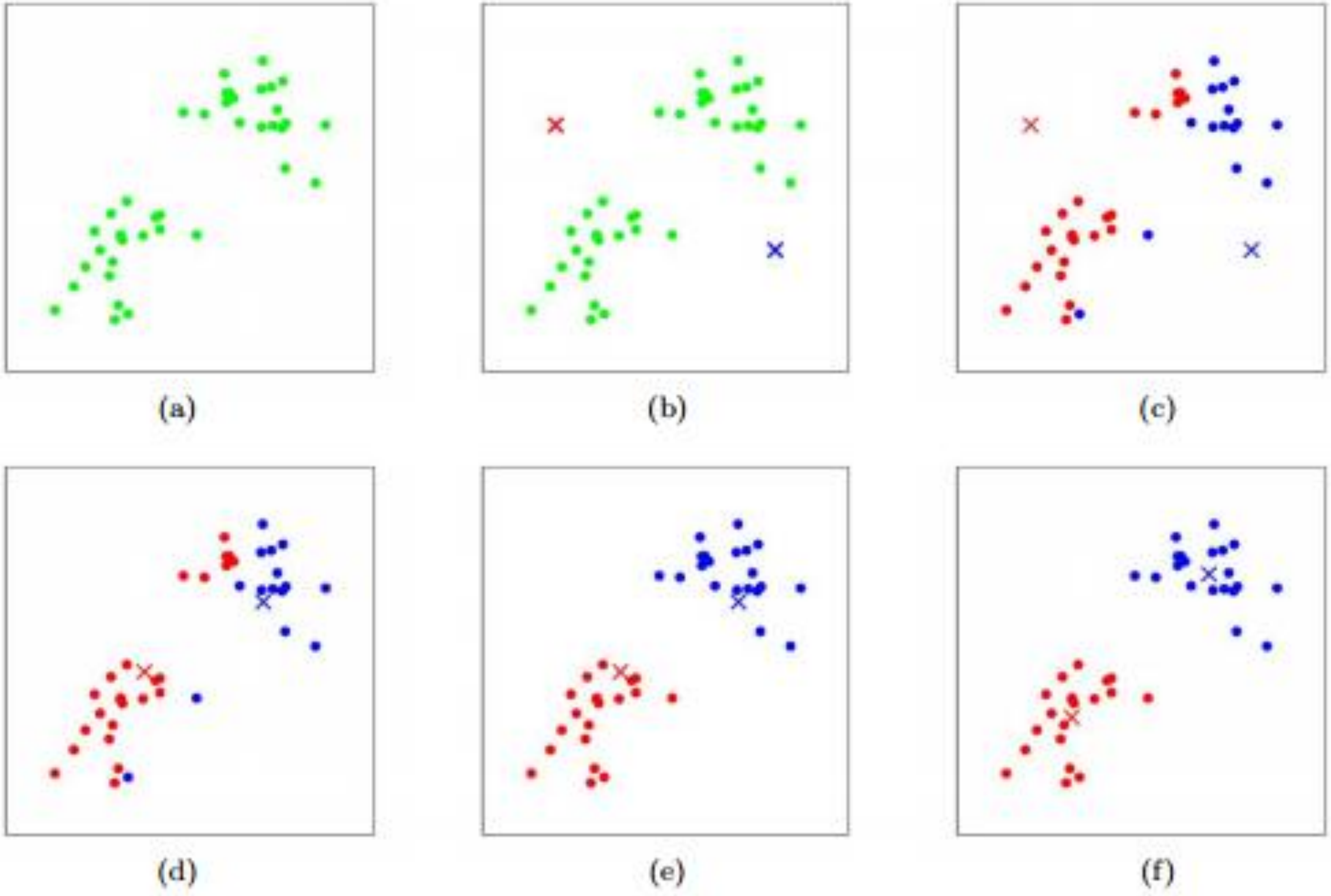
50.007 Machine Learning

K-Means & K-Medoids

Yixiao Wang

Assistant Professor, ISTD/DAI, SUTD

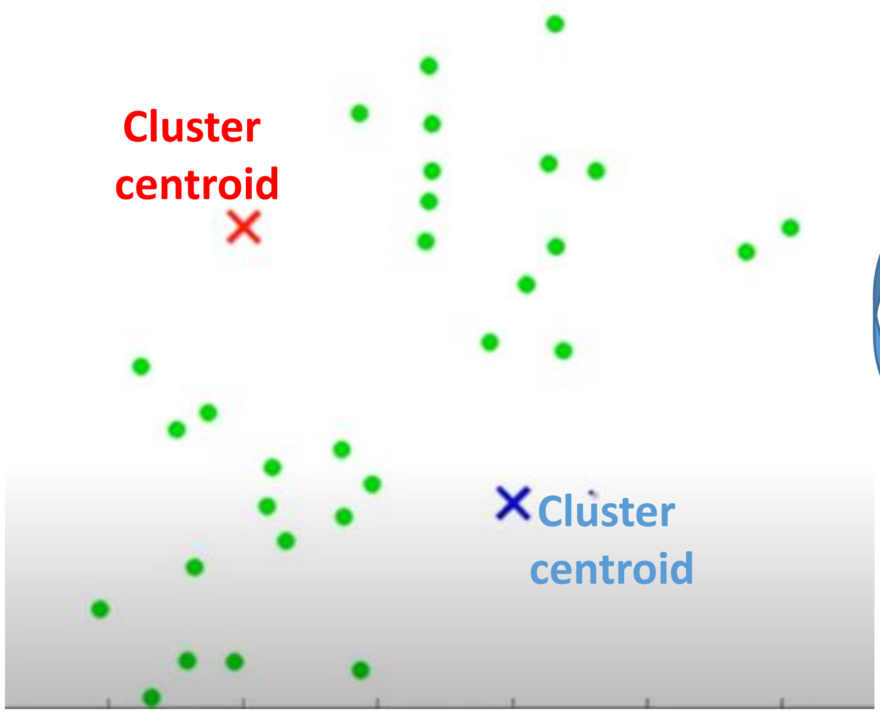
K-Means finds the best centroids by alternating between (1) assigning data points to clusters based on the current centroids (2) choosing centroids (points which are the center of a cluster) based on the current assignment of data points to clusters.



$K = 2$

K-Means (illustration)

- We have 2 cluster centroids, because we would like to group the data into 2 clusters.



K-Means is an iterative algorithm and it does 2 things:

1. **Cluster assignment step:** algorithm will go through each of the examples (green dots) and depending on whether it is closer to red cluster centroid, or blue; algorithm will assign each of the data points in blue or red cluster.

2. **Move centroid step:** calculate the mean of the new clusters, and move the cluster centroids accordingly.

K-Means Algorithm

Input: unlabeled data, K (number of clusters)

Random mean values as centroids

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

Repeat {

for i = 1 to m **For all the data points**

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

Cluster assignment step: assign all the data points to the clusters

for k = 1 to K

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

Move centroids step: recalculate the mean of the cluster

$c^{(i)}$ is the cluster index for each data point i

$c^{(i)}$ can have values from 1 to K.

K-Means Algorithm

For every i, set

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

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

Repeat {

Cluster assignment step:

for i = 1 to m

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

Move centroids step:

for k = 1 to K

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

For each j, set

$$\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}.$$

$c^{(i)}$ is the cluster index for each data point i

$c^{(i)}$ can have values from 1 to K.

K-Means – Mathematical Formulation

By now you should all know the basic of K-means, and how to implement it.

Let's study the mathematical Representation.

K-means has an objective function (cost function) that we need to minimize.

K-Means - Objective Function Optimization

Notations:

K = total number of clusters.

$c^{(i)}$ = cluster index of $x^{(i)}$. Note that $c^{(i)}$ can take values from 1 to K.

μ_k = centroid of cluster k

$\mu_{c^{(i)}}$ = centroid of cluster $c^{(i)}$, cluster that $x^{(i)}$ is assigned.

Example:

Let's assume $x^{(1)}$ belongs to cluster 6, then $c^{(1)}$ will be equal to 6 and $\mu_{c^{(1)}}$ will be μ_6.

K-Means - Objective Function Optimization

Notations:

K = total number of clusters.

$c^{(i)}$ = cluster index of $x^{(i)}$. Note that $c^{(i)}$ can take values from 1 to K.

μ_k = centroid of cluster k

$\mu_{c^{(i)}}$ = centroid of cluster $c^{(i)}$.

Optimization objective:

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

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

K-Means - Objective Function Optimization

Notations:

K = total number of clusters.

$c^{(i)}$ = cluster index of $x^{(i)}$. Note that $c^{(i)}$ can take values from 1 to K.

μ_k = centroid of cluster k

$\mu_{c^{(i)}}$ = centroid of cluster $c^{(i)}$.

We are trying to find the **cluster assignments** and **centroids** that minimize this cost function.

Optimization objective:

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

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

K-Means - Objective Function vs Algorithm

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

Repeat {

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

for i = 1 to m

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

for k = 1 to K

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

Cluster assignment step: we assign each point to clusters (centroids do not change!)

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

Move centroid step: we calculate the new centroids (assignments do not change!)

K-Means – Random Initialization

We talk about k-means a lot; especially the algorithm and objective function. But we never discussed about how to randomly initialize k cluster centroids.

How to initialize K-means?

A popular way is to randomly pick **K training examples**, and set $\mu_1, \ldots, \mu_K$ equal to these k training examples.

K-Means – Random Initialization

We talk about k-means a lot; especially the algorithm and objective function. But we never discussed about how to randomly initialize k-means.

How to initialize K-means?

A popular way is to randomly pick **K training examples**, and set $\mu_1, \ldots, \mu_K$ equal to these k training examples.

Sometimes this can be a poor choice. It is possible that points tend to group too densely in some areas, and thus initializing K-means with randomly chosen patches leads to a large number of centroids starting close together.

K-Means Algorithm

Input: unlabeled data, K (number of clusters)

Our focus!!

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

Repeat {

for i = 1 to m **For all the data points**

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

for k = 1 to K

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

$c^{(i)}$ is the cluster index for each data point i, cluster centroid that is closest to data point i

$c^{(i)}$ can have values from 1 to K.

K-Means – Random Initialization

An example:

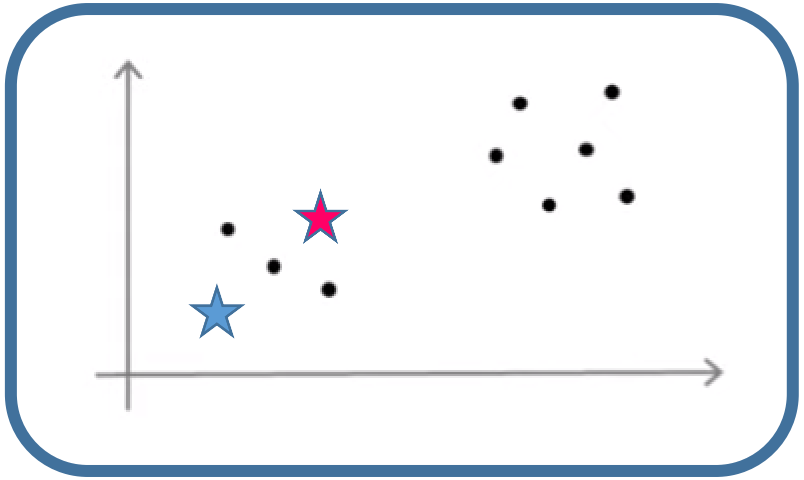
Your data

K-Means – Random Initialization

An example:

Your data

Random initialization 1

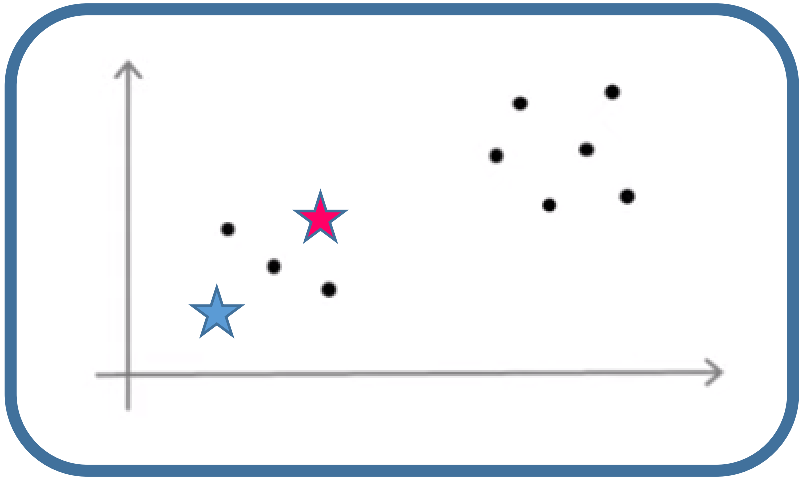


K-Means – Random Initialization

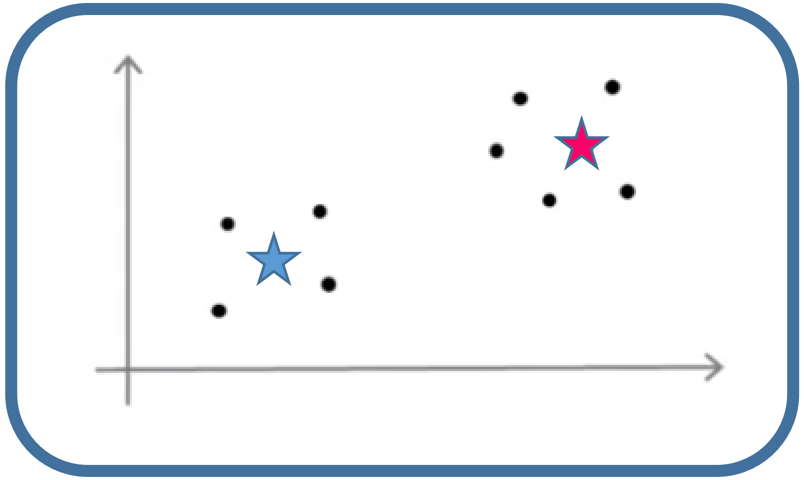
An example:

Your data

Random initialization 1



Random initialization 2

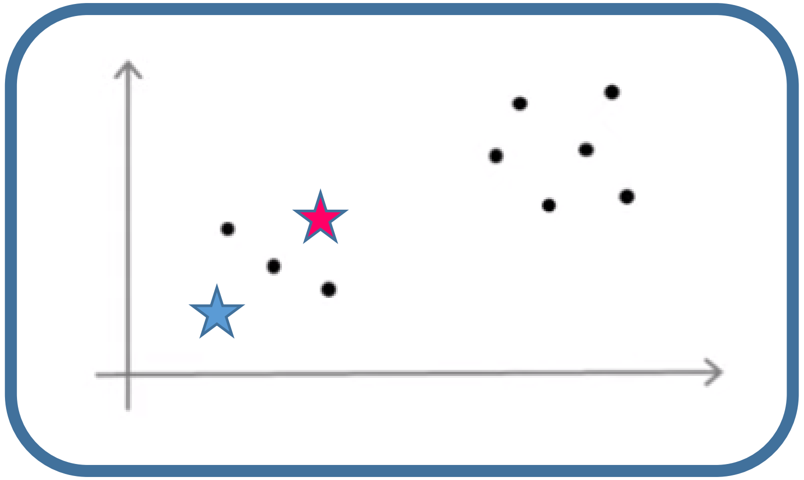


K-Means – Random Initialization

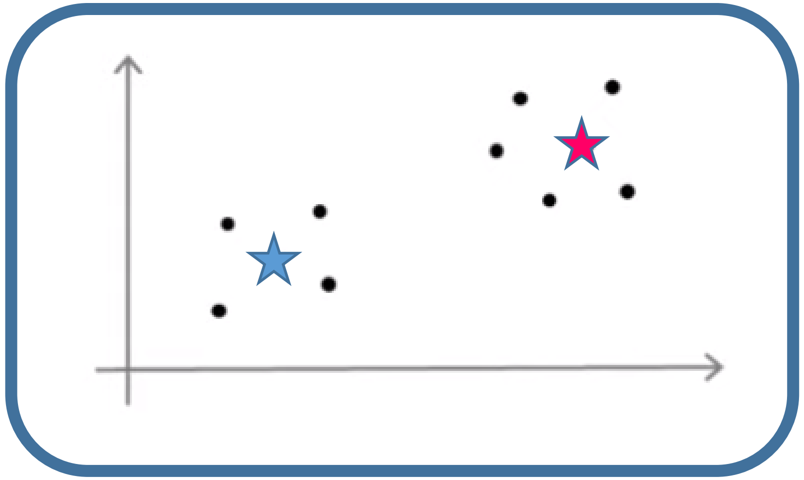
An example:

Your data

Random initialization 1



Random initialization 2



- This is the recommended way if you are using k-means.
- K-means can end up converging to different solutions depending on the random initialization.
- If you're interested to find the best clustering with k-means, try multiple initializations, and run k-means lots of times (for example 100 times). You can use the best solution.

K-Means – Random Initialization

Random initialization

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 (distortion)

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

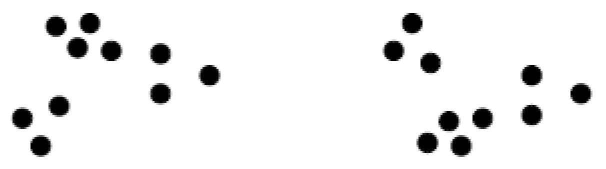
}

Pick clustering that gave lowest cost J

If you're interested to find the best clustering with k-means, try multiple initializations, and run k-means lots of times (for example 100 times). You can use the best solution

K-Means – Number of Clusters

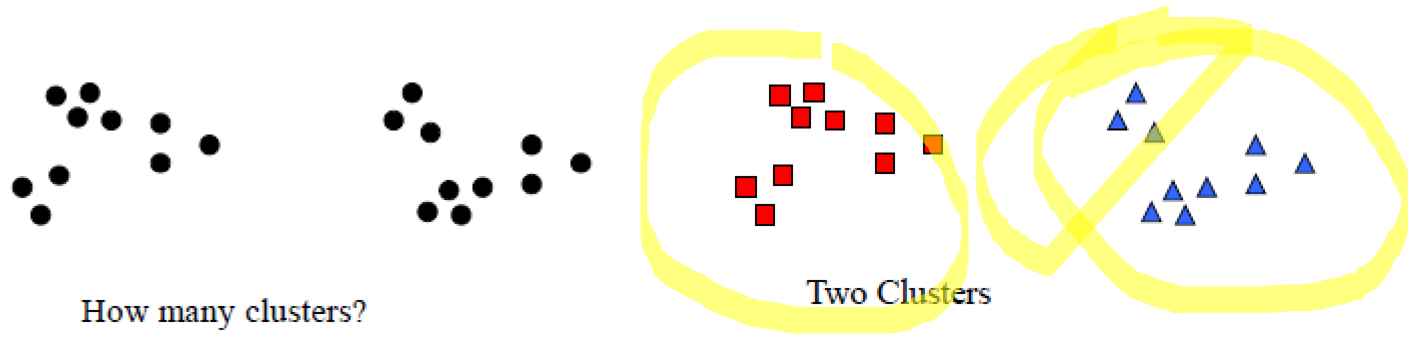
- Notion of a cluster can be ambiguous



How many clusters?

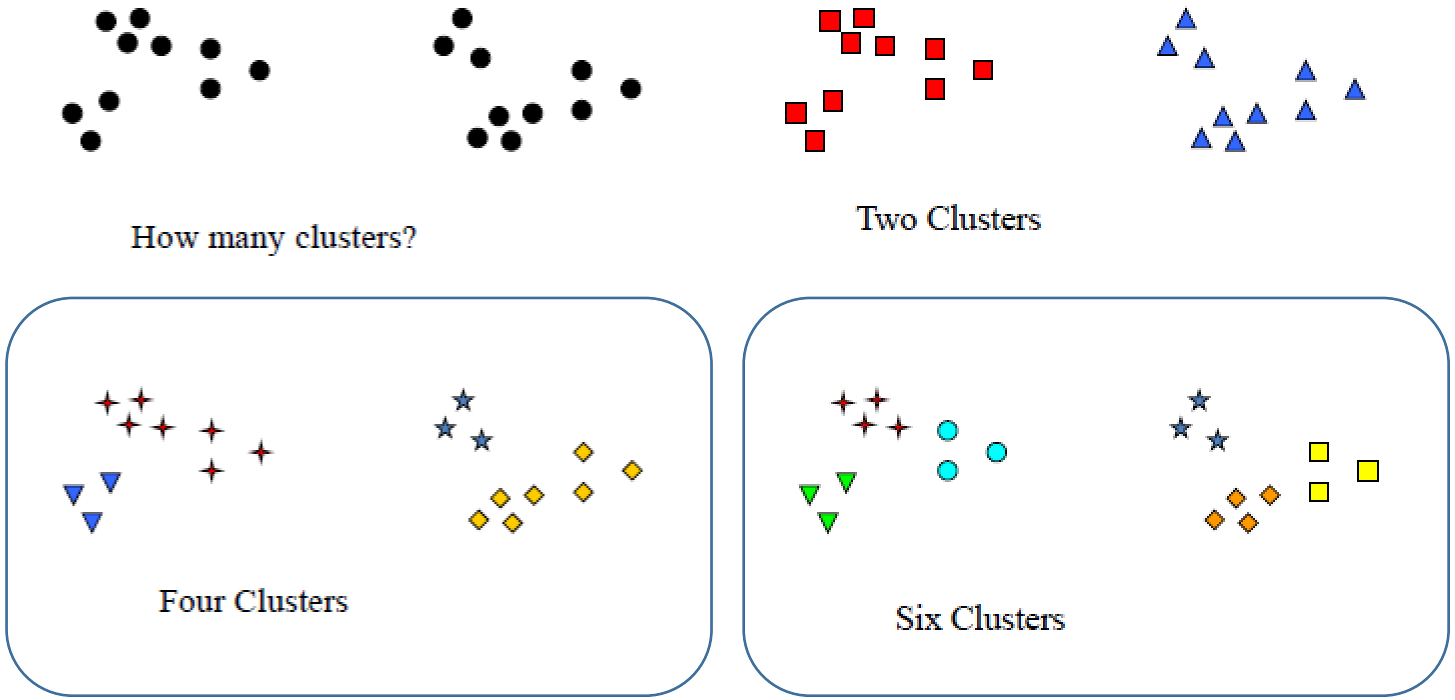
K-Means – Number of Clusters

- Notion of a cluster can be ambiguous



K-Means – Number of Clusters

- Notion of a cluster can be ambiguous

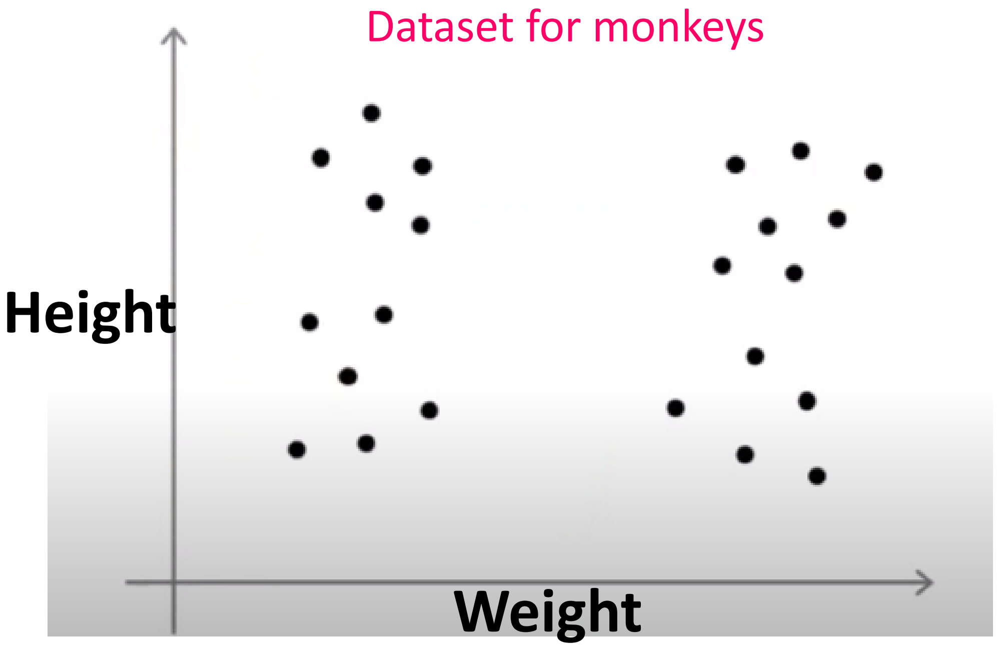


K-Means – Number of Clusters

- How to choose the number of clusters, K?
  - Not an easy question.
- The most popular way is to choose manually (by hand), or look at the data visually.

For example, when you look at this data,

How many clusters you see?

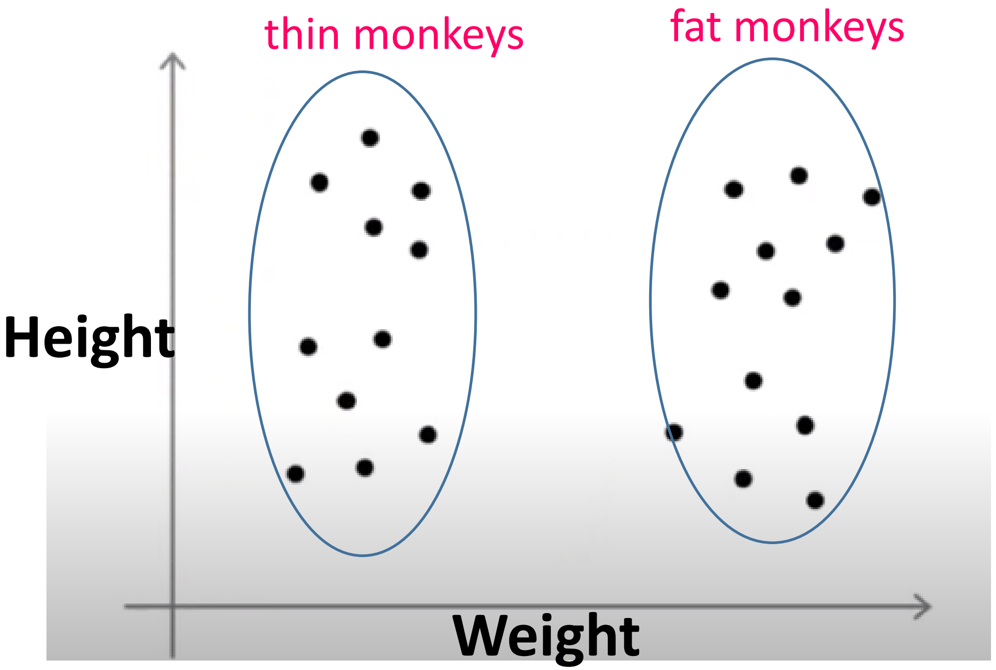


K-Means – Number of Clusters

- How to choose the number of clusters, K?
  - Not an easy question.
- The most popular way is to choose manually (by hand), or look at the data visually.

For example, when you look at this data,

How many clusters you see?



2 clusters?

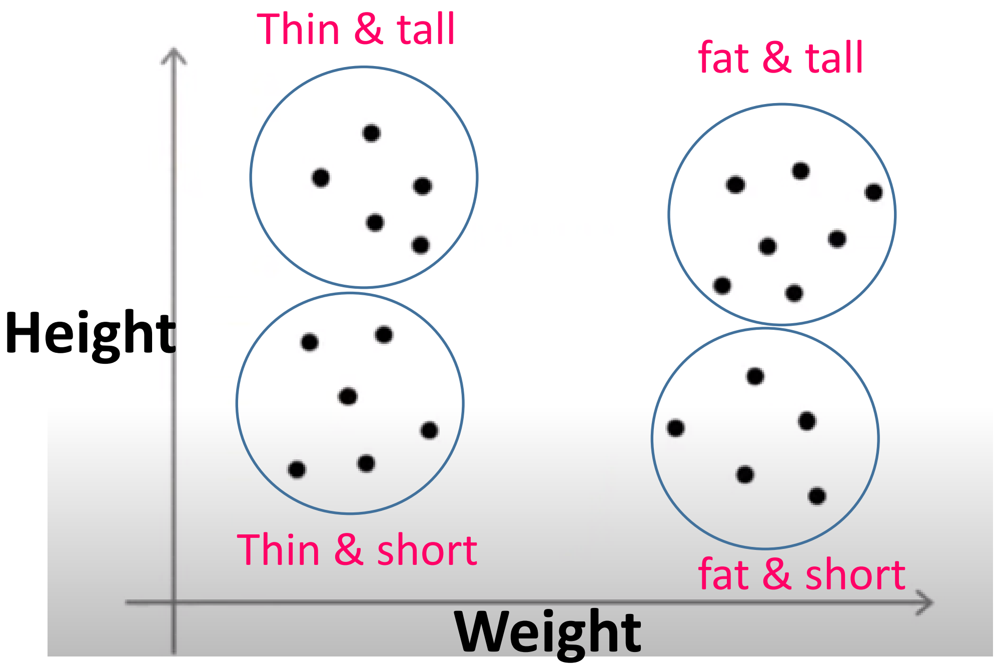
$$K = 2$$

K-Means – Number of Clusters

- How to choose the number of clusters, K?
  - Not an easy question.
- The most popular way is to choose manually (by hand), or look at the data visually.

For example, when you look at this data,

How many clusters you see?



4 clusters?

$$\boldsymbol{K = 4}$$

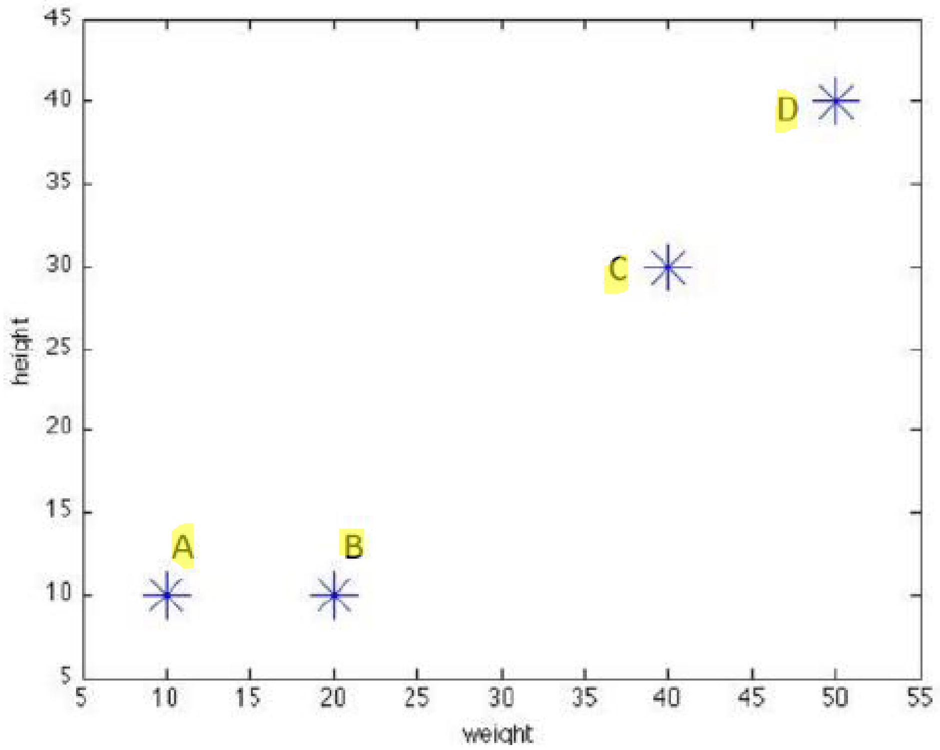
There is no clear answer for the number of clusters. This is unsupervised learning, so we do not have labels.

K-means

Let’s solve some examples together… ☺

Example - 1

Suppose we have 4 boxes with different heights and weights, and we want to divide them into 2 clusters. Please note that each box represents one point with two attributes (X, Y):



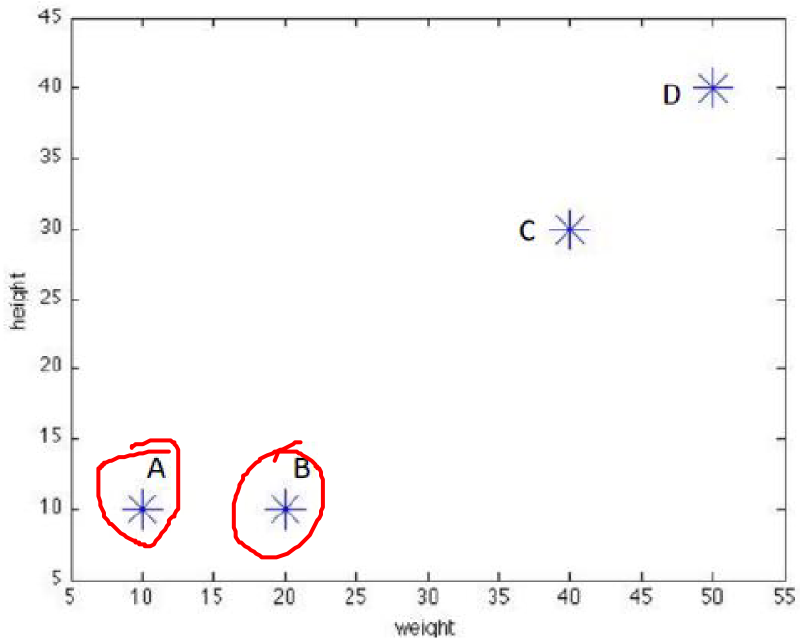
$A = (10,10)$
$B = (20,10)$
$C = (40,30)$
$D = (50,40)$

Example - 1

$k=2$

Initial centers: suppose we choose points A and B as the initial centers, so $c1 = (10, 10)$ and $c2 = (20, 10)$.

Object centre distance: Calculate the Euclidean distance between cluster centers and the objects.



$A = (10,10)$
$B = (20,10)$
$C = (40,30)$
$D = (50,40)$

Example - 1

$c1 \rightarrow \boldsymbol{A} = (\mathbf{10}, \mathbf{10})$

$c2 \rightarrow \boldsymbol{B} = (\mathbf{20}, \mathbf{10})$

$C = (40,30)$

$D = (50,40)$

We can obtain the following distance table:

	A	B	C	D
A Centre 1	$\boxed{0}$	10	36.06	50
B Centre 2	10	0	28.28	43.43

Object clustering: We assign each object to one of the clusters based on the minimum distance from the centre.

	A	B	C	D
Centre 1	$\boxed{1}$	0	0	0
Centre 2	0	1	1	1

It is not over ☺ we should determine centers based on the group membership, we compute the new centers.

Example - 1

$c1 \rightarrow A = (\mathbf{10}, \mathbf{10})$
$c2 \rightarrow B = (\mathbf{20}, \mathbf{10})$
$C = (40{,}30)$
$D = (50{,}40)$

It is not over ☺ we should determine centers based on the group membership, we compute the new centers.

$$c1 = (10{,}10)$$

$$c2 = \left(\frac{20+40+50}{3}, \frac{10+30+40}{3}\right) = (36.7, 26.7)$$

Re-compute the object center distances: We compute the distances of each data point from the new centers:

	A	B	C	D
$c1 = (10, 10)$ Centre 1	0	10	36.06	50
$c2 = (36.7, 26.7)$ Centre 2	31.4	23.6	4.7	18.9

	A	B	C	D
Centre 1	1	1	0	0
Centre 2	0	0	1	1

Example - 1

Determine the new centers:

$$c_1 = \left(\frac{10+20}{2}, \frac{10+10}{2}\right) = (15, 10)$$
$$c_2 = \left(\frac{40+50}{2}, \frac{30+40}{2}\right) = (45, 35)$$

Recompute the object-centers distances

	A	B	C	D
(15, 10) Centre 1	5	5	32	46.1
(45, 35) Centre 2	43	35.4	7.1	7.1

Object clustering

	A	B	C	D
Centre 1	1	1	0	0
Centre 2	0	0	1	1

The cluster membership did not change from one iteration to another. So the k-means computation terminates.

K-medoids

From K-means to K-medoids

- I would like to introduce you another interesting k partitioning clustering method called the k-medoids clustering.
- K-medoids is actually very similar to K-means algorithm…

Why k-medoids?

From K-means to K-medoids

- I would like to introduce you another interesting k partitioning clustering method called the k-medoids clustering.
- K-medoids is actually very similar to K-means algorithm…

Why k-medoids?

- In real-life applications, K-means algorithm is actually sensitive to the outliers due to the mean calculation. Very basically speaking, just imagine that you are trying to calculate the mean of the salaries in company X. If there is one **very** high salary (outlier), the mean may increase a lot.

Instead of taking the mean value as the centroid, we can use the most centrally located object in the cluster (or we called medoids).

From K-means to K-medoids Algorithm

- We had previously defined the cost function for the **k-means algorithm** in terms of squared Euclidean distance of each point $x^{(i)}$ to the closest cluster representative.
- We showed that, for any given cluster, the best representative to choose is the **mean of the points in the cluster**.
- The resulting cluster mean typically does not correspond to any point in the original dataset.

From K-means to K-medoids Algorithm

- We had previously defined the cost function for the **k-means algorithm** in terms of squared Euclidean distance of each point $x^{(i)}$ to the closest cluster representative.
- We showed that, for any given cluster, the best representative to choose is the **mean of the points in the cluster**.
- The resulting cluster mean typically does not correspond to any point in the original dataset.

The k-medoids algorithm operates exactly like k-means but, instead of choosing the cluster mean as a representative, it chooses one of the original points as a representative. This point is called an exemplar.

K-medoids Applications

Selecting *exemplars* rather than *cluster means* as representatives can be important in applications.
For example, Google News, where a single article is used to represent a news cluster. Blending articles together to evaluate the "mean" would not make sense in this context.

K-medoids Algorithm

Choose K points as the initial representative objects (as initial k-medoids).

Repeat:

1. Assign each point to the cluster with the closest medoid.
2. In each cluster, randomly select a non-representative object.
3. Compute the cost of swabbing the medoid to the selected non-representative object for each cluster.
4. If the cost is lower, then swab the medoid to the non-representative object.

K-medoids Algorithm

The algorithm:

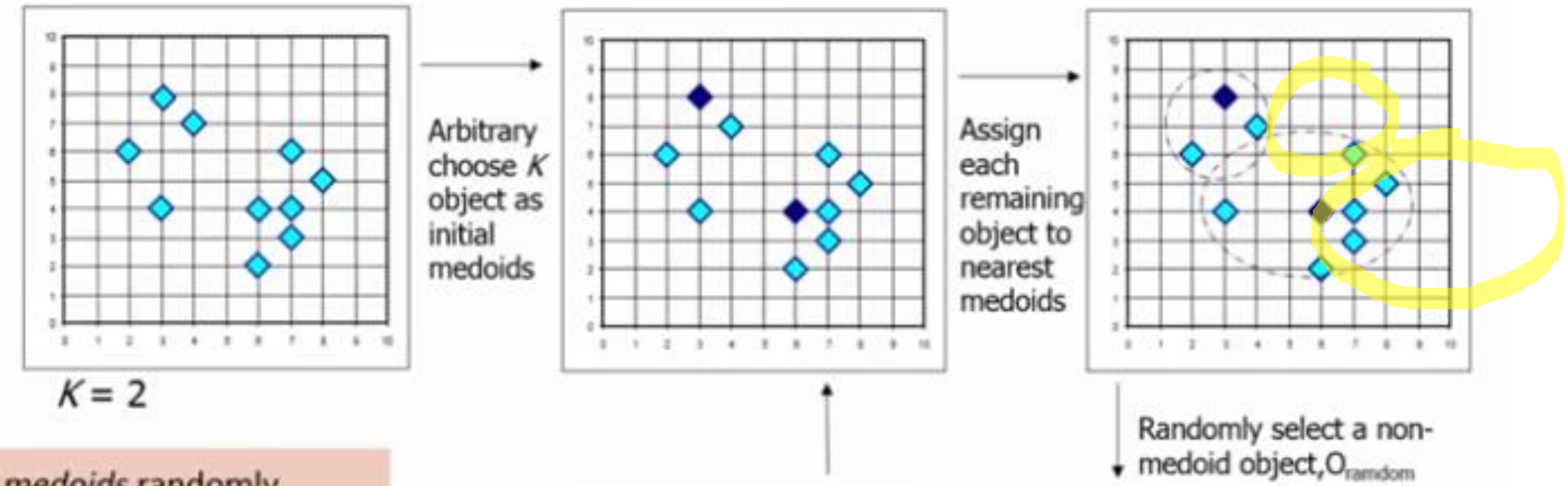
1. Initialize exemplars: $\{z^{(1)}, \ldots, z^{(k)}\} \subseteq \{x^{(1)}, \ldots, x^{(n)}\}$ (exemplars are k points from the original dataset)

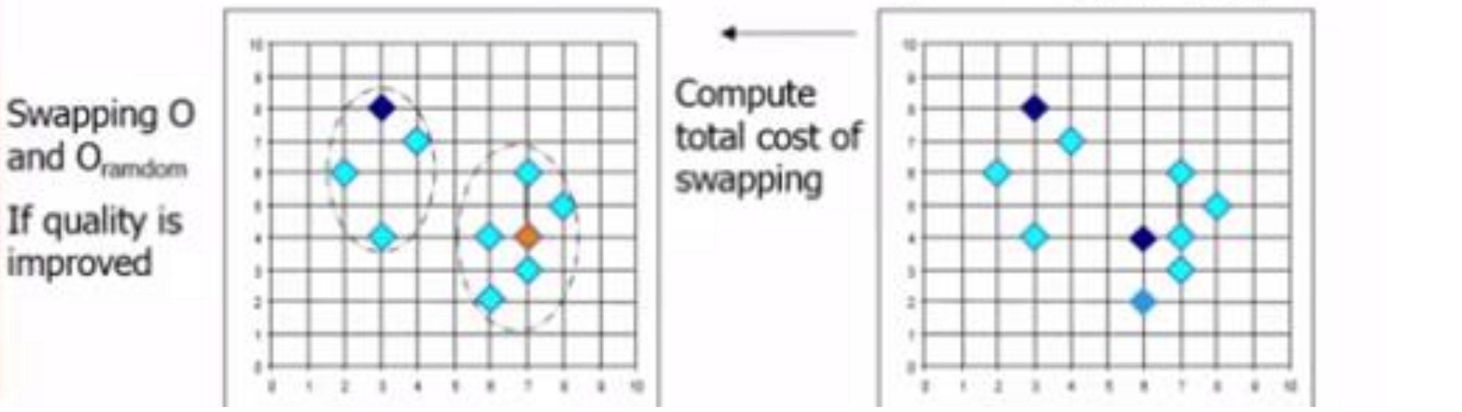
2. Repeat until there is no further change in cost:

   (a) for each j : $C^j = \{i : x^{(i)}$'s closest exemplar is $z^{(j)}\}$

   (b) for each j : set $z^{(j)}$ to be the point in C^j that minimizes $\sum_{i \in C^j} d(x^{(i)}, z^{(j)})$

PAM: A Typical *K-Medoids* Algorithm





Select initial *K medoids* randomly

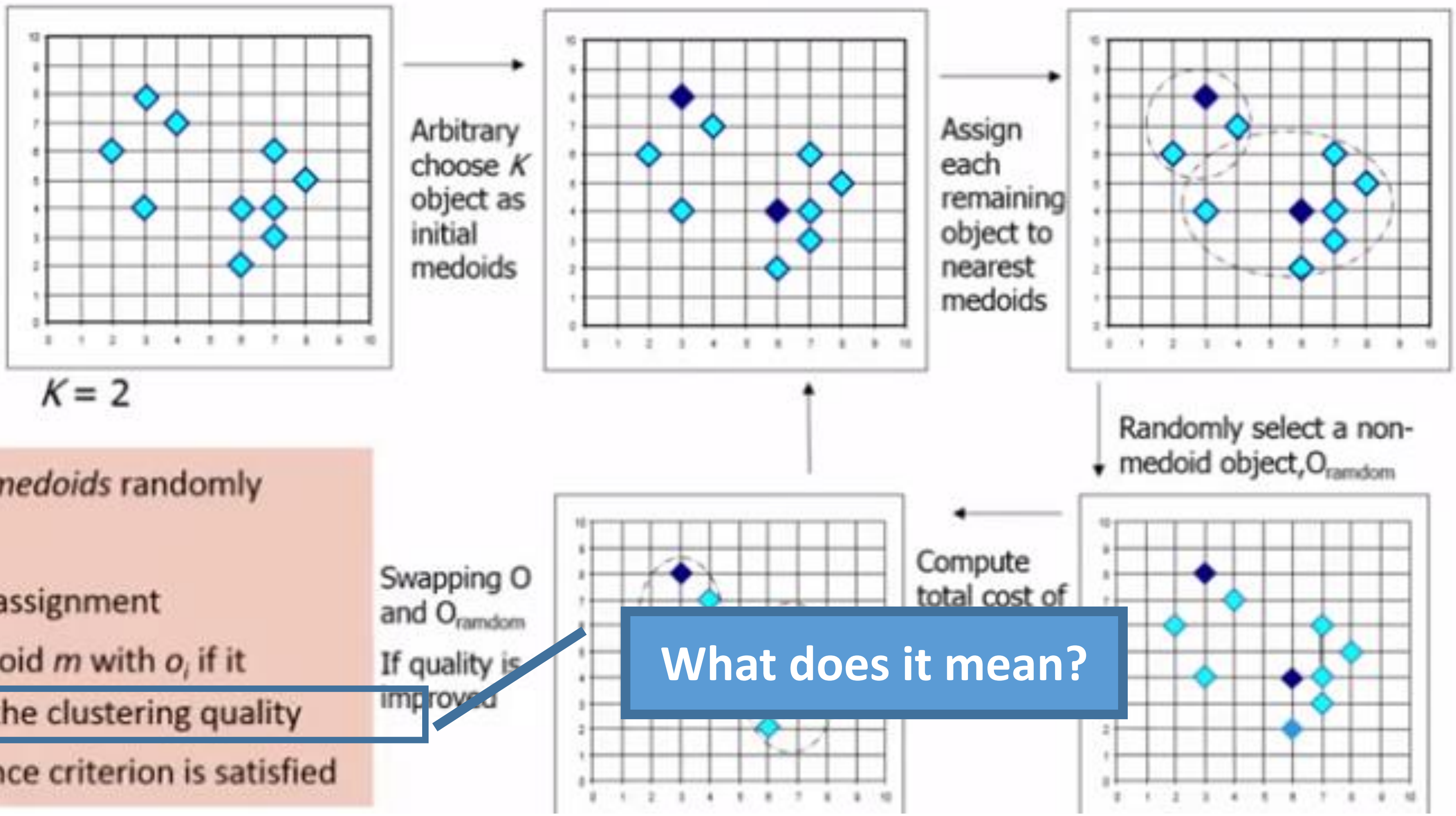
Repeat

Object re-assignment

Swap medoid *m* with o_i if it improves the clustering quality

Until convergence criterion is satisfied

PAM: A Typical *K-Medoids* Algorithm



Select initial *K medoids* randomly

Repeat

Object re-assignment

Swap medoid *m* with o_i if it improves the clustering quality

Until convergence criterion is satisfied

What does it mean?

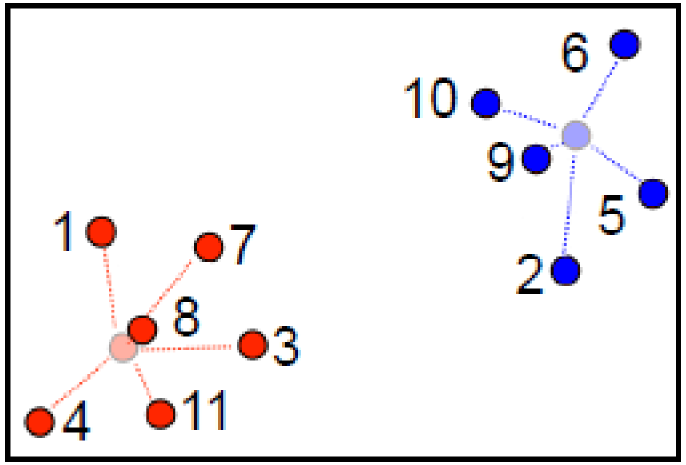
How to Specify a Cluster

Using a representative:

1) K-means: A point in the center of cluster (mean, centroid)

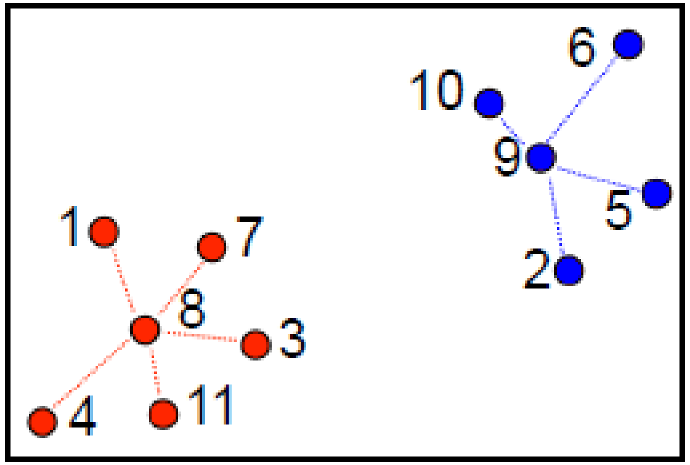
2) K-medoids: A point in the training data (exemplar)

Each point $x^{(i)}$ will be assigned the closest representative.



centroid

K-means



exemplar

K-medoids

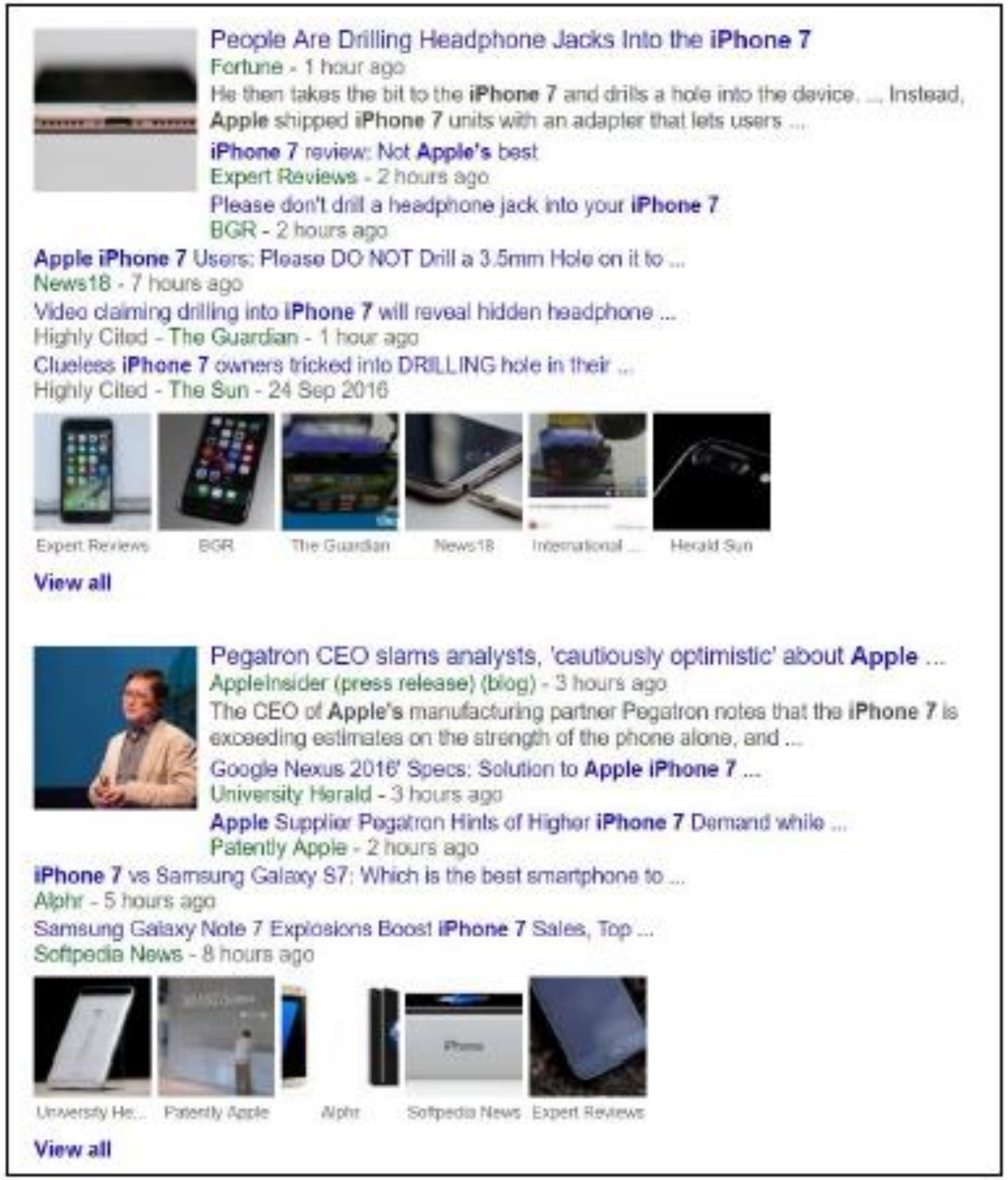
K-medoids – Example 1

Use exemplars
instead of centroids.

e.g. Google News.

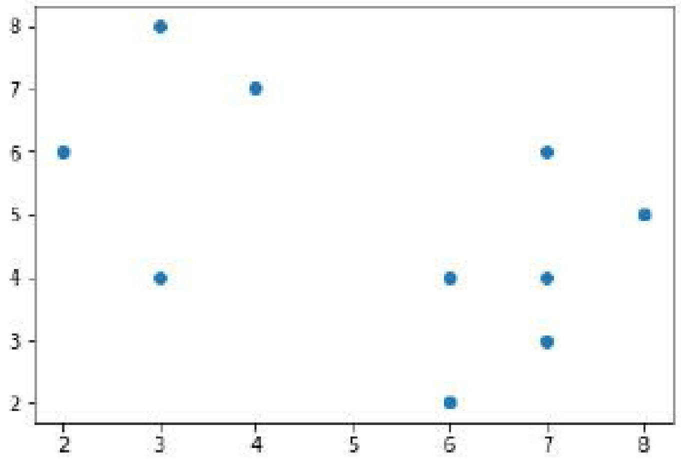
Repeat until convergence:

- Find best clusters given exemplars
- Find best exemplars given clusters



K-Medoids – Example 2

- Consider the following set of points

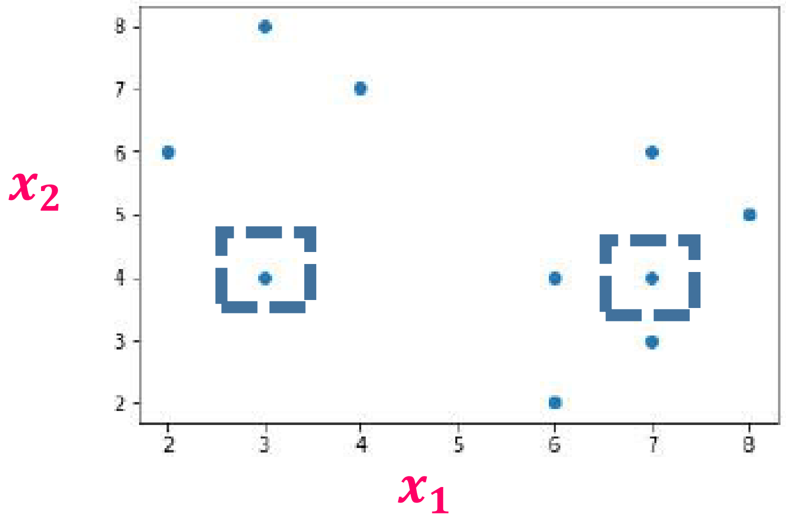


	x_1	x_2
$x^{(1)}$	2	6
$x^{(2)}$	3	4
$x^{(3)}$	3	8
$x^{(4)}$	4	7
$x^{(5)}$	6	2
$x^{(6)}$	6	4
$x^{(7)}$	7	3
$x^{(8)}$	7	4
$x^{(9)}$	8	5
$x^{(10)}$	7	6

- We will consider L1 distance $d(x^{(i)}, z^{(j)}) = |x^{(i)} - z^{(j)}|$

K-Medoids – Example 2

- Consider the following set of points



	x_1	x_2	
$x^{(1)}$	2	6	
$x^{(2)}$	3	4	Medoid 1
$x^{(3)}$	3	8	
$x^{(4)}$	4	7	
$x^{(5)}$	6	2	
$x^{(6)}$	6	4	
$x^{(7)}$	7	3	
$x^{(8)}$	7	4	Medoid 2
$x^{(9)}$	8	5	
$x^{(10)}$	7	6	

- We will consider L1 distance $d(x^{(i)}, z^{(j)}) = |x^{(i)} - z^{(j)}|$

K-Medoids – Example 2

▪

- Let the randomly selected 2 medoids be

$$z^{(1)} = (3,4)$$
$$z^{(2)} = (7,4)$$

- The cost of each non-medoid point with the medoids is calculated and tabulated:

Data object		Distance to	
i	$x^{(i)}$	$z^{(1)} = (3,4)$	$z^{(2)} = (7,4)$
1	(2, 6)	3	7
2	(3, 4)	0	4
3	(3, 8)	4	8
4	(4, 7)	4	6
5	(6, 2)	5	3
6	(6, 4)	3	1
7	(7, 3)	5	1
8	(7, 4)	4	0
9	(8, 5)	6	2
10	(7, 6)	6	2
Cost			

K-Medoids – Example 2

-

- Let the randomly selected 2 medoids be

$$z^{(1)} = (3,4)$$
$$z^{(2)} = (7,4)$$

- The total cost of this clustering is:

Cluster 1: (3+0+4+4) = 11

Cluster 2: (3+1+1+0+2+2) = 9

Total: 20

Data object		Distance to	
i	$x^{(i)}$	$z^{(1)} = (3,4)$	$z^{(2)} = (7,4)$
1	(2, 6)	3	7
2	(3, 4)	0	4
3	(3, 8)	4	8
4	(4, 7)	4	6
5	(6, 2)	5	3
6	(6, 4)	3	1
7	(7, 3)	5	1
8	(7, 4)	4	0
9	(8, 5)	6	2
10	(7, 6)	6	2
Cost		11	9

Cluster 1

Cluster 2

Cost of cluster 1

Cost of cluster 2

K-Medoids – Example 2

- Updating Z_2 with a non-medoid point, O' in Cl

$$z^{(1)} = (3,4)$$
$$O' = (7,3)$$

- The total cost of this clustering is:

Cluster 2: (2+2+0+1+3+3) = 11

Distance from data points to medoid 1

i	$z^{(1)}$		$x^{(i)}$		dist
1	3	4	2	6	**3**
3	3	4	3	8	**4**
4	3	4	4	7	**4**
5	3	4	6	2	5
6	3	4	6	4	3
8	3	4	7	4	4
9	3	4	8	5	6
10	3	4	7	6	6

Cluster 1

i	O'		$x^{(i)}$		dist
1	7	3	2	6	8
3	7	3	3	8	9
4	7	3	4	7	7
5	7	3	6	2	**2**
6	7	3	6	4	**2**
8	7	3	7	4	**1**
9	7	3	8	5	**3**
10	7	3	7	6	**3**

Cluster 2

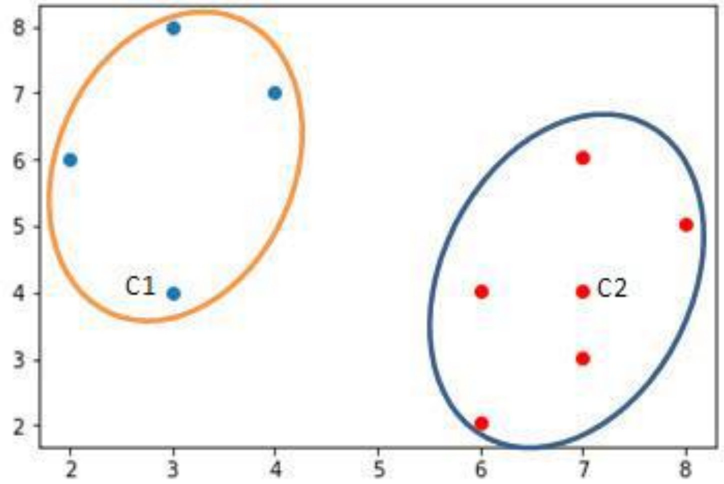
Distance from data points to the new medoid 2

K-Medoids – Example 2

- The total cost of this clustering is:

Cluster 2: (2+2+0+1+3+3) = 11

11 > 9 (our original cost before updating Z_2), we will not update (7, 3) to be our medoid. We will keep randomly try non-medoids in **Cluster 2** until we find the lowest cost non-medoid, and then make it our medoid.



After having the new medoid, we will do the clustering step again with **all the points (in both C1 and C2).**

k-medoids vs k-means

- The K-medoids algorithm shares the properties of K-means that we discussed:
    - each iteration decreases the cost;
    - the algorithm always converges;
    - different starts gives different final answers;
    - it does not achieve the global minimum.
- K-medoids is computationally harder than K-means:
    - because of step 2: computing the medoid is harder than computing the average
- Remember, K-medoids has the (potentially important) property that the centers are located among the data points themselves.

k-medoids vs k-means

☹ A problem with the K-means clustering is that the final centroids are not interpretable or in other words, centroids are not the actual point but the mean of points present in that cluster.

☺ The idea of K-medoids clustering is to make the final centroids as actual data-points.

- This result makes the centroids interpretable.

Summary of this lecture

- Unsupervised learning:
  - Goal: Discover hidden structure in data without prior labels or observations of that structure
  - Challenging but necessary • many applications
- Clustering
  - Goal: Segment data points into similar groups
  - many applications
- k-means
  - Simple, popular, canonical approach to clustering
  - Great diversity of applications
  - Drawbacks and opportunities for improvement (Objective, choice of k, initialization)
  - k-medoids

Summary of this lecture

- Unsupervised learning:
  - Goal: Discover hidden structure in data without prior labels or observations of that structure
  - Challenging but necessary • many applications
- Clustering
  - Goal: Segment data points into similar groups
  - many applications
- k-means
  - Simple, popular, canonical approach to clustering

MUST: Please study the lecture notes!!

  - Drawbacks and opportunities for improvement (Objective, choice of k, initialization)

Suggestion: If you're not very clear or want to learn more, please do read:
Clustering and K-means part of "C. Bishop: Pattern Recognition and Machine Learning. Springer, 2006" ***(recommended text book).*** **It will help you to understand the concept.**

Thank you :)

Acknowledgement

- National University of Singapore – Pattern Recognition Module (EE5907R)
- National University of Singapore – Neural Networks Module (EE5904R)
- University of Edinburgh, United Kingdom – Machine Learning And Pattern Recognition (MLPR, INFR11130)
- Stanford University – Machine Learning, Andrew Ng (CS229)