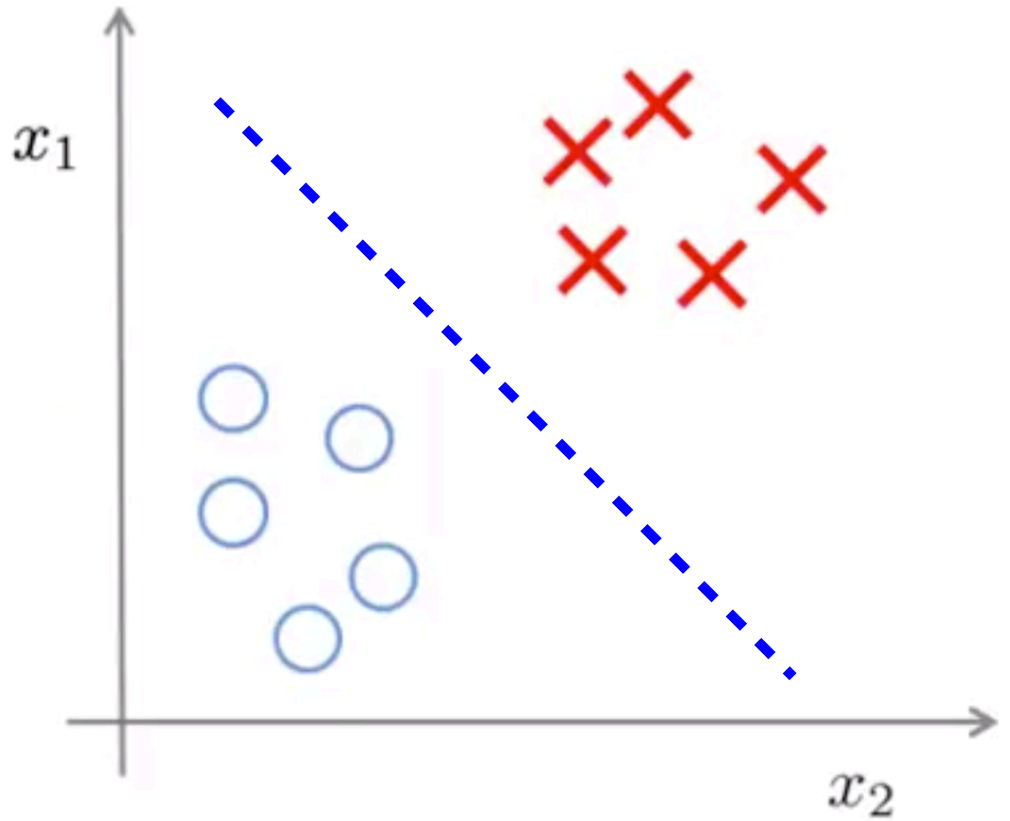
Unsupervised Learning (for Clustering)

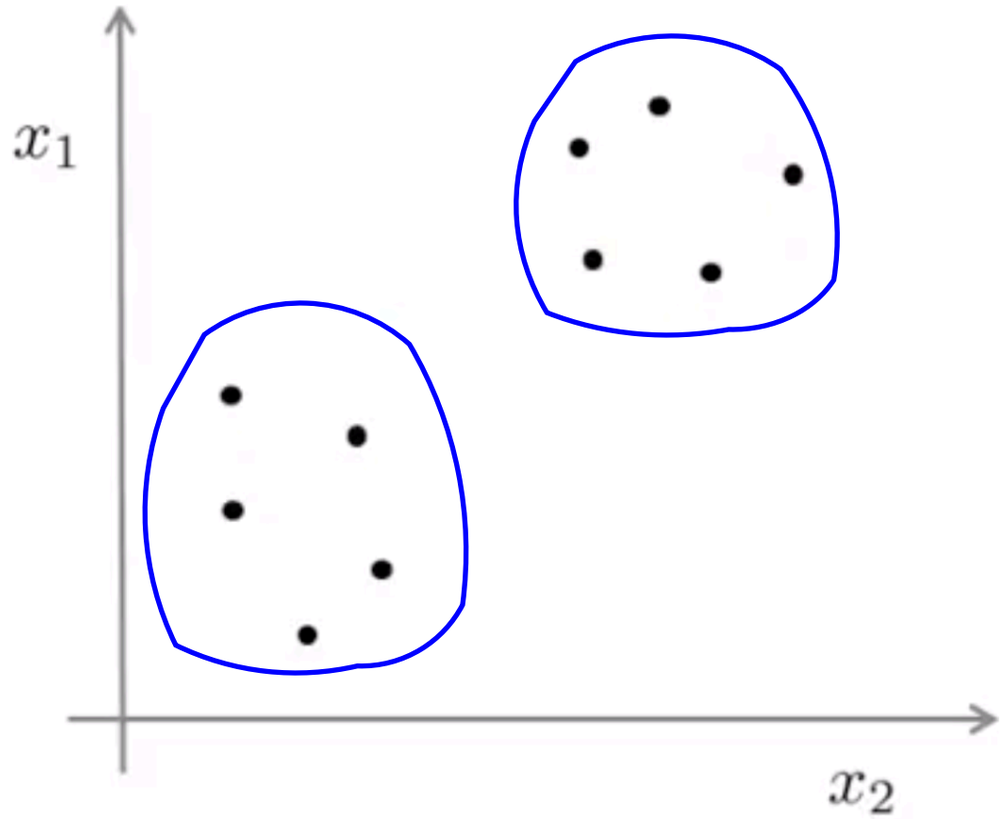
Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

Supervised Learning (Recap)



Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

Unsupervised Learning



We ask an unsupervised learning algorithm to find some structure in the data for us

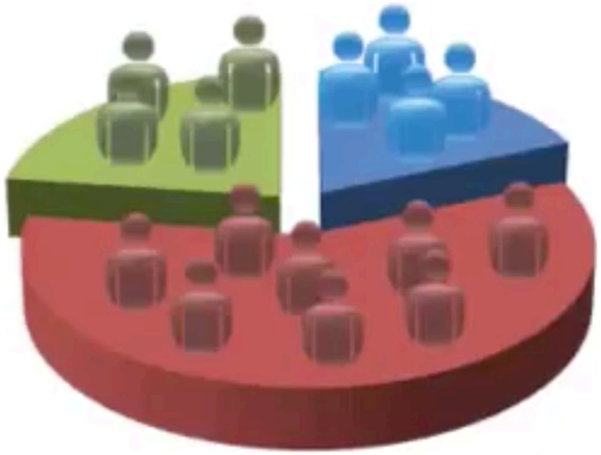
Given this dataset, a type of structure we might have an algorithm to find is:

- clustering

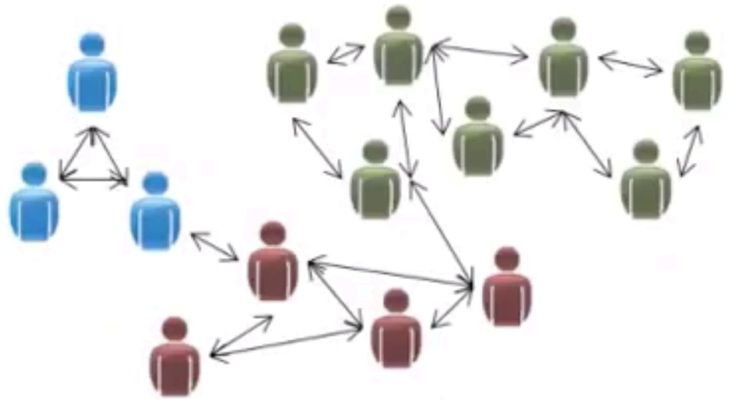
There are also other types of structures that we may find; but,
let's talk about that later !

Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

What's clustering good for?



Market Segmentation



Social Network Analysis

Organizing Computing Clusters

Astronomical Data Analysis

Question

- Which of the following statements are true? Circle all that apply.

(i) In unsupervised learning, the training set is of the form $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ without labels $y^{(i)}$.

(ii) Clustering is an example of unsupervised learning.

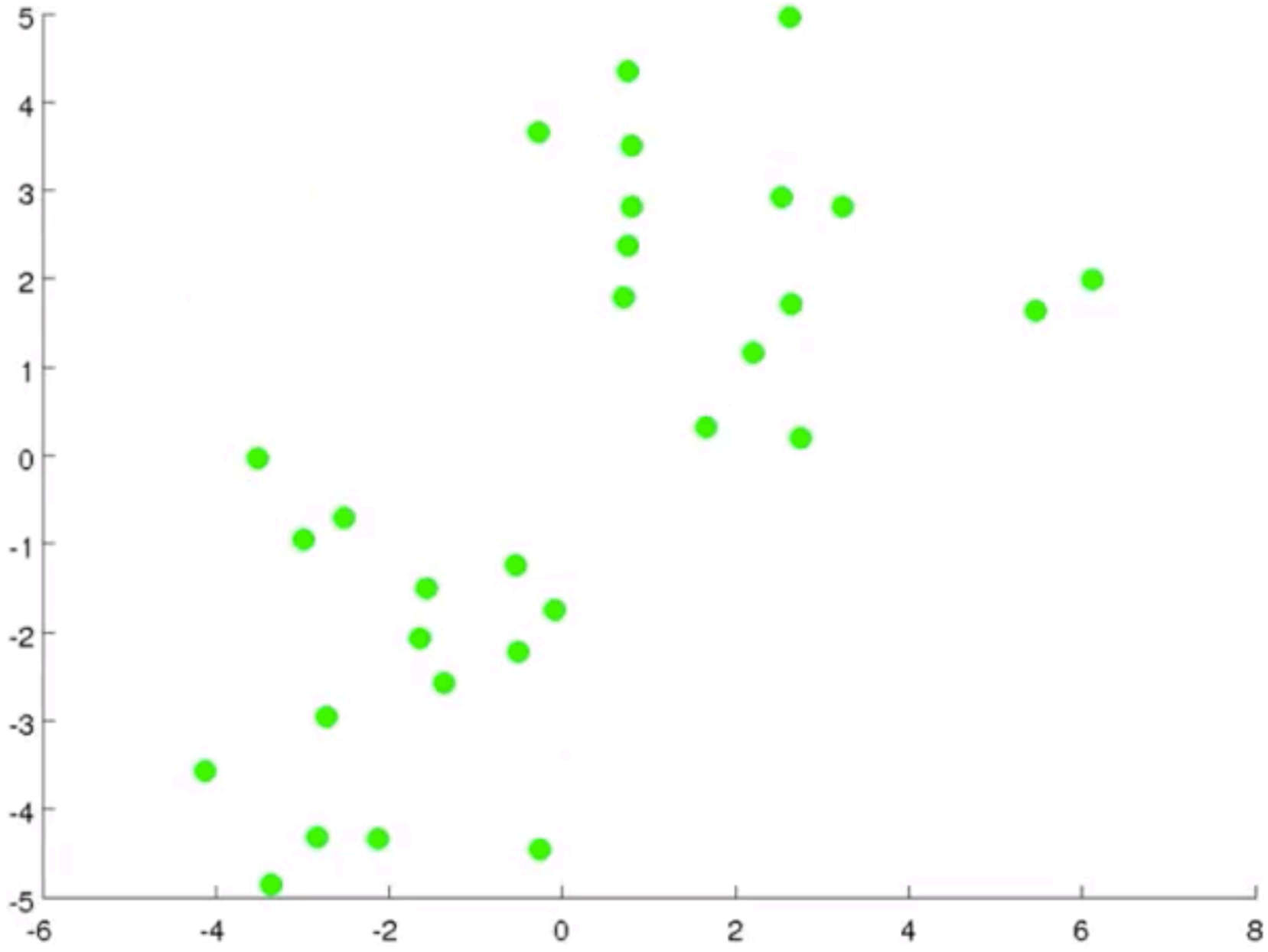
(iii) In unsupervised learning, you are given an unlabeled dataset and are asked to find 'structure' in the data.

(iv) Clustering is the only unsupervised learning algorithm.

K-means Algorithm

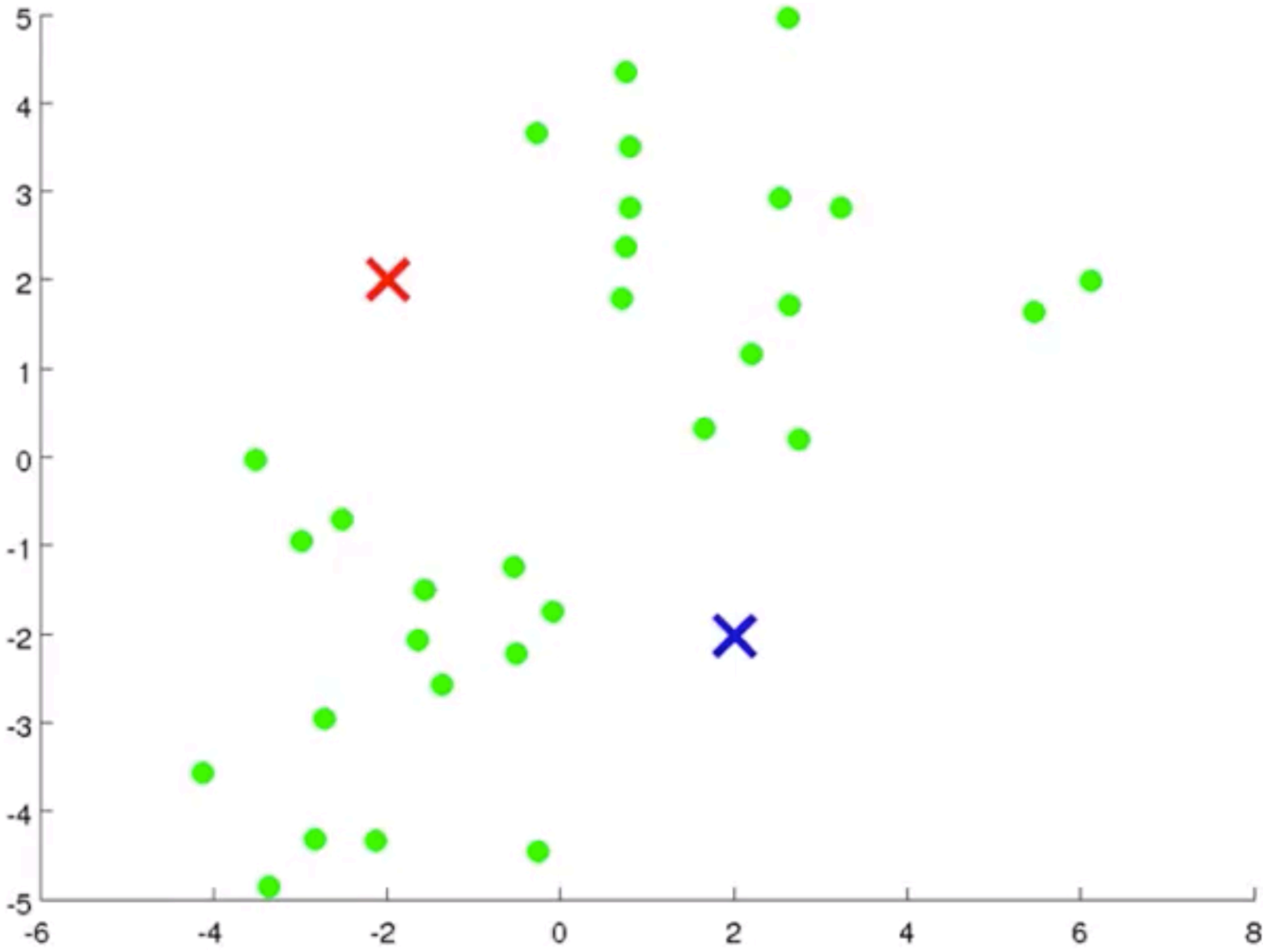
K-means (Intuition)

Suppose that we want to group the data into two clusters



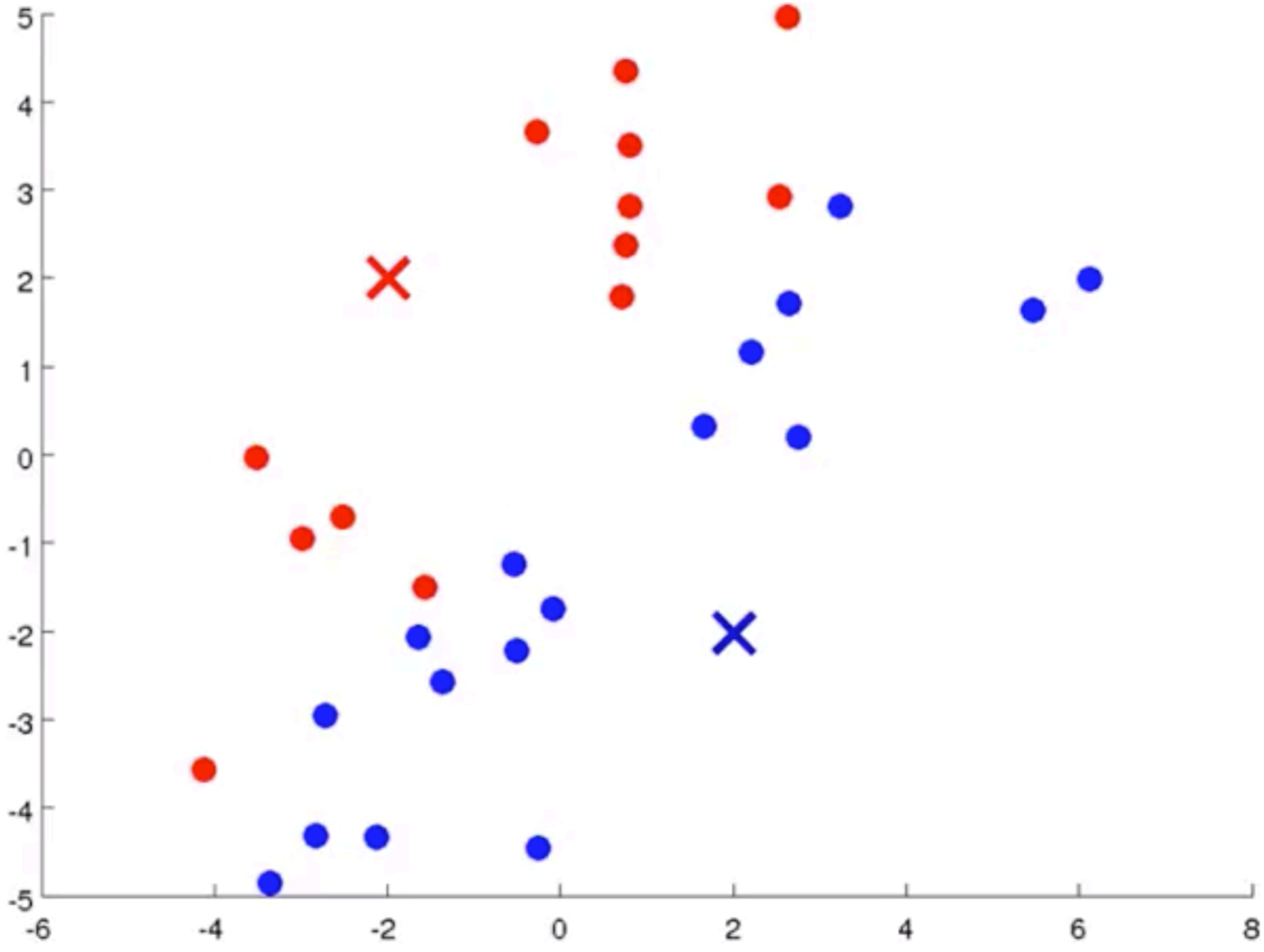
K-means (Intuition)

First, randomly initialize two points called the ‘cluster centroids’.



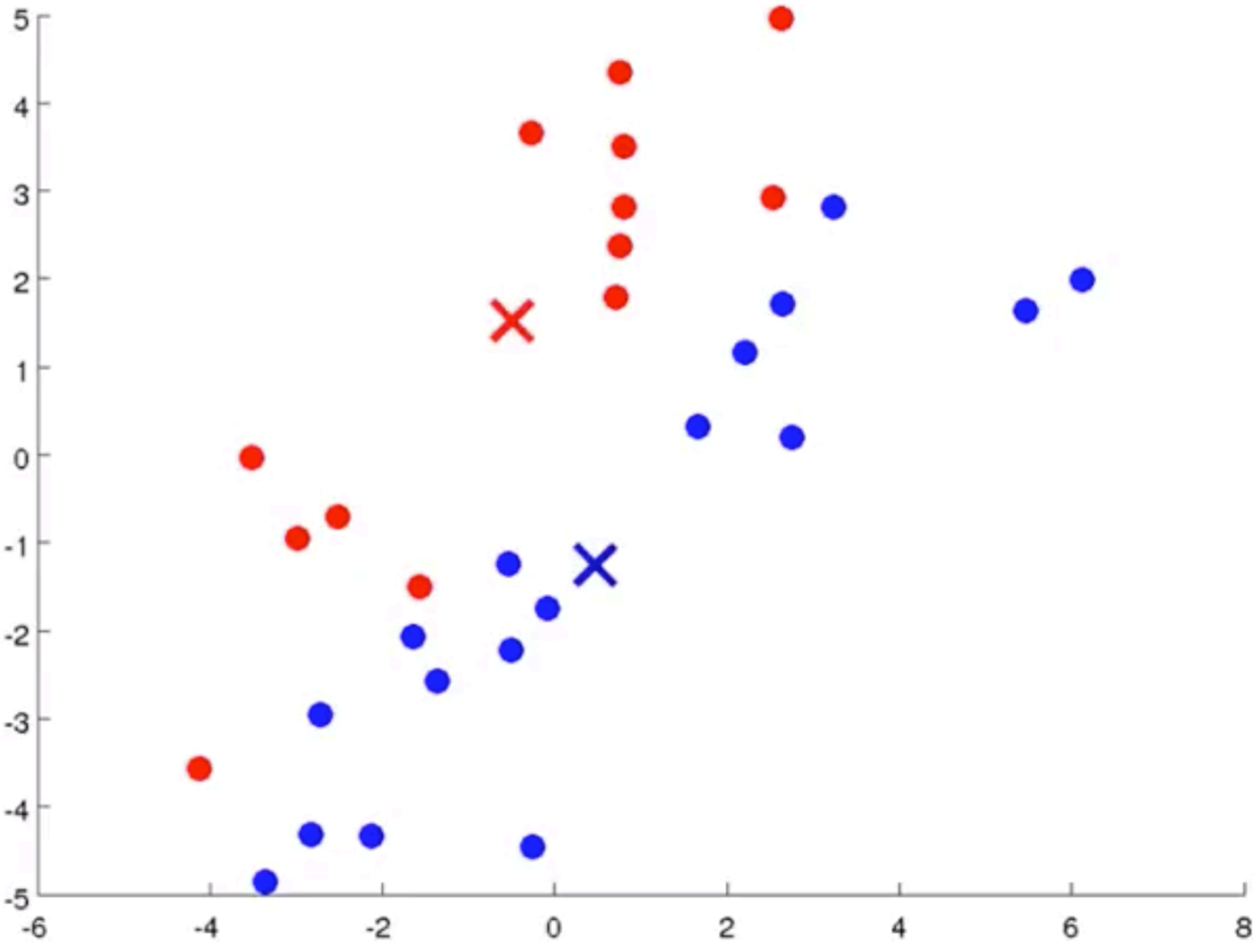
K-means (Intuition)

Second, color each example depending on its closer cluster centroid.



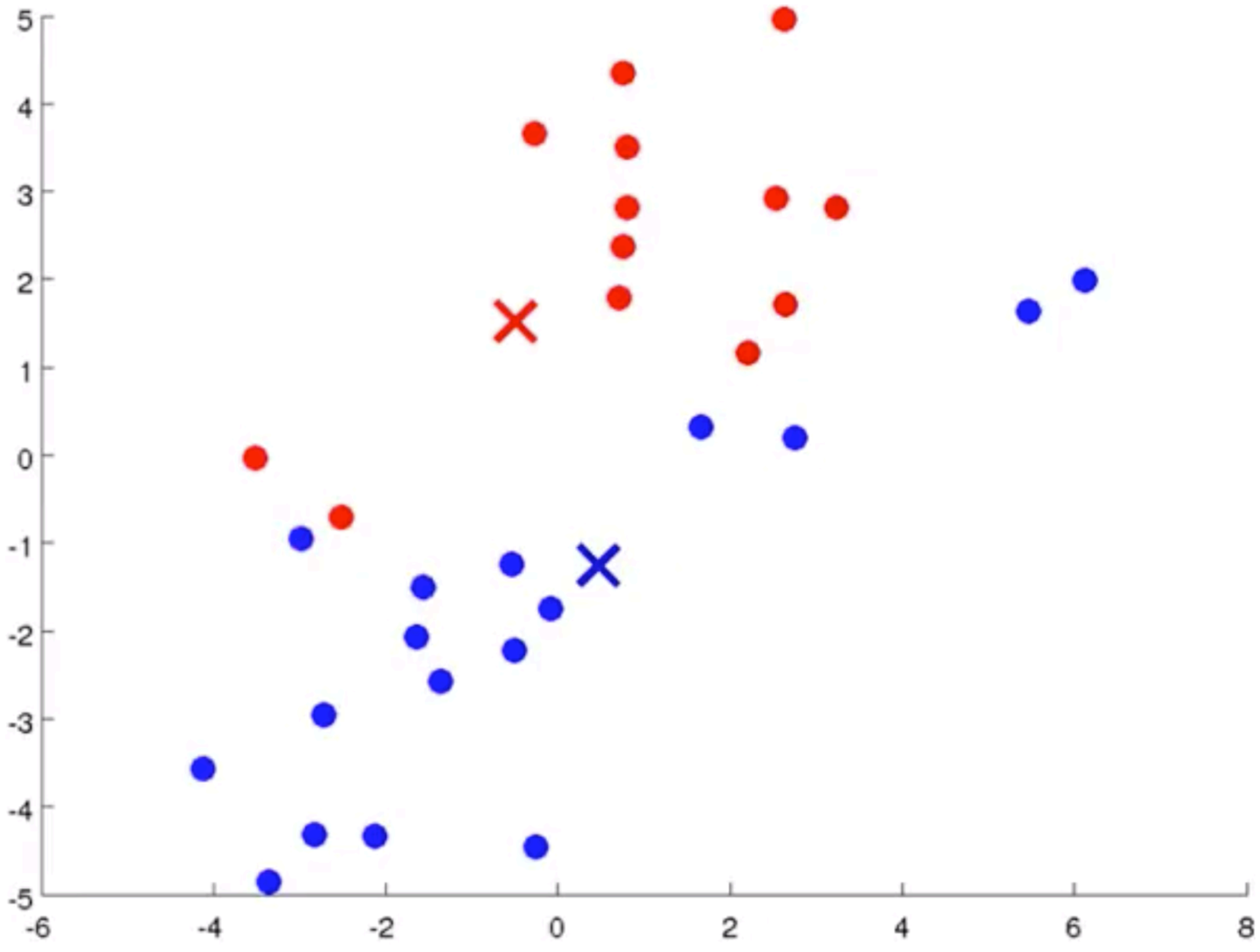
K-means (Intuition)

Third, move cluster centroids to their average of the same color.



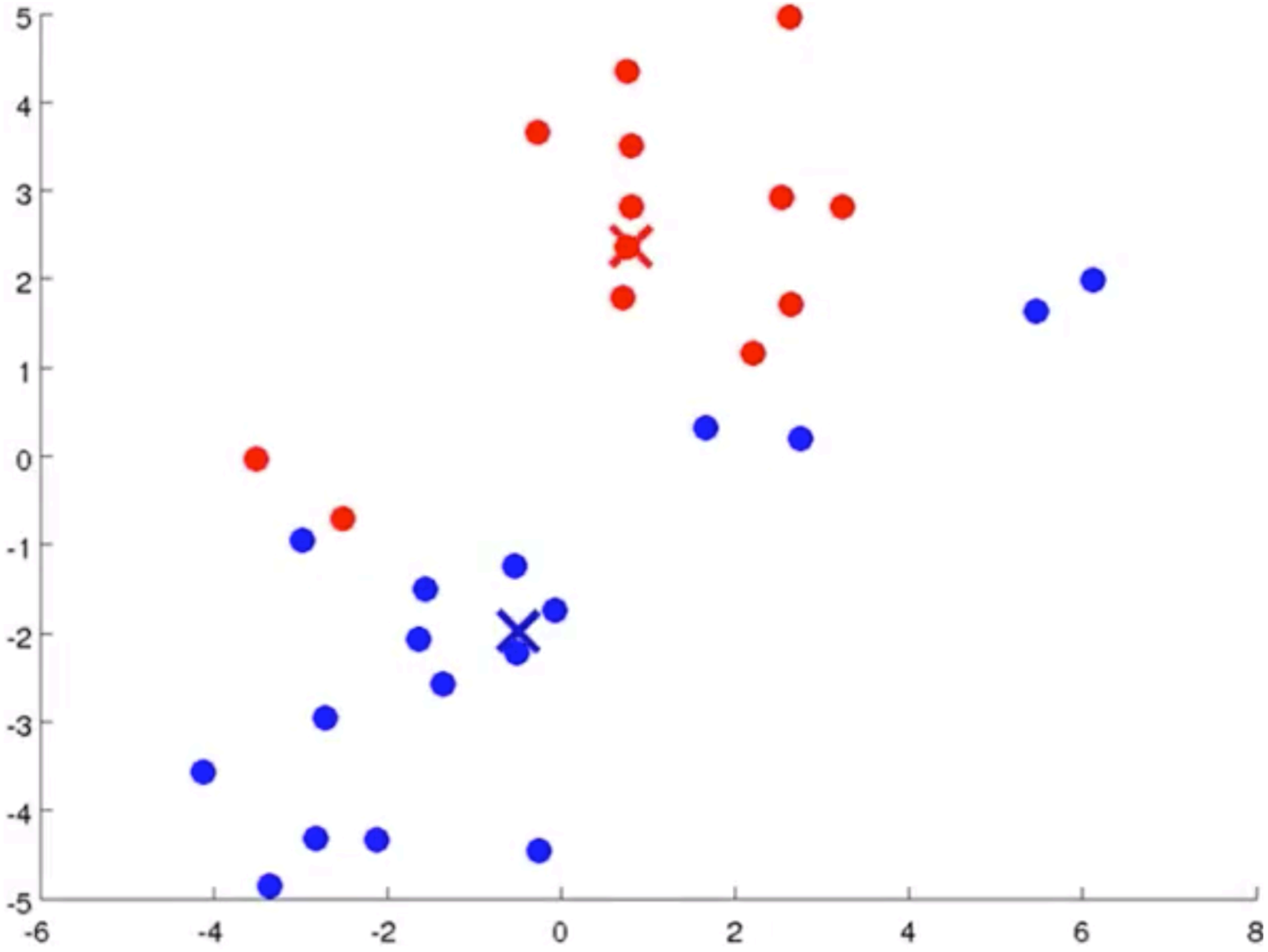
K-means (Intuition)

Next, repeat the same iterative steps and obtain new coloring as follows:



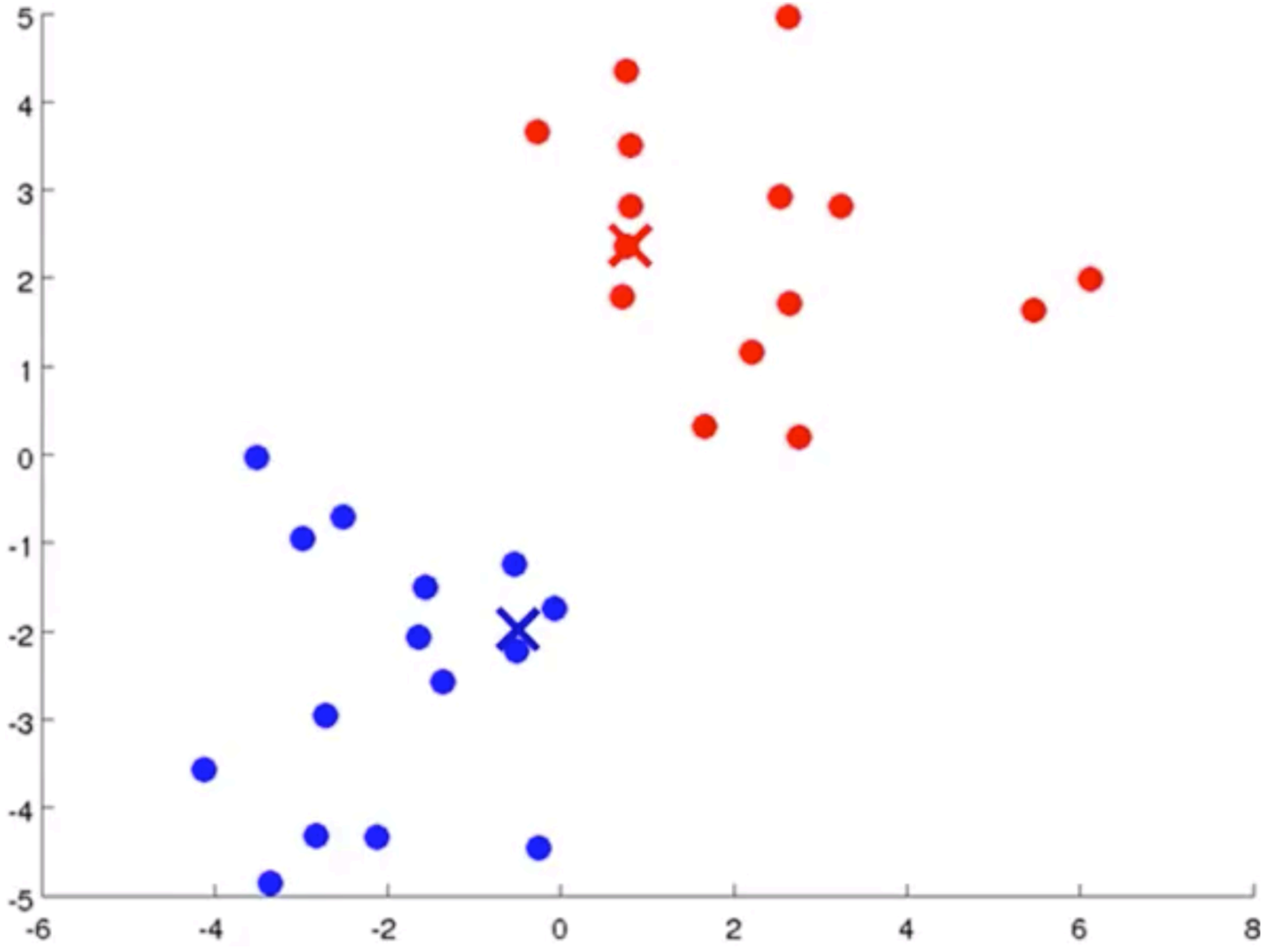
K-means (Intuition)

Next, repeat the same iterative steps and obtain new coloring as follows:



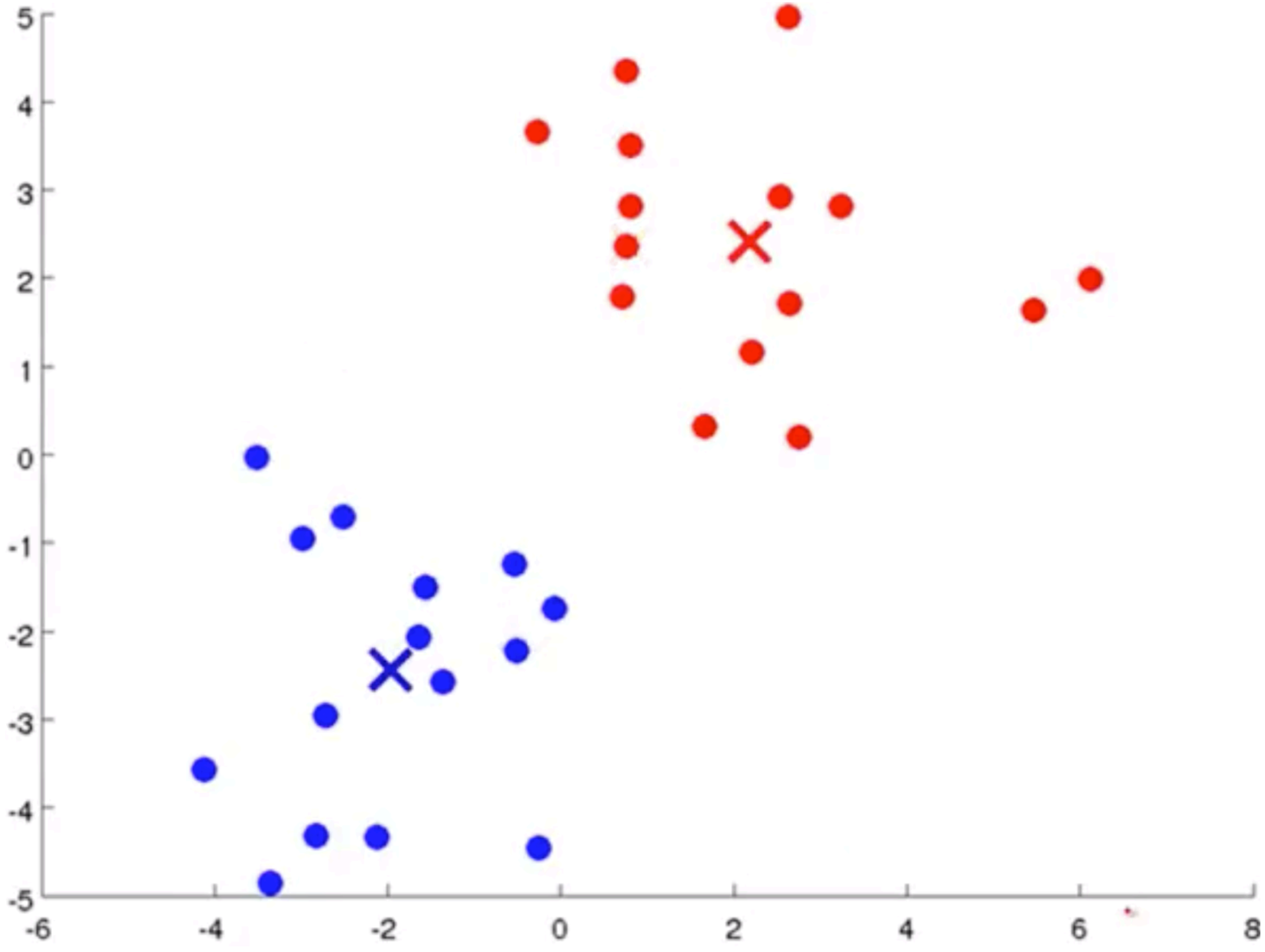
K-means (Intuition)

Next, repeat the same iterative steps and obtain new coloring as follows:



K-means (Intuition)

Now, we are done. Running an iteration from here will not change the centroids.



K-means Algorithm

Input: - K (number of clusters)

- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$

(drop $x_0 = 1$ convention)

Algorithm:

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

for $i = 1$ to m ($c^{(i)} := \min_k \|x^{(i)} - \mu_k\|^2$)

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

for $k = 1$ to K what if there're no points assigned to cluster k?

μ_k := average (mean) of points assigned to cluster k

}

Question

- Suppose you run k-means and after the algorithm converges, you have:

$c^{(1)} = 3,\ c^{(2)} = 3,\ c^{(3)} = 5,\ldots$ c1 อยู่กลุ่ม 3 , c2 กลุ่ม 3, c3 กลุ่ม 5

Which of the following statements are true? Circle all that apply.

(i) The third example $x^{(3)}$ has been assigned to cluster 5.

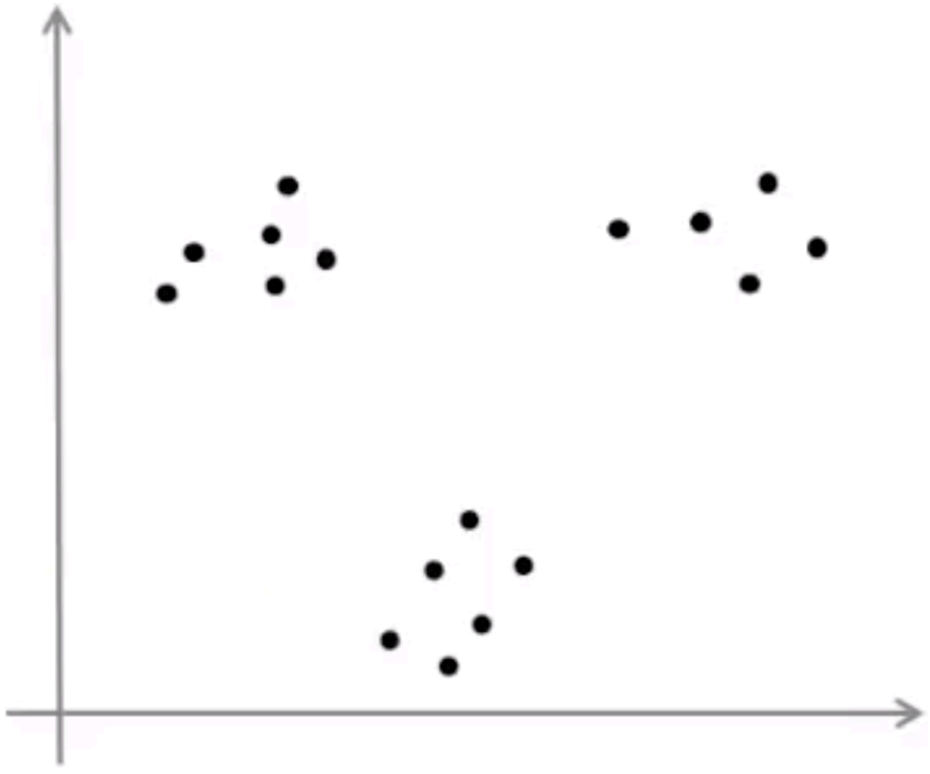
(ii) The first and second training examples $x^{(1)}$ and $x^{(2)}$ have been assigned to the same cluster.

(iii) The second and third training examples have been assigned to the same cluster.

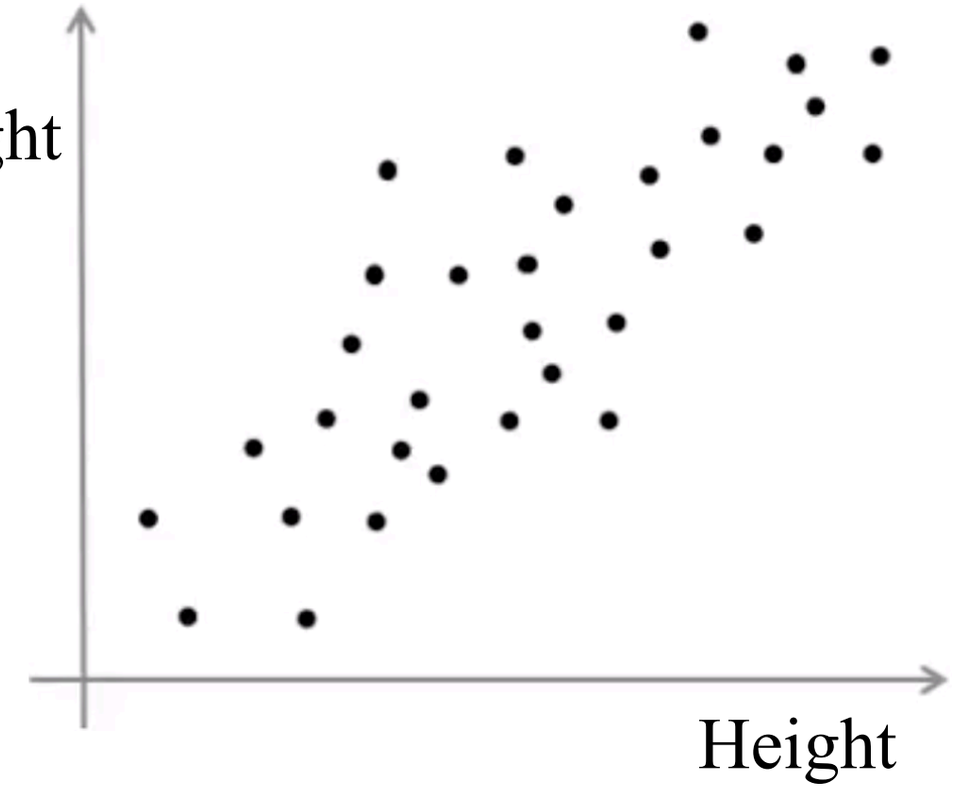
(iv) Out of all the possible values of $k \in \{1, 2, \ldots, K\}$ the value $k = 3$ minimizes $\|x^{(2)} - \mu_k\|^2$

K-means for Non-Separated Clusters

Separable Clusters



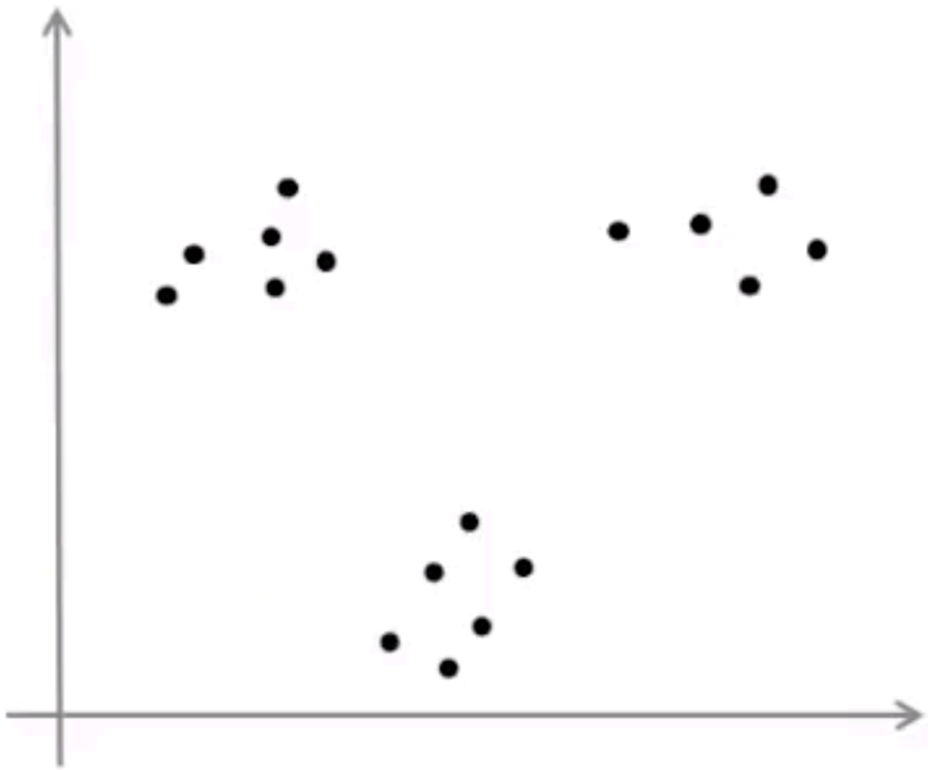
Non-separable Clusters



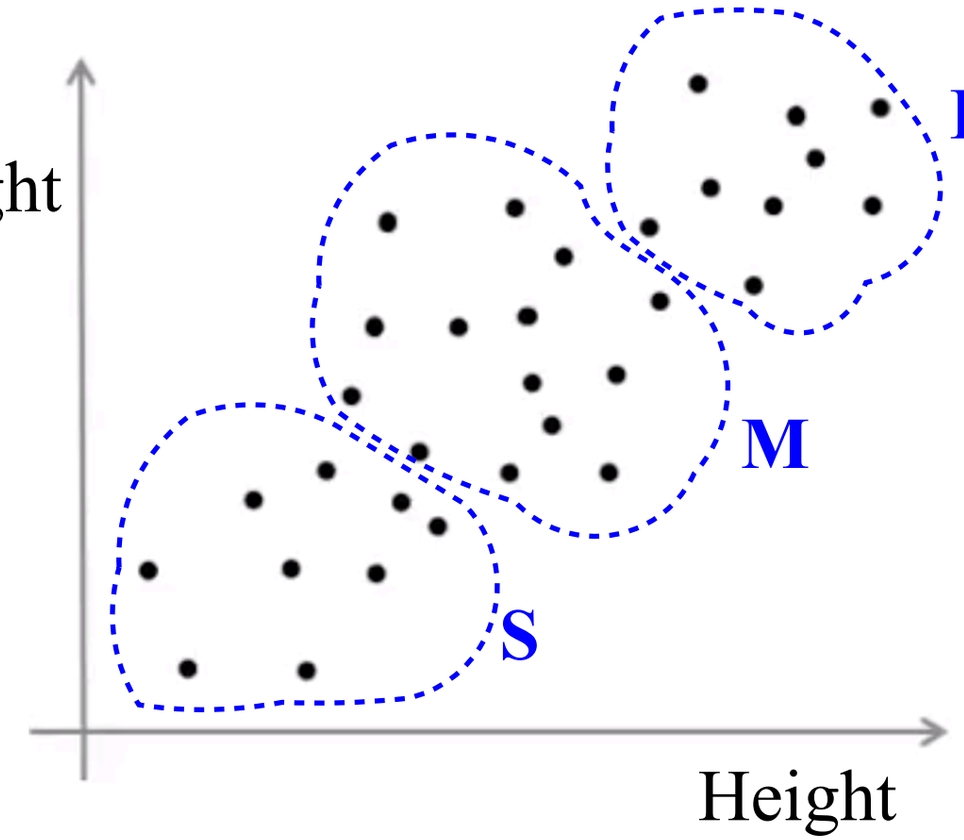
T-shirt Sizing

K-means for Non-Separated Clusters

Separated Clusters



Non-separated Clusters



T-shirt Sizing

Optimization Objective

K-means Optimization Objectives

Formal Notation

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

μ_k := cluster centroid k ($\mu_k \in \mathbb{R}^n$)

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

Cost Function (sometimes called 'distortion')

$$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K) = \frac{1}{m}\Sigma_{i=1}^{m}\|x^{(i)} - \mu_{c^{(i)}}\|^2$$

Objective Function

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

K-means Algorithm

Input: - K (number of clusters)

- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$

(drop $x_0 = 1$ convention)

Algorithm:

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$ **(Other variables remain fixed)**

Repeat {

  for i = 1 to m

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

  for k = 1 to K

    μ_k := average (mean) of points assigned to cluster k

}

'Cluster assignment step': $\min_{c^{(1)},\ldots,c^{(m)}} J(c^{(1)}, \ldots, c^{(m)})$
we can show that it does:

'Move centroid step': $\min_{\mu_1,\ldots,\mu_K} J(\mu_1, \ldots, \mu_K)$ **(Other variables remain fixed)**
we can show that it does:

Question

- Suppose you have implemented K-means and to check that it's running correctly, you plot the cost function $J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$ as a function of the number of iterations. Your plot looks like this.
  What does it mean?

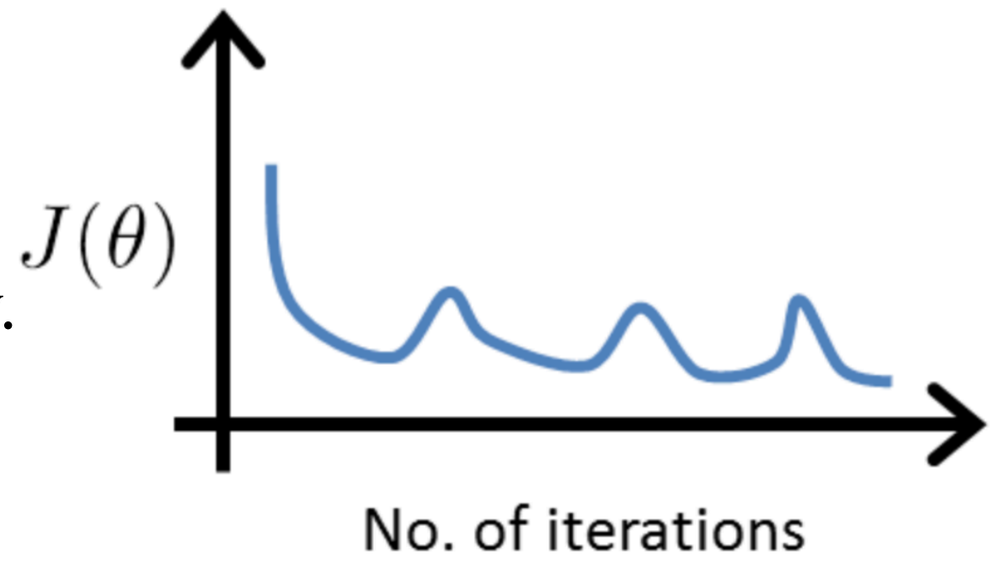
learning rate ไม่ได้ set

(i) The learning rate is too large.

(ii) The algorithm is working correctly.

(iii) The algorithm is working, but k is too large.

(iv) It is not possible to sometimes increase. There must be a bug in the code.



ถูกเขียน code ถูก J ต้องลดลงเรื่อยๆ

Random Initialization

K-means Algorithm

Input: - K (number of clusters)

- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$, $x^{(i)} \in \mathbb{R}^n$

(drop $x_0 = 1$ convention)

Algorithm:

Randomly initialize K cluster centroids $\mu_1, \mu_2, \ldots, \mu_K \in \mathbb{R}^n$

Repeat {

for $i = 1$ to m

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

for $k = 1$ to K

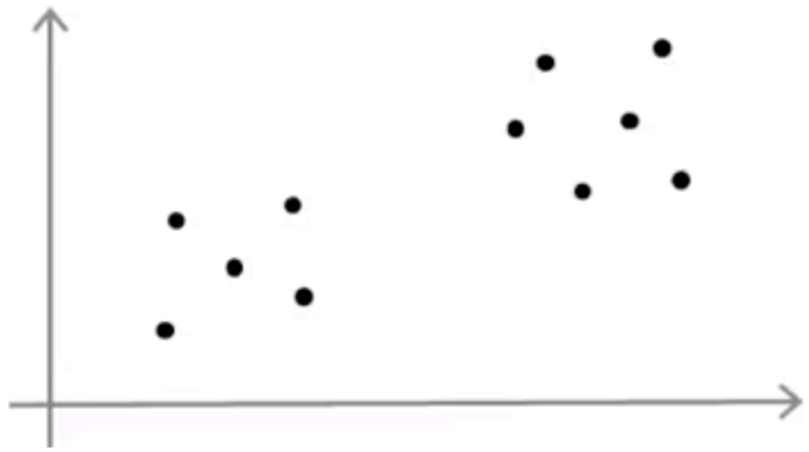
μ_k := average (mean) of points assigned to cluster k

}

Random Initialization

Idea:

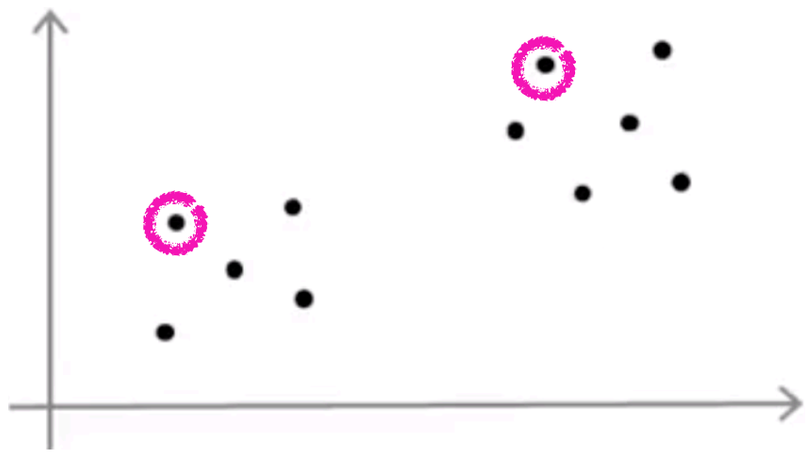
- Should have $K < m$
- Randomly pick K training examples
- Set $\mu_1, \ldots, \mu_K$ equal to these K examples



Random Initialization

Idea:

- Should have $K < m$
- Randomly pick K training examples
- Set $\mu_1, \ldots, \mu_K$ equal to these K examples

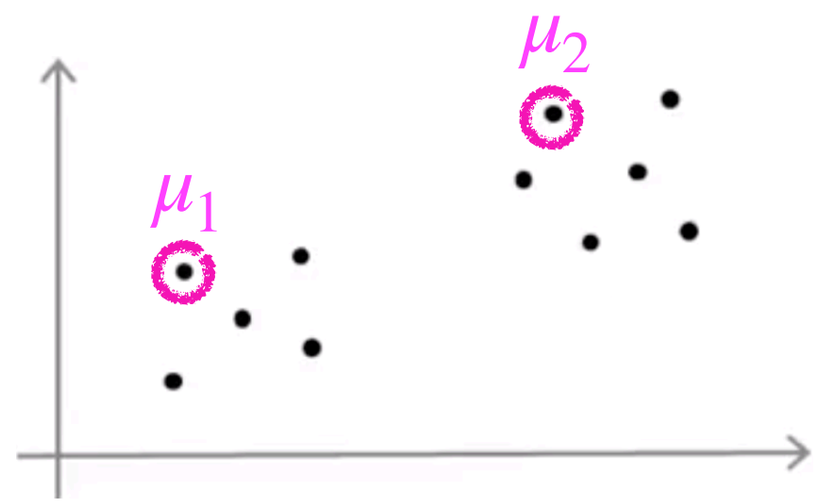


Example 1: let's say $K = 2$

Random Initialization

Idea:

- Should have $K < m$
- Randomly pick K training examples
- Set $\mu_1, \ldots, \mu_K$ equal to these K examples



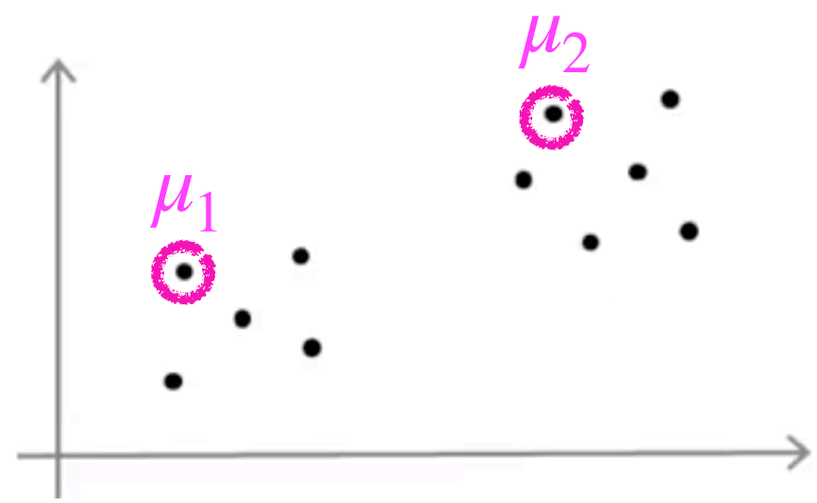
This might be a lucky case !

Example 1: let's say $K = 2$

Random Initialization

Idea:

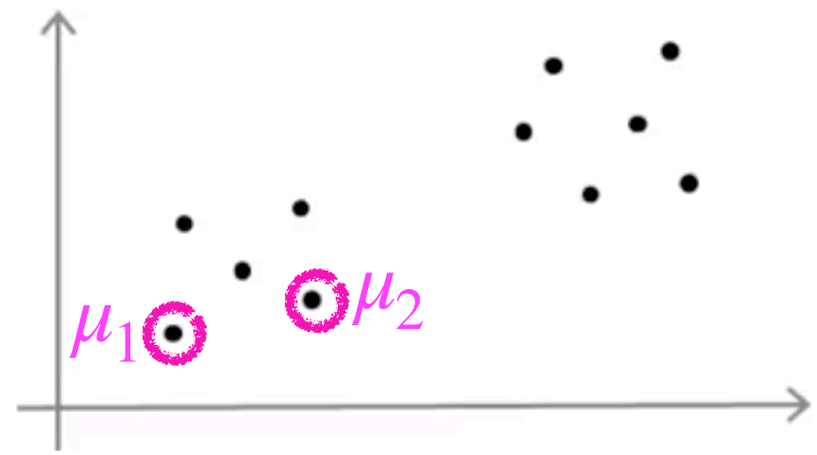
- Should have $K < m$
- Randomly pick K training examples
- Set $\mu_1, \ldots, \mu_K$ equal to these K examples



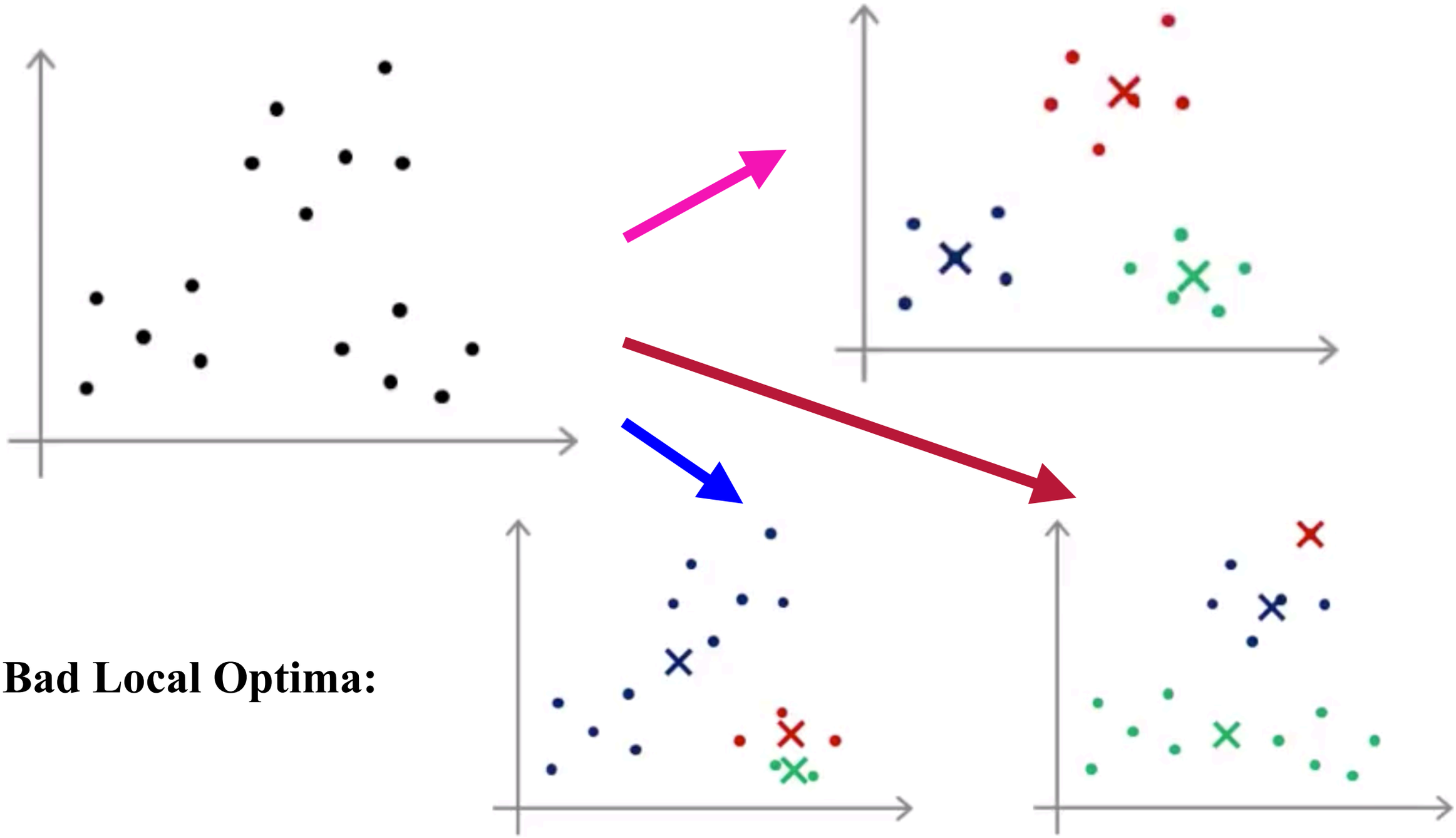
Example 1: let's say $K = 2$

Example 2: let's say $K = 2$

Caution: running k-means may converge to different solutions depending on how $\mu_1, \ldots, \mu_K$ are initialized !

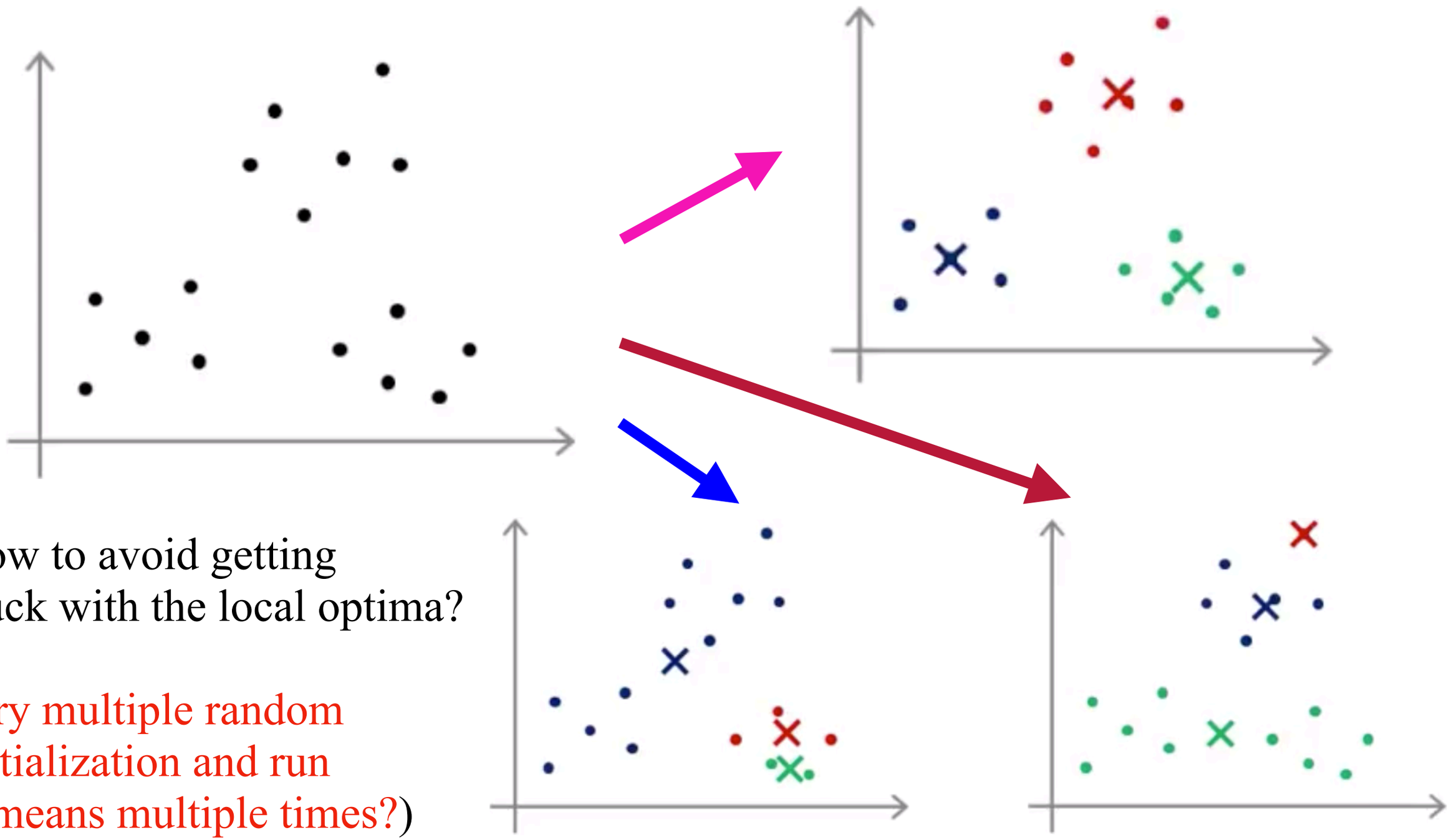


Local Optima



Bad Local Optima:

Local Optima



How to avoid getting stuck with the local optima?

(Try multiple random initialization and run k-means multiple times?)

Random Initialization

This number can be 10 - 1,000 times

For $i = 1$ to 100 {

Randomly initialize K-means.

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

Compute cost function (*aka.* distortion)

$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

}

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

Random Initialization

This number can be 10 - 1,000 times

For $i = 1$ to 100 {

Randomly initialize K-means.

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

Compute cost function (*aka.* distortion)

$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$

}

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

If you are running k-means with a fairly small number of clusters (*e.g.* $K = 2$ - 10), doing multiple random initialization can sometimes yield the better local optima. Otherwise, multiple randomization may still yield the better local optima — but not much !

Question

- Which of the following is the recommended way to initialize k-means?

(i) Pick a random integer i from $\{1, \ldots, K\}$.
Set $\mu_1 = \mu_2 = \ldots = \mu_K = x^{(i)}$.

(ii) Pick k distinct random integers $i_1, \ldots, i_k$ from $\{1, \ldots, K\}$.
Set $\mu_1 = x^{(i_1)}, \mu_2 = x^{(i_2)}, \ldots, \mu_K = x^{(i_k)}$.

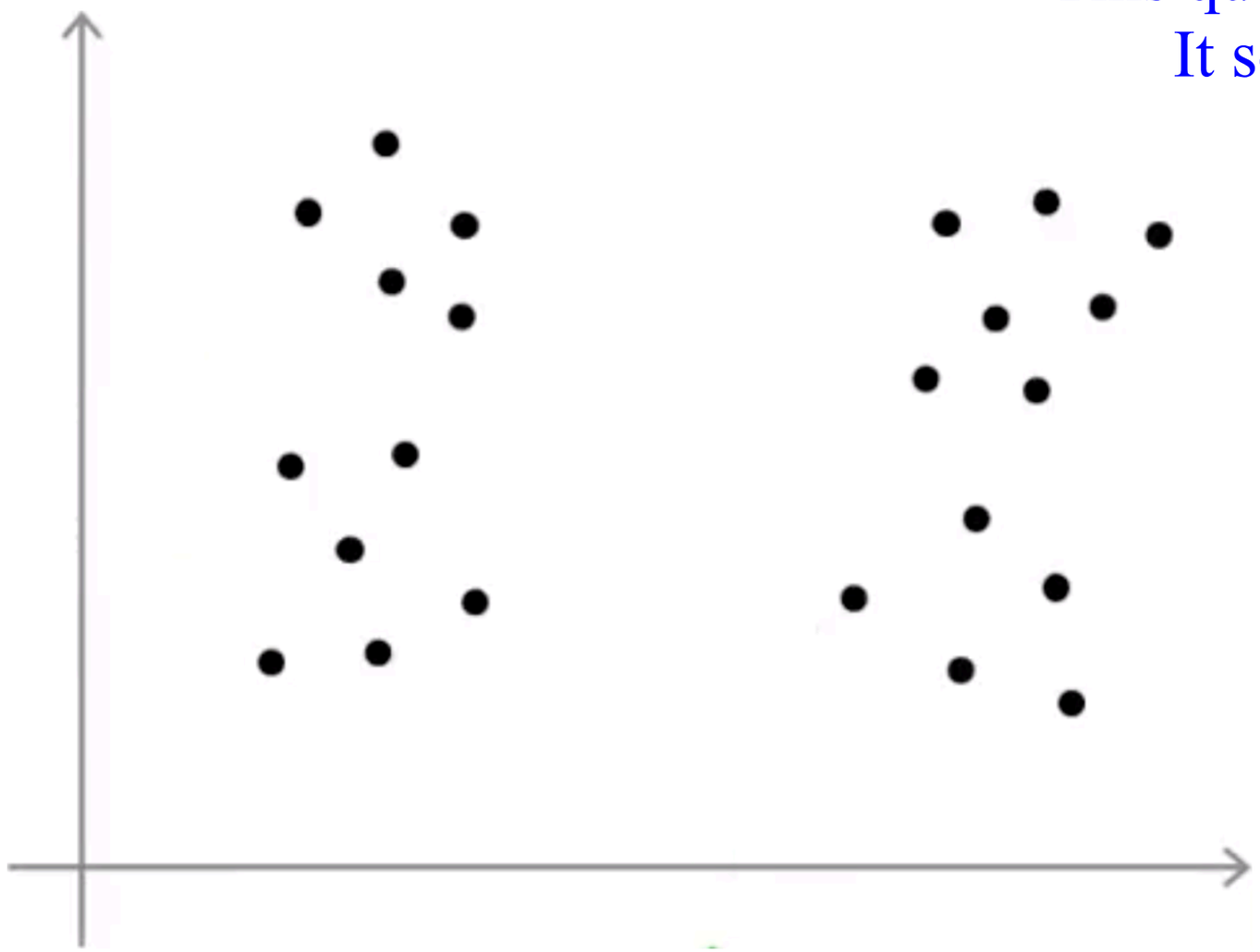
(iii) Pick k distinct random integers $i_1, \ldots, i_k$ from $\{1, \ldots, m\}$.
Set $\mu_1 = x^{(i_1)}, \mu_2 = x^{(i_2)}, \ldots, \mu_K = x^{(i_k)}$.

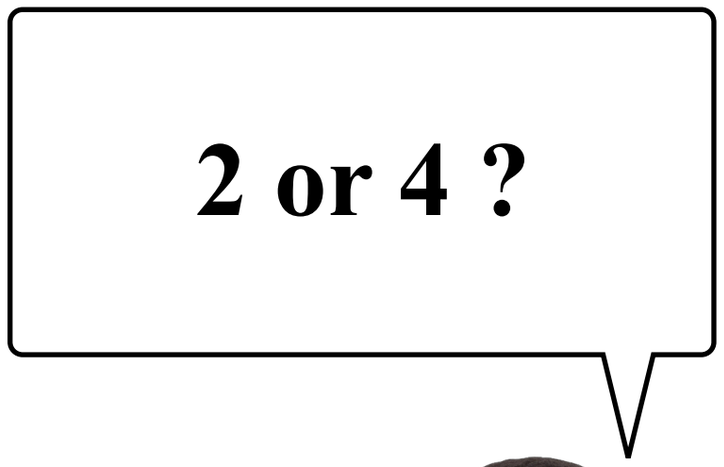
(iv) Set every element of $\mu_i \in \mathbb{R}^n$ to a random value between $-\epsilon$ and ϵ, for some small ϵ.

Choosing the Number of Clusters

What is the right value of K?

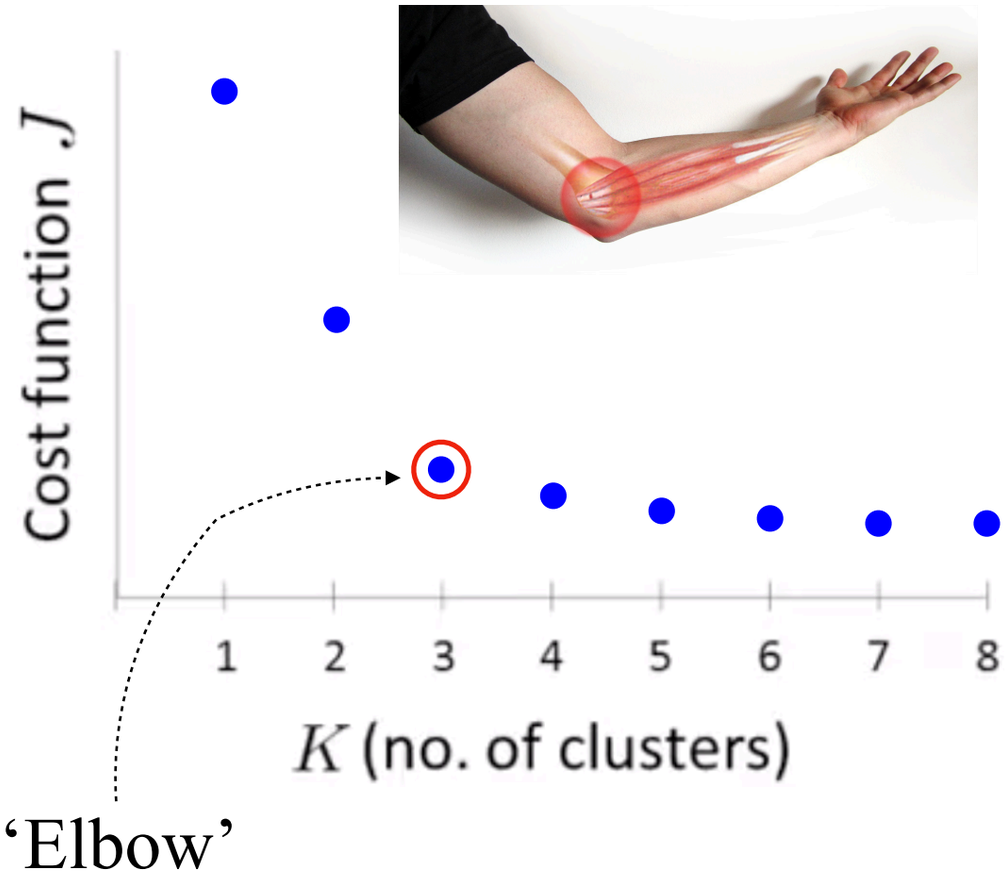
This question is quite subjective to answer.
It seems that there is no the right one !

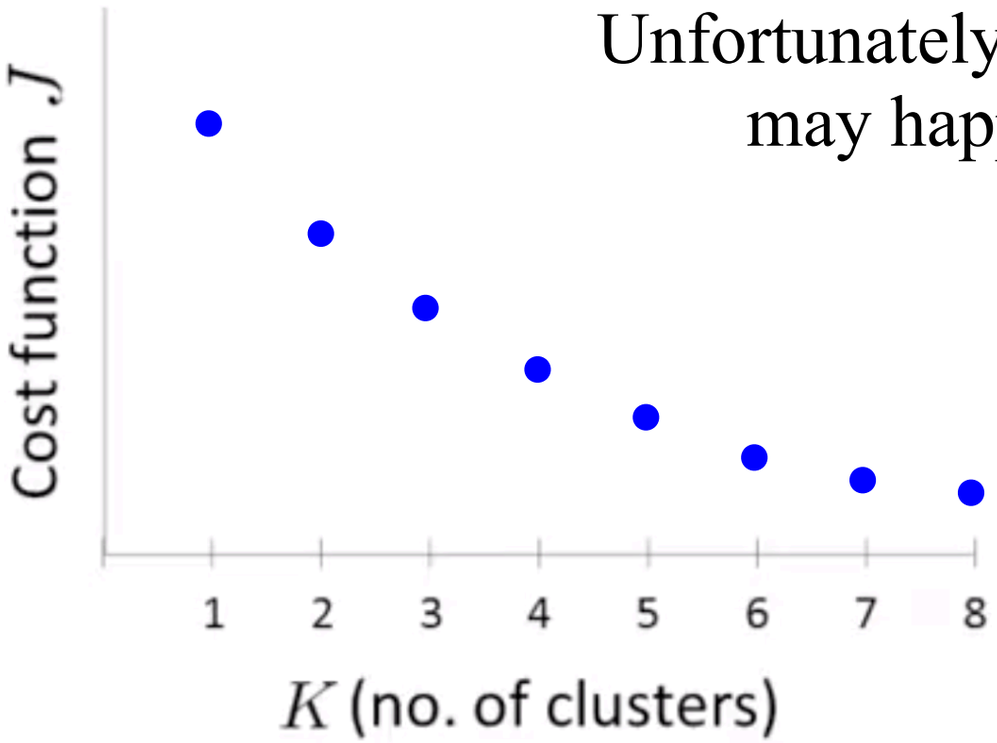




Choosing the value of K

Elbow method: Varying K, starting from 1, 2, …. Then, pick the elbow for K





Unfortunately, this may happen !

กรณีทีไม่มี Elbow ก็ต้องอิงตามฝั่งธุรกิจ

Question

Suppose you run K-means using $K = 3$ and $K = 5$.
You find that the cost function J is much higher for $K = 5$ than for $K = 3$.
What can you conclude?

(i) This is mathematically impossible.
There must be a bug in the code.

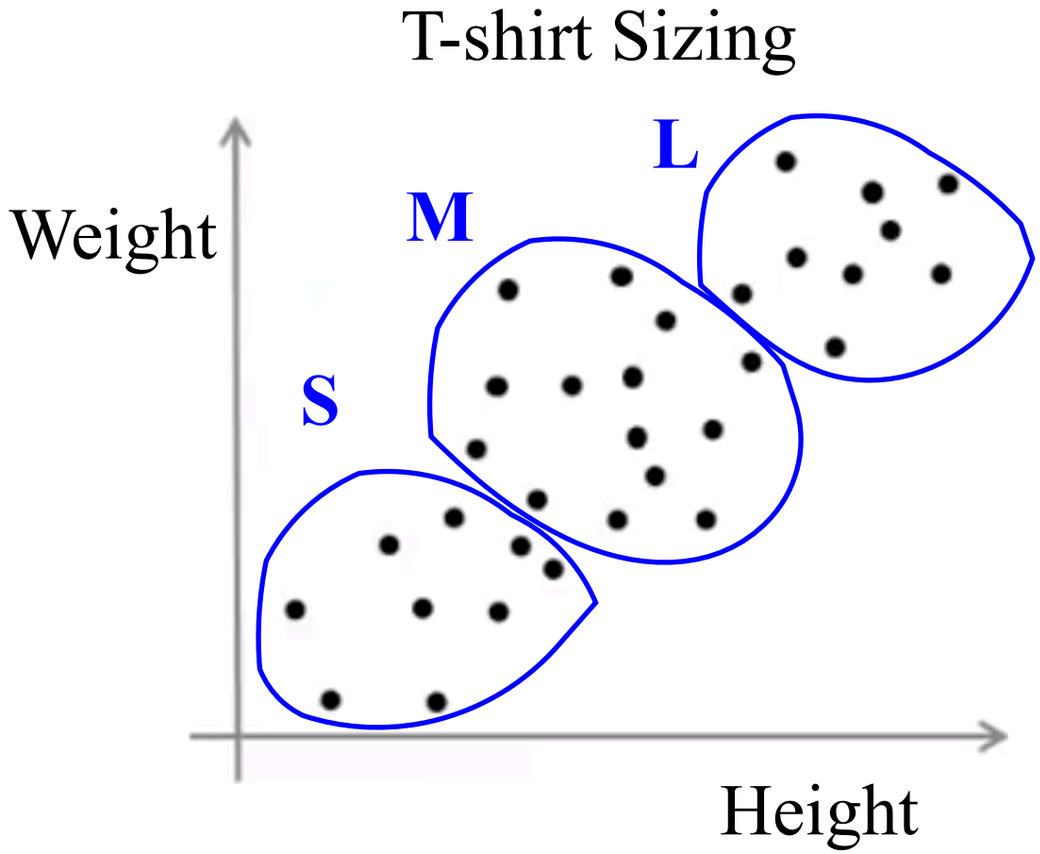
(ii) The correct number of clusters is $K = 3$.

(iii) In the run with $K = 5$, K-means got stuck in a bad local minima.
You should try re-running K-means with multiple random initialization.

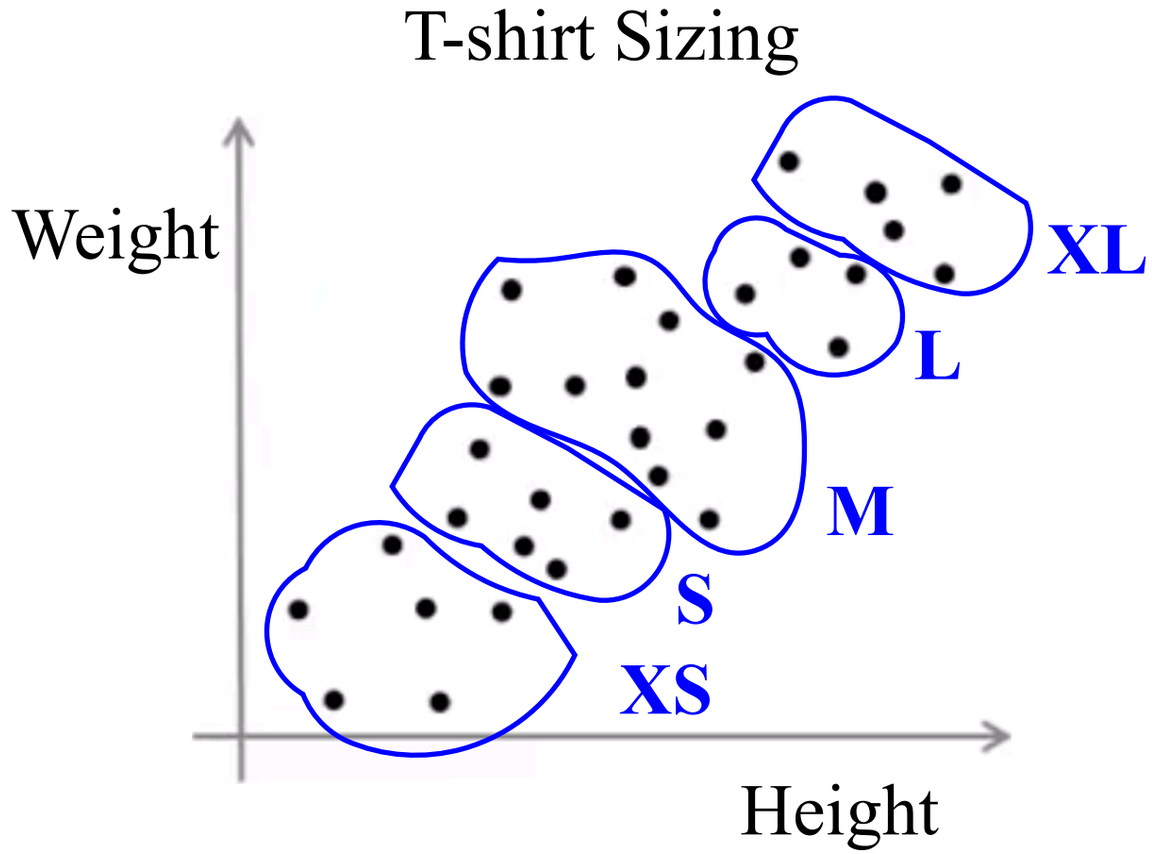
(iv) In the run with $K = 3$, k-means got lucky. You should try re-running k-means with $K = 3$ and different random initializations until it performs no better with $K = 5$.

Choosing the Value of K

Sometimes, you are running K-means to get clusters to use for some later/ downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose *e.g.*



$K = 3$



$K = 5$

Summary

- Often, the number of K is chosen by hands.
- One way to do so is to use Elbow method.
  But, it may not always work well.
- A more appropriate way is ask:
  "for what purposes, you are running k-means?".
  Then, choose K that suitably answer that question.