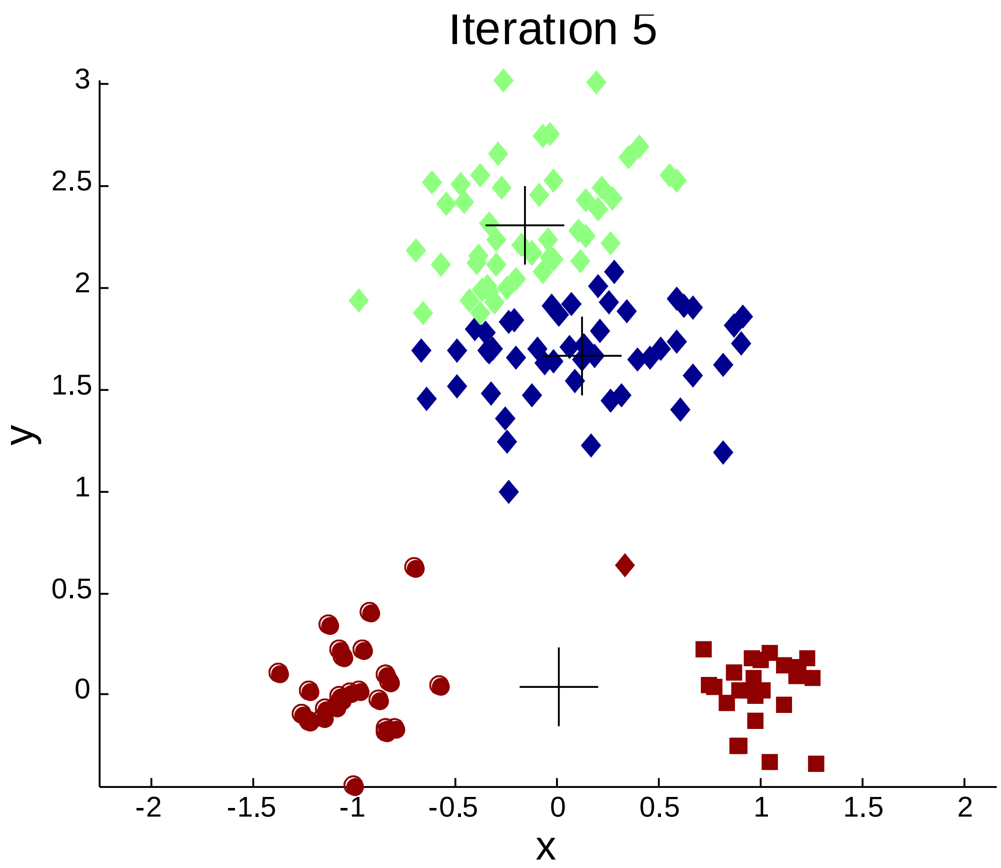
Reference point



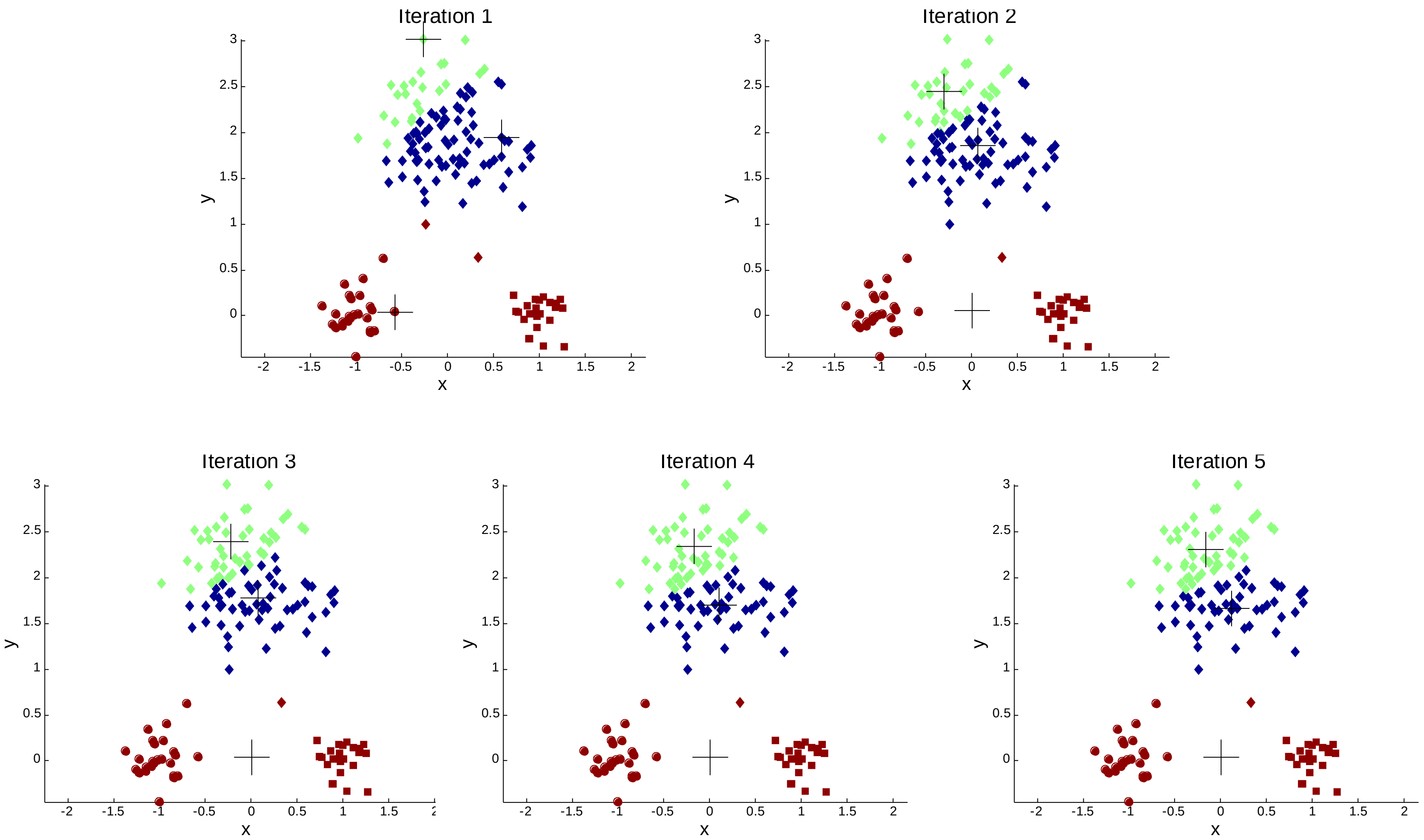
Voroni cell = set of points that are closer to a reference point than any other

http://www.cs.cornell.edu/home/chew/Delaunay.html

Importance of Choosing Initial Centroids …



Importance of Choosing Initial Centroids …



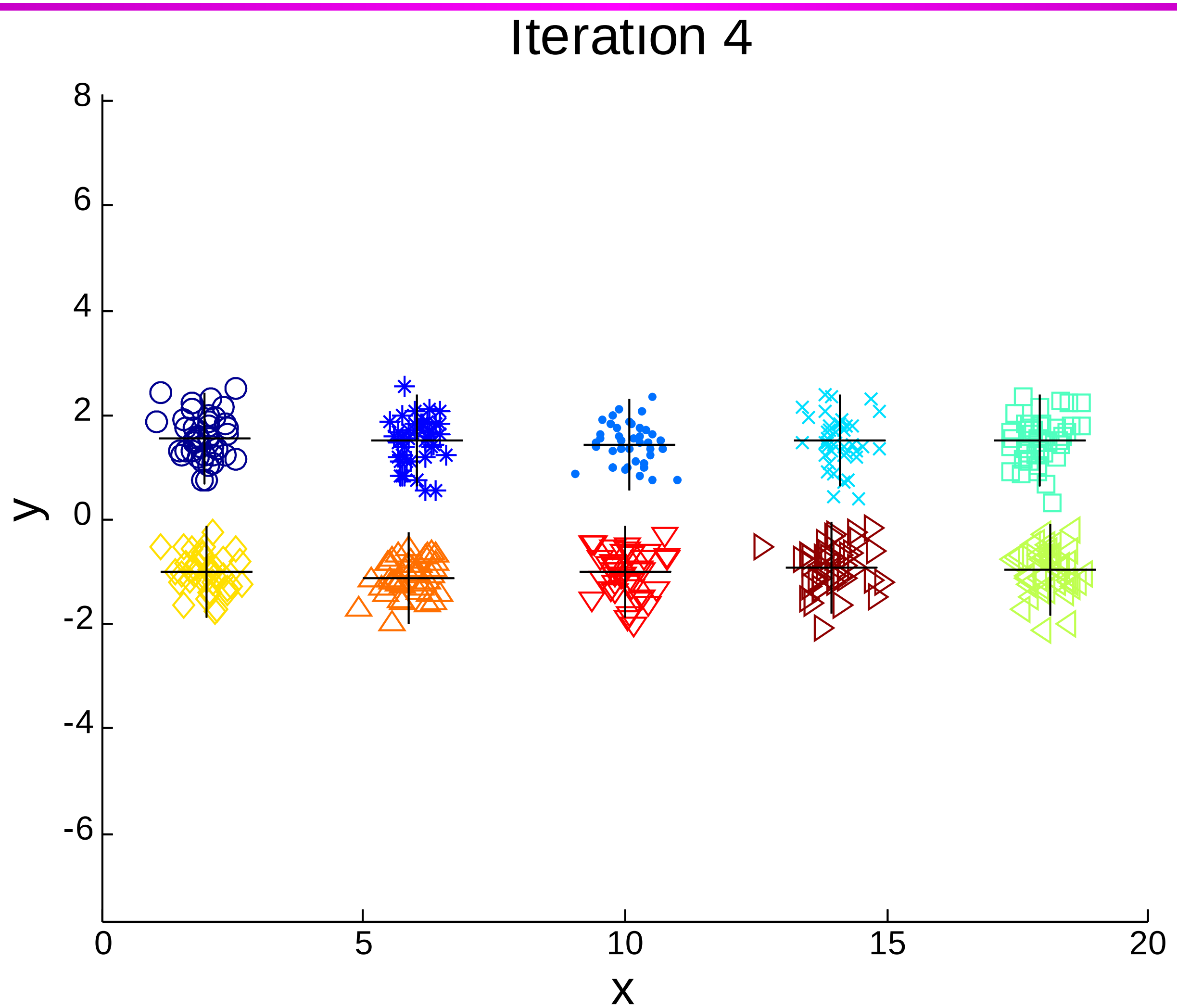
Problems with Selecting Initial Points

- 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

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

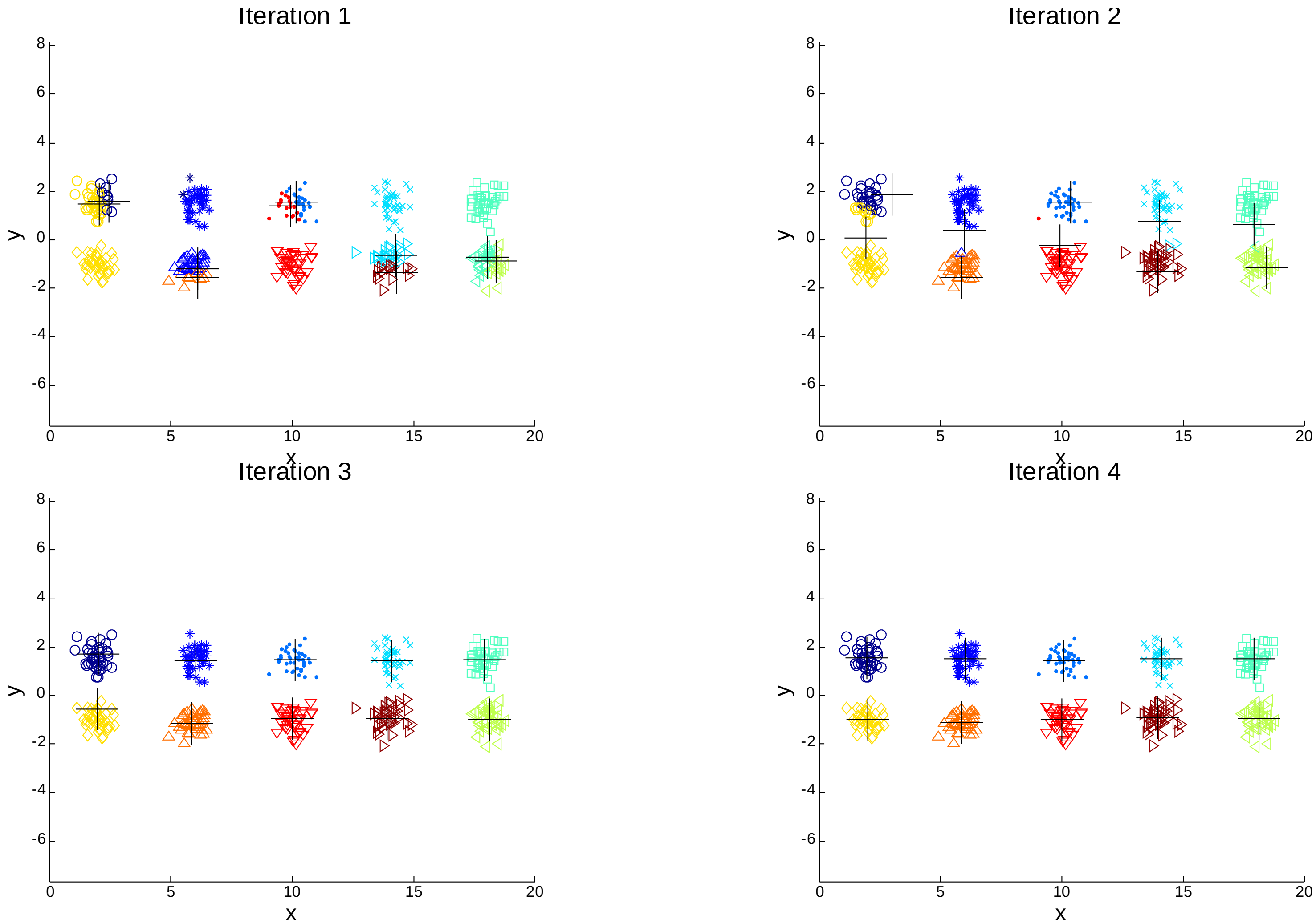
  - For example, if K = 10, then probability = $10!/10^{10}$ = 0.00036
  - Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't
  - Consider an example of five pairs of clusters

10 Clusters Example



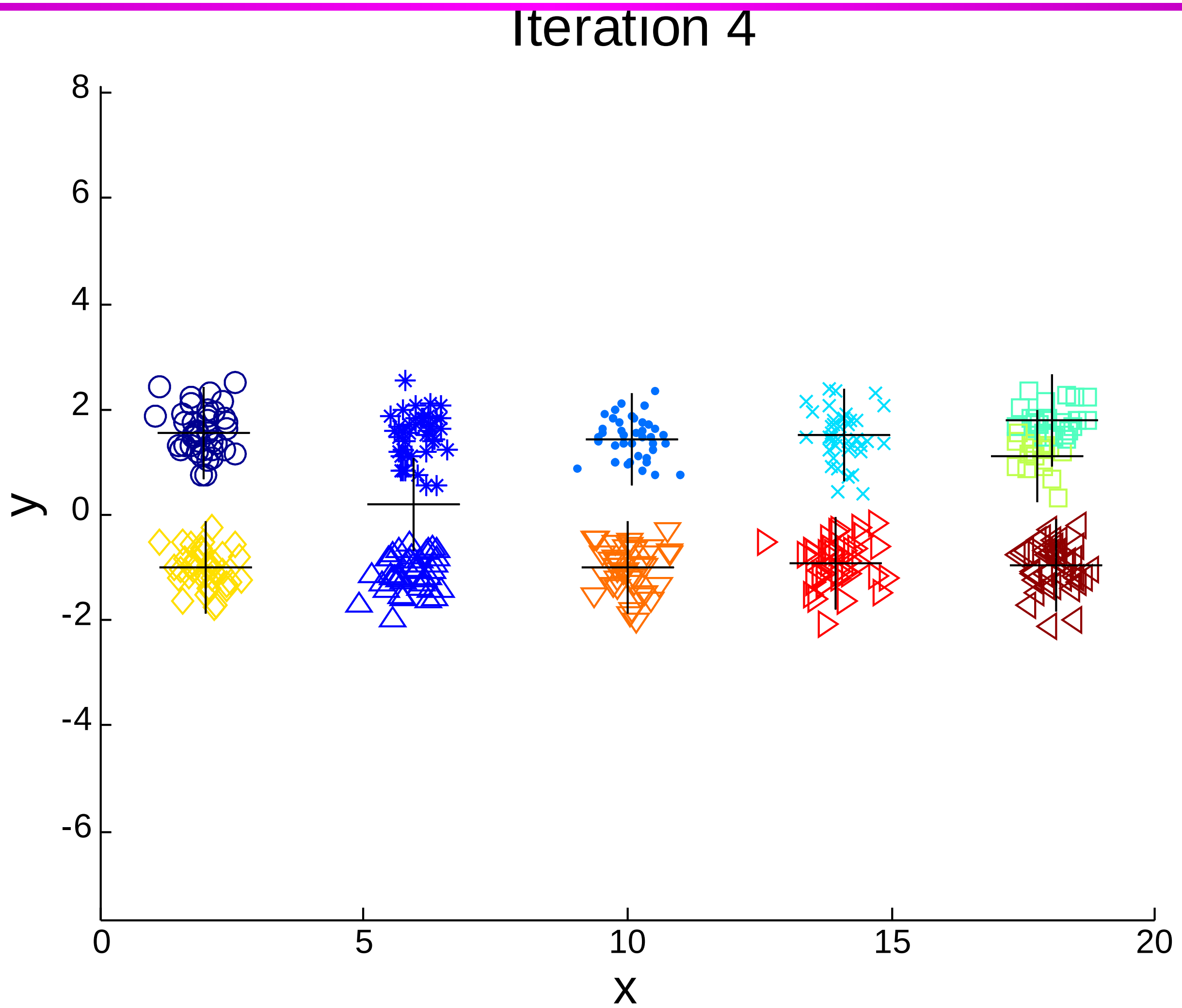
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



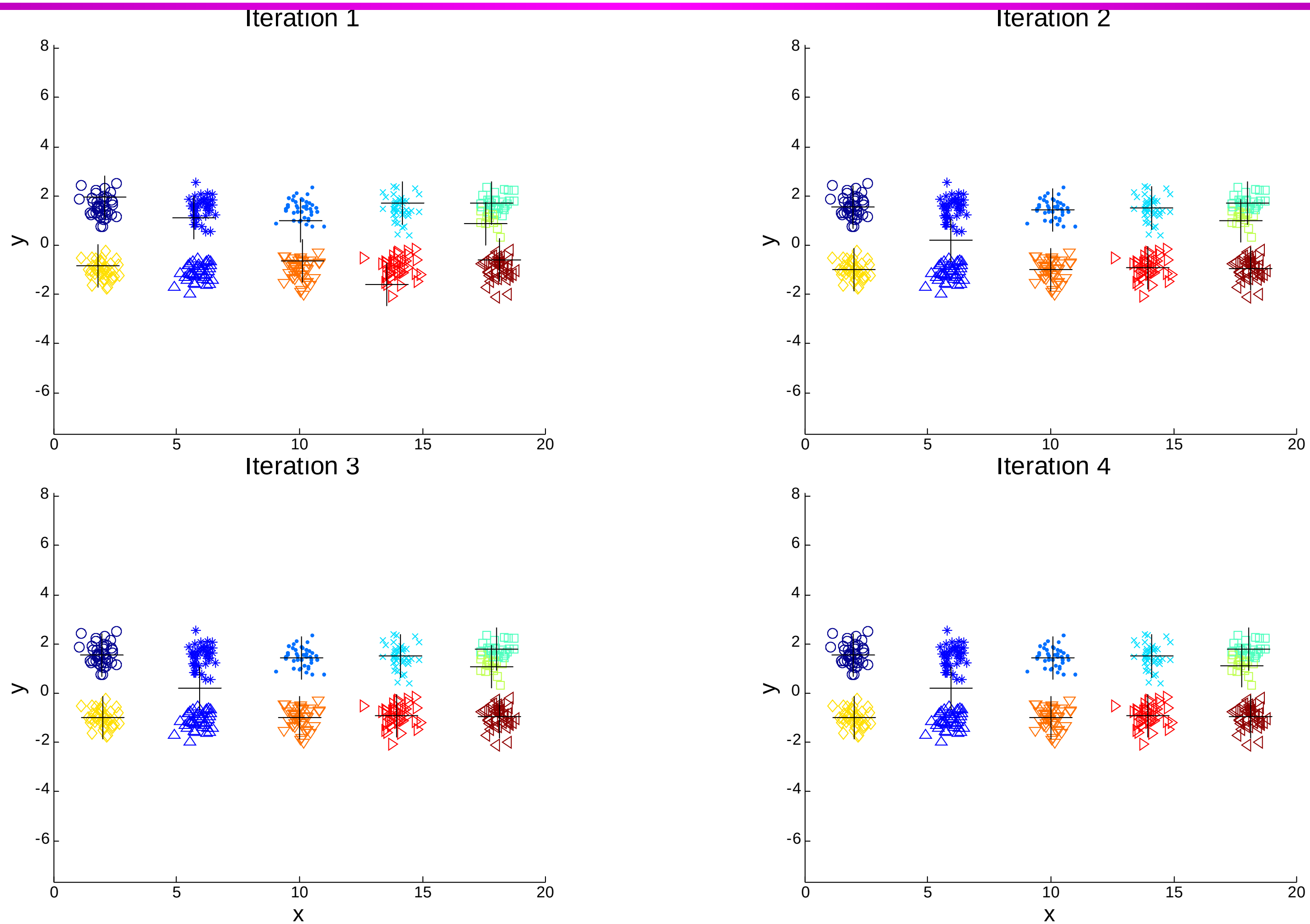
Starting with two initial centroids in one cluster of each pair of clusters

10 Clusters Example



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

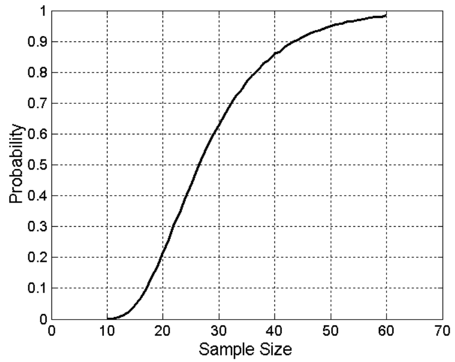
10 Clusters Example



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

Solutions to Initial Centroids Problem

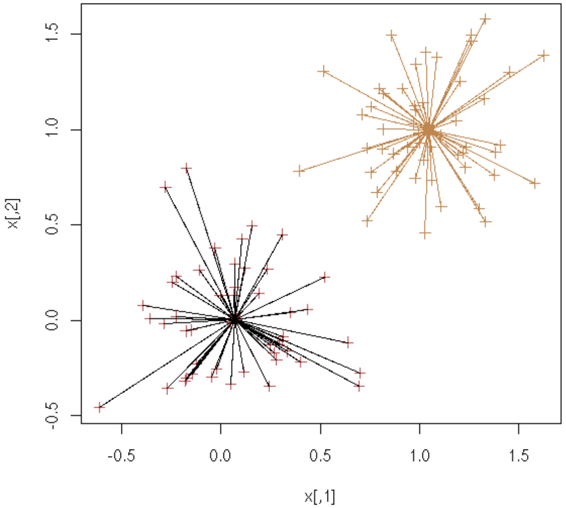
- Multiple runs
  - Helps, but probability is not on your side
- Sample and use hierarchical clustering to determine initial centroids
- Select more than k initial centroids and then select among these initial centroids
  - Select most widely separated
- Postprocessing
- Bisecting K-means
  - Not as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



Evaluating K-means Clusters

- Most common measure is Sum of Squared Errors (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

Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters

- Several strategies
  - Choose a point and assign it to the cluster
    - The point that contributes most to SSE
    - A random point from the cluster with highest SSE
  - If there are several empty clusters, the above can be repeated several times.

Pre-processing and Post-processing

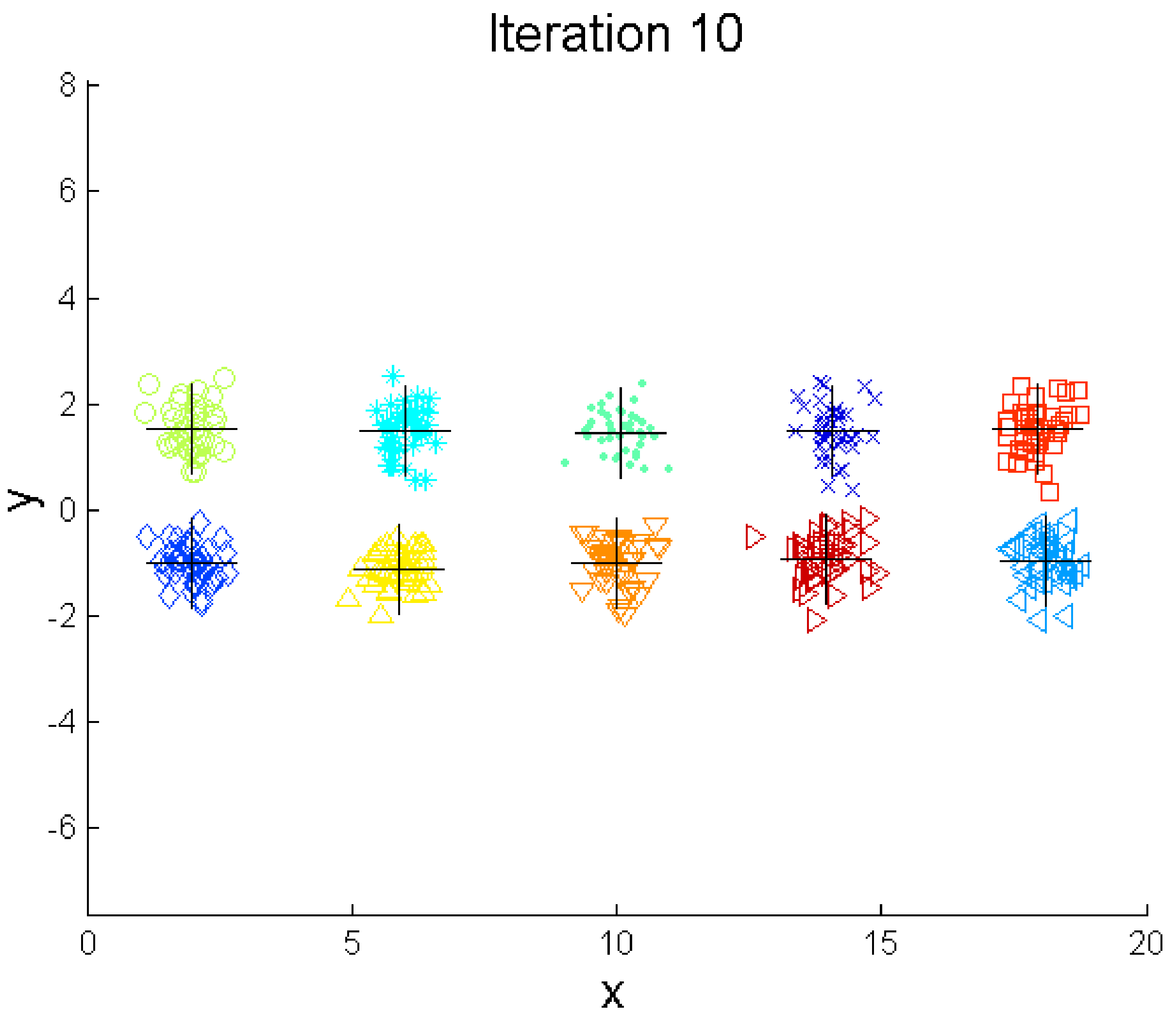
- Pre-processing
  – Normalize the data
  – Eliminate outliers
- Post-processing
  – Eliminate small clusters that may represent outliers
  – Split ‘loose’ clusters, i.e., clusters with relatively high SSE
  – Merge clusters that are ‘close’ and that have relatively low SSE
  – Can use these steps during the clustering process
    - ISODATA

Bisecting K-means

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

```
Initialize the list of clusters to contain the cluster containing all points.
repeat
   Select a cluster from the list of clusters
   for i = 1 to number_of_iterations do
      Bisect the selected cluster using basic K-means        <---- Bisect => K=2
   end for
   Add the two clusters from the bisection with the lowest SSE to the list of clusters.
until Until the list of clusters contains K clusters
```

Bisecting K-means Example

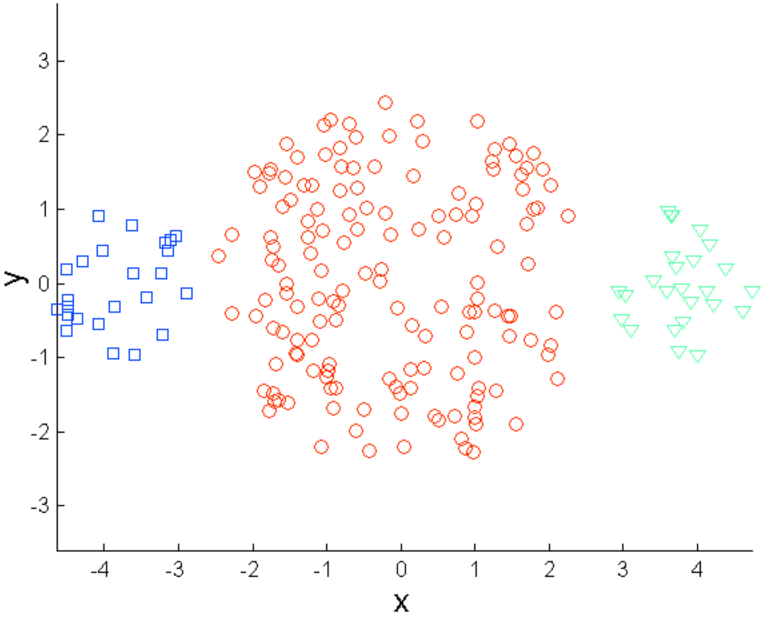


Limitations of K-means

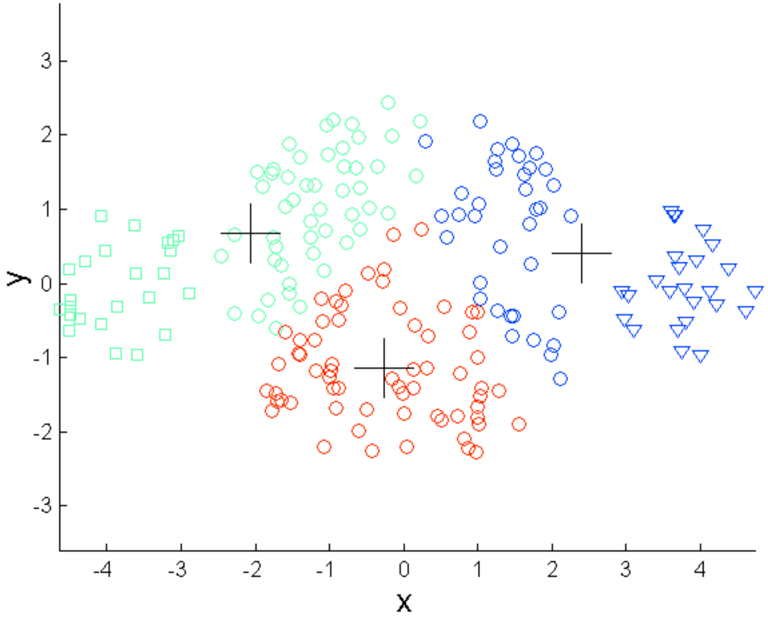
- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

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

Limitations of K-means: Differing Sizes

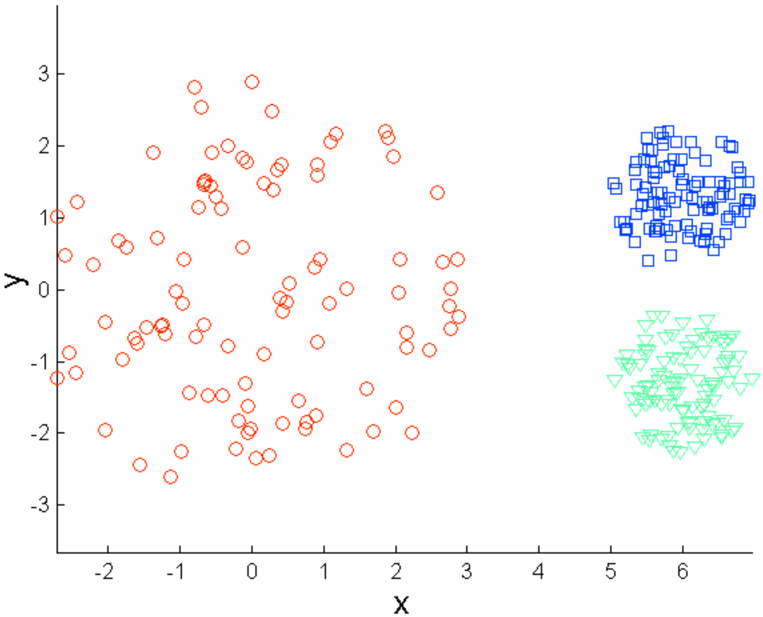


Original Points

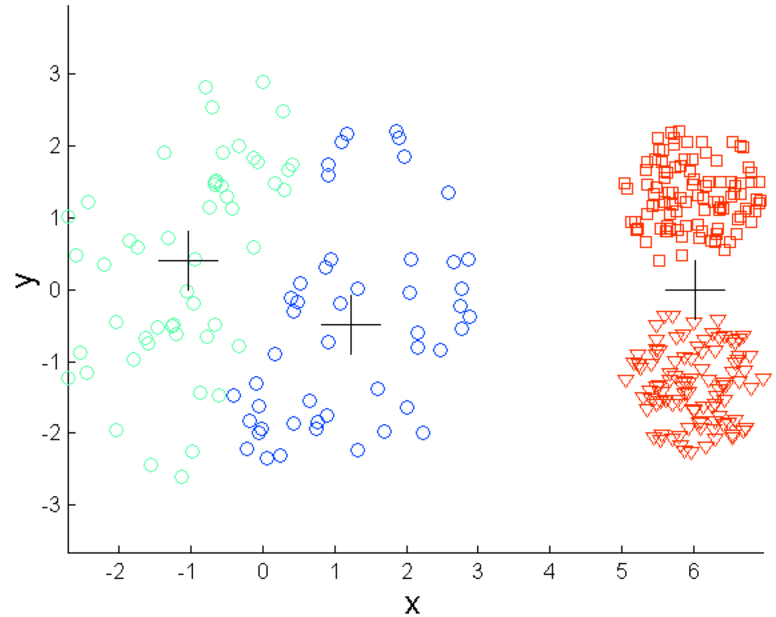


K-means (3 Clusters)

Limitations of K-means: Differing Density

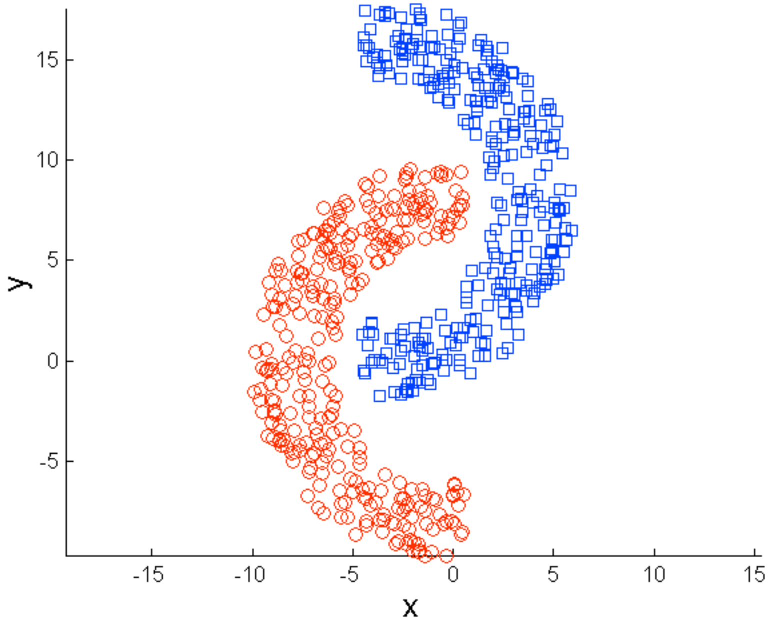


Original Points

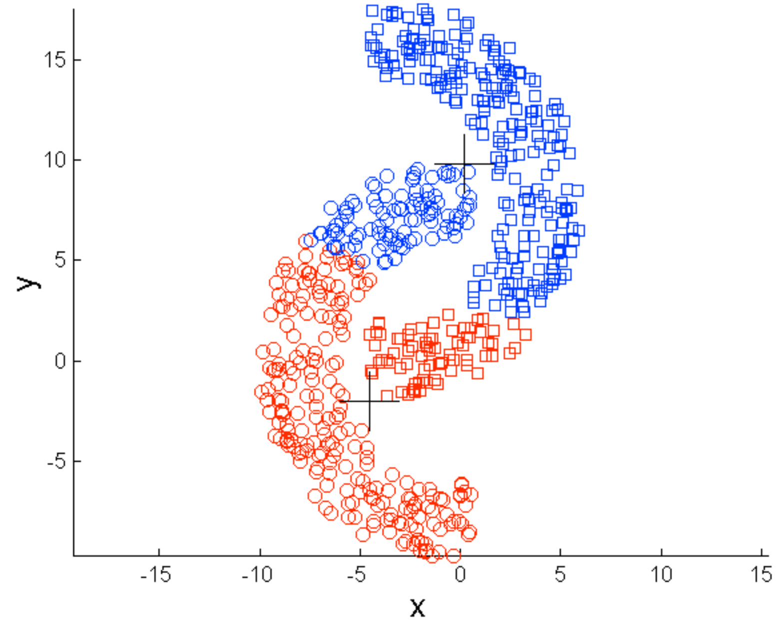


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

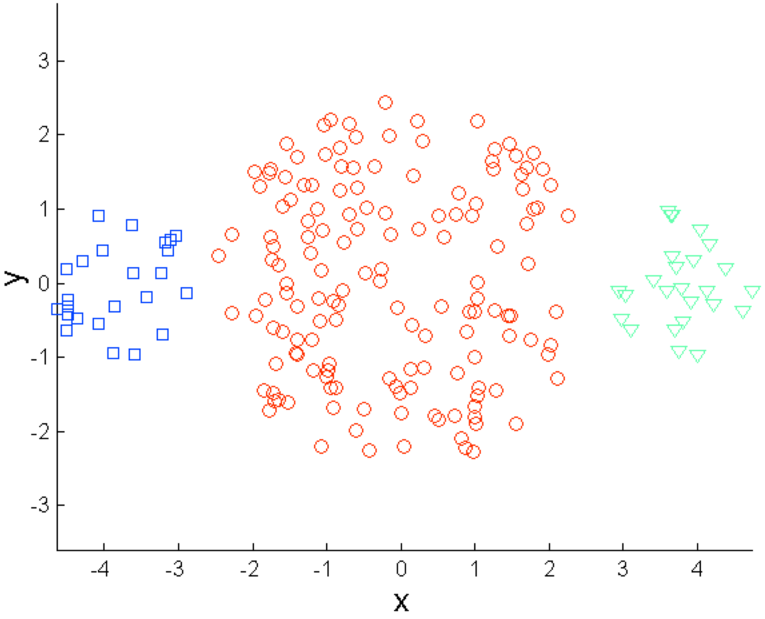


Original Points

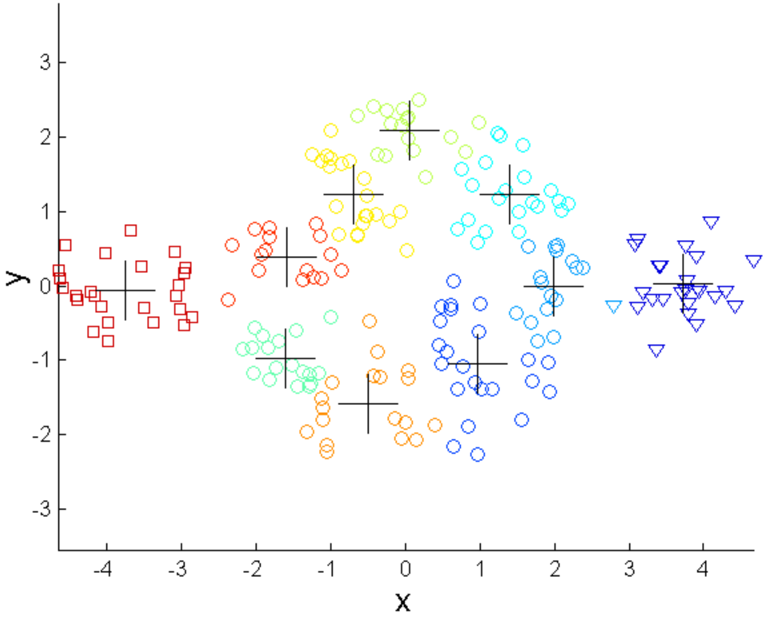


K-means (2 Clusters)

Overcoming K-means Limitations



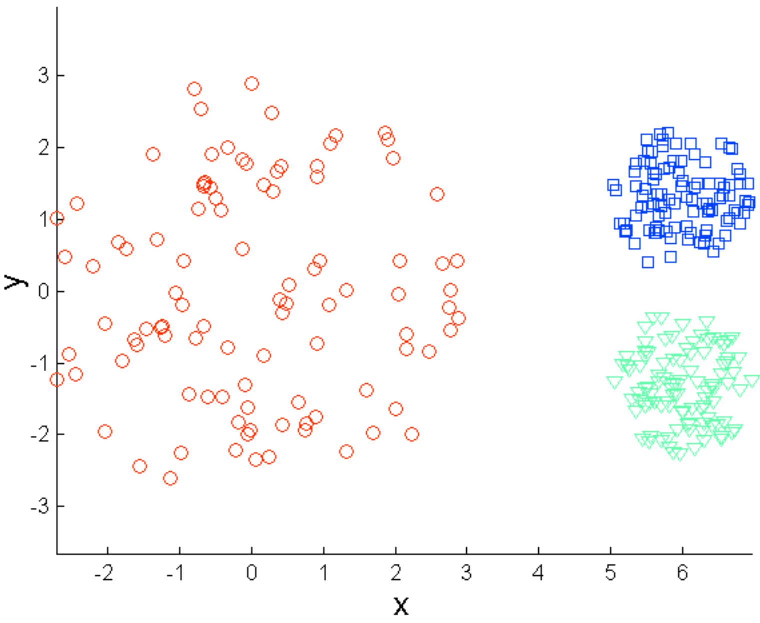
Original Points

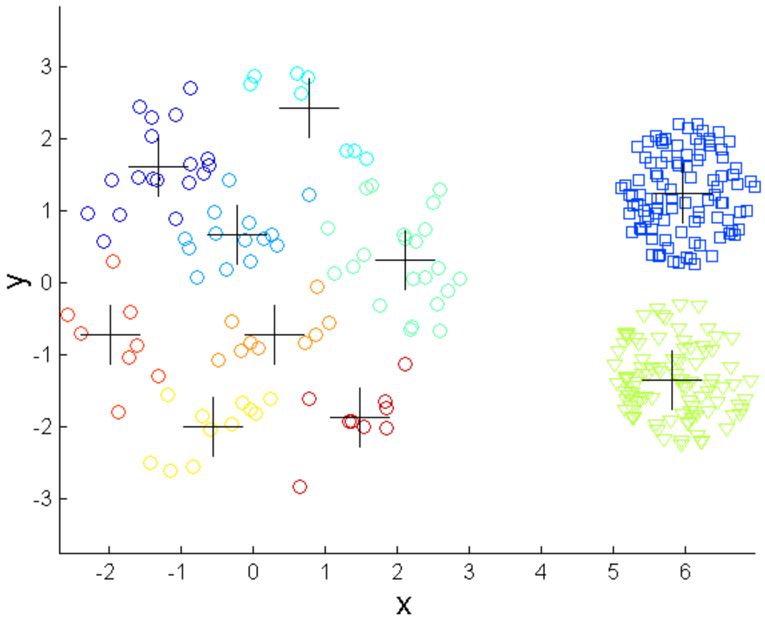


K-means Clusters

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

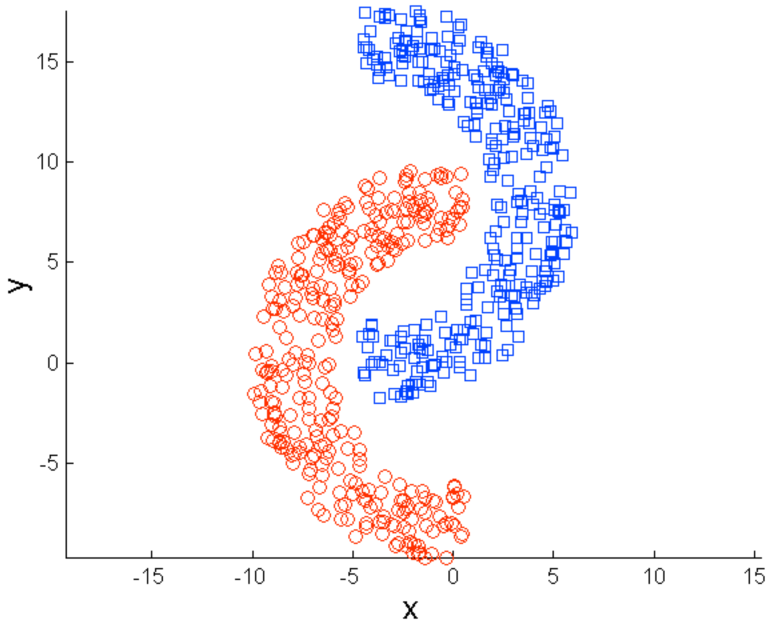
Overcoming K-means Limitations

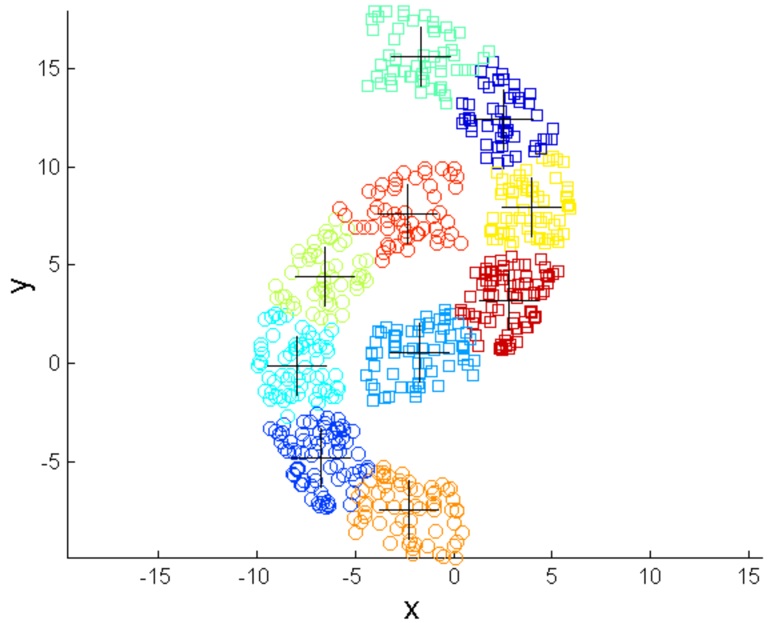




Original Points | **K-means Clusters**

Overcoming K-means Limitations





Original Points

K-means Clusters