

# ITCS 6156/8156 Spring 2024

## Machine Learning

# Clustering

Instructor: Hongfei Xue

Email: [hongfei.xue@charlotte.edu](mailto:hongfei.xue@charlotte.edu)

Class Meeting: Mon & Wed, 4:00 PM – 5:15 PM, Denny 109



Some content in the slides is based on Dr. Raquel Urtasun's lecture

# Unsupervised Learning

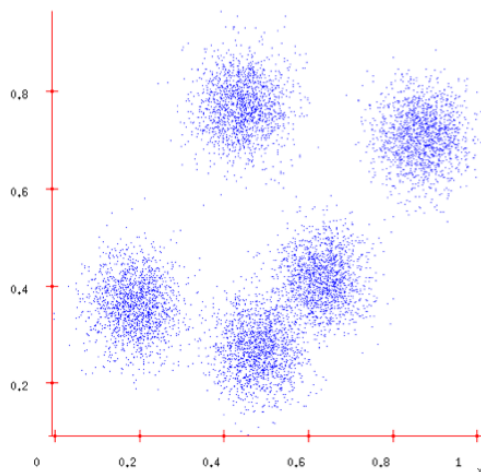
- **Supervised learning** algorithms have a clear goal: produce desired outputs for given inputs.
  - ▶ You are given  $\{(x^{(i)}, t^{(i)})\}$  during training (inputs and targets)
- Goal of **unsupervised learning** algorithms (no explicit feedback whether outputs of system are correct) less clear.
  - ▶ You are given the inputs  $\{x^{(i)}\}$  during training, labels are unknown.
- Tasks to consider:
  - ▶ Reduce dimensionality
  - ▶ Find clusters
  - ▶ Model data density
  - ▶ Find hidden causes
- Key utility
  - ▶ Compress data
  - ▶ Detect outliers
  - ▶ Facilitate other learning

# Major Types

- Primary problems, approaches in unsupervised learning fall into three classes:
  1. **Dimensionality reduction**: represent each input case using a small number of variables (e.g., principal components analysis, factor analysis, independent components analysis)
  2. **Clustering**: represent each input case using a prototype example (e.g., k-means, mixture models)
  3. **Density estimation**: estimating the probability distribution over the data space

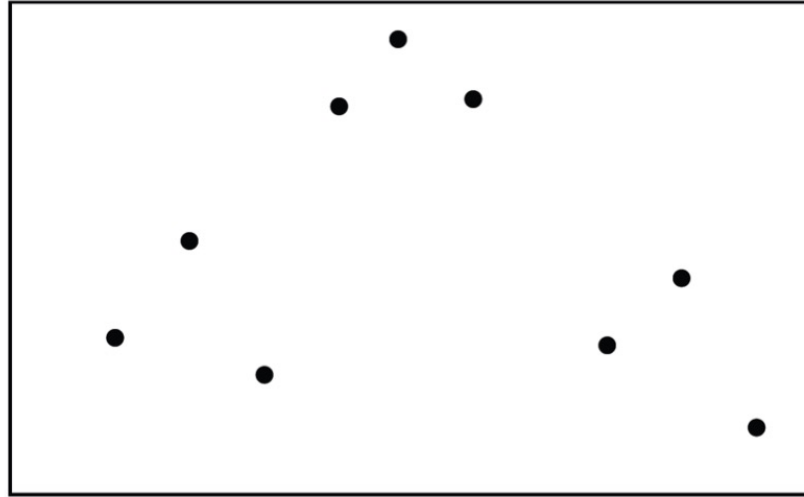
# Clustering

- Grouping  $N$  examples into  $K$  clusters one of canonical problems in unsupervised learning



- Motivation: prediction; lossy compression; outlier detection
- We assume that the data was generated from a number of different classes. The aim is to cluster data from the same class together.
  - ▶ How many classes?
  - ▶ Why not put each datapoint into a separate class?
- What is the objective function that is optimized by sensible clustering?

# Clustering

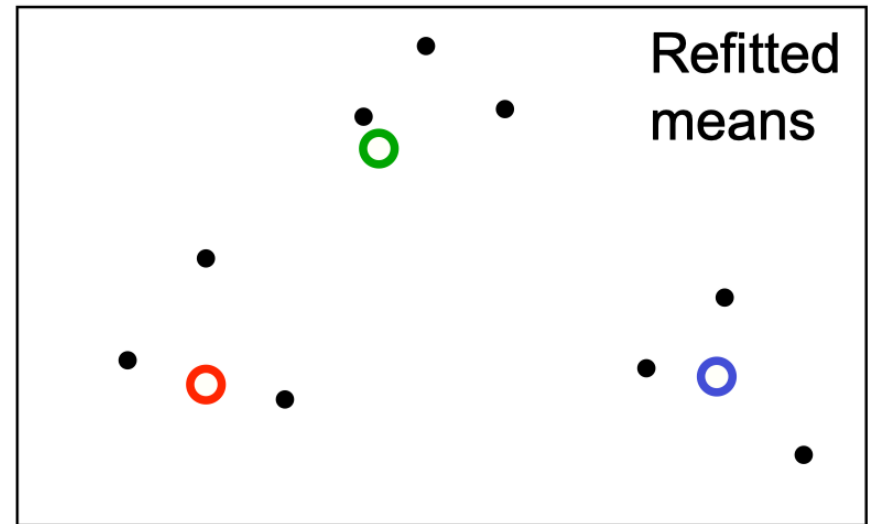
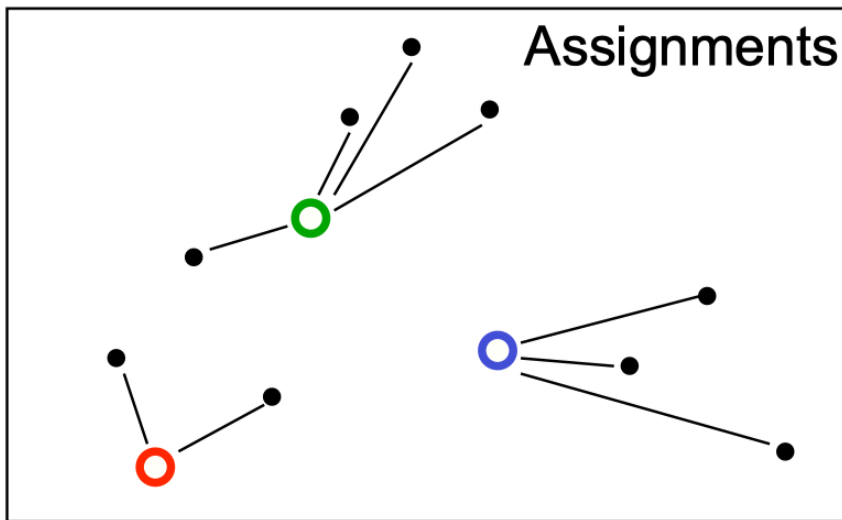


- Assume the data  $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(N)}\}$  lives in a Euclidean space,  $\mathbf{x}^{(n)} \in \mathbb{R}^d$ .
- Assume the data belongs to  $K$  classes (patterns)
- How can we identify those classes (data points that belong to each class)?

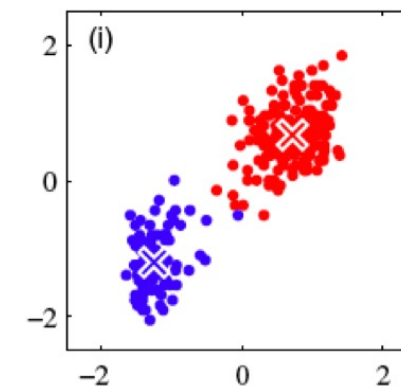
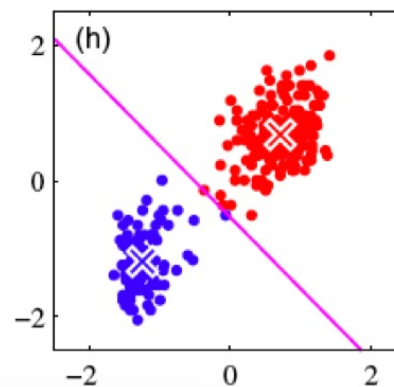
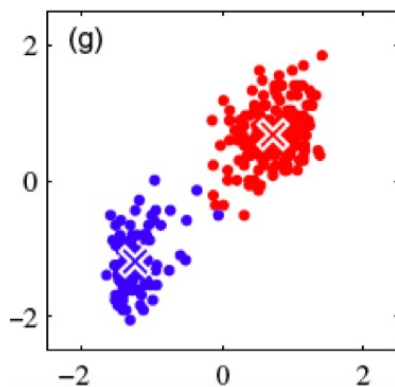
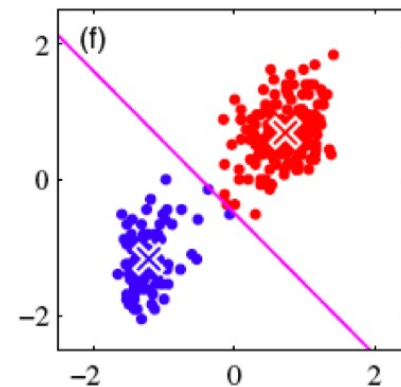
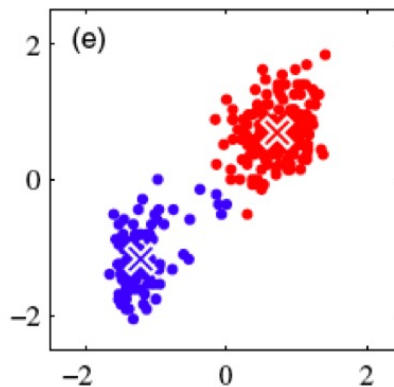
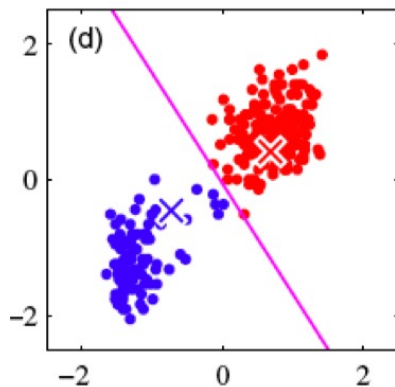
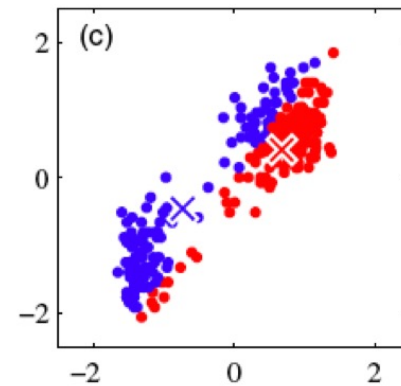
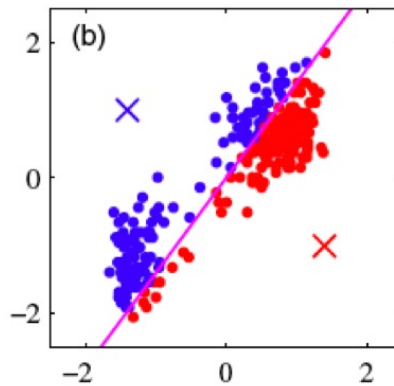
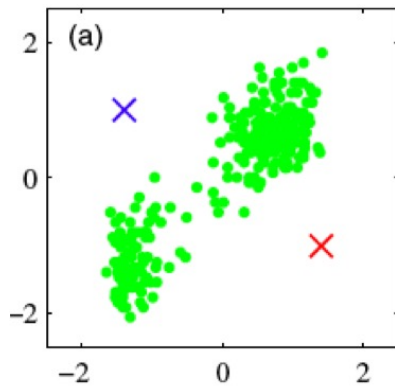


# K-means

- **Initialization**: randomly initialize cluster centers
- The algorithm iteratively alternates between two steps:
  - ▶ **Assignment step**: Assign each data point to the closest cluster
  - ▶ **Refitting step**: Move each cluster center to the center of gravity of the data assigned to it



# K-means



# K-means Objective

What is actually being optimized?

## K-means Objective:

Find cluster centers  $\mathbf{m}$  and assignments  $\mathbf{r}$  to minimize the sum of squared distances of data points  $\{\mathbf{x}^{(n)}\}$  to their assigned cluster centers

$$\min_{\{\mathbf{m}\}, \{\mathbf{r}\}} J(\{\mathbf{m}\}, \{\mathbf{r}\}) = \min_{\{\mathbf{m}\}, \{\mathbf{r}\}} \sum_{n=1}^N \sum_{k=1}^K r_k^{(n)} \|\mathbf{m}_k - \mathbf{x}^{(n)}\|^2$$
$$\text{s.t. } \sum_k r_k^{(n)} = 1, \forall n, \quad \text{where } r_k^{(n)} \in \{0, 1\}, \forall k, n$$

where  $r_k^{(n)} = 1$  means that  $\mathbf{x}^{(n)}$  is assigned to cluster  $k$  (with center  $\mathbf{m}_k$ )

- **Optimization method** is a form of coordinate descent ("block coordinate descent")
  - ▶ Fix centers, optimize assignments (choose cluster whose mean is closest)
  - ▶ Fix assignments, optimize means (average of assigned datapoints)



# The K-means Algorithm

- **Initialization:** Set K cluster means  $\mathbf{m}_1, \dots, \mathbf{m}_K$  to random values
- Repeat until convergence (until assignments do not change):
  - ▶ **Assignment:** Each data point  $\mathbf{x}^{(n)}$  assigned to nearest mean

$$\hat{k}^n = \arg \min_k d(\mathbf{m}_k, \mathbf{x}^{(n)})$$

(with, for example, L2 norm:  $\hat{k}^n = \arg \min_k \|\mathbf{m}_k - \mathbf{x}^{(n)}\|^2$ )

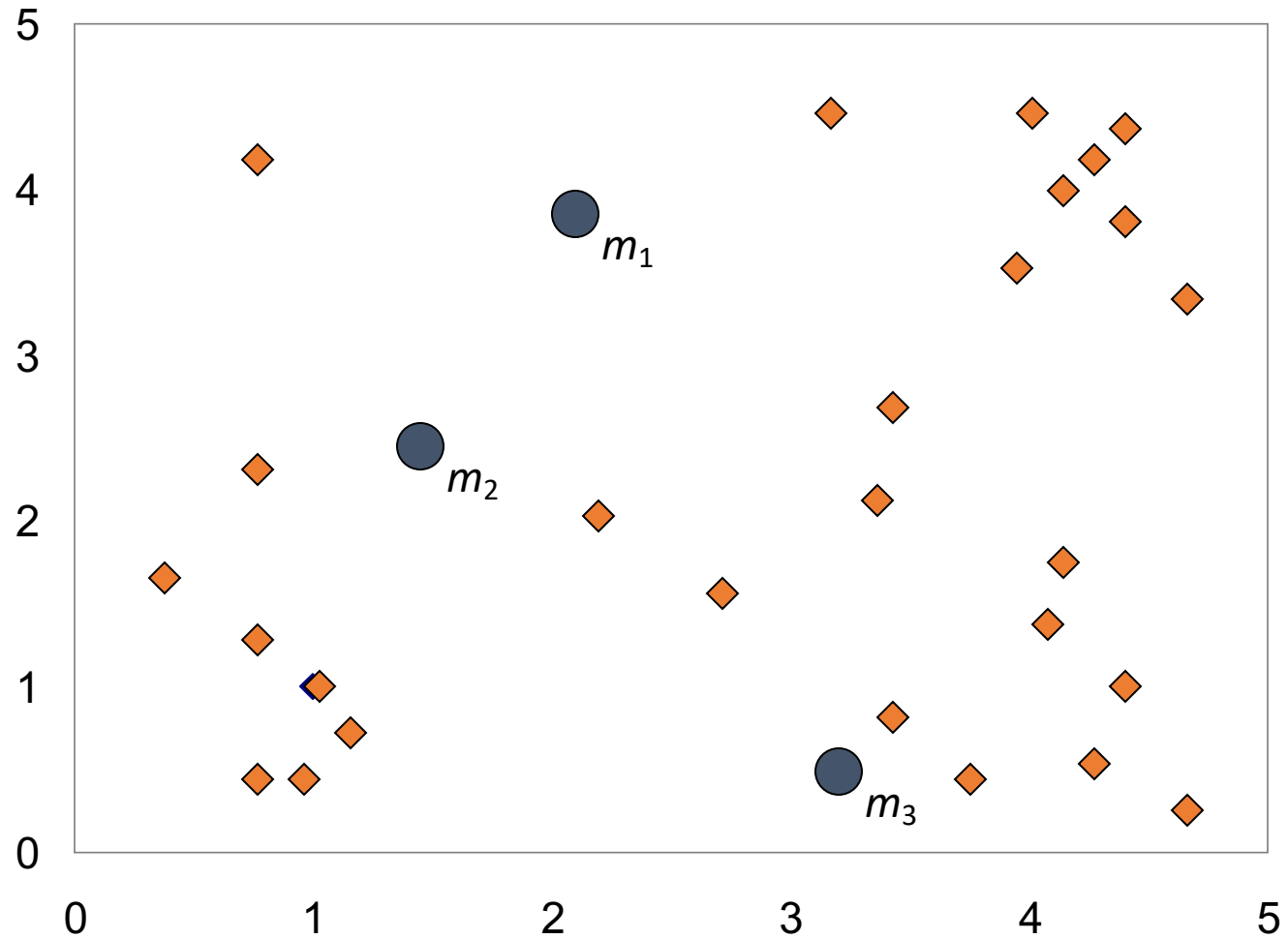
and **Responsibilities** (1 of k encoding)

$$r_k^{(n)} = 1 \iff \hat{k}^{(n)} = k$$

- ▶ **Update:** Model parameters, means are adjusted to match sample means of data points they are responsible for:

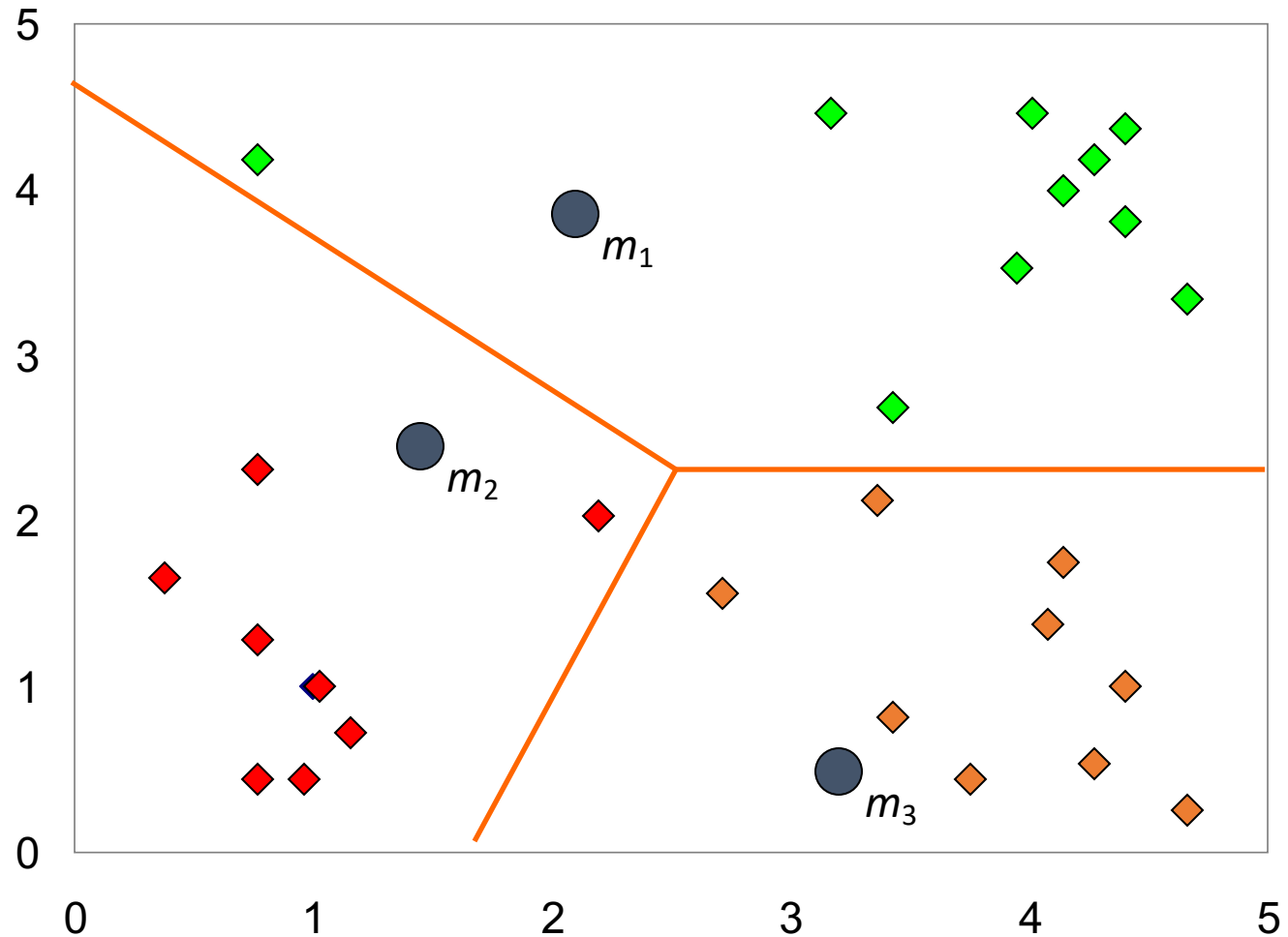
$$\mathbf{m}_k = \frac{\sum_n r_k^{(n)} \mathbf{x}^{(n)}}{\sum_n r_k^{(n)}}$$

# Why K-means Converges



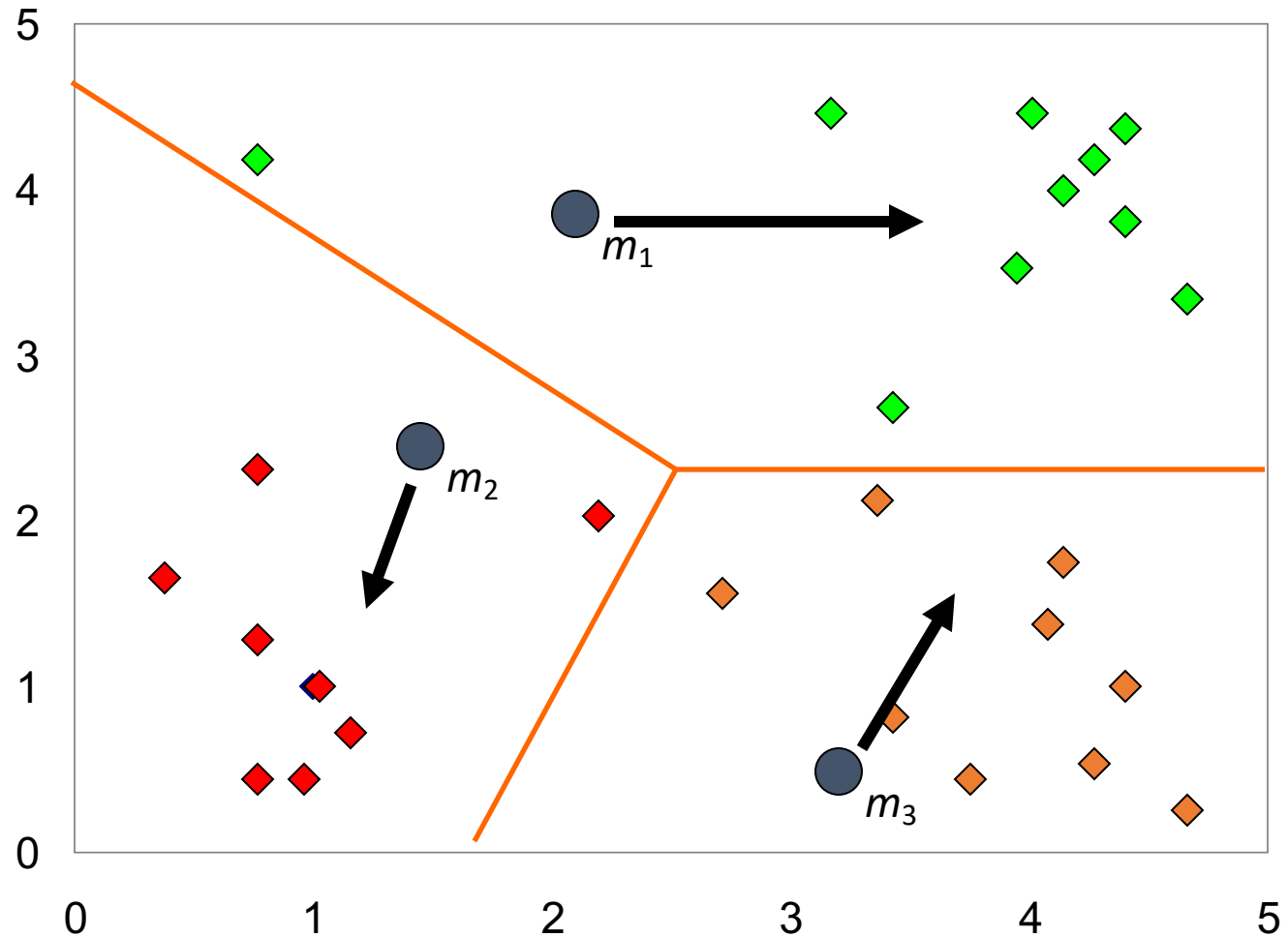
- Initialize

# Why K-means Converges



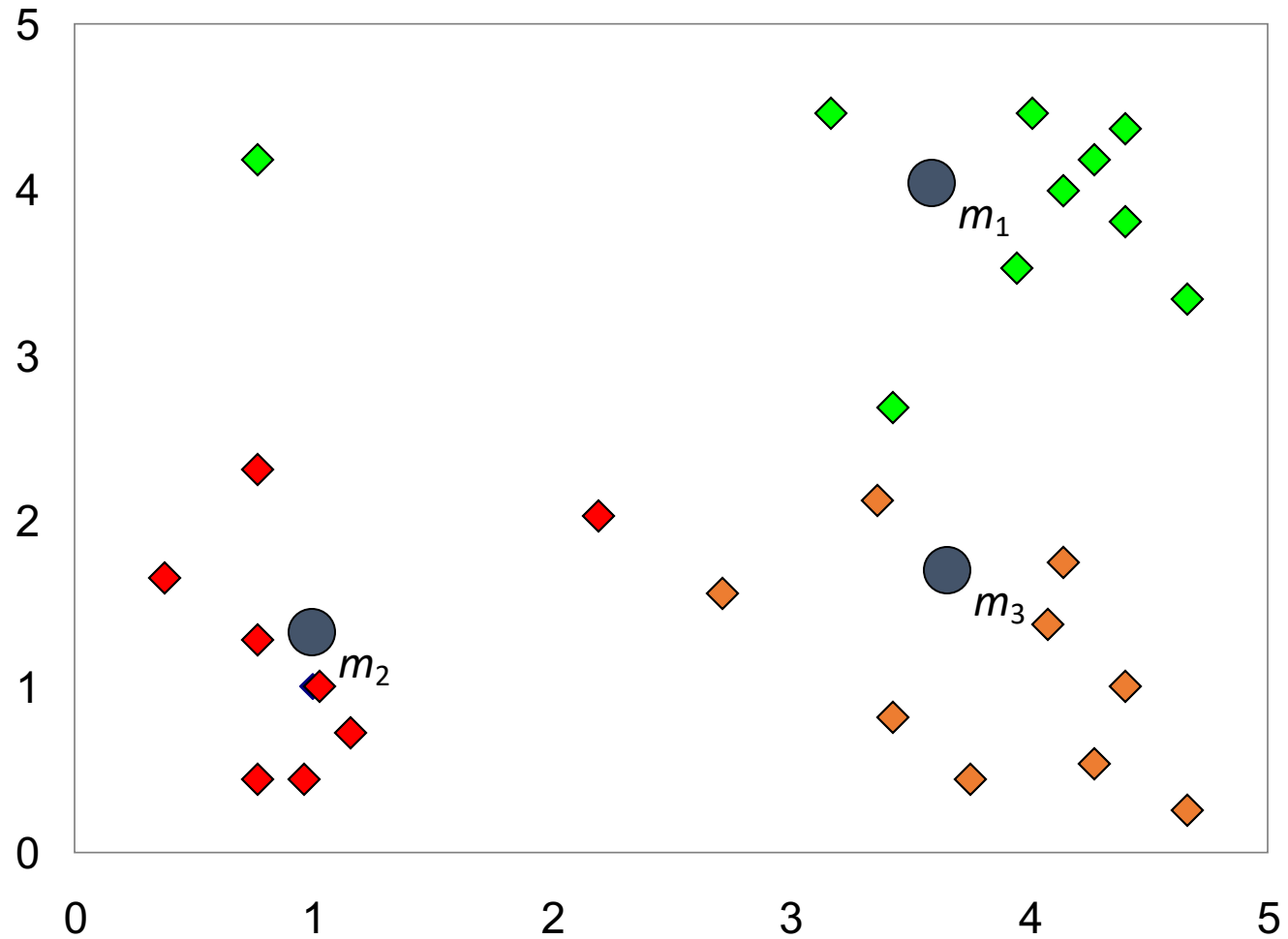
- Split into clusters.

# Why K-means Converges

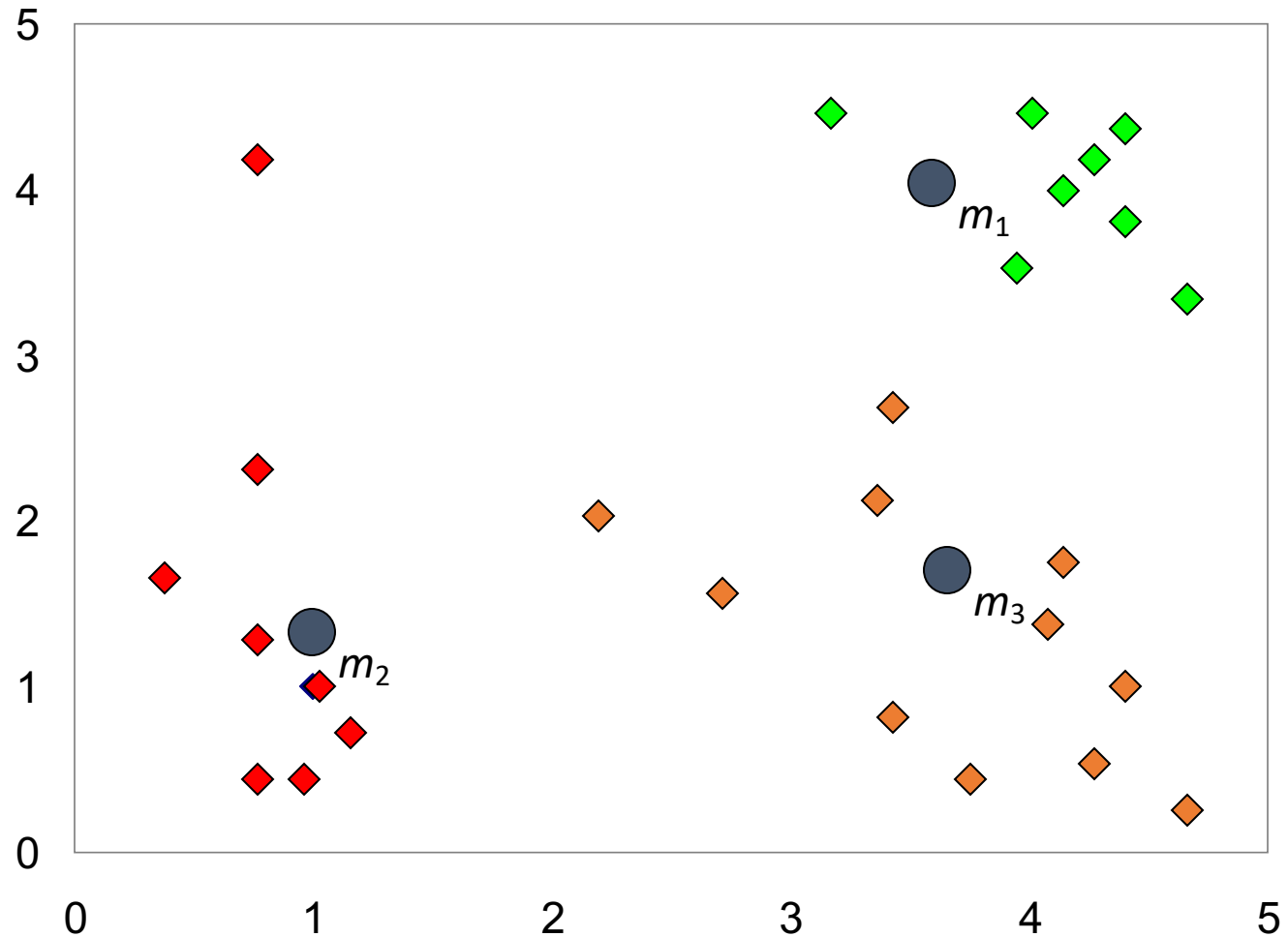


- Overall distances are reduced since the distances in each cluster is reduced.

# Why K-means Converges



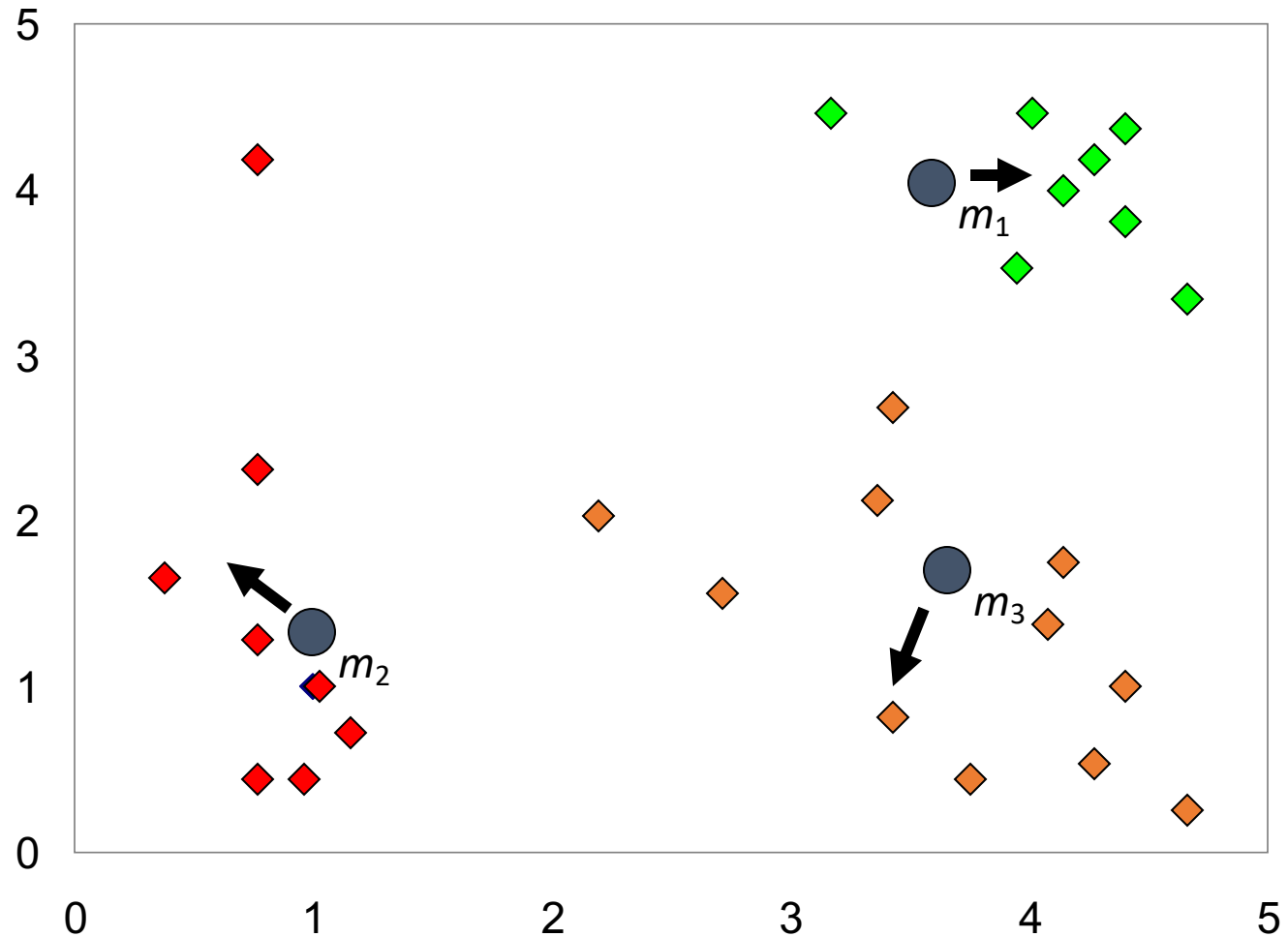
# Why K-means Converges



- Overall distances are reduced since the points are assigned to closer centroids.

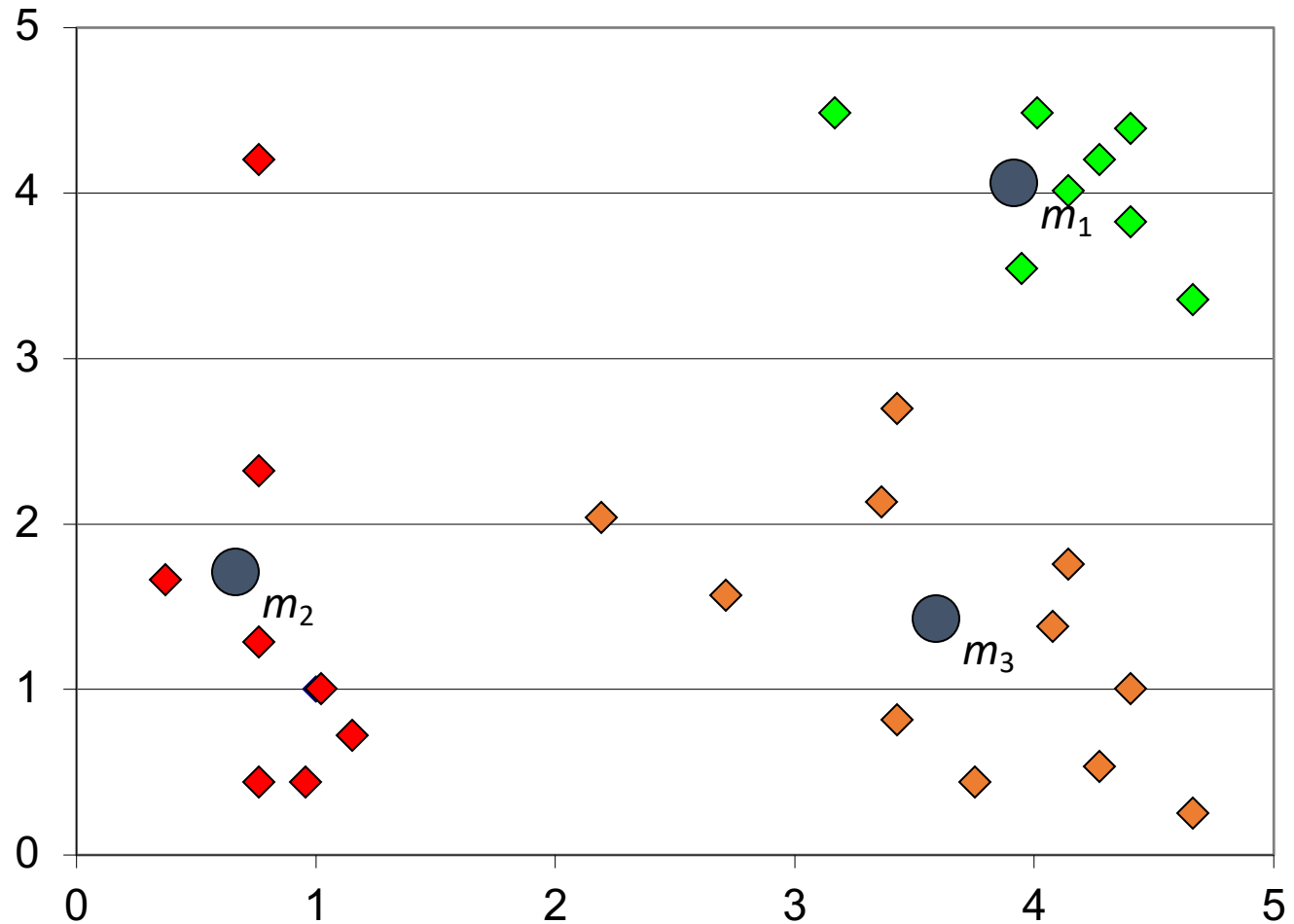


# Why K-means Converges



- Overall distances are reduced since the distances in each cluster is reduced.

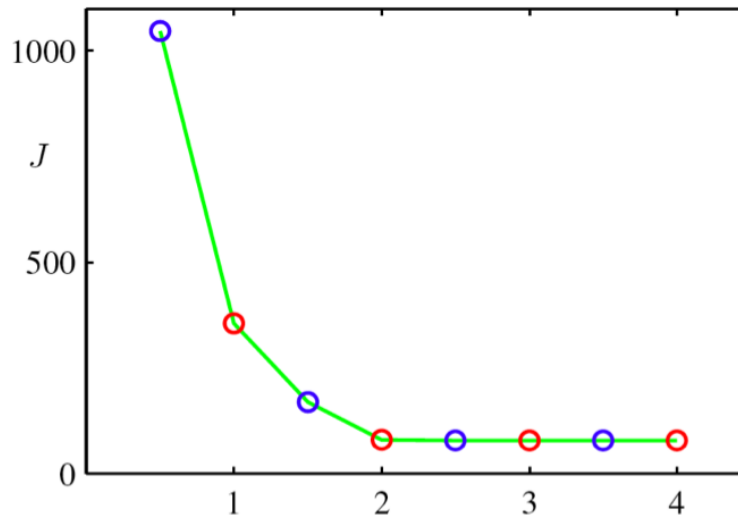
# Why K-means Converges



- Converge

# Why K-means Converges

- Whenever an assignment is changed, the sum squared distances  $J$  of data points from their assigned cluster centers is reduced.
- Whenever a cluster center is moved,  $J$  is reduced.
- **Test for convergence:** If the assignments do not change in the assignment step, we have converged (to at least a local minimum).



- K-means cost function after each E step (blue) and M step (red). The algorithm has converged after the third M step

# Comments on the K-Means Method

- **Strength**

- Efficient:  $O(tkn)$ , where  $n$  is # objects,  $k$  is # clusters, and  $t$  is # iterations.  
Normally,  $k, t \ll n$
- Easy to implement

- **Issues**

- Need to specify  $K$ , the number of clusters
- Local minimum– Initialization matters
- Empty clusters may appear

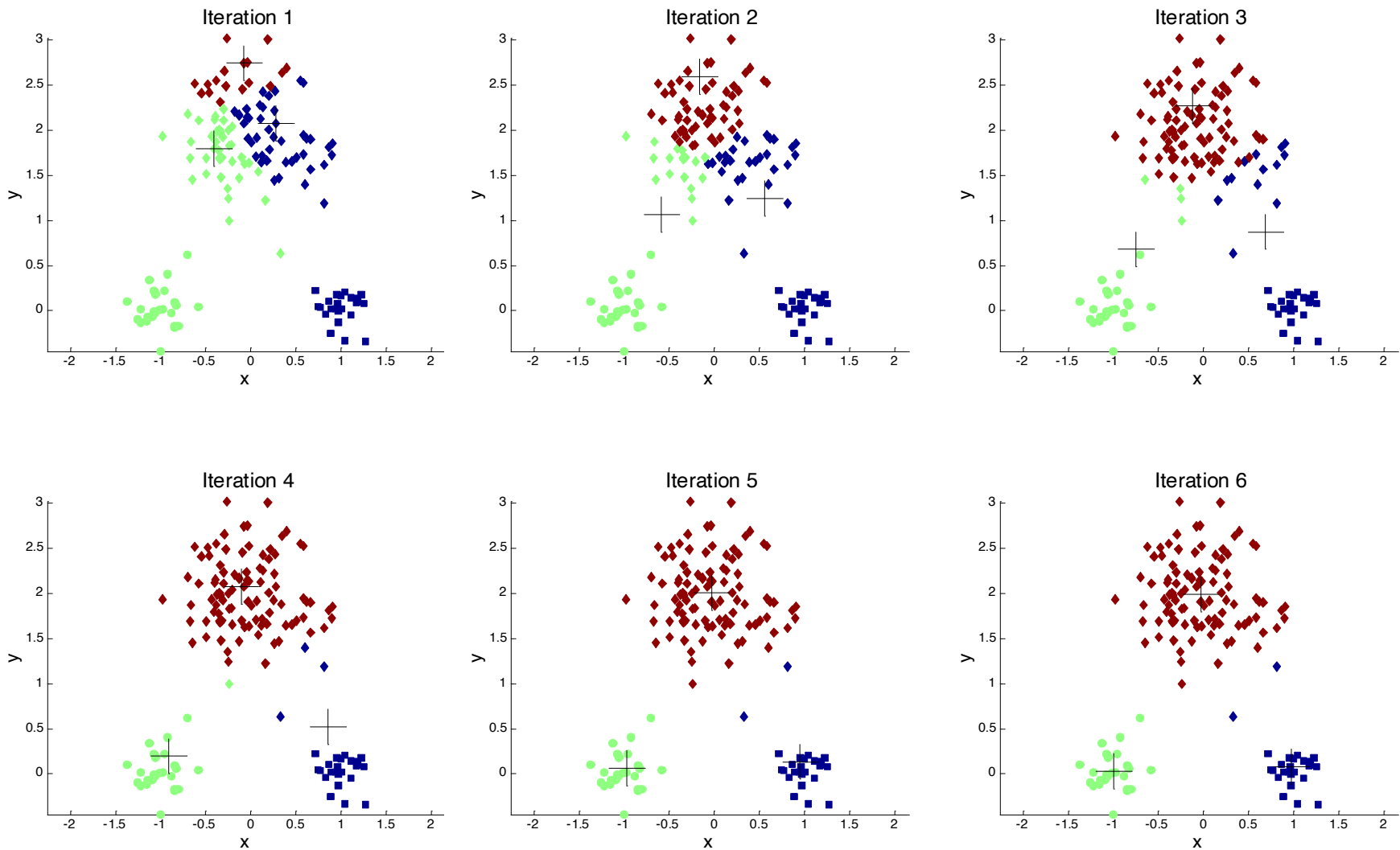
# Problems with Selecting Initial Points

- 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

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

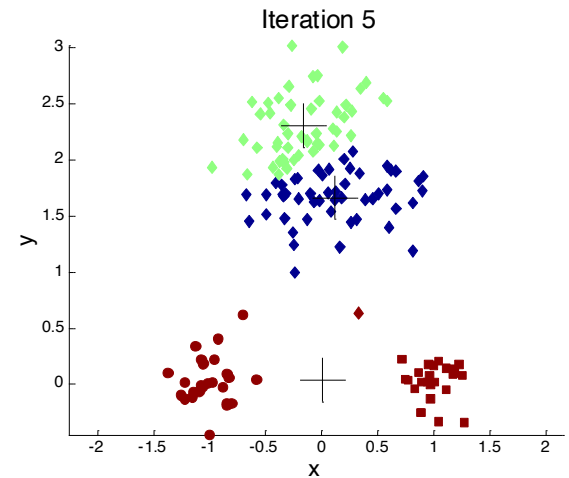
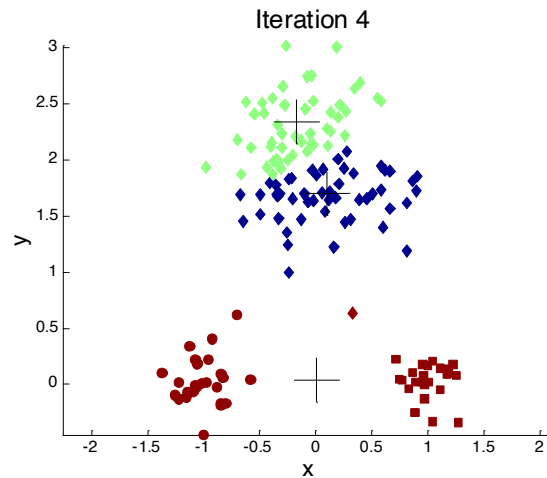
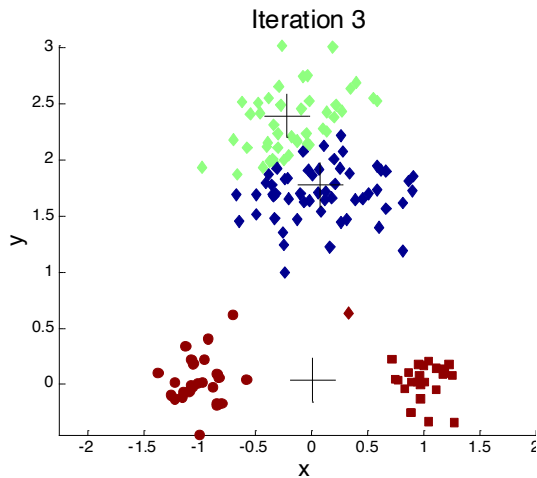
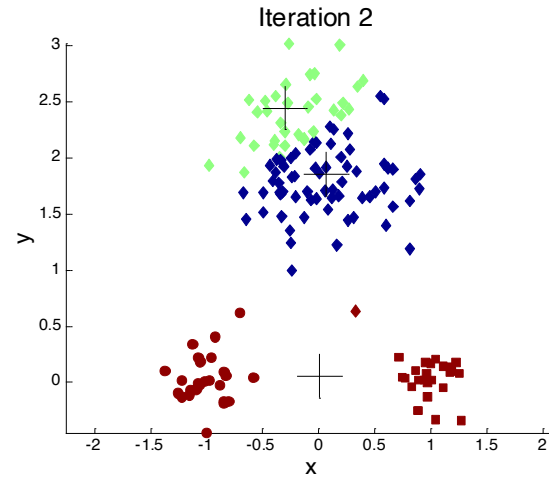
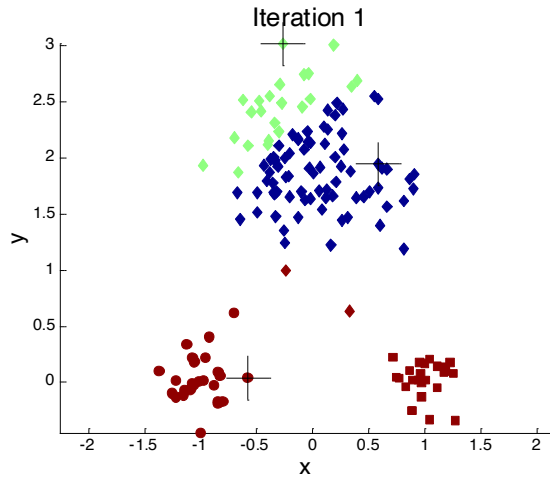
- For example, if  $K = 10$ , then probability =  $10!/10^{10} = 0.00036$
- Sometimes the initial centroids will readjust themselves in ‘right’ way, and sometimes they don’t

# Problems with Selecting Initial Points



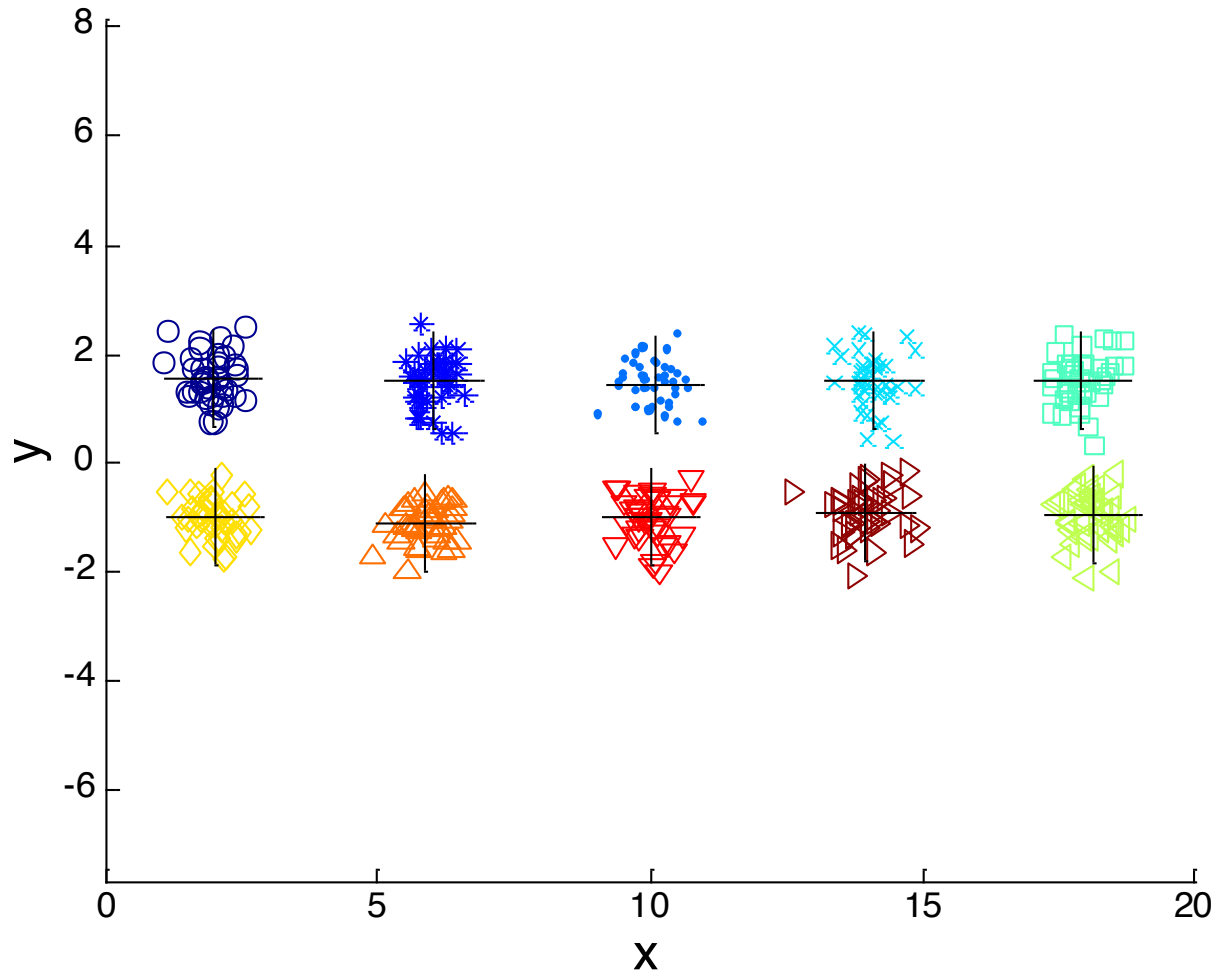


# Problems with Selecting Initial Points



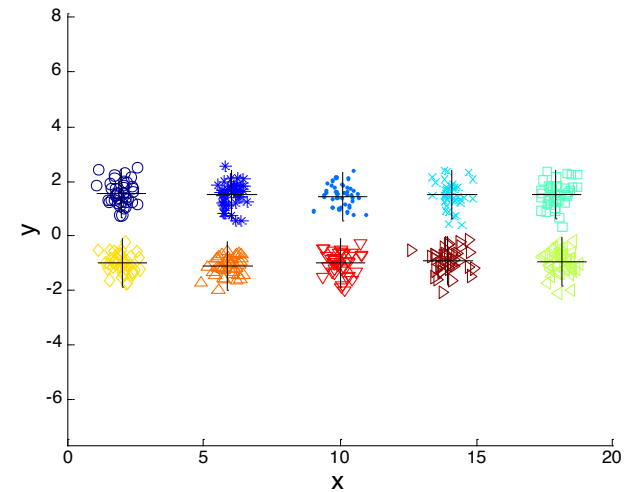
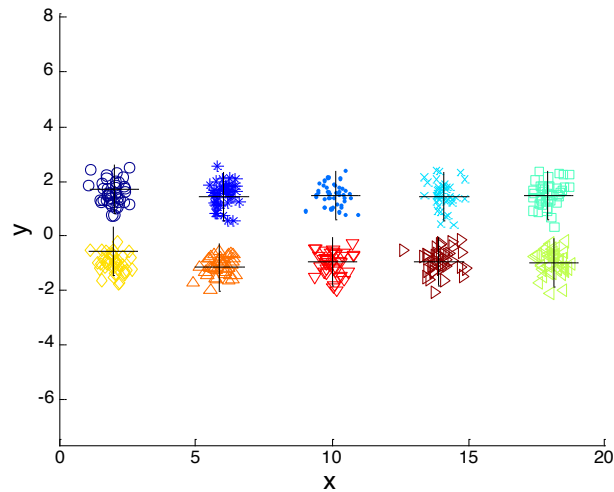
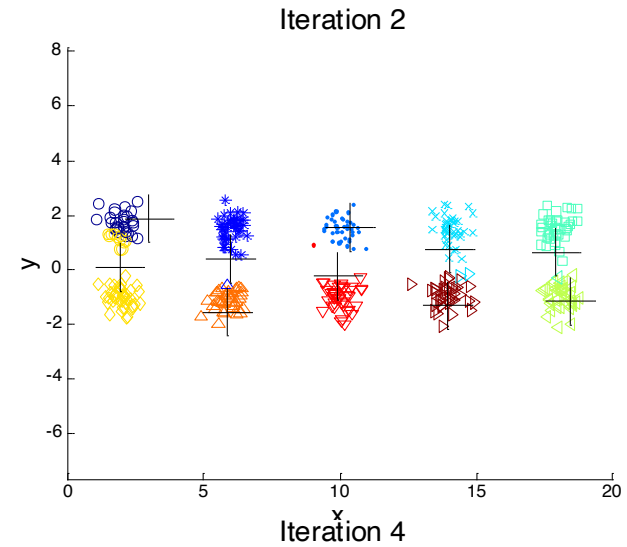
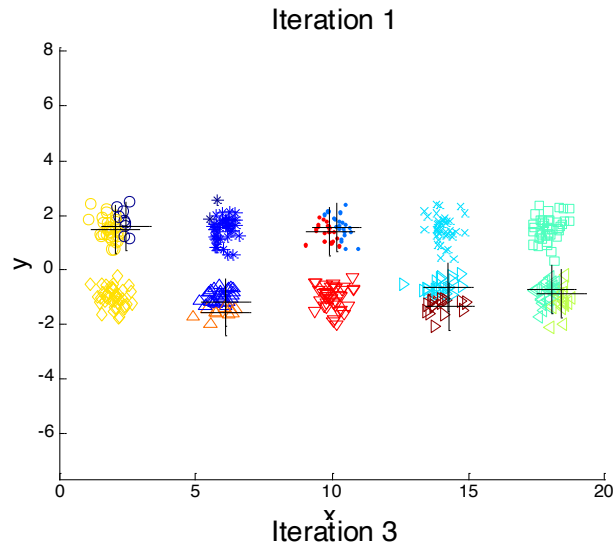
# 10 Clusters Example

Iteration 4



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

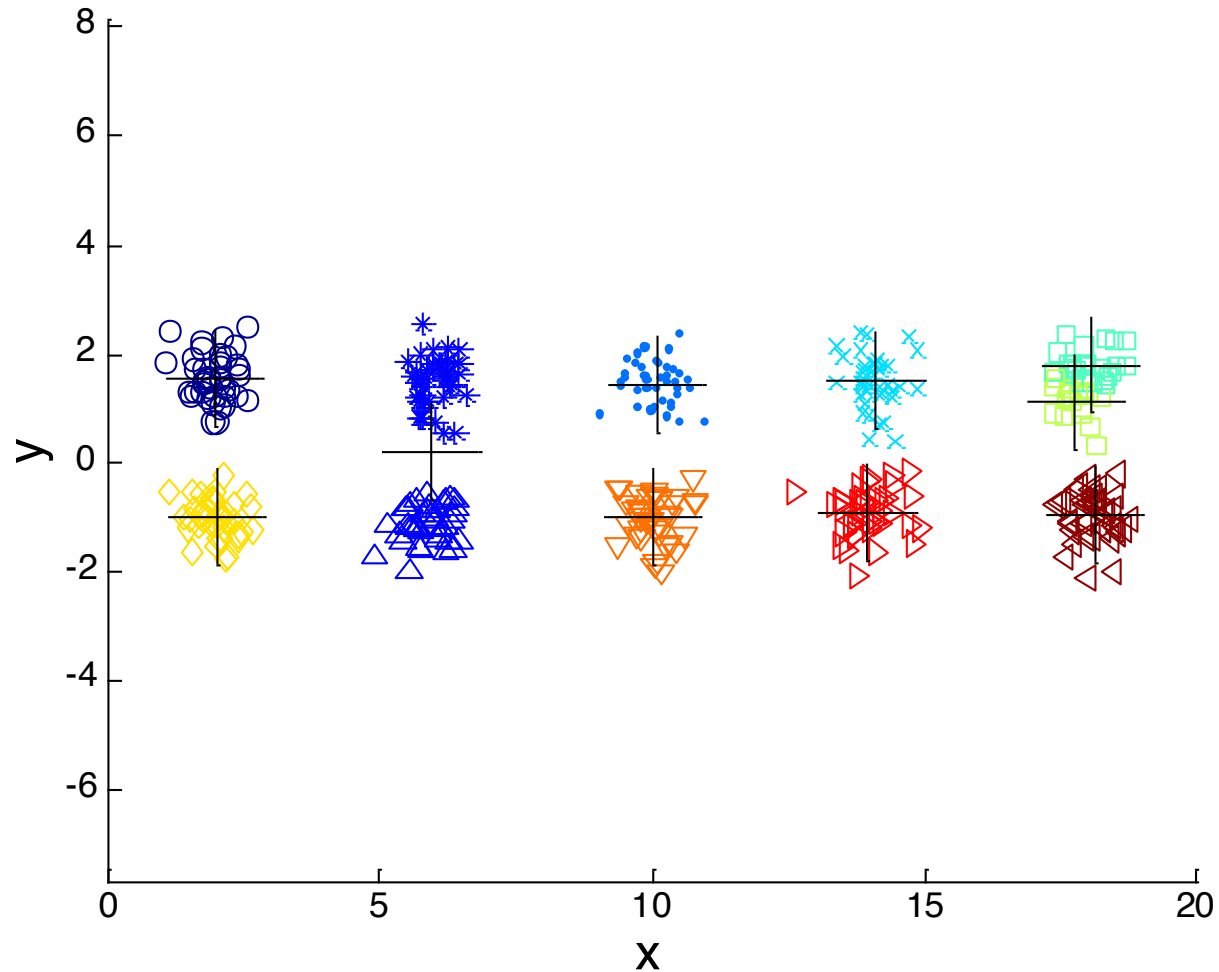
# 10 Clusters Example



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

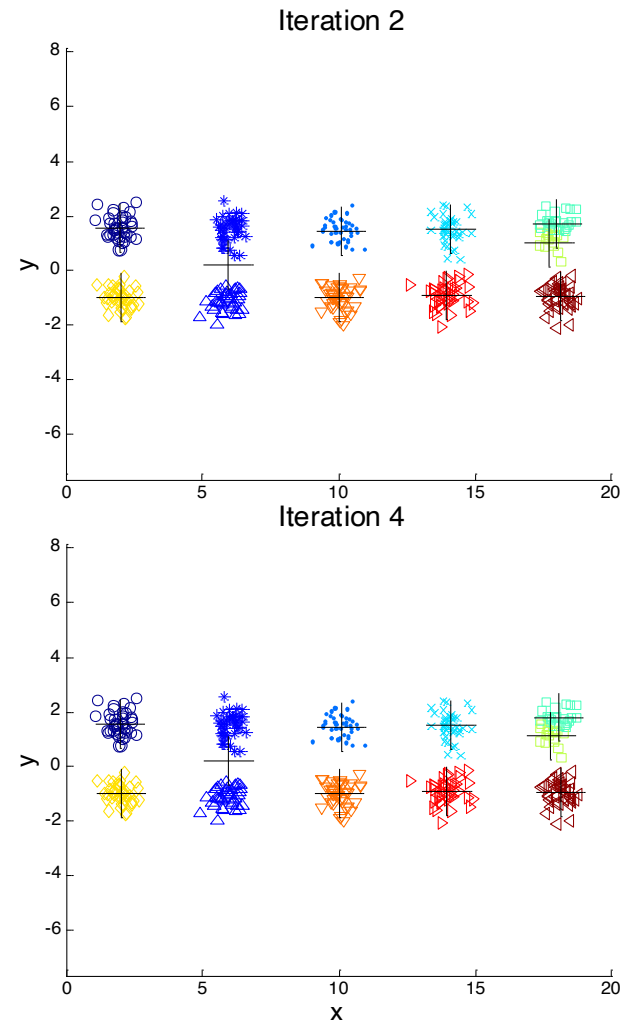
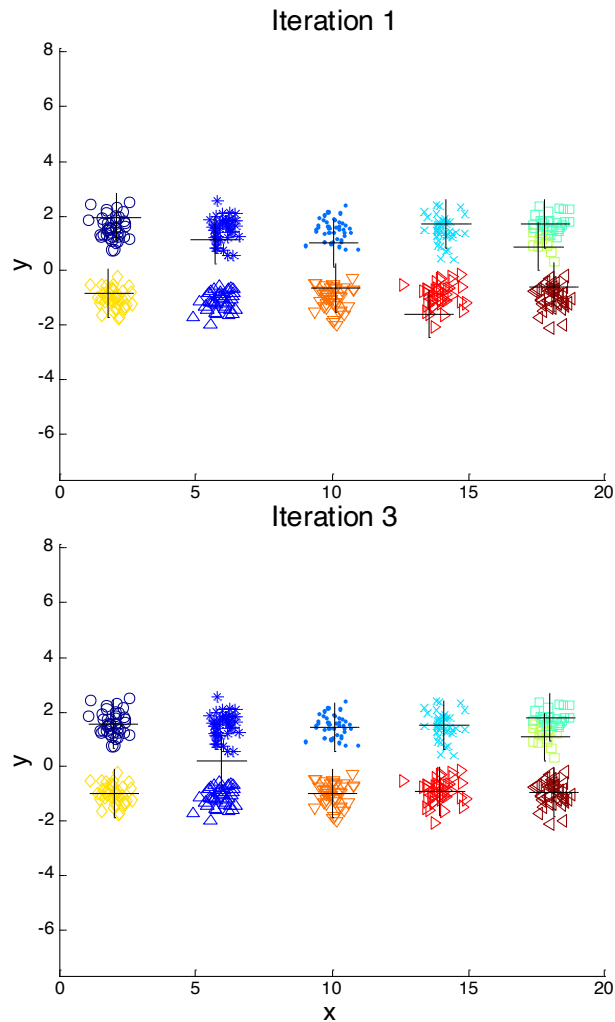
# 10 Clusters Example

Iteration 4



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

# 10 Clusters Example

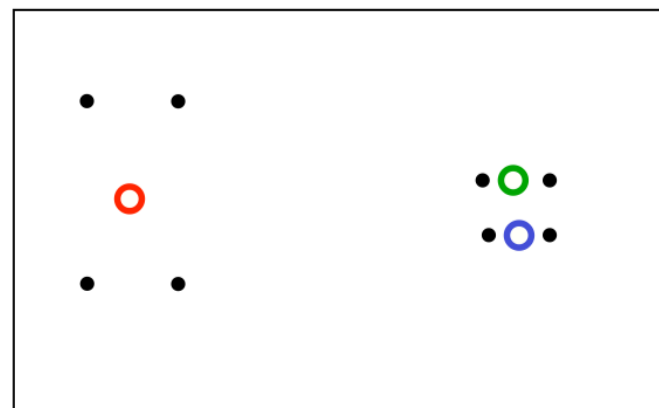


- Starting with some pairs of clusters having three initial centroids, while other have only one.

# Local Minima

- The objective  $J$  is non-convex (so coordinate descent on  $J$  is not guaranteed to converge to the global minimum)
- There is nothing to prevent k-means getting stuck at local minima.
- We could try many random starting points
- We could try non-local split-and-merge moves:
  - ▶ Simultaneously **merge** two nearby clusters
  - ▶ and **split** a big cluster into two

A bad local optimum





# Handling Empty Clusters

- Basic K-means algorithm can yield empty clusters
- Several strategies
  - Choose a point that has the highest distance to a cluster center, set the point as the centroid of the empty cluster.
  - Split a cluster with the highest overall distance by assigning the centroid of the empty cluster within that cluster.
  - 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

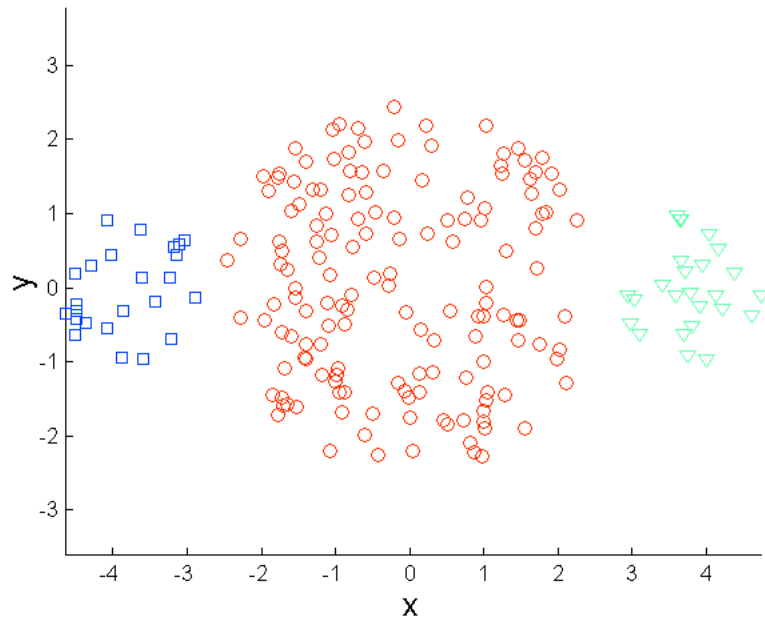
- **Post-processing**

- Eliminate small clusters that may represent outliers
- Split 'loose' clusters, i.e., clusters with relatively high error
- Merge clusters that are 'close' and that have relatively low error

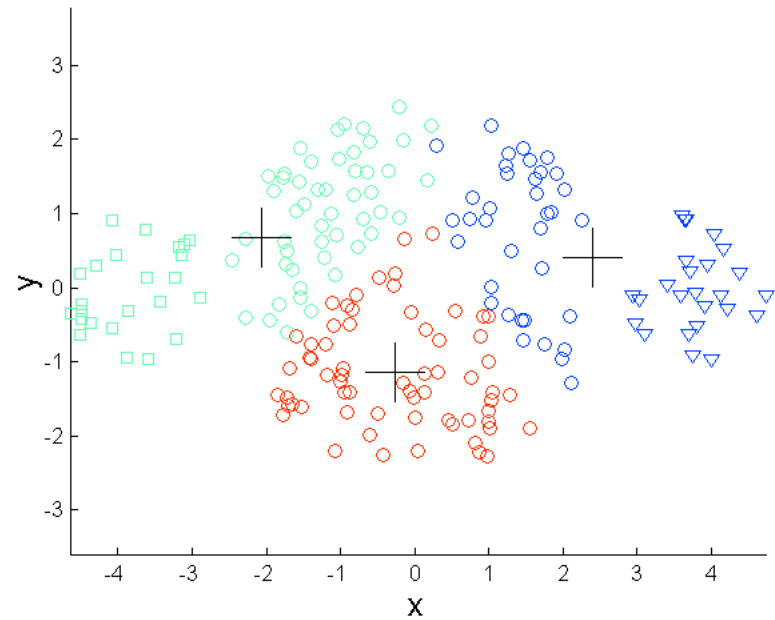
# Limitations of K-means

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

# Limitations of K-means: Differing Sizes

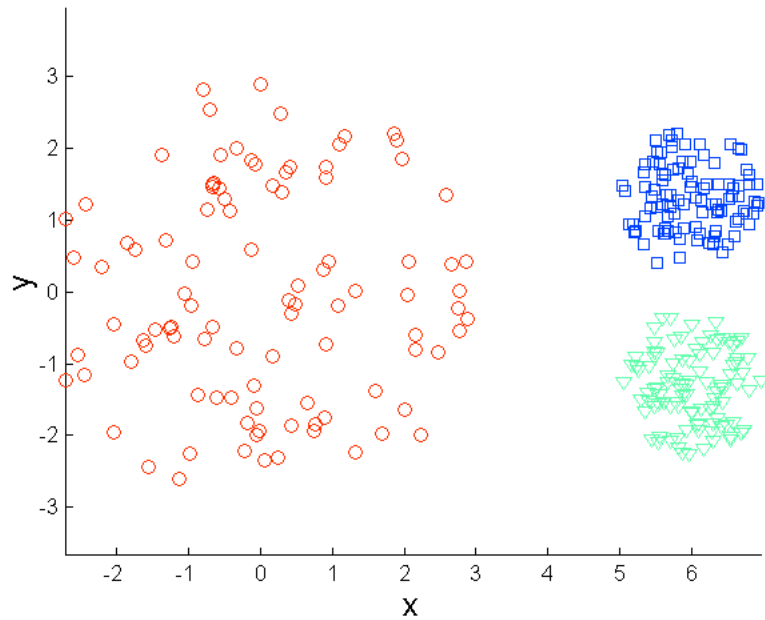


Original Points

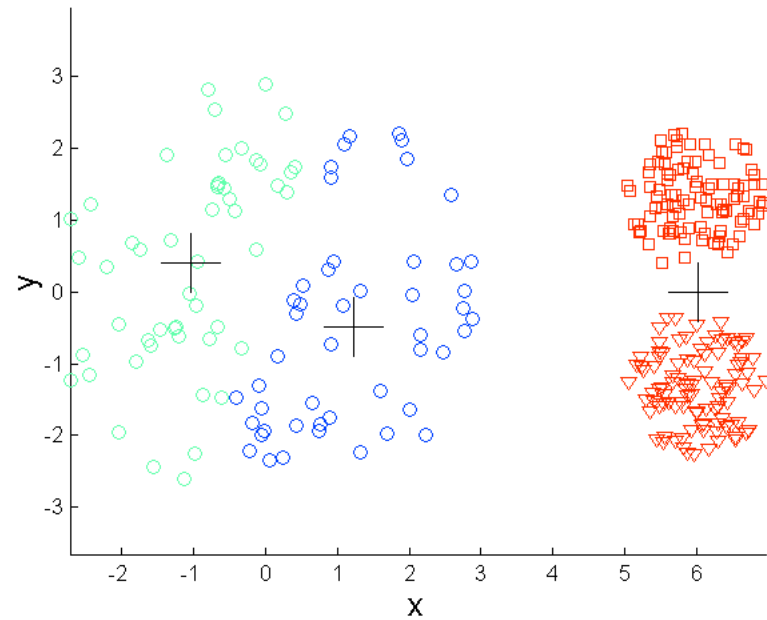


K-means (3 Clusters)

# Limitations of K-means: Differing Density

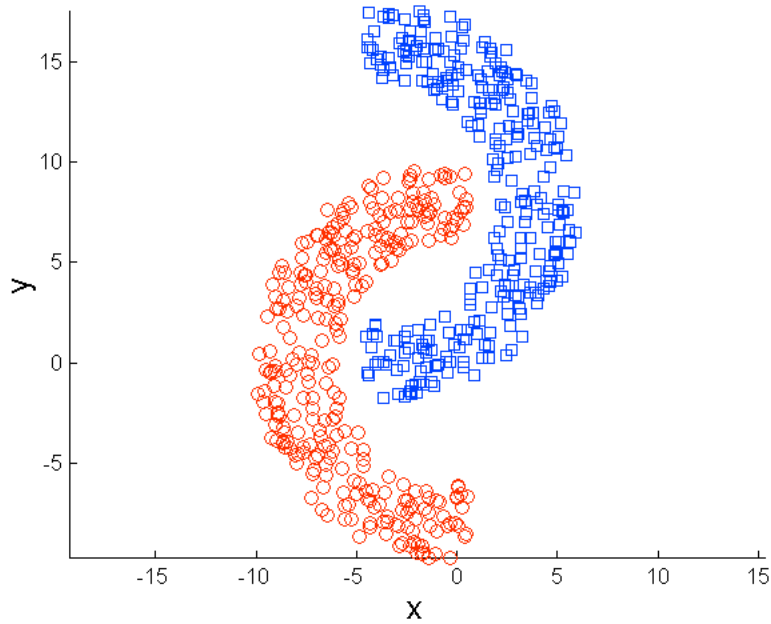


Original Points

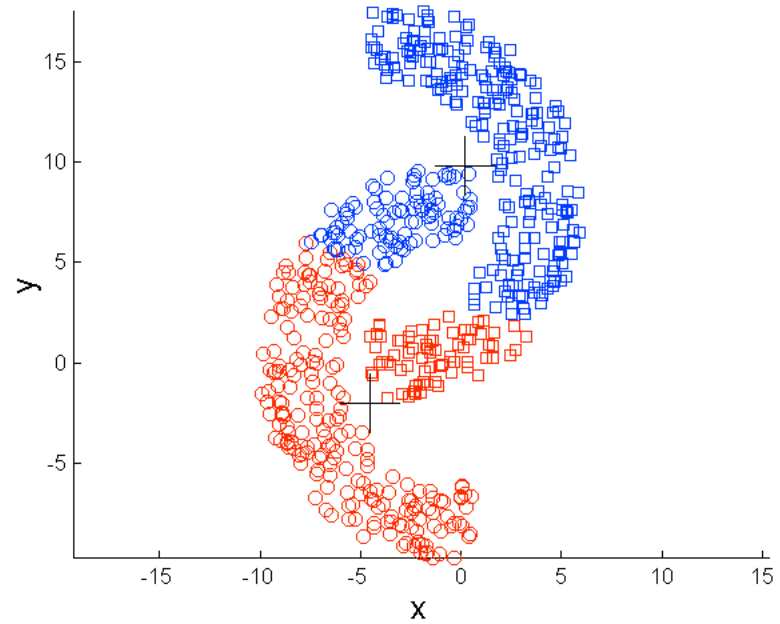


K-means (3 Clusters)

# Limitations of K-means: Irregular Shapes



Original Points



K-means (2 Clusters)



# K-means for Vector Quantization

$K = 2$



$K = 3$



$K = 10$

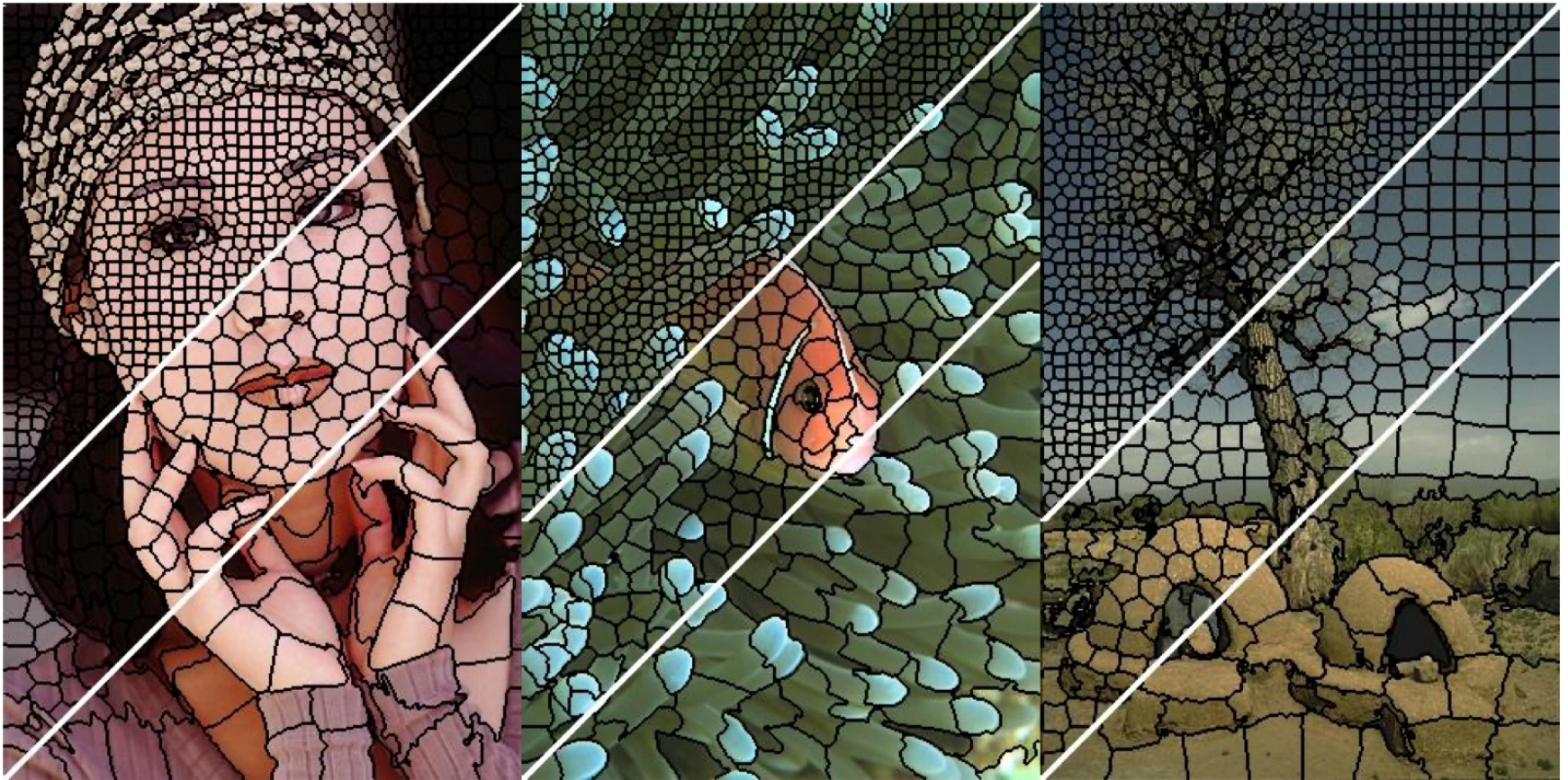


Original image





# K-means for Image Segmentation



- How would you modify k-means to get super pixels?

# Questions?