Machine Learning
NETW 1013

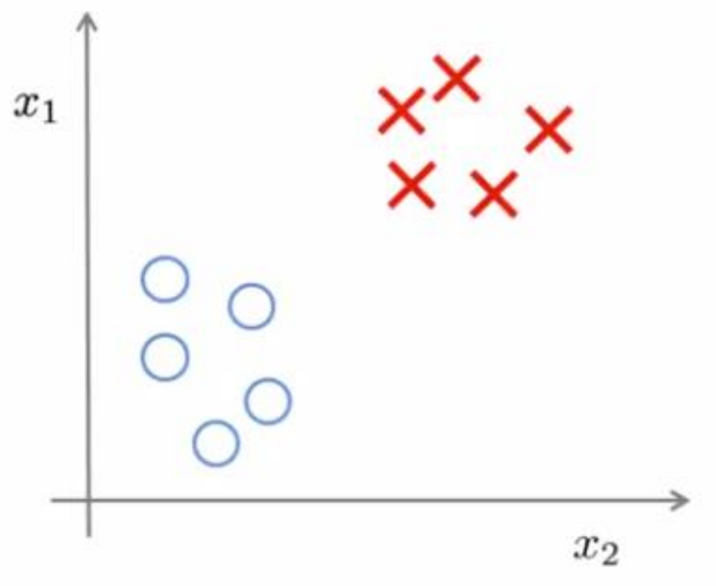
Lecture 2
Unsupervised Learning: Clustering

Course Instructors: Associate Prof. Dr.-Ing. Maggie Mashaly
maggie.ezzat@guc.edu.eg
C3.220

Contents

Supervised vs. Unsupervised Learning

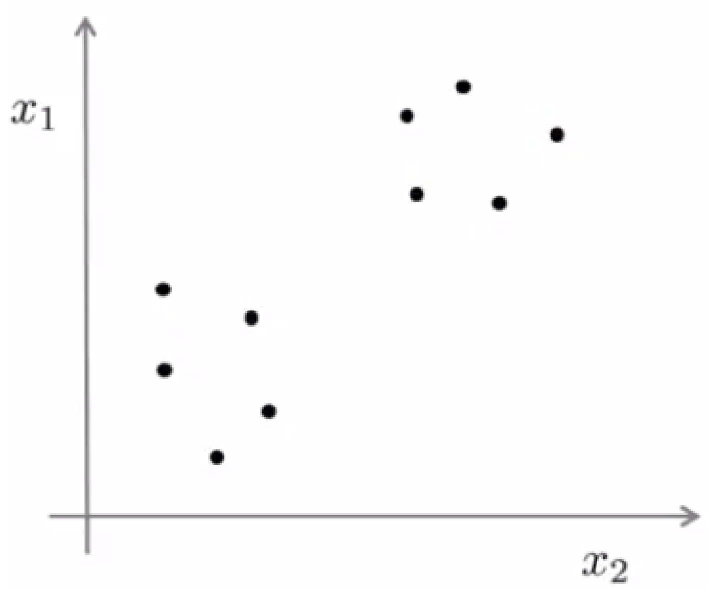
Supervised Learning



Training set:

$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \dots, (x^{(m)}, y^{(m)})\}$

Unsupervised Learning



Training set:

$\{x^{(1)}, x^{(2)}, x^{(3)}, \dots, x^{(m)}\}$

Supervised Learning: Classification

Training set of labeled docs

SPORTS

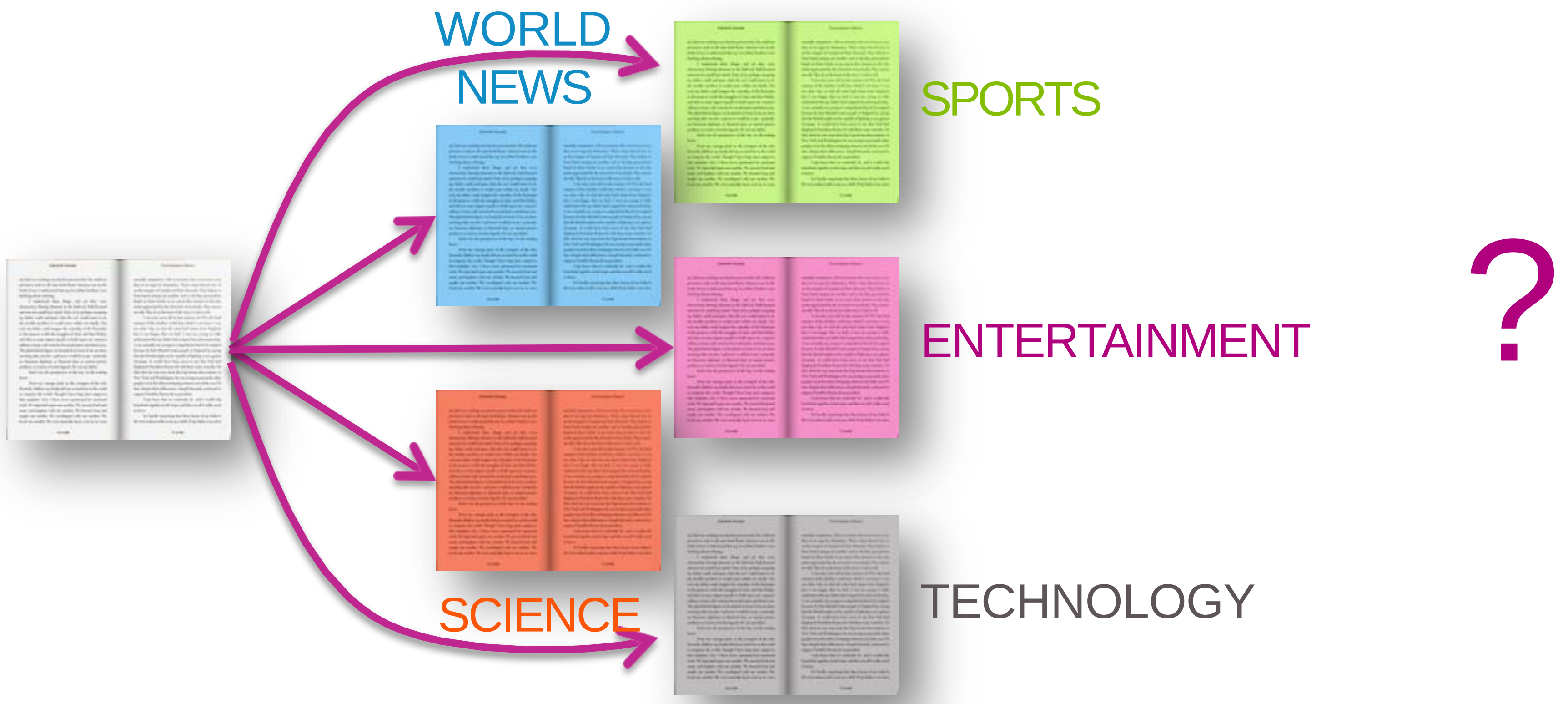
WORLD NEWS

ENTERTAINMENT

SCIENCE

https://www.coursera.org/learn/ml-clustering-and-retrieval/

Supervised Learning: Multi-class Classification



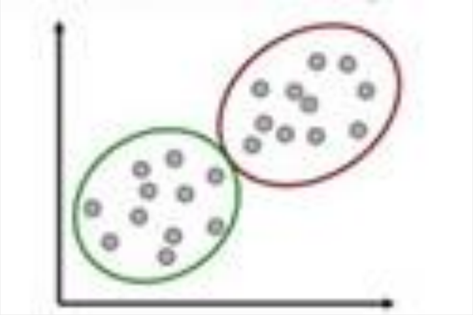
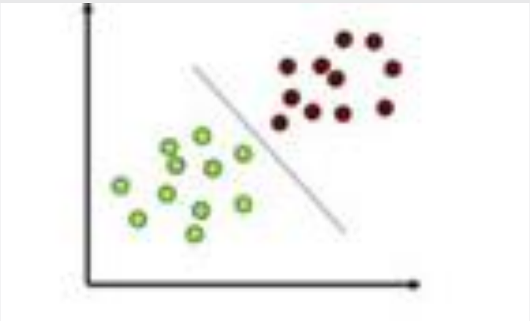
But what if labels were not provided...?

Unsupervised Learning: Clustering

- **Goal:**

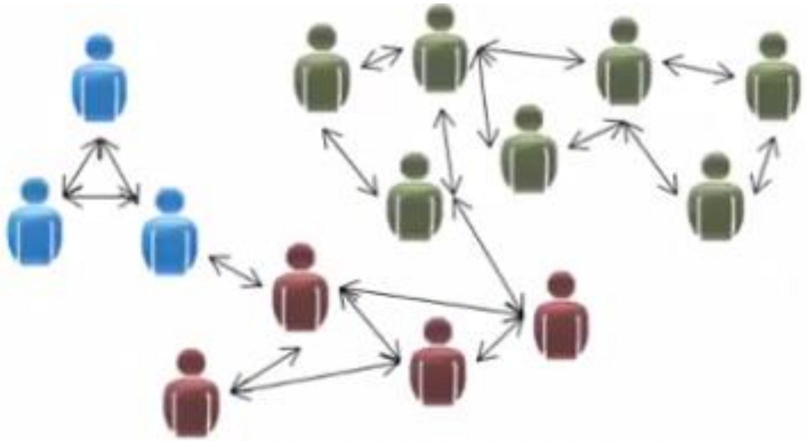
Finding structure within the data, usually by dividing it into Clusters

- **But mind the difference:**

Clustering	Classification
• Data is not labeled • Group points that are "close" to each other • Identify structure or patterns in data • Unsupervised learning	• Labeled data points • Want a "rule" that assigns labels to new points • Supervised learning
	

Clustering: Use cases

Market segmentation



Social network analysis

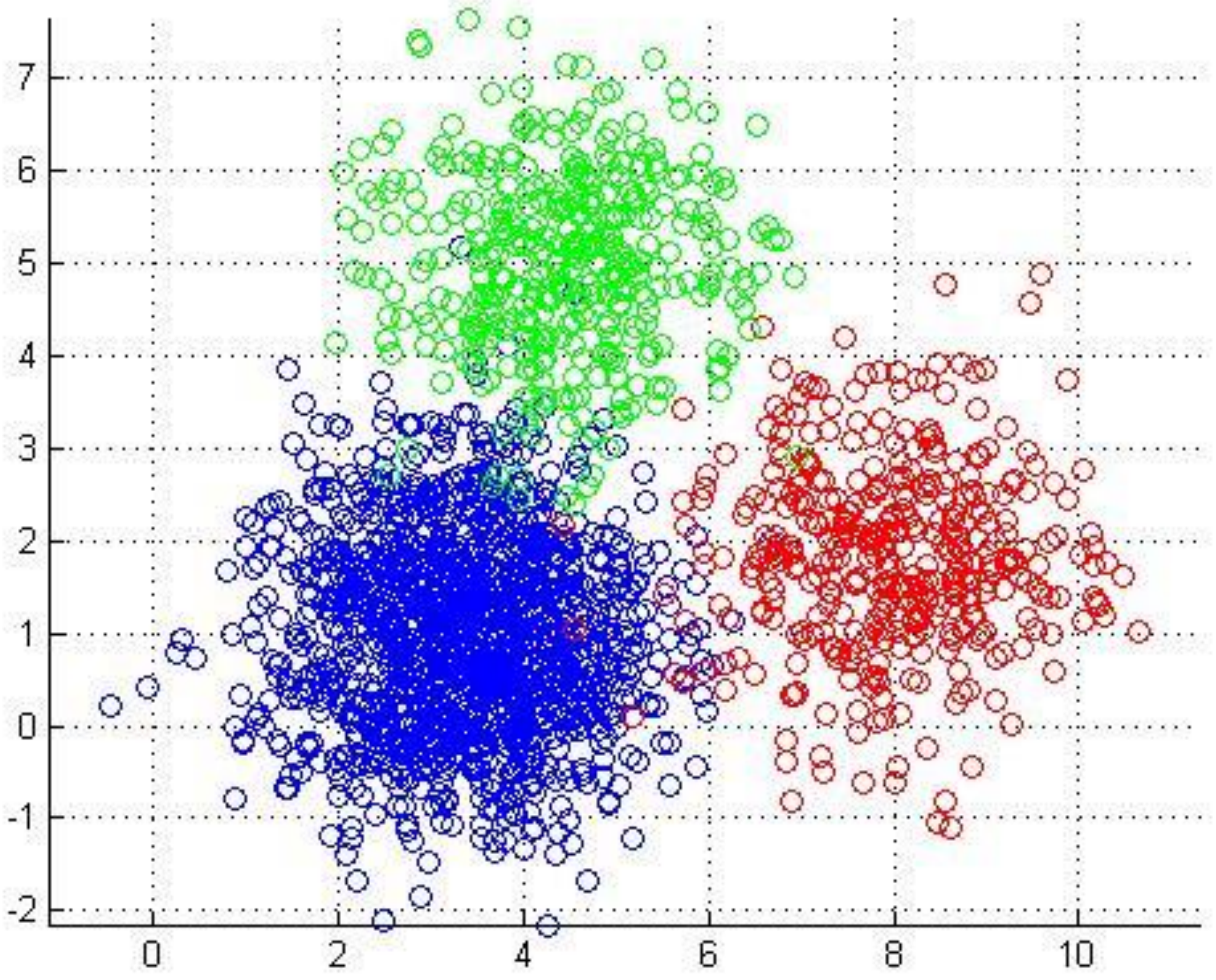
Organize computing clusters

Astronomical data analysis

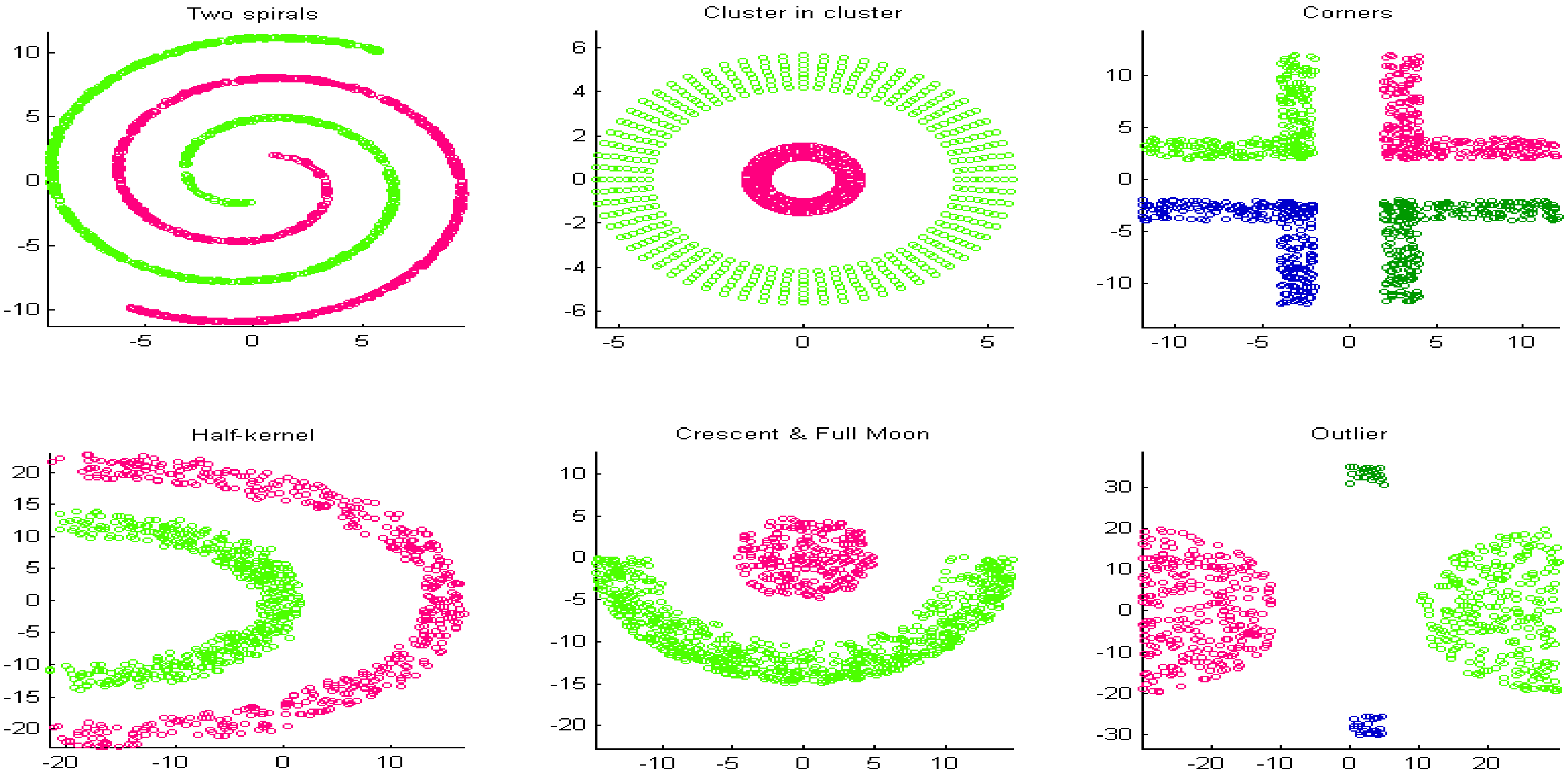
Clustering: Use cases

- Data summarization, compression, and reduction
  - *Examples: Image processing or vector quantization*
- Collaborative filtering, recommendation systems, or customer segmentation
  - *Finding like-minded users or similar products*
- Dynamic trend detection
  - *Clustering stream data and detecting trends and patterns*
- Multimedia data analysis, biological data analysis, and social network analysis
  - *Example: Clustering images or video/audio clips, gene/protein sequences, etc.*
- A key intermediate step for other data mining tasks
  - *Generating a compact summary of data for classification, pattern discovery, and hypothesis generation and testing*
- Outlier detection: Outliers - those “far away” from any cluster

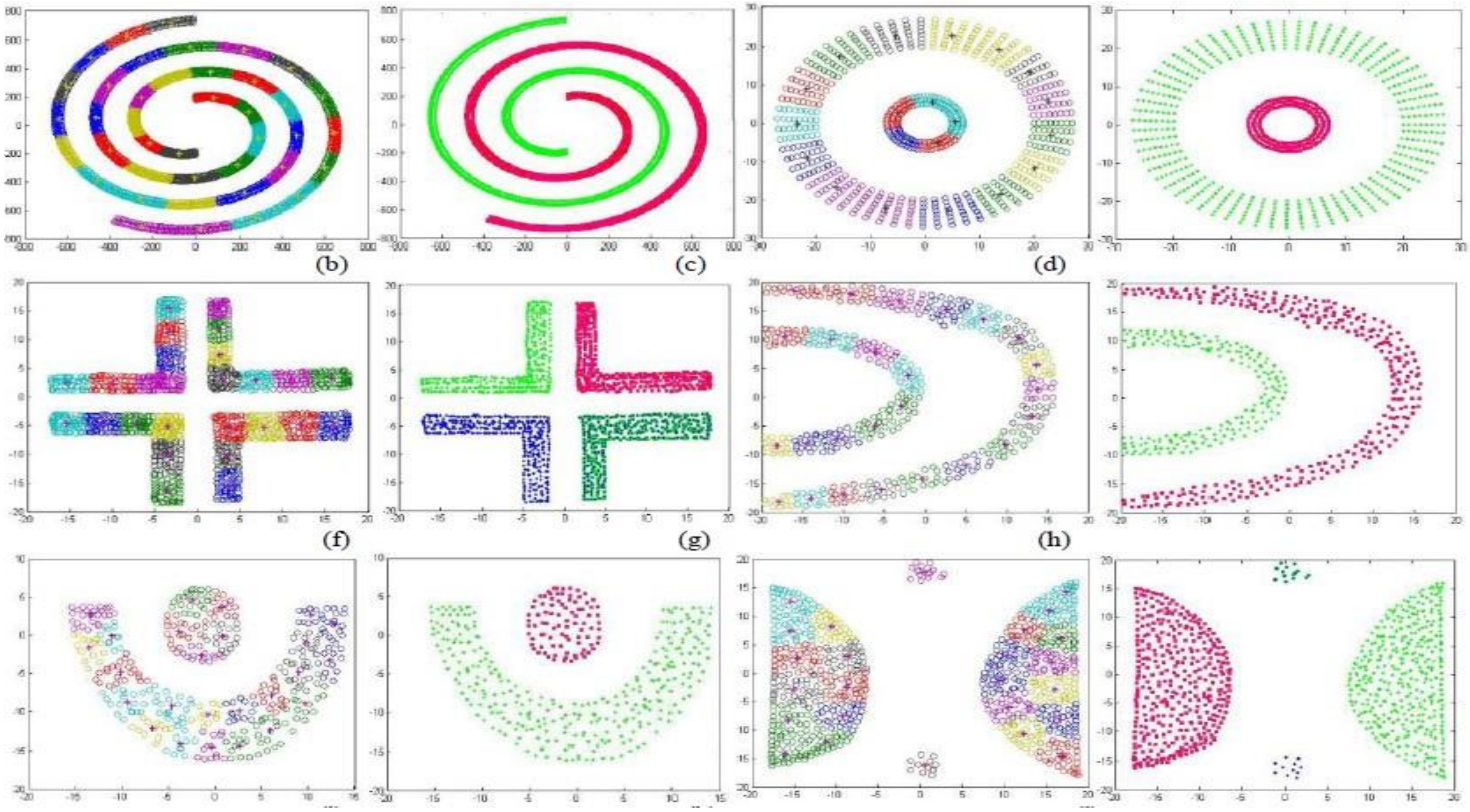
Clustering: an easy task?



Clustering: Challenging Clusters to discover!



Clustering: Challenging Clusters to discover!



Clustering: Definition & Formulation

- Clustering is the task of partitioning the data points into natural groups called clusters, such that points within a group are very similar, whereas points between different groups are as dissimilar as possible.
- $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$
- $C_i = \{x_j | x_j \in C_i\}$
- $\boldsymbol{C_i} \cap \boldsymbol{C_j} = \emptyset$ ➔ disjoint cluster no overlapping
- Clustering is an unsupervised learning approach since it does not require a separate training dataset to learn the model parameters.

Clustering: Different Data Types

- Numerical data
- Categorical data (including binary data)
  - *Discrete data, no natural order (e.g., gender, zip-code, and market-basket)*
- Text data: Popular in social media, Web, and social networks
  - *Features: High-dimensional, sparse, value corresponding to word frequencies*
- Multimedia data: Image, audio, video (e.g., on Flickr, YouTube)
  - *Multi-modal (often combined with text data)*
- Time-series data: Sensor data, stock markets, temporal tracking, forecasting, etc.
- Sequence data: Weblogs, biological sequences, system command sequences
- Stream data

Clustering: Portioning Problem

- Problem definition: Given K, find a partition of K clusters that optimizes the chosen partitioning criterion
- A brute-force or exhaustive algorithm for finding a good clustering is simply to
  - *generate all possible partitions of n points into k clusters*
  - *evaluate clusters*
  - *Choose the best clusters*
- However, this is clearly infeasible, since there are $O(k_n/k!)$ clusterings of n points into k groups.
- Global optimal: Needs to exhaustively enumerate all partitions
- Heuristic methods (i.e., greedy algorithms): K-Means, K-Medians, K-Medoids, etc.

Clustering Paradigms

- Representative-based
  - *K-means Clustering*
  - *Expectation-Maximization (EM) Algorithms*
- Hierarchical
- Density-based
- Graph-based
- Spectral clustering

Clustering: K-means Algorithm

K-means Clustering

- One of the simplest and most popular unsupervised machine learning algorithms
- K-means is a greedy algorithm that minimizes the squared distance of points from their respective cluster means
- It performs hard clustering, that is, each point is assigned to only one cluster.
- We also show how kernel K-means can be used for nonlinear clusters

Clustering: K-means Algorithm

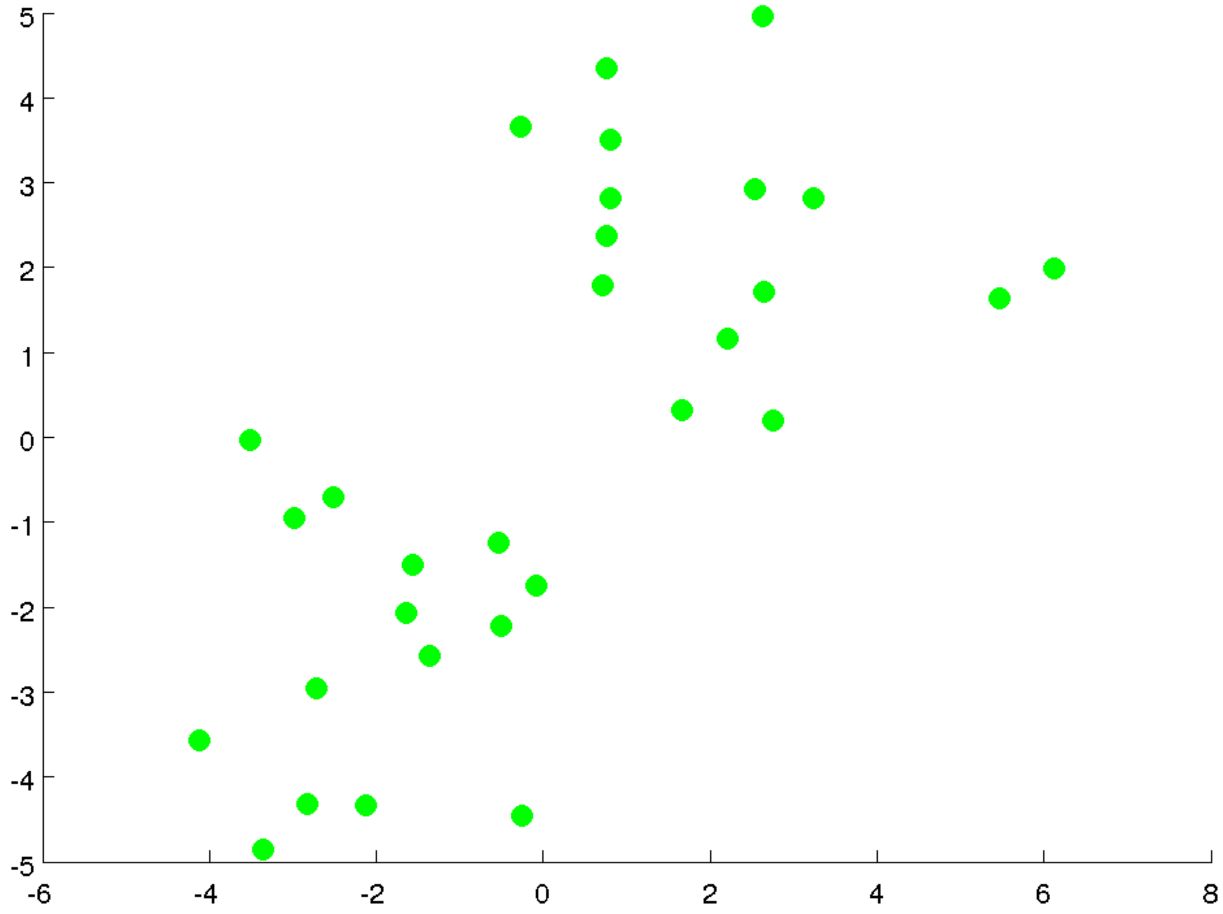
K-means Clustering

- **How does it work:**

1. K is chosen as the number of clusters we wish to cluster our data into
2. Randomly choose centroid for each cluster
3. Iterate over the following two steps:
   i. Cluster Assignment:
      Assign data to the cluster whose centroid is closest
   ii. Centroid Adjustment:
      Move each centroid to the mean of data assigned to its cluster

Clustering: K-means Algorithm

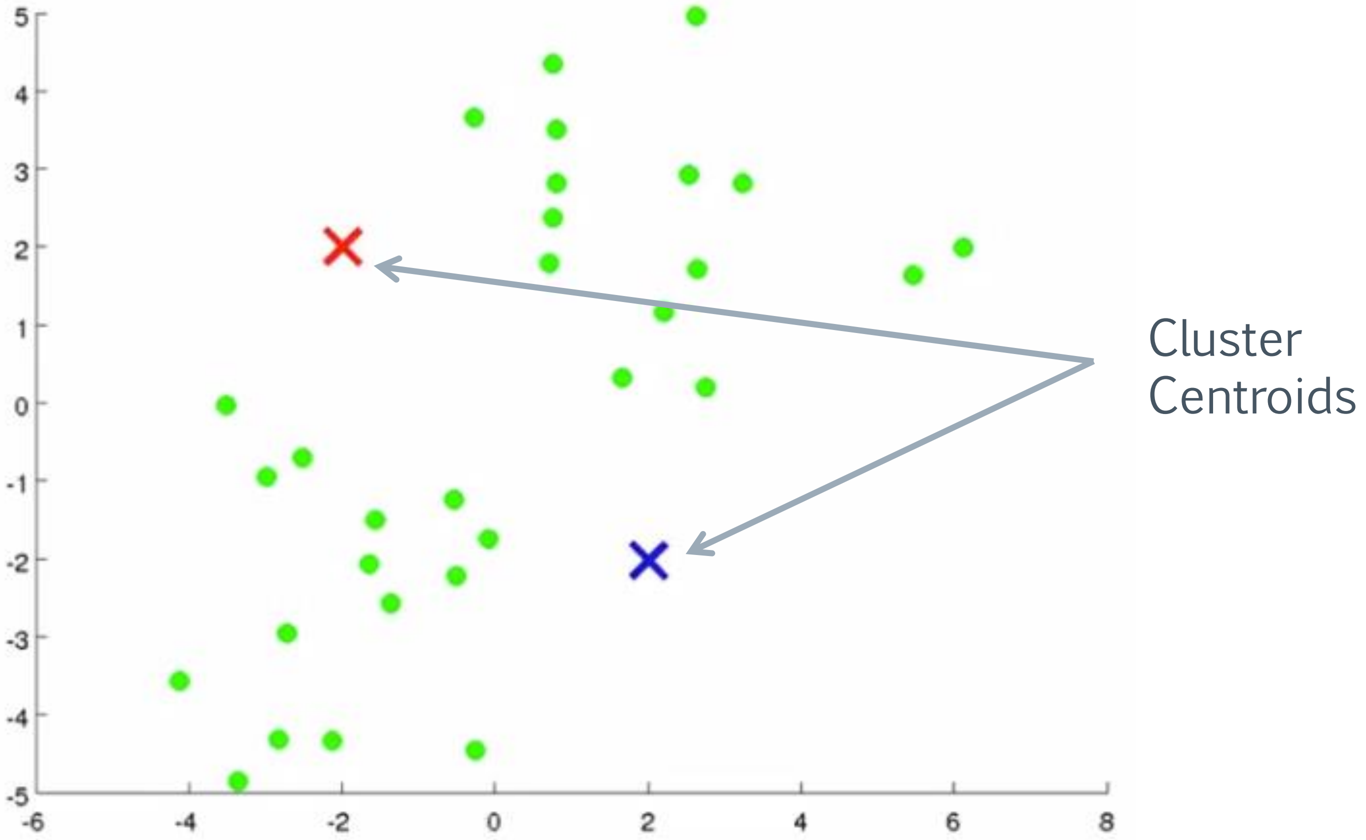
K-means Clustering: Example (K=2)



Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

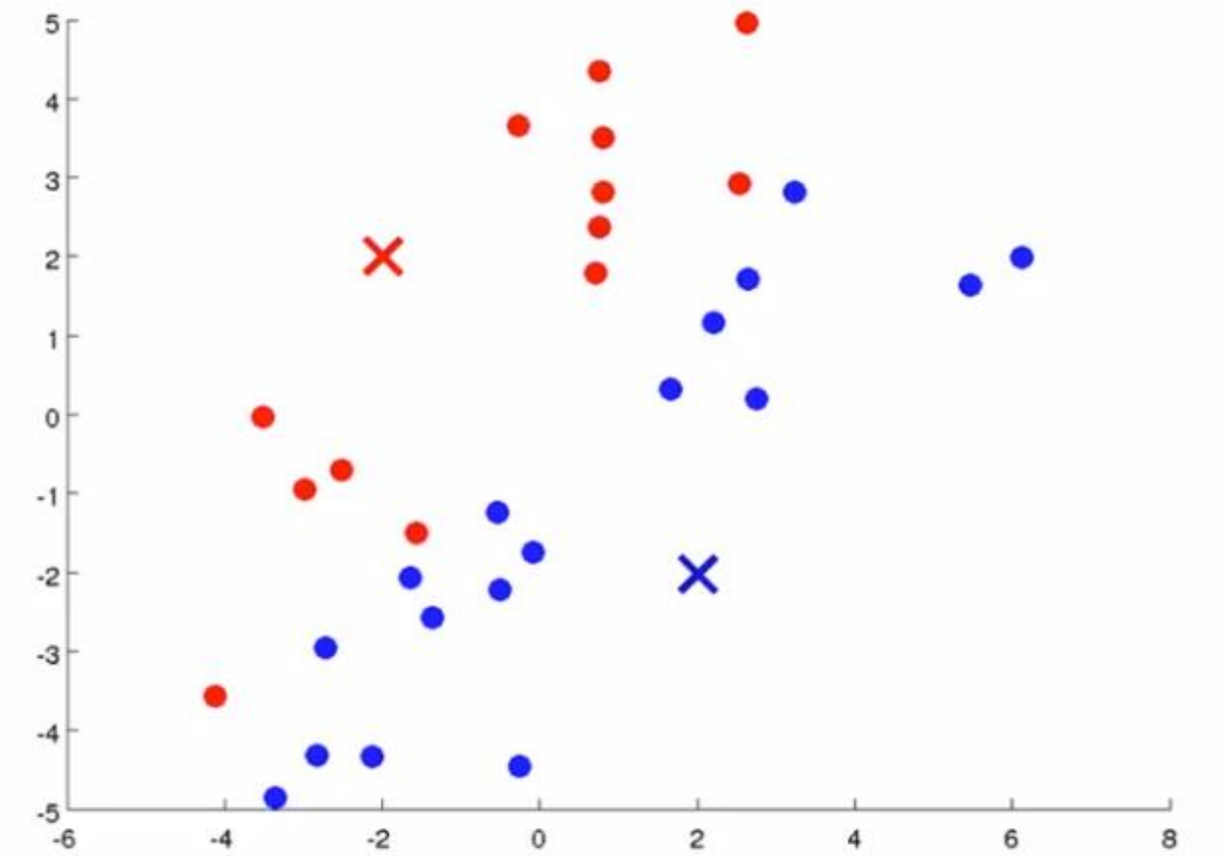
Step 1: Randomly assign cluster centroids



Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

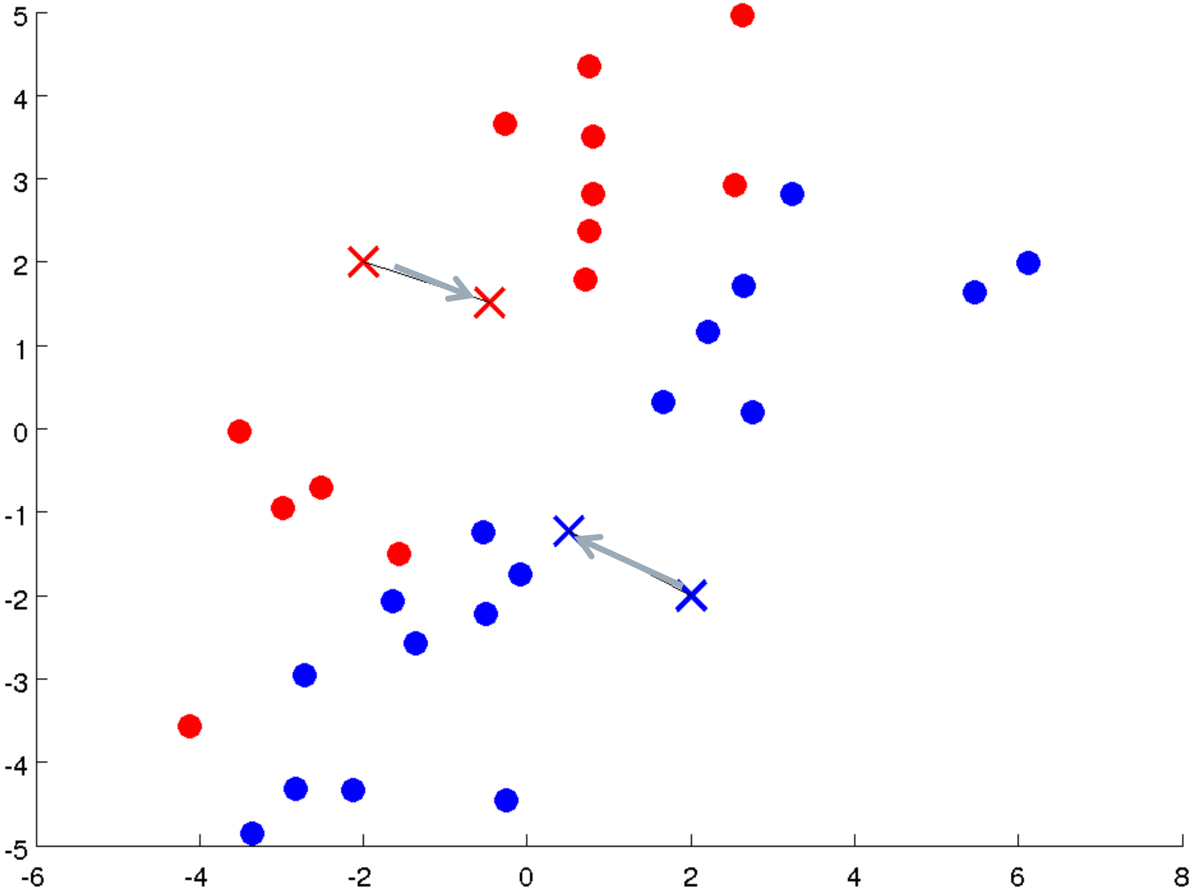
Step 2.1: Cluster assignment



Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

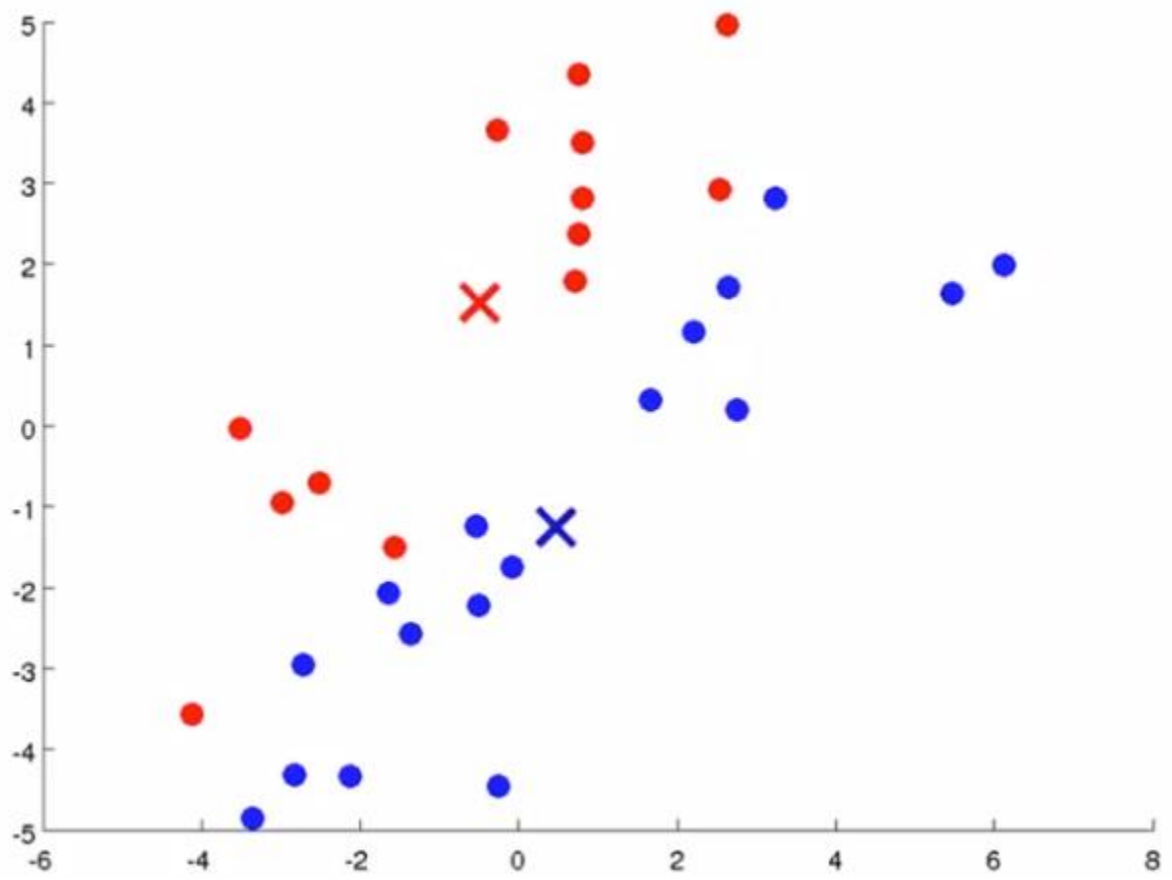
Step 2.2: Centroid Adjustment



Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

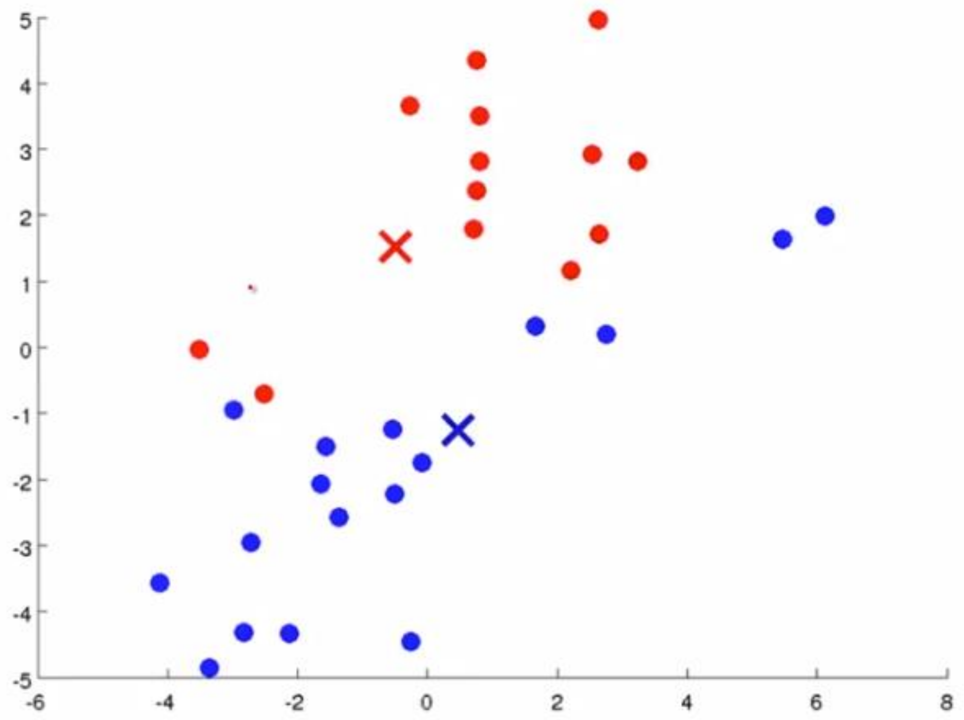
Step 2.2: Centroid Adjustment

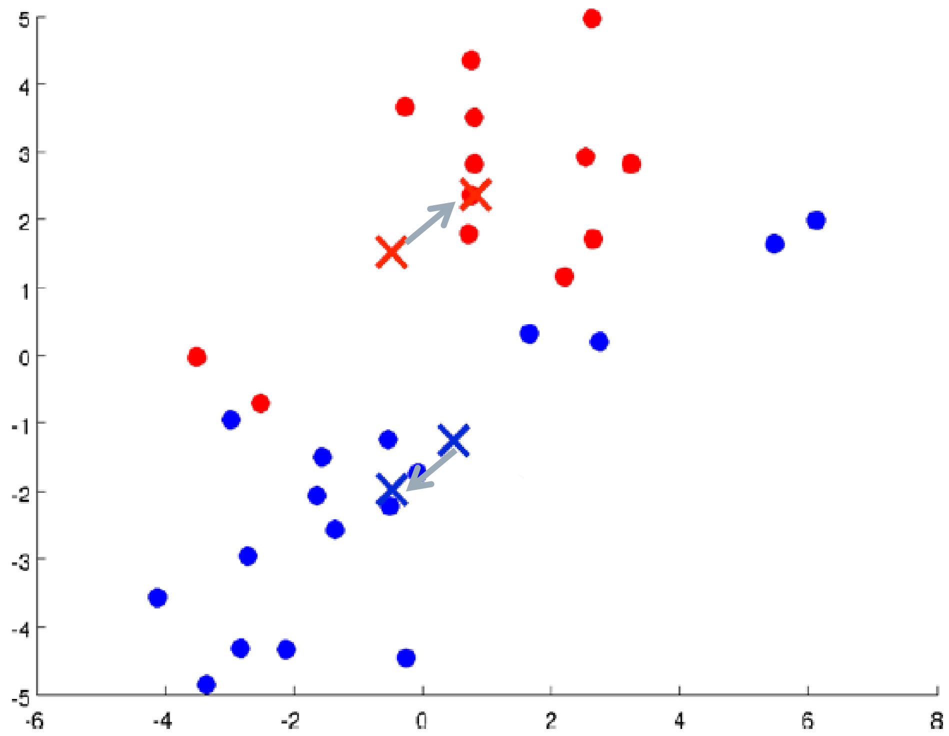


Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

Repeat: Cluster Assignment & Centroid Adjustment

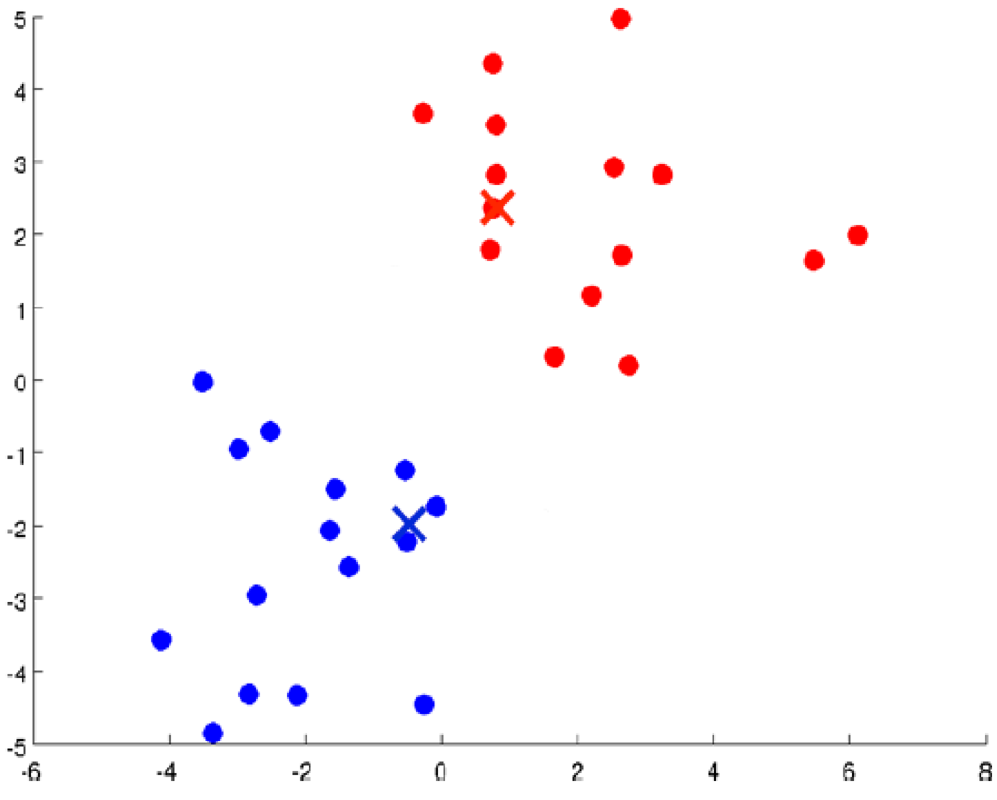


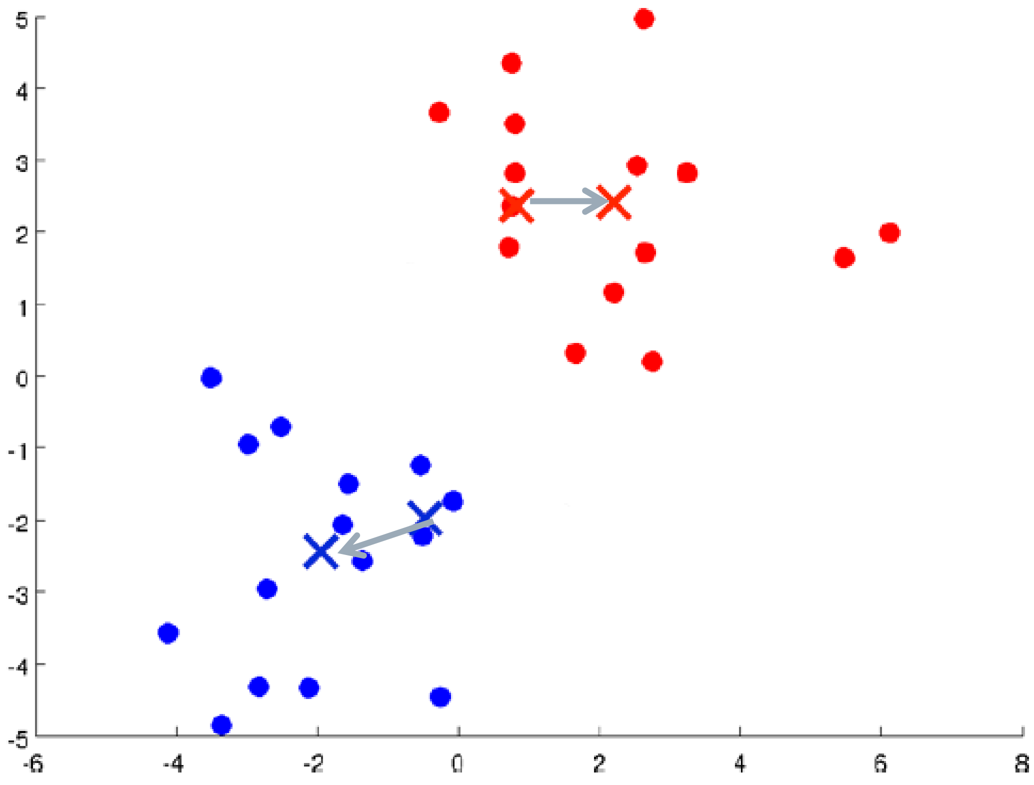


Clustering: K-means Algorithm

K-means Clustering: Example (K=2)

Repeat: Cluster Assignment & Centroid Adjustment

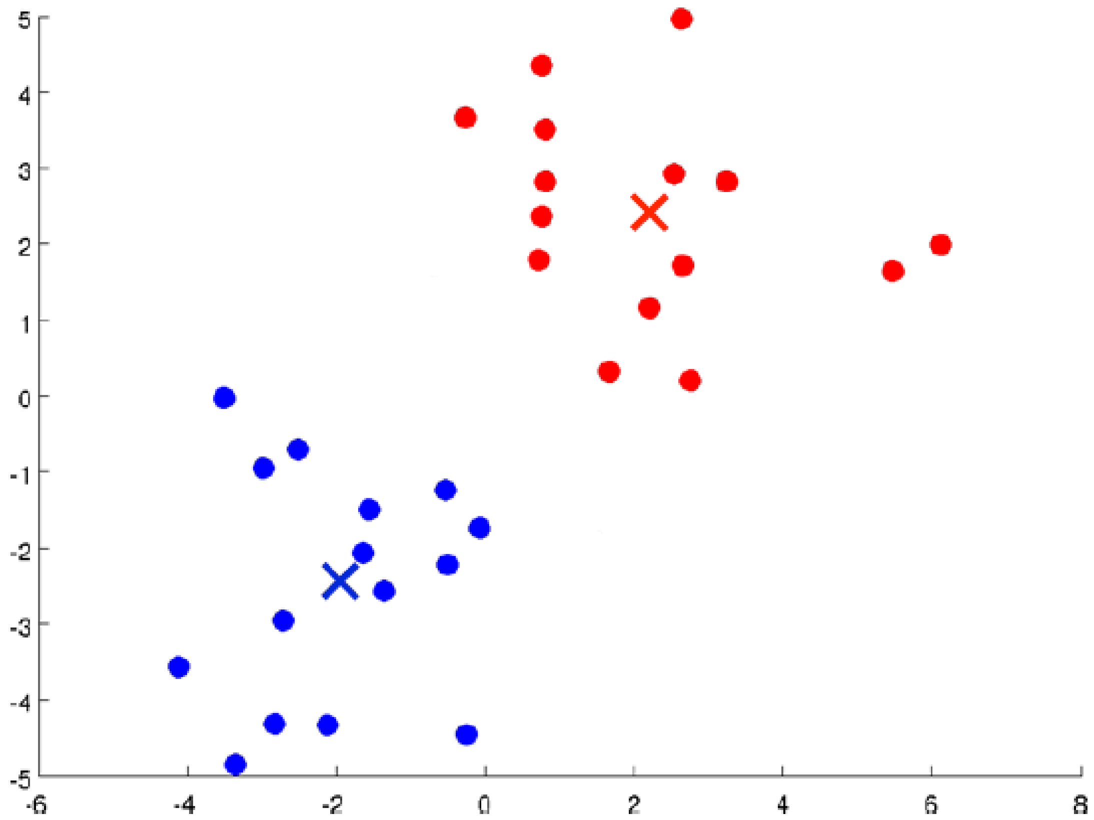




Clustering: K-means Algorithm

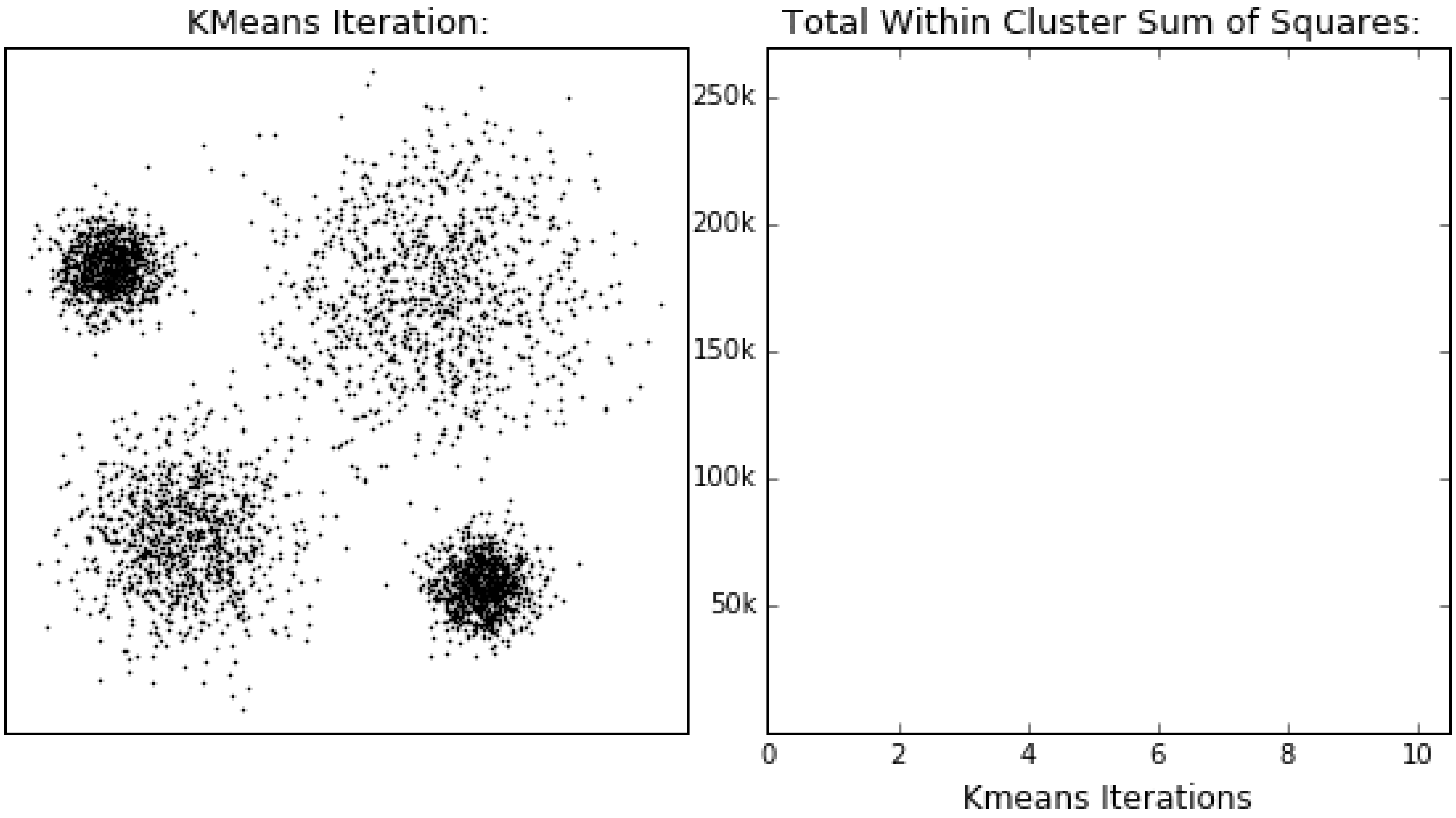
K-means Clustering: Example (K=2)

Repeat: Cluster Assignment & Centroid Adjustment



Stop when there is no change in the cluster centroid

Clustering: K-means Algorithm



Clustering: K-means Algorithm

Input:

- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, x^{(3)}, \dots, x^{(m)}\}$

Algorithm:

- Randomly initialize K cluster centroids $\mu_1, \mu_2, \dots, \mu_K$
- Repeat until convergence

{ for i=1 to m

$c^{(i)}$=index (from 1 to K) of cluster centroid closest to $x^{(i)}$

Calculated as : $\min_k \left\| x^{(i)} - \mu_K \right\|^2$

for k=1 to K

μ_K= average of points assigned to cluster k

}

Clustering: K-means Algorithm

Optimization Objective

$c^{(i)}$= index of cluster (1,2,...,K) to which $x^{(i)}$ is currently assigned

μ_K = cluster centroid k

$\mu_{c^{(i)}}$= cluster centroid of cluster to which the example $x^{(i)}$ has been assigned

$$\min_{\substack{c^{(1)},\dots,c^{(m)} \\ \mu_1,\dots,\mu_K}} J(c^{(1)},\dots,c^{(m)},\mu_1,\dots,\mu_K)$$

where

$$J(c^{(1)},\dots,c^{(m)},\mu_1,\dots,\mu_K) = \frac{1}{m}\sum_{i=1}^{m}\left\|x^{(i)} - \mu_{c^{(i)}}\right\|^2$$

and this objective function is also known as Distortion Function

Example: K-means in 1D

- Consider the following one-dimensional data. Assume that we want to cluster the data into k = 2 groups. {2, 3,4,10, 11,12,20,25,30}.

1) Initial centroids **μ1 = 2** and **μ2 = 4**.

2) **Loop until convergence**

   A. ***First iteration***

   a) *Cluster assignment, assigning each point to the closest mean:*

   C1 = {2, 3} C2 = {4, 10,11,12,20,25,30}

   b) *Centroid update, update the means*

   $$\mu 1 = \frac{2+3}{2} = \frac{5}{2} = 2.5 \qquad \mu 2 = \frac{4+10+11+12+20+25+30}{7} = \frac{112}{7} = 16$$

   B. ***Second iteration***

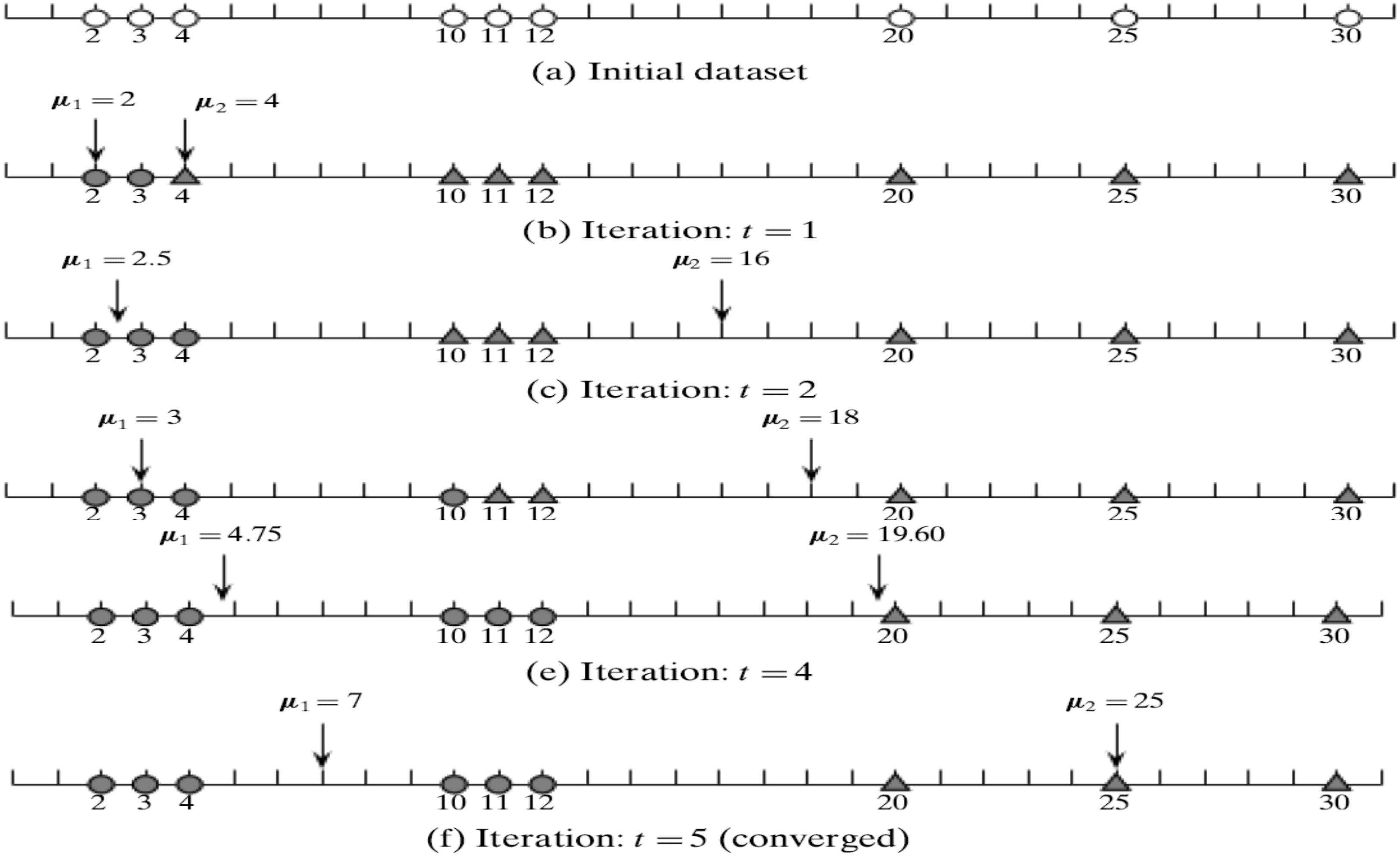
   a) *assigning each point to the closest mean:*

   C1 = {2, 3,4} C2 = {10,11,12,20,25,30}

   b) *update the means*

   $$\mu 1 = \frac{2+3+4}{3} = \frac{9}{3} = 3 \qquad \mu 2 = \frac{10+11+12+20+25+30}{7} = \frac{108}{6} = 18$$

Example: K-means in 1D



C1 = {2, 3,4, 10, 11, 12} C2 = {20,25,30} with centroids μ1 = 7 and μ2 = 25.

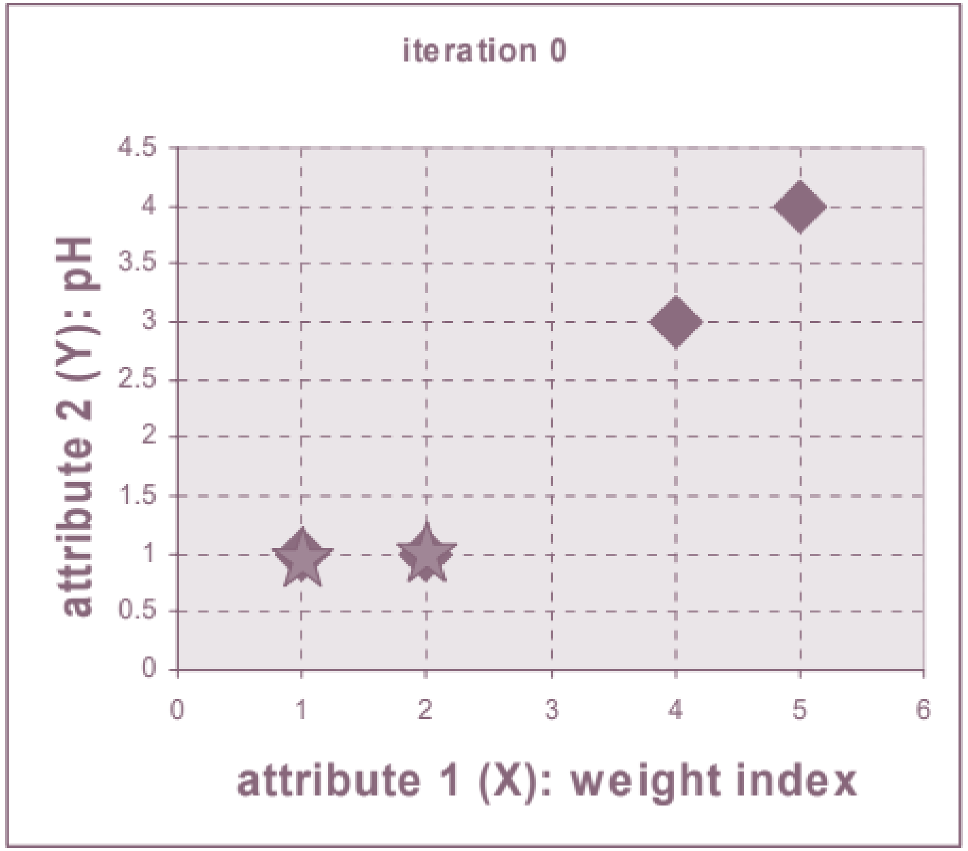
Example: K-means in 2D

- Suppose we have several objects (4 types of medicines) and each object have two attributes or features as shown in table below. Our goal is to group these objects into K=2 group of medicine based on the two features (pH and weight index).

Object	Attribute 1 (x1): weight index	Attribute 2 (x2): pH
A	1	1
B	2	1
C	4	3
D	5	4

Example: K-means in 2D

- Each medicine represents one point with two features (X, Y) that we can represent it as coordinate in a feature space as shown in the figure.



Object	Attribute 1 (x1): weight index	Attribute 2 (x2): pH
A	1	1
B	2	1
C	4	3
D	5	4

Example: K-means in 2D

- Initial value of centroids: **μ1** = (1,1) and **μ2** = (2,1)
- **μ1** = (1,1) with Medicine C
- $=\sqrt{(4-1)^2+(3-1)^2}$
- $=\sqrt{(3)^2+(2)^2}$
- $=\sqrt{9+4}$
- $\sqrt{13}=3.61$

Object	Attribute 1 (x1): weight index	Attribute 2 (x2): pH
A	1	1
B	2	1
C	4	3
D	5	4

http://people.revoledu.com/kardi/tutorial/kMean/index.html

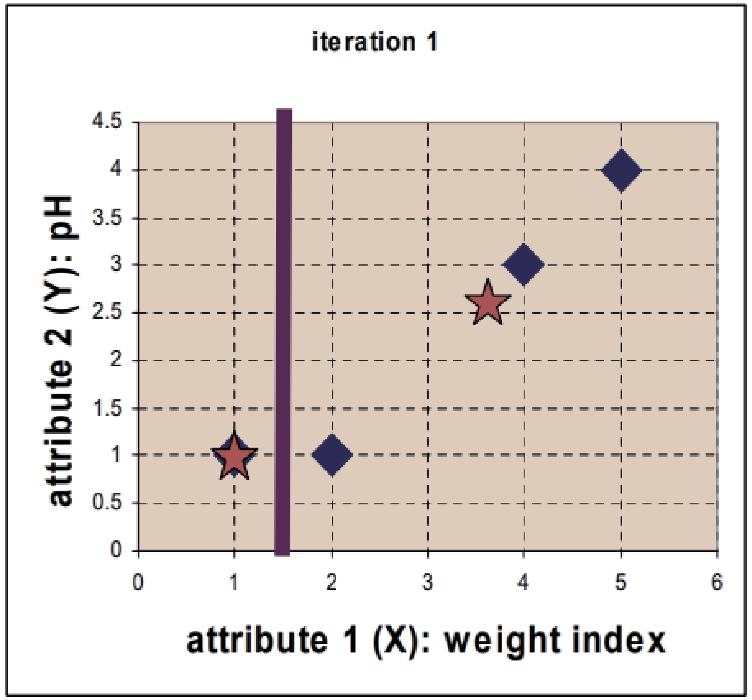
Example: K-means in 2D

- **Cluster assignment**

	μ1	μ2	Cluster
A	0	1	1
B	1	0	2
C	3.605551	2.828427	2
D	5	4.242641	2

- **Update Centroid**
- **μ1=(1,1)**
- **μ2=** $\frac{2+4+5}{3}, \frac{1+3+4}{3} = \frac{11}{3}, \frac{8}{3}$

Object	weight index	pH
A	1	1
B	2	1
C	4	3
D	5	4



Example: K-means in 2D (2)

- Use the k-means algorithm and Euclidean distance to cluster the following 8 examples into 3 clusters:
- A1=(2,10), A2=(2,5), A3=(8,4), A4=(5,8), A5=(7,5), A6=(6,4), A7=(1,2), A8=(4,9). The distance matrix based on the Euclidean distance is given below

	A1	A2	A3	A4	A5	A6	A7	A8
A1	0	$\sqrt{25}$	$\sqrt{36}$	$\sqrt{13}$	$\sqrt{50}$	$\sqrt{52}$	$\sqrt{65}$	$\sqrt{5}$
A2		0	$\sqrt{37}$	$\sqrt{18}$	$\sqrt{25}$	$\sqrt{17}$	$\sqrt{10}$	$\sqrt{20}$
A3			0	$\sqrt{25}$	$\sqrt{2}$	$\sqrt{2}$	$\sqrt{53}$	$\sqrt{41}$
A4				0	$\sqrt{13}$	$\sqrt{17}$	$\sqrt{52}$	$\sqrt{2}$
A5					0	$\sqrt{2}$	$\sqrt{45}$	$\sqrt{25}$
A6						0	$\sqrt{29}$	$\sqrt{29}$
A7							0	$\sqrt{58}$
A8								0

Application case: K-Means for Segmentation

Goal of Segmentation is to partition an image into regions each of which has reasonably homogenous visual appearance.

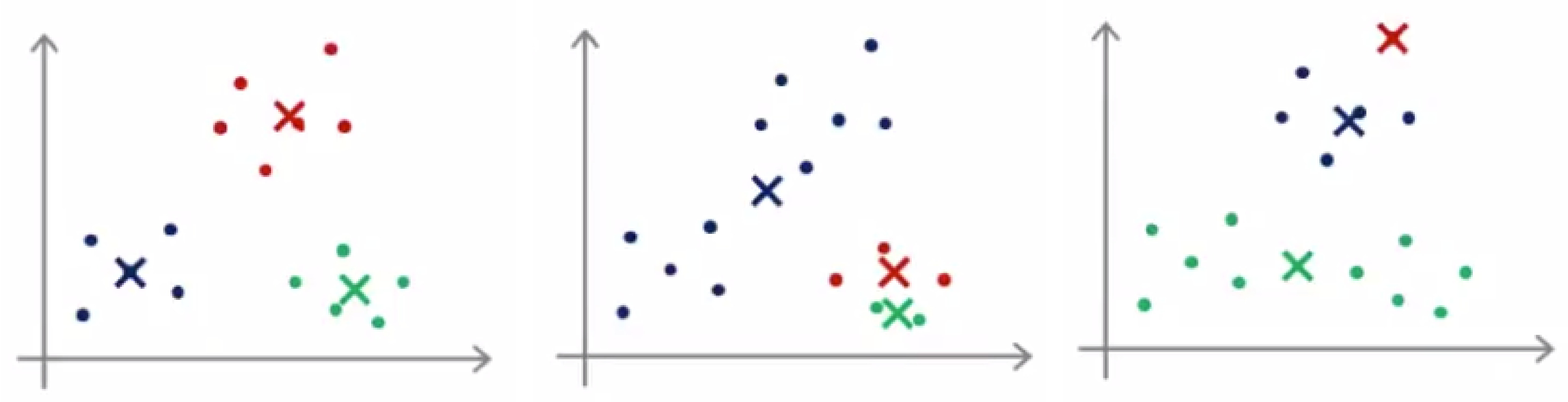
Original **K=2** **K=3** **K=10**

K-means Algorithm: Random Initialization

How to randomly initialize cluster centroids?

1. Choose K<m
2. Randomly pick K training examples
3. Set $\mu_1, \ldots, \mu_K$ equal to these K examples

Seems correct, but will it always work?



No, because K-means can get stuck at different local optimas

K-means Algorithm: Random Initialization

Solution:

Instead of initializing K-means once and hoping that it works;

- Initialize and run K-means many times, and use the solution that gives best local or global optima as possible.

So we do the following:

for i=1 to 100

{ Randomly initialize K-means

Run K-means to get $c^{(1)}, \ldots c^{(m)}, \mu_1, \ldots, \mu_K$

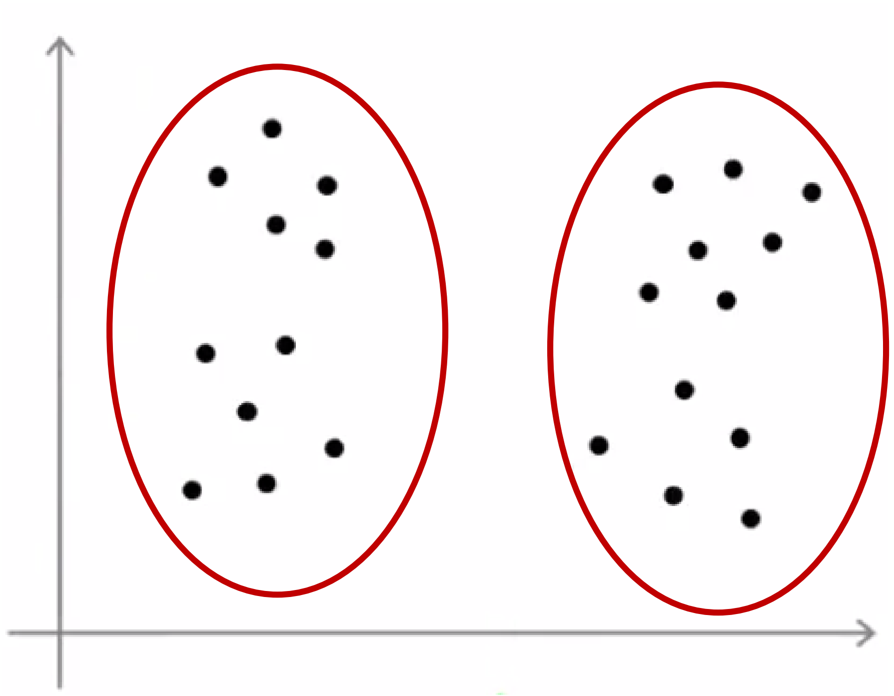
Compute cost function (distortion) $J\left(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K\right)$

}

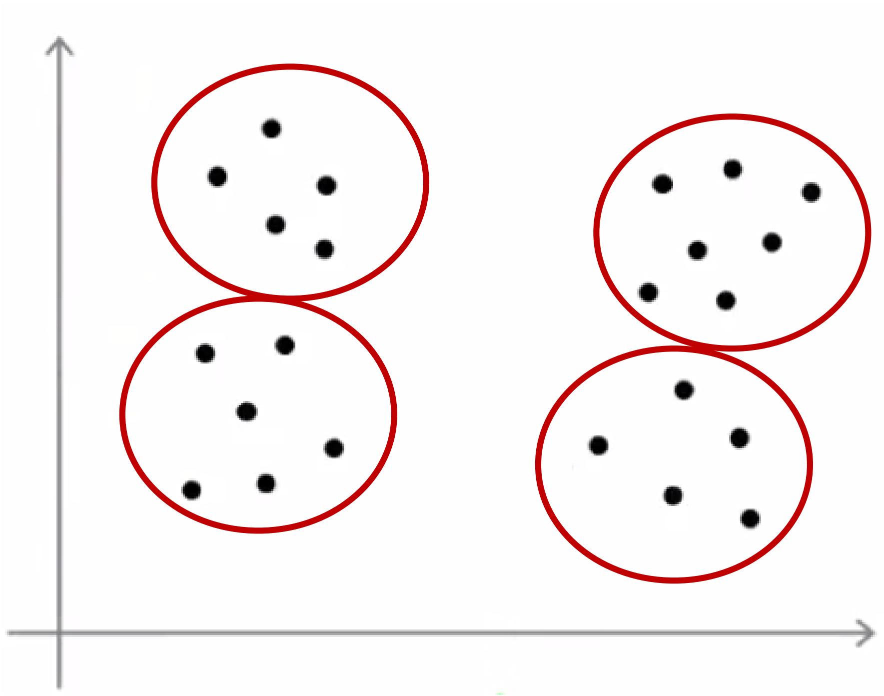
- **Pick clustering that gave lowest cost $J\left(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K\right)$**

K-means Algorithm: Choosing number of clusters

K=2



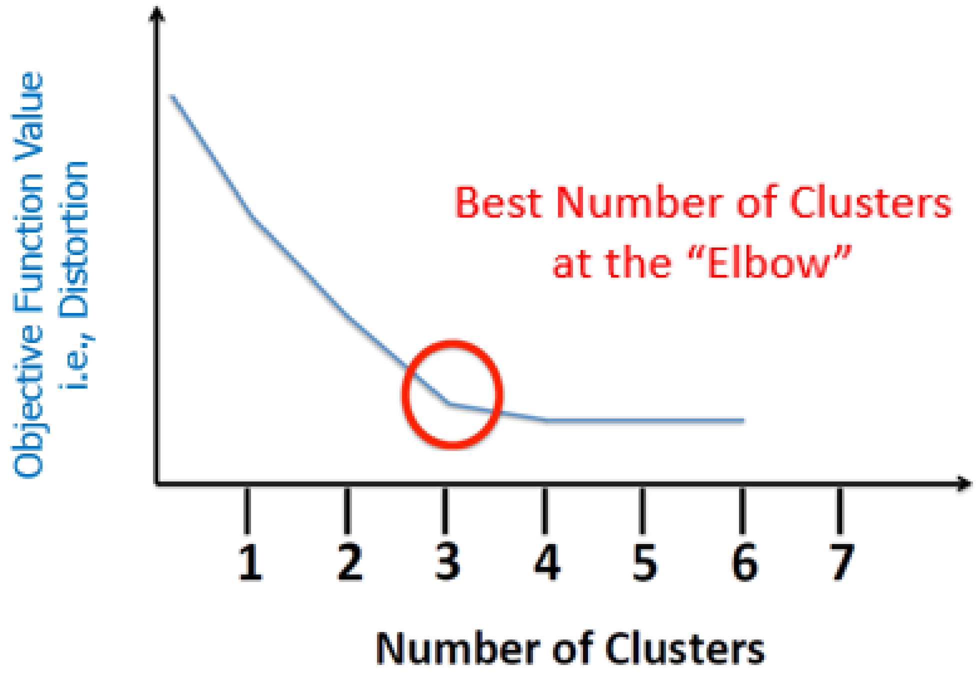
K=4



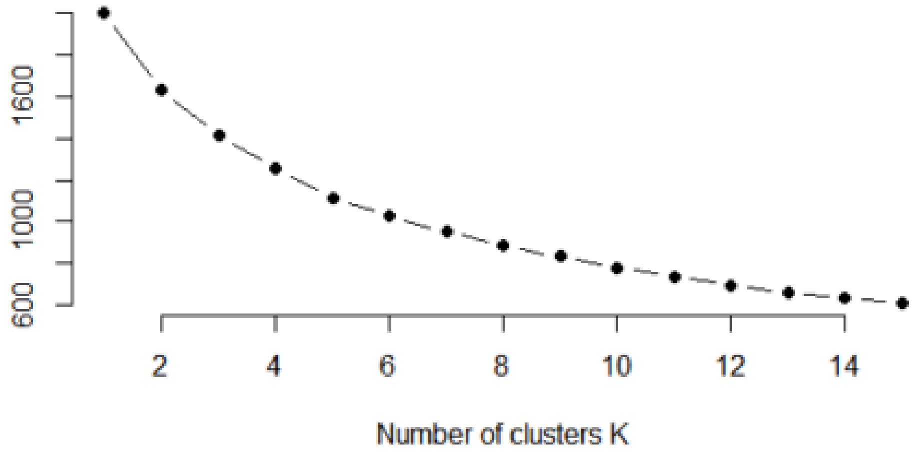
K-means Algorithm: Choosing number of clusters

One possible way is using Elbow Method

- Plotting cost function J verses number of clusters

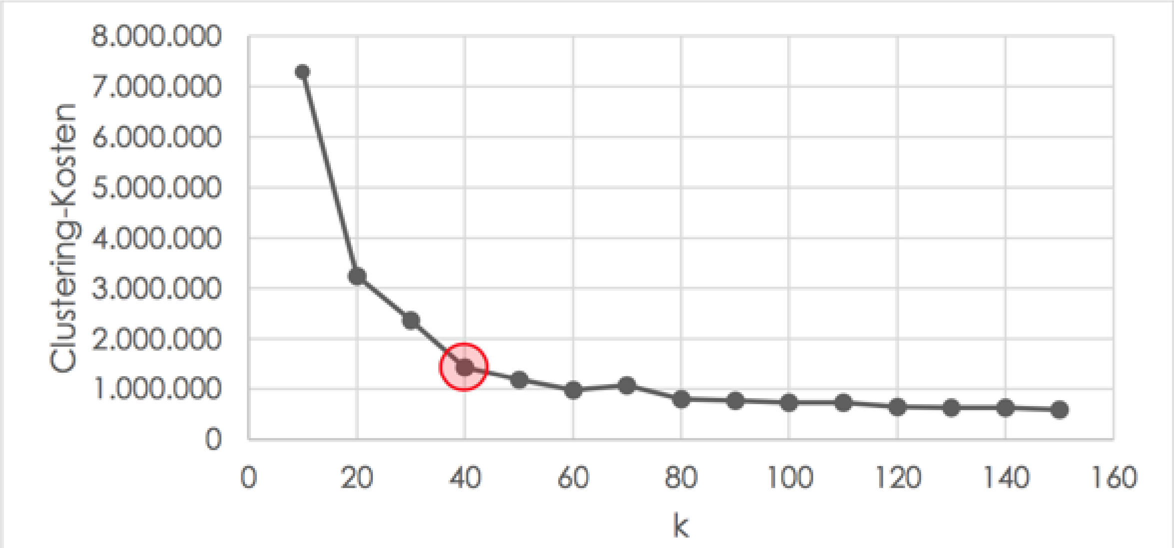


But, you may not always find an "elbow"...



K-means Algorithm: Choosing number of clusters

- Using the elbow method we run k-means clustering for a range of values of k. (e.g. 1 to 150).
- For each value of k we then compute the sum of squared errors (SSE) and add both into a line plot.
- Illustration 1 shows an exemplary curve of a range of values of k and the corresponding SSE.



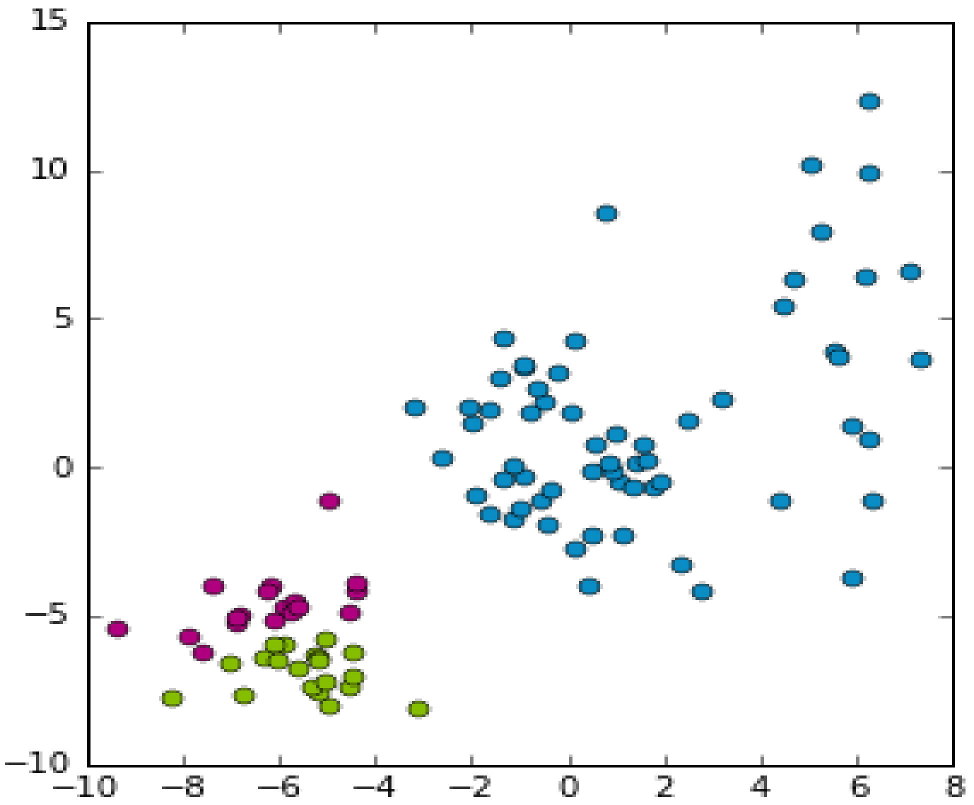
K-means Algorithm: Evaluation

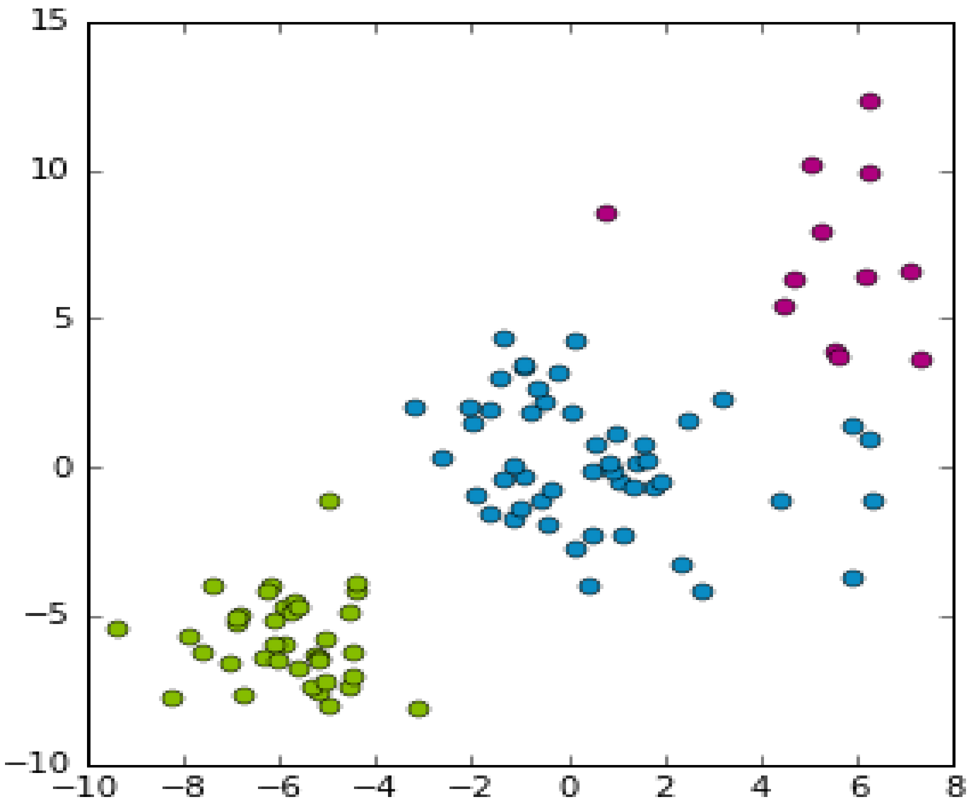
- Guaranteed to converge in a finite number of iterations

- Running time per iteration:
  1. Assign data points to closest cluster center: O(KN) time
  2. Change the cluster center to the average of its assigned points O(N)

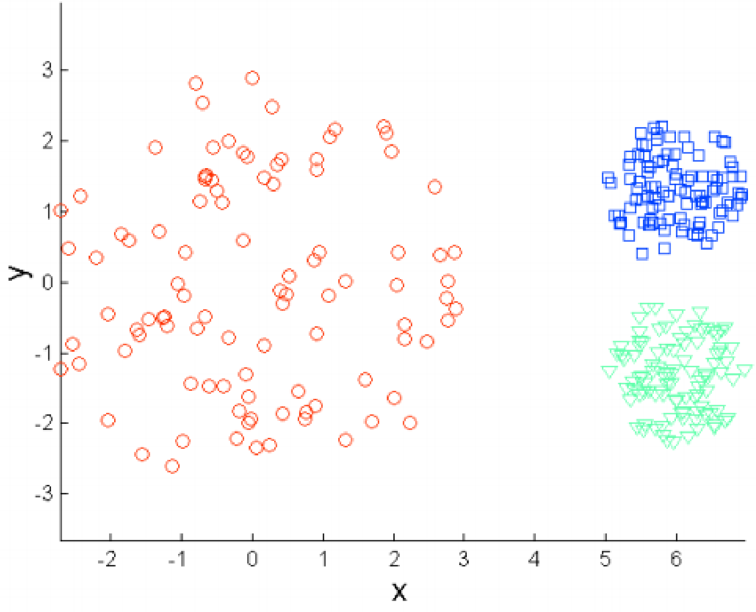
K-means Algorithm: Assessing Clustering Quality

- Which clustering is better?

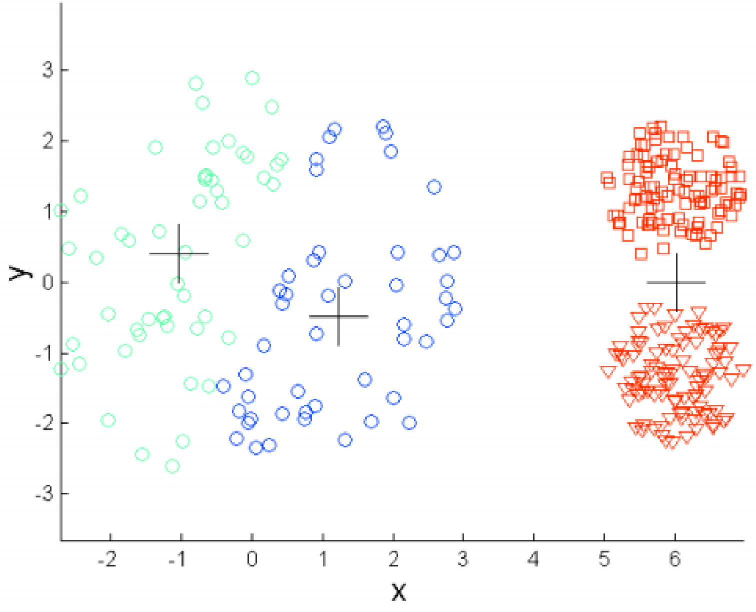




K-means Algorithm: Assessing Clustering Quality



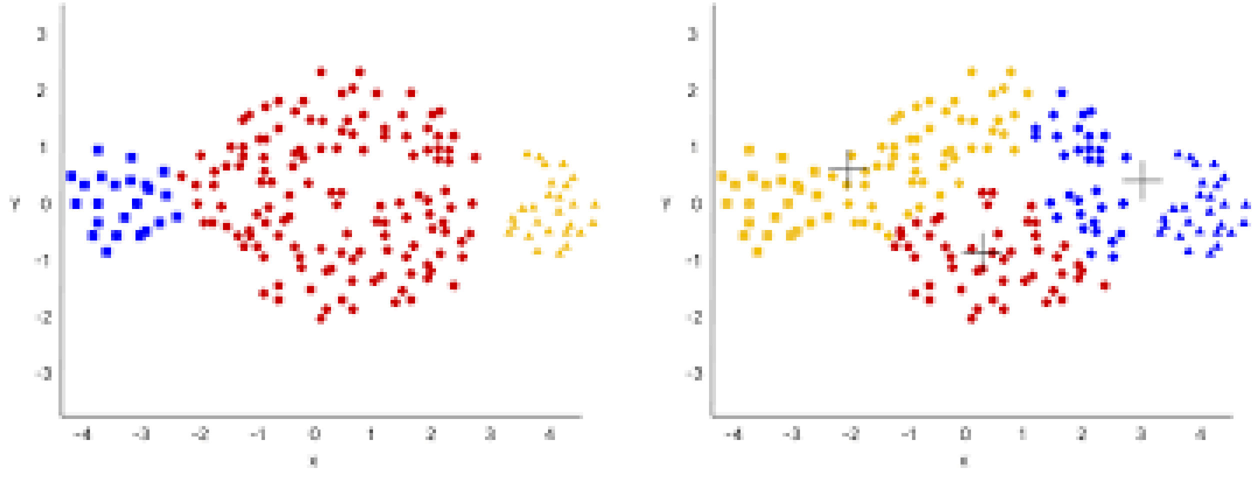
Original Points

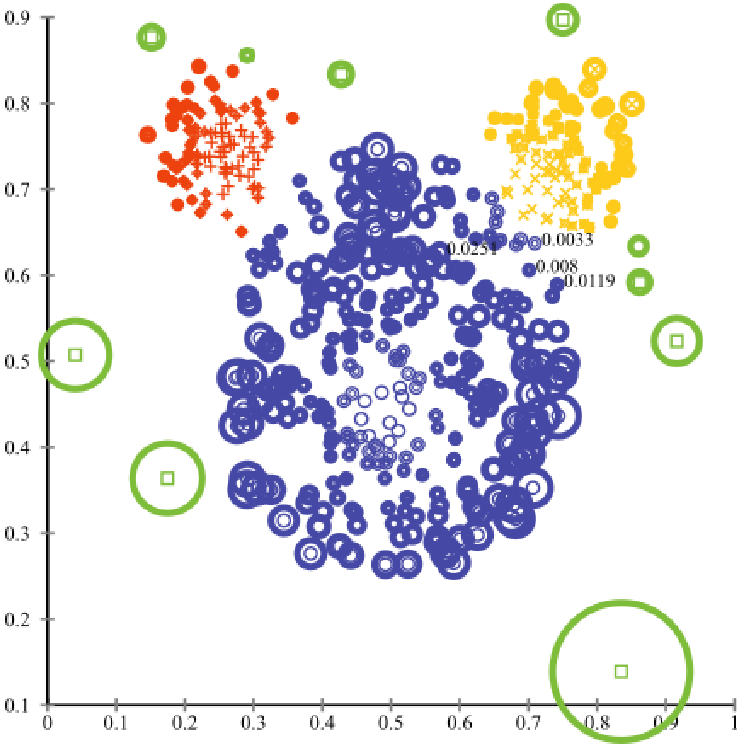


K-means (3 Clusters)

K-means Algorithm: Clustering with Outliers

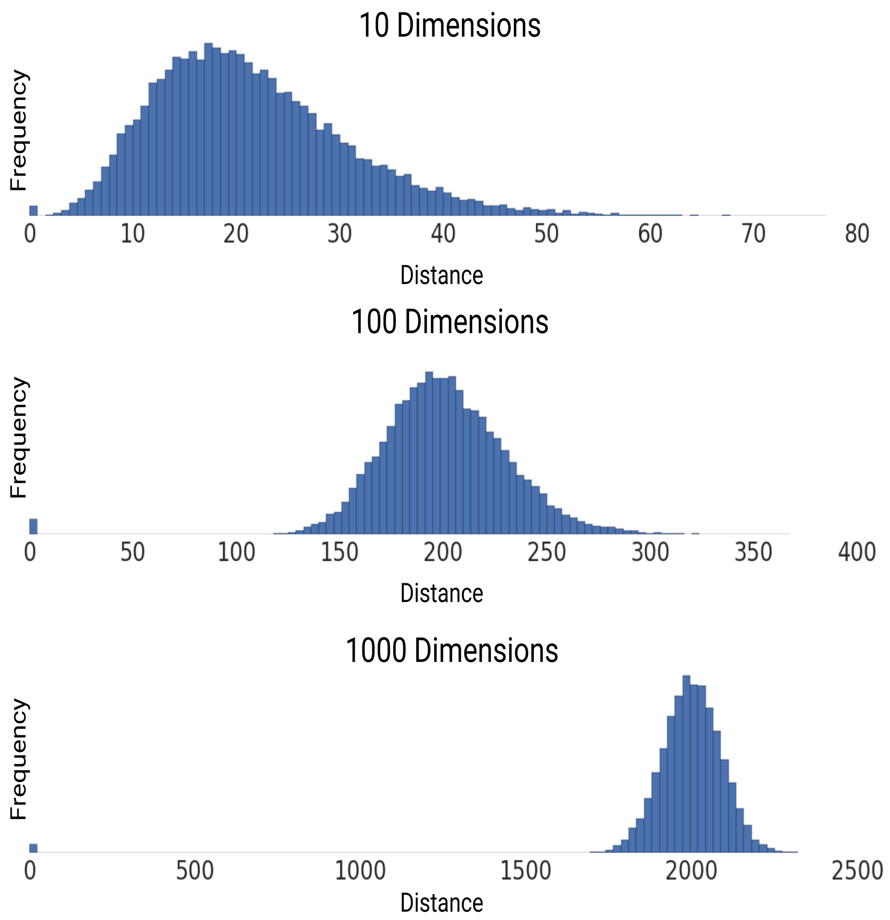
- Centroids can be dragged by outliers, or outliers might get their own cluster instead of being ignored.
- Consider removing or clipping outliers before clustering.





K-means Algorithm: Curse of Dimensionality

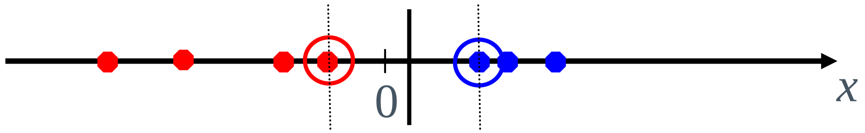
- ➢ These plots show how the ratio of the standard deviation to the mean of distance between examples decreases as the number of dimensions increases.
- ➢ This convergence means k-means becomes less effective at distinguishing between examples.
- ➢ This negative consequence of high-dimensional data is called the curse of dimensionality.



https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages

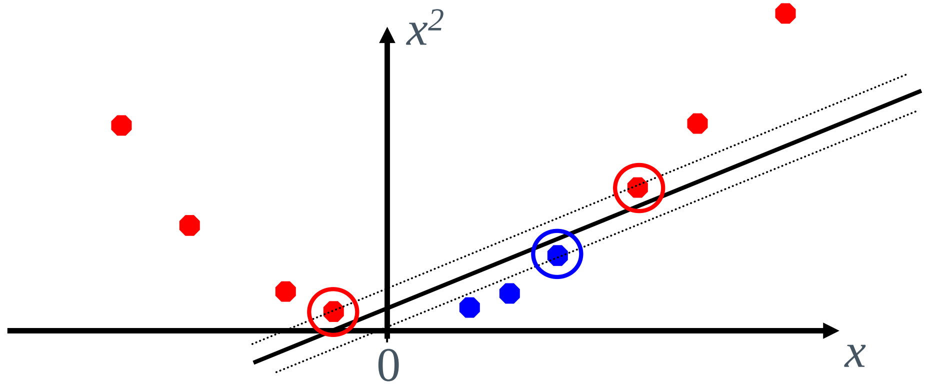
K-means Algorithm: Non-linear Separators

- Data that is linearly separable works out great for linear decision rules:



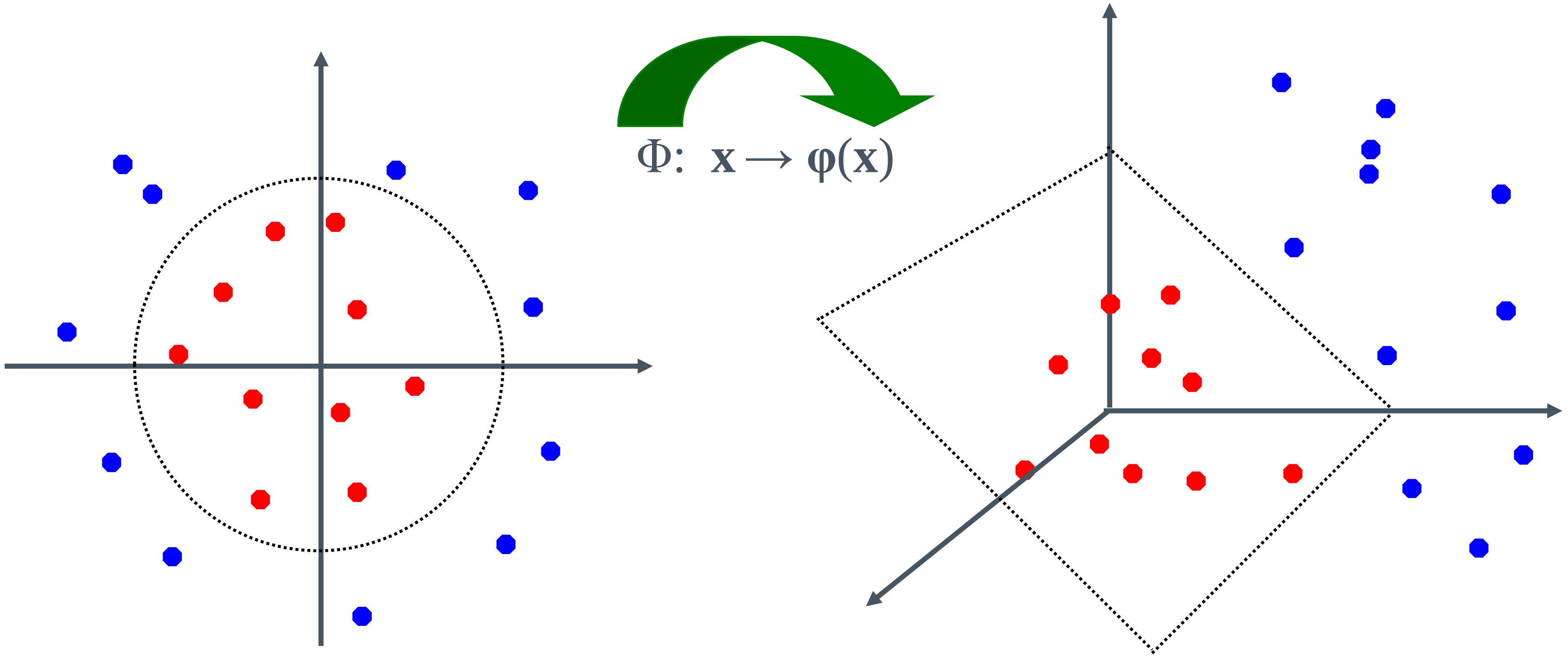
- But what are we going to do if the dataset is just too hard?

- How about… mapping data to a higher-dimensional space:



K-means Algorithm: Non-linear Separators

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable



http://ai.berkeley.edu/lecture_slides.html