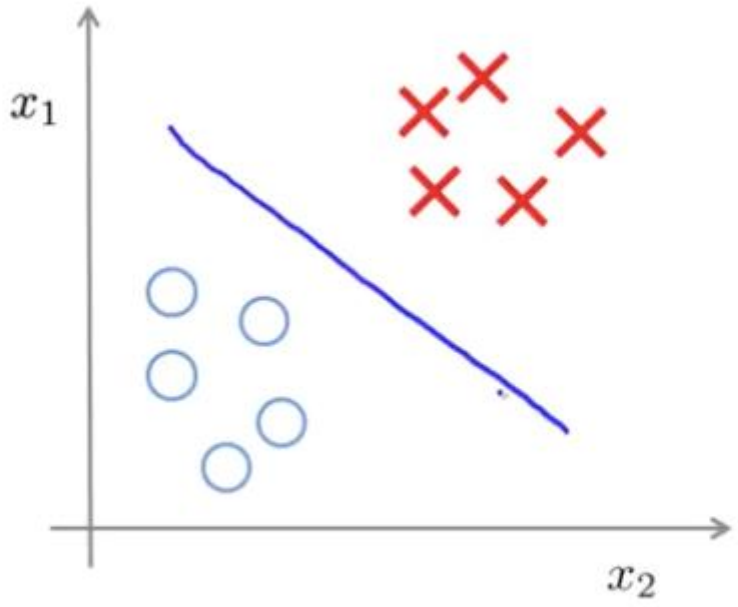
Unsupervised Machine Learning

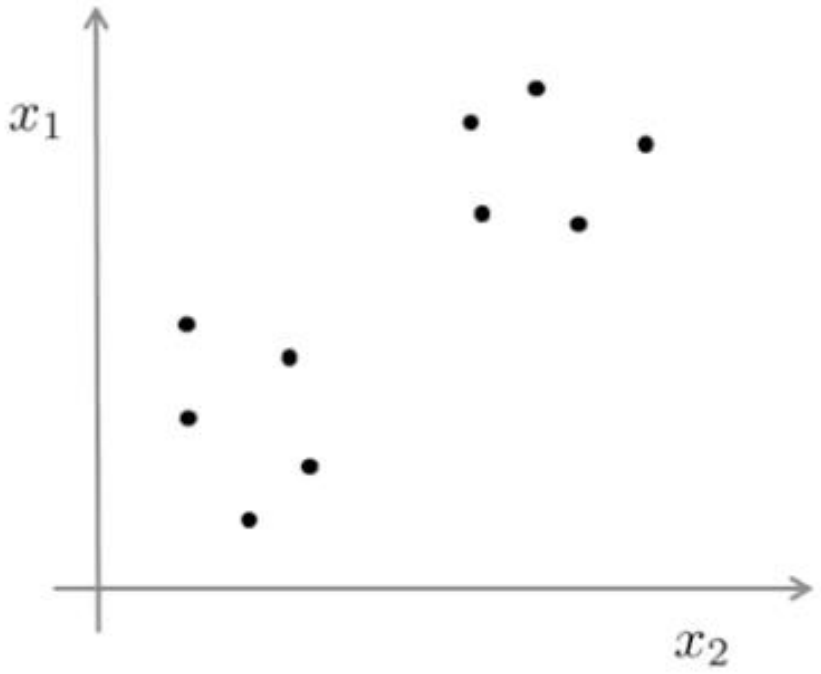
Clustering

Supervised learning



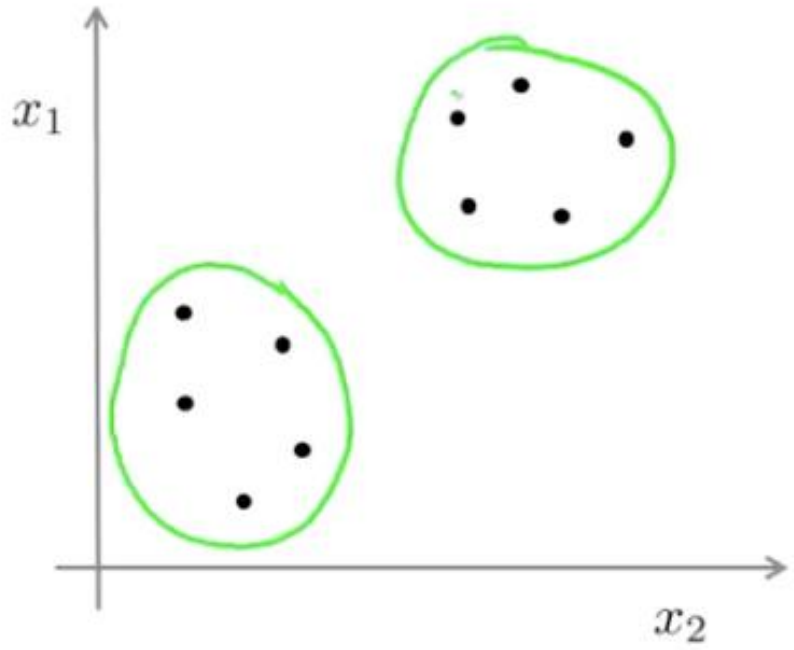
Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), (x^{(3)}, y^{(3)}), \ldots, (x^{(m)}, y^{(m)})\}$

Unsupervised learning



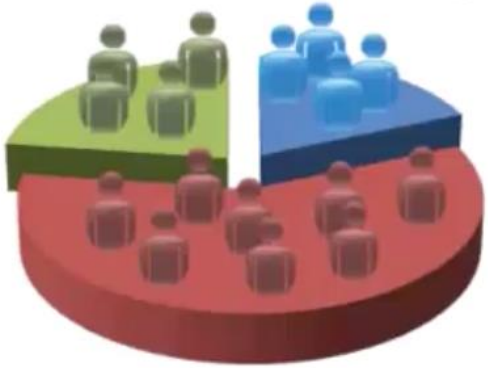
Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

Unsupervised learning

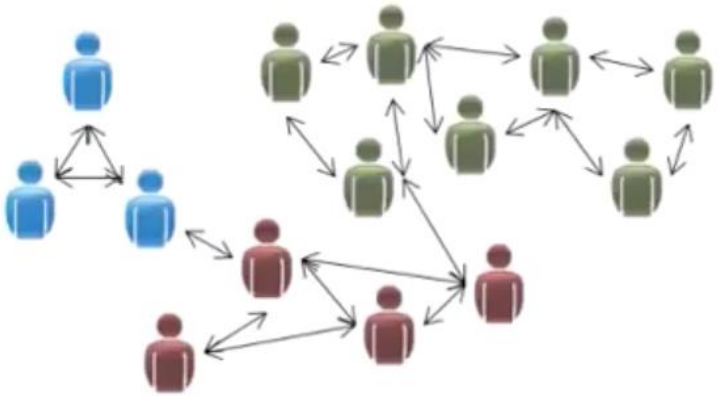


Training set: $\{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}\}$

Applications of clustering



Market segmentation



Social network analysis

Organize computing clusters

Astronomical data analysis

Unsupervised Machine Learning

Unsupervised learning is where you only have input data (X) and no corresponding output variables.- UNLABELED DATASET

The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.

Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behaviour.

- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

- **Dimensionality Reduction**: Data Compression

What Is Cluster Analysis?

- **Cluster analysis** or simply **clustering** is the process of partitioning a set of data objects (or observations) into subsets.
- Each subset is a **cluster**, such that objects in a cluster are similar to one another, yet dissimilar to objects in other clusters
- The set of clusters resulting from a cluster analysis can be referred to as a **clustering**.
- In this context, different clustering methods may generate different clustering on the same dataset.
- The partitioning is not performed by humans, but by the clustering algorithm.
  - Hence, **clustering is useful in that it can lead to the discovery of previously unknown groups within the data.**

The main elements of cluster analysis

1. Data presentation.
2. Choice of objects.
3. Choice of variables.
4. What to cluster: data units or variables.
5. Normalization of variables.
6. Choice of (dis)similarity measures.
7. Choice of clustering criterion (objective function).
8. Choice of missing data strategy.
9. Algorithms and computer implementation (and their reliability, e.g., convergence)
10. Number of clusters.
11. Interpretation of results.

Basic Clustering Methods

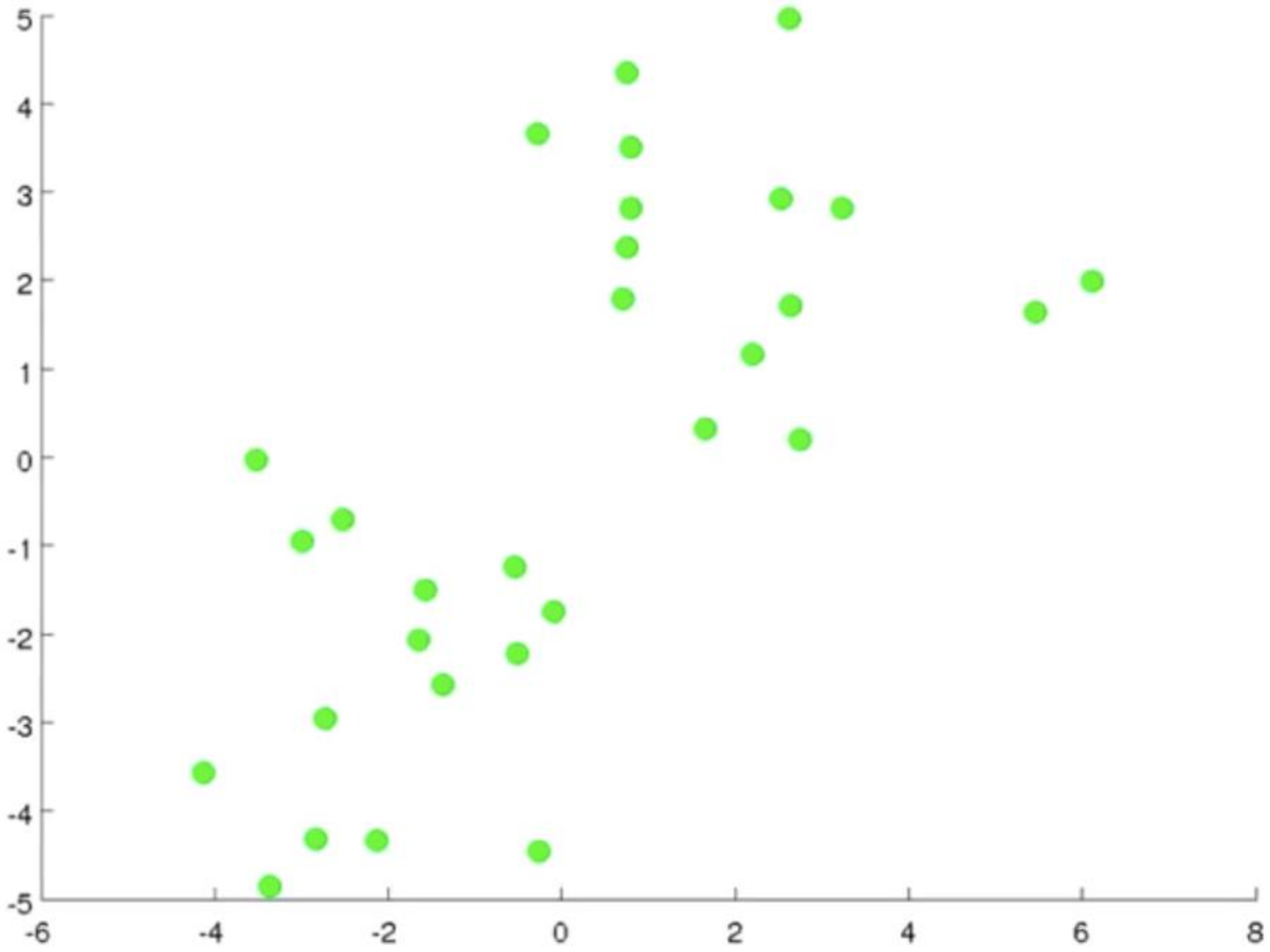
Method	General Characteristics
Partitioning methods	– Find mutually exclusive clusters of spherical shape – Distance-based – May use mean or medoid (etc.) to represent cluster center – Effective for small- to medium-size data sets
Hierarchical methods	– Clustering is a hierarchical decomposition (i.e., multiple levels) – Cannot correct erroneous merges or splits – May incorporate other techniques like microclustering or consider object "linkages"
Density-based methods	– Can find arbitrarily shaped clusters – Clusters are dense regions of objects in space that are separated by low-density regions – Cluster density: Each point must have a minimum number of points within its "neighborhood" – May filter out outliers
Grid-based methods	– Use a multiresolution grid data structure – Fast processing time (typically independent of the number of data objects, yet dependent on grid size)

Partitioning Methods

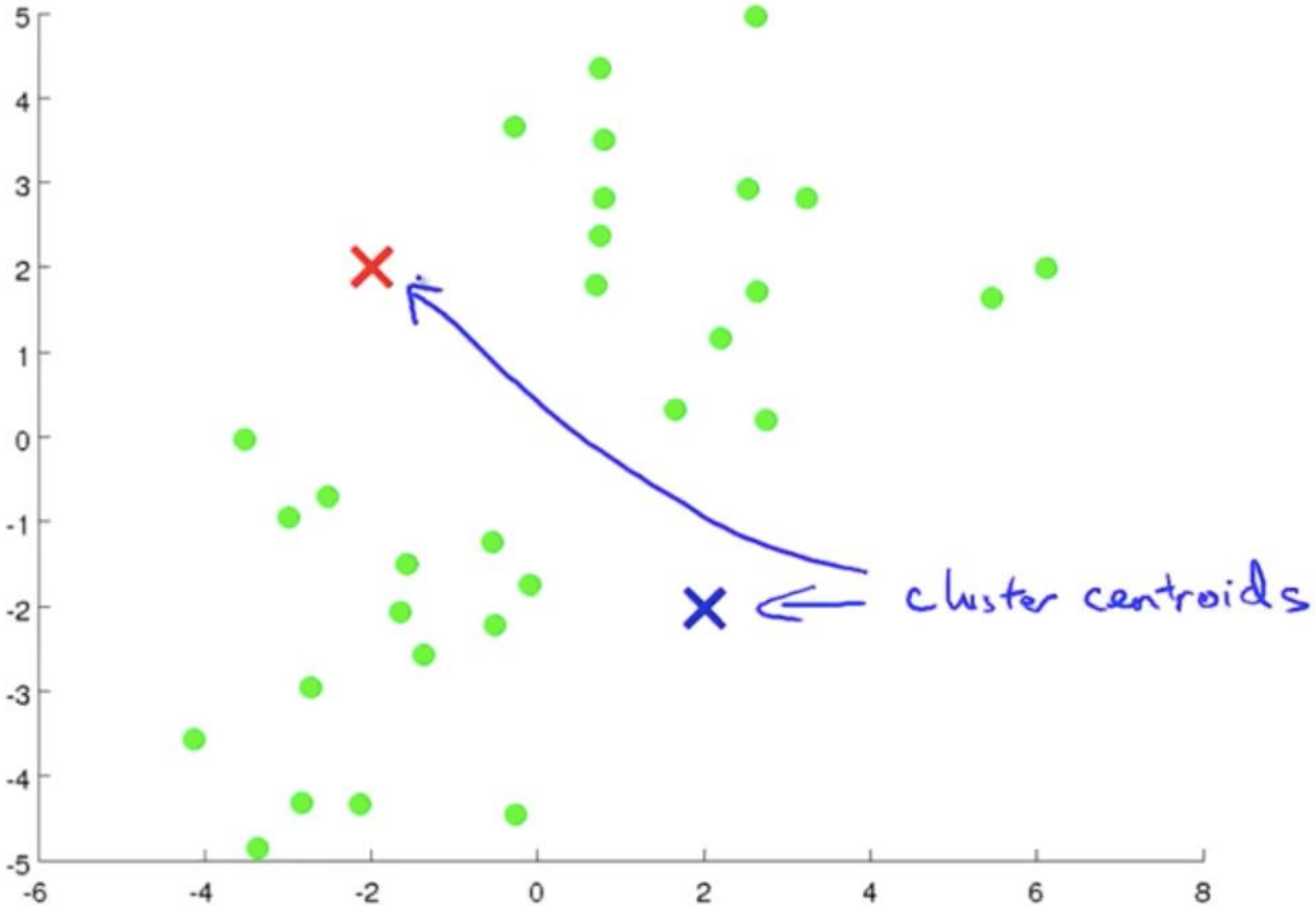
- Given a set of n objects, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$.
- That is, it divides the data into k groups such that each group must contain at least one object.
- The basic partitioning methods typically adopt *exclusive cluster separation*. That is, each object must belong to exactly one group.
- Most partitioning methods are **distance-based**.
- Given k, the number of partitions to construct, a partitioning method creates an initial partitioning.
- It then uses an **iterative relocation technique** that attempts to improve the partitioning by moving objects from one group to another.
- The general criterion of a good partitioning is that objects in the same cluster are "close" or related to each other, whereas objects in different clusters are "far apart" or very different.

Example: ***K-means clustering algorithm***

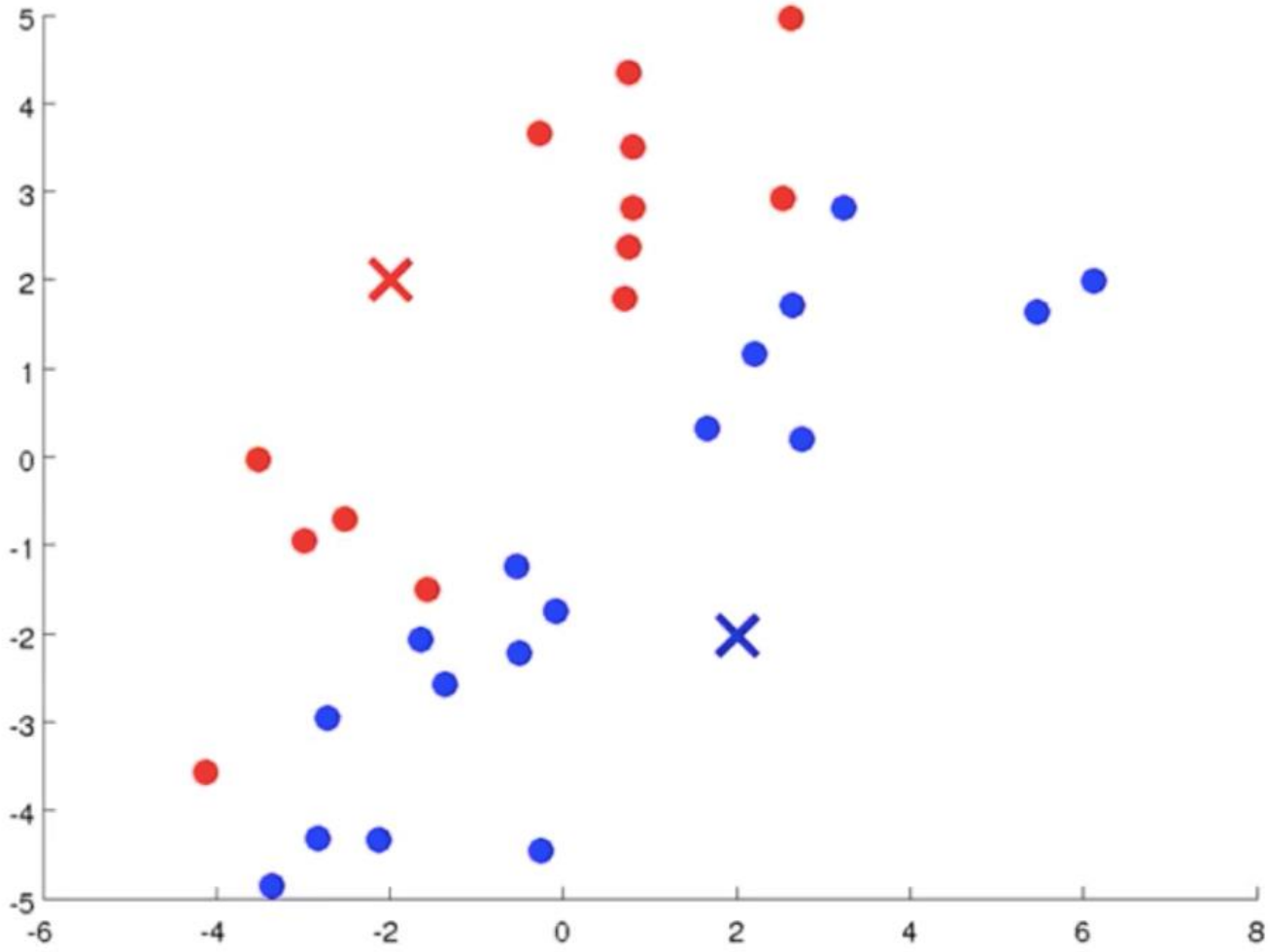
K-means Clustering Algorithm



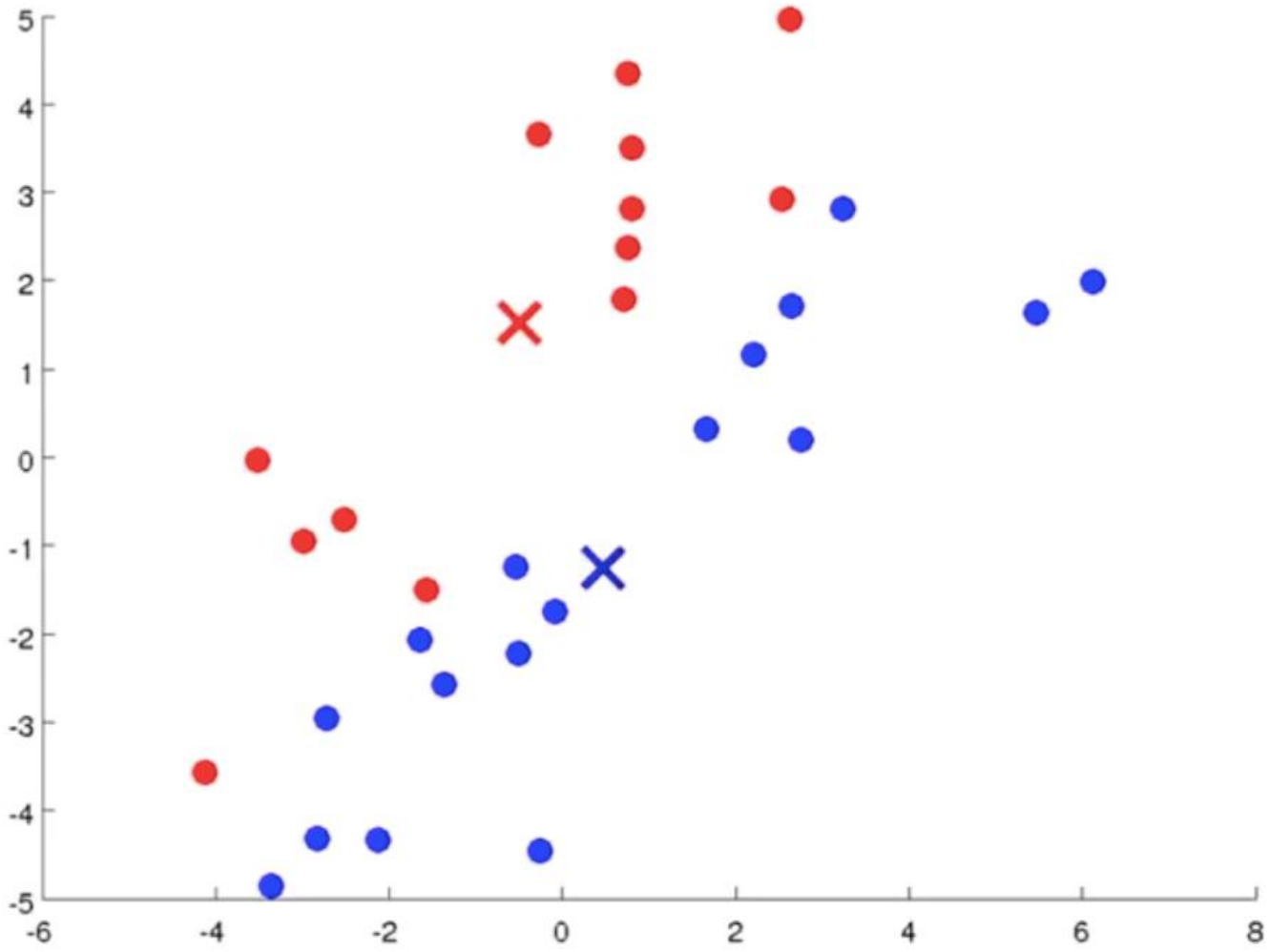
K-means Clustering Algorithm



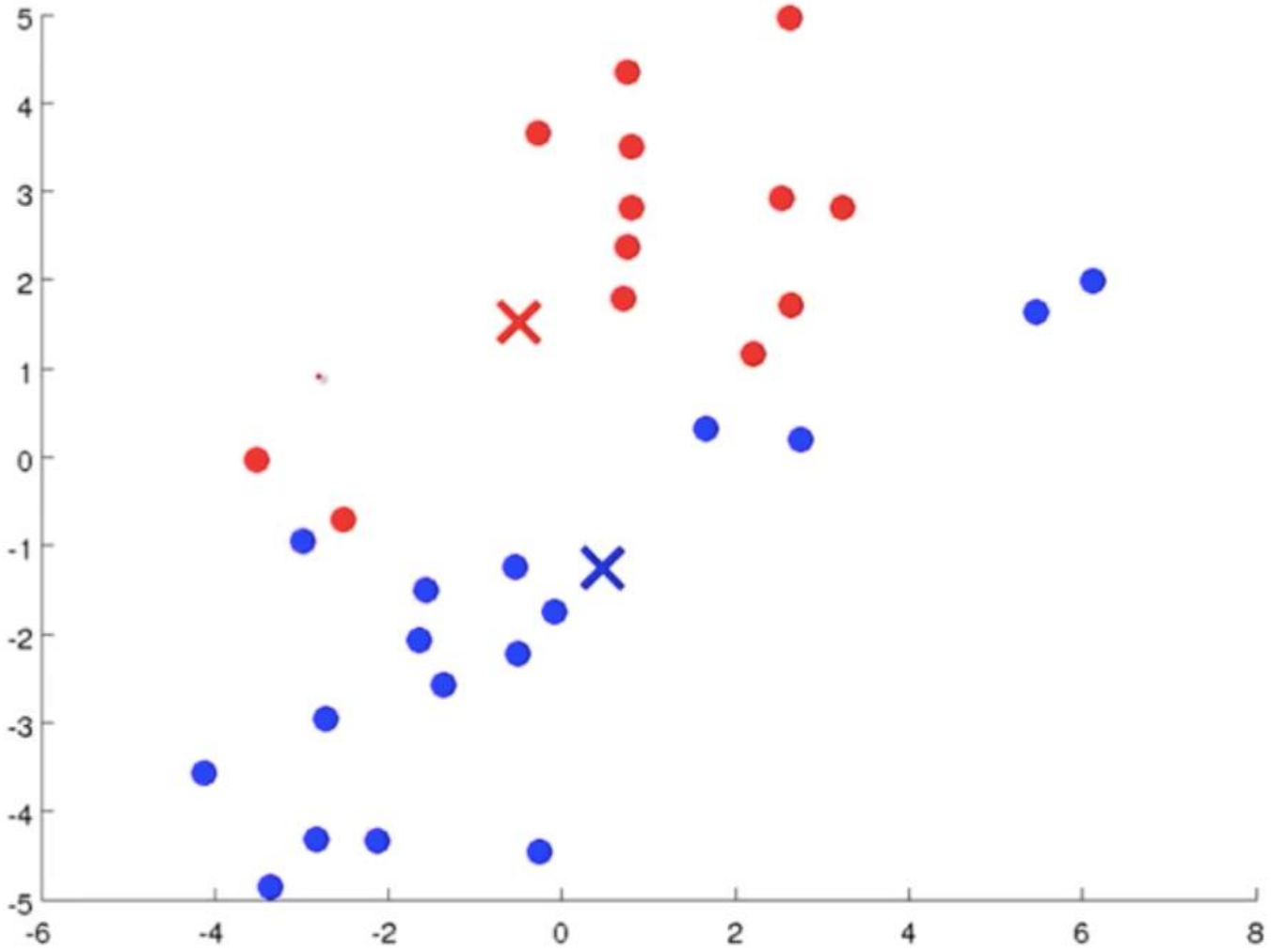
K-means Clustering Algorithm



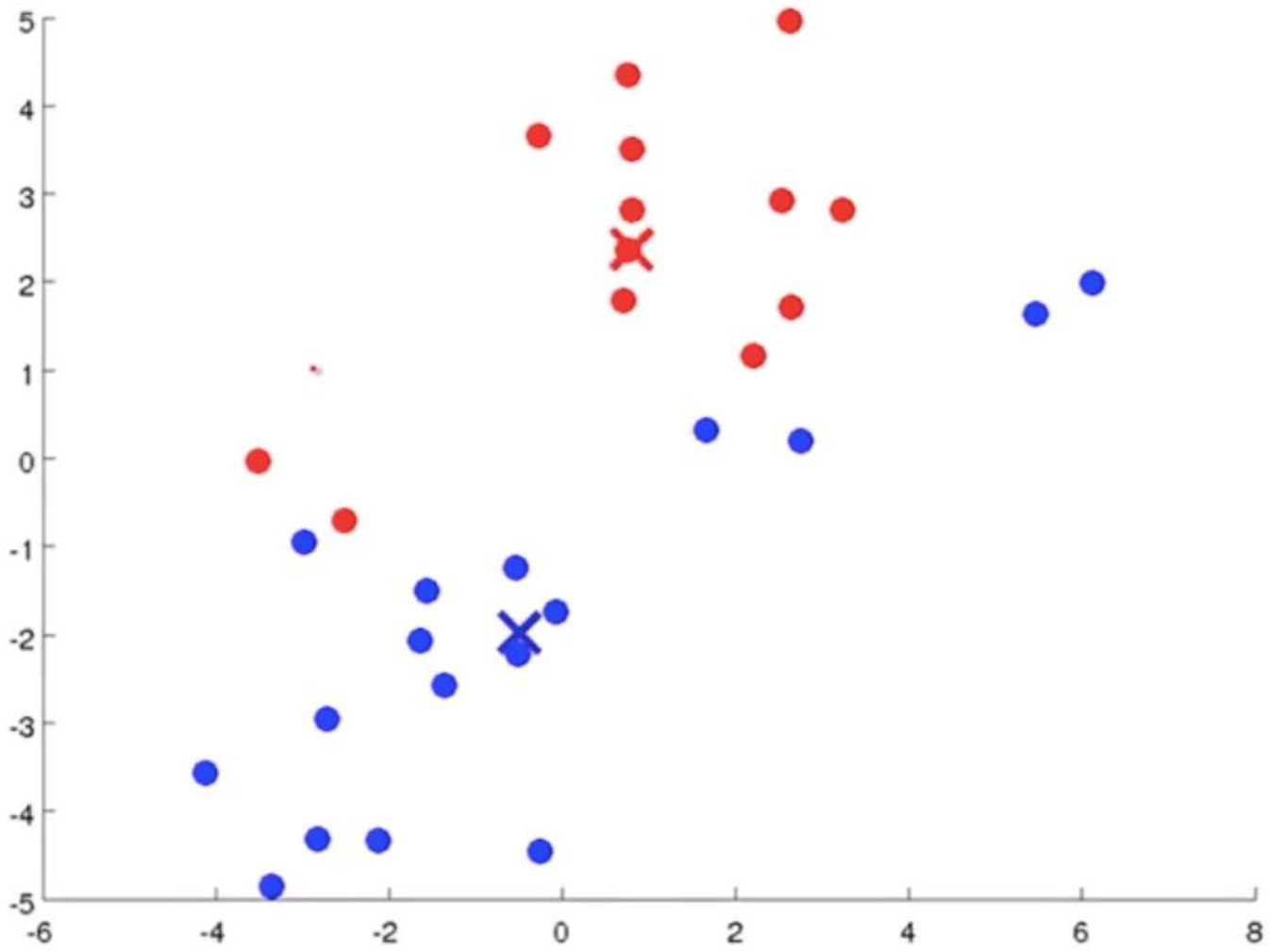
K-means Clustering Algorithm



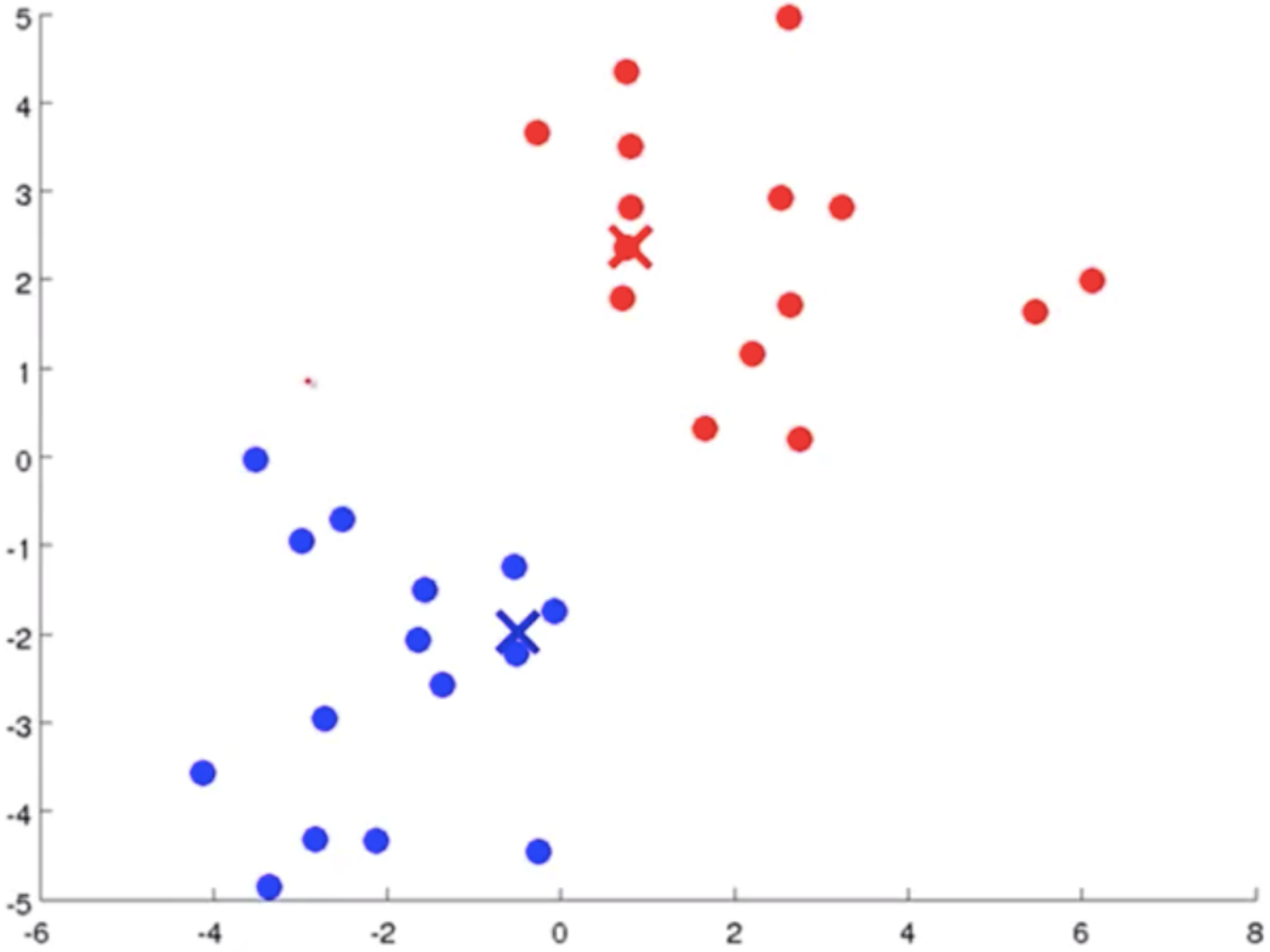
K-means Clustering Algorithm



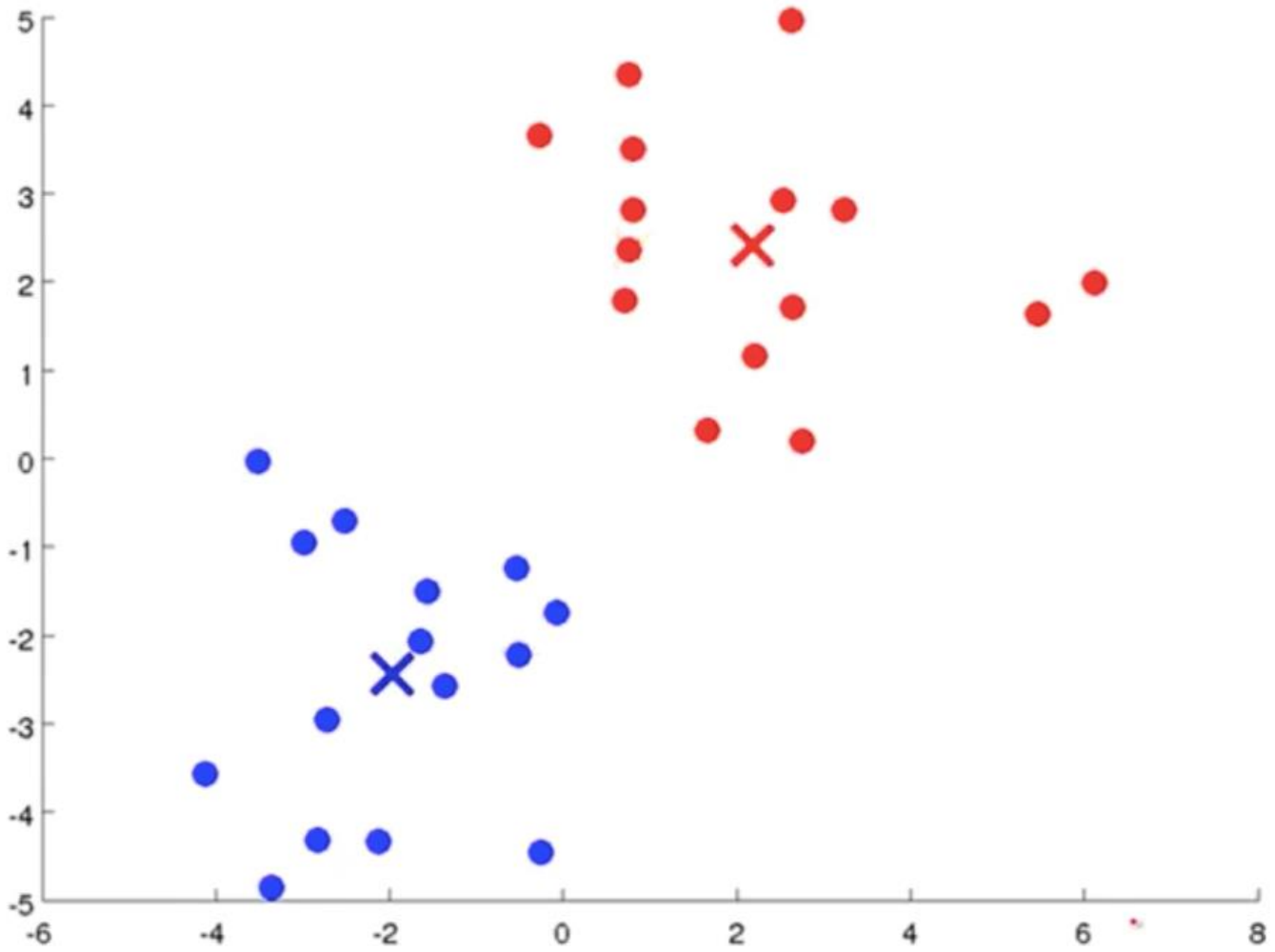
K-means Clustering Algorithm



K-means Clustering Algorithm



K-means Clustering Algorithm



K-means Clustering Algorithm

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)

K-means Clustering Algorithm

K-means 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 closest to $x^{(i)}$

for k = 1 to K

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

}

K-means Clustering Algorithm

K-means algorithm

μ_1 μ_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 $x^{(i)}$

$\min_k \|x^{(i)} - \mu_k\|^2$ → $c^{(i)}$

Move centroid

for k = 1 to K

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

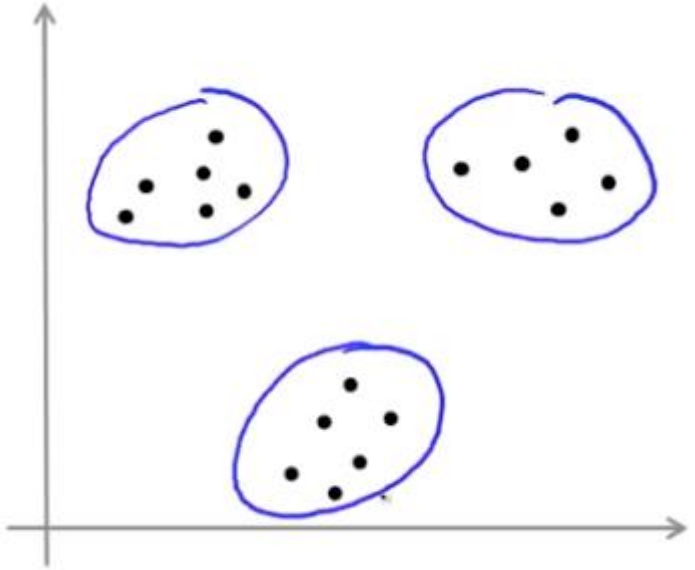
$x^{(1)}, x^{(5)}, x^{(6)}, x^{(10)}$ → $c^{(1)}=2$, $c^{(5)}=2$, $c^{(6)}=2$, $c^{(10)}=2$

$$\mu_2 = \frac{1}{4}\left[x^{(1)} + x^{(5)} + x^{(6)} + x^{(10)}\right] \in \mathbb{R}^n$$

}

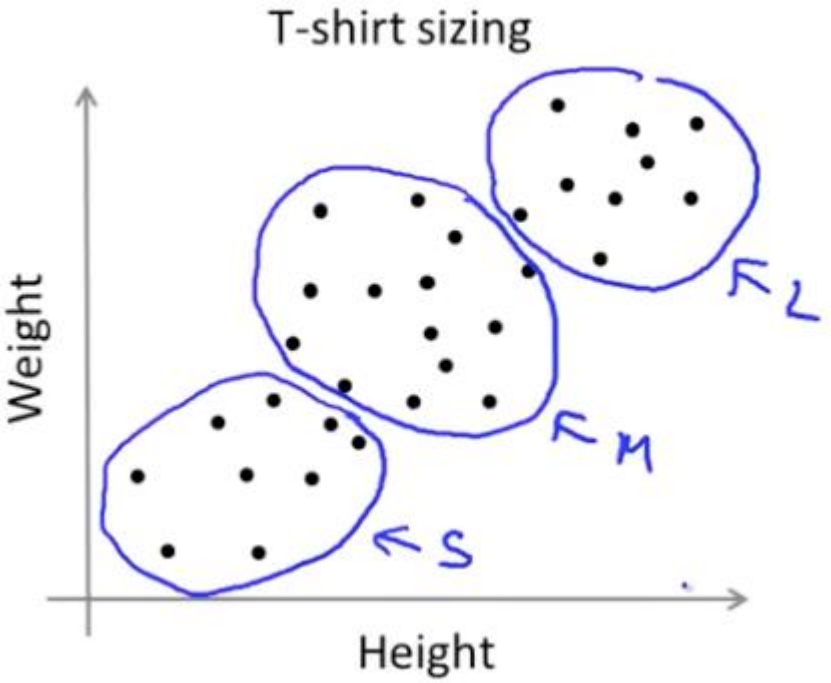
K-means Clustering Algorithm

K-means for non-separated clusters



K-means Clustering Algorithm

K-means for non-separated clusters



K-means Clustering Algorithm

K-means optimization objective

- $c^{(i)}$ = 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

$x^{(i)} \rightarrow 5$ $\quad c^{(i)} = 5$ $\quad \mu_{c^{(i)}} = \mu_5$

K-means Clustering Algorithm

Optimization objective:

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

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

K-means Clustering Algorithm

K-means 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 closest to $x^{(i)}$

for k = 1 to K

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

}

K-means Clustering Algorithm

Random Initialization: K-means

It will lead into a discussion of how to make K-means avoid local optima !

K-means 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 closest to $x^{(i)}$

for k = 1 to K

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

}

K-means Clustering Algorithm

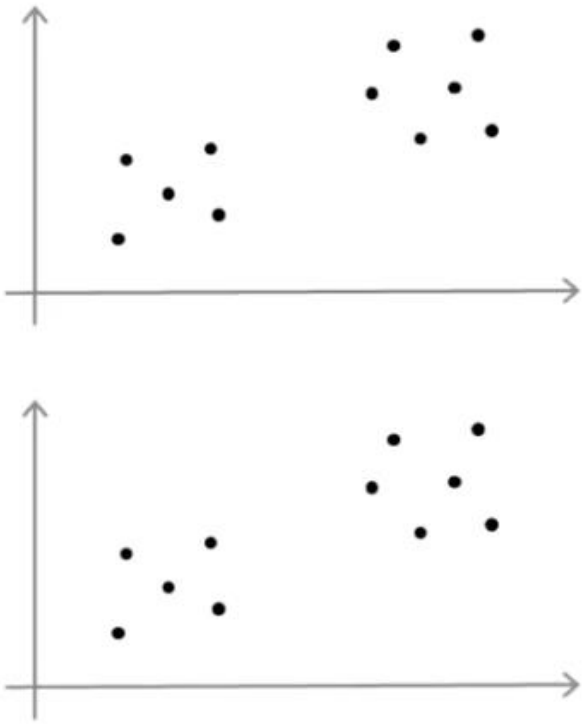
How to initialize cluster centroids?

Random initialization

Should have $K < m$

Randomly pick K training examples.

Set $\mu_1, \ldots, \mu_K$ equal to these K examples.



K-means Clustering Algorithm

How to initialize cluster centroids?

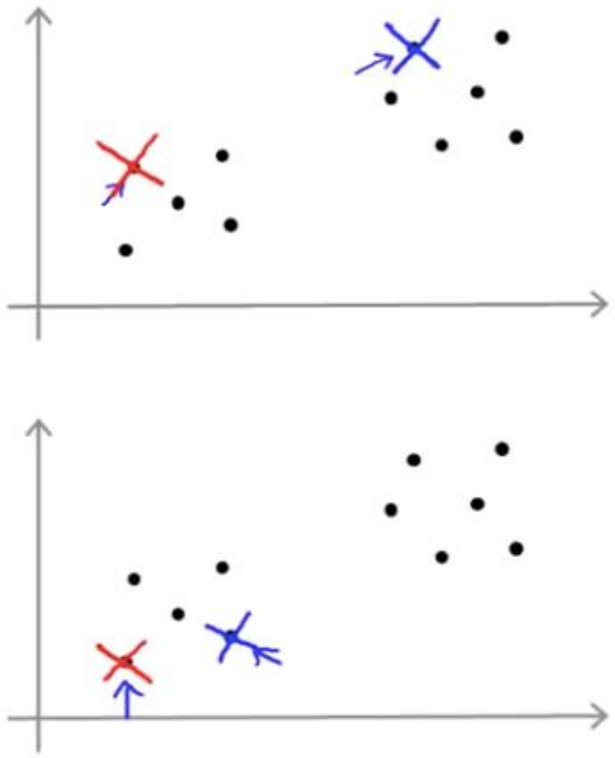
Random initialization

Should have $K < m$

$K=2$

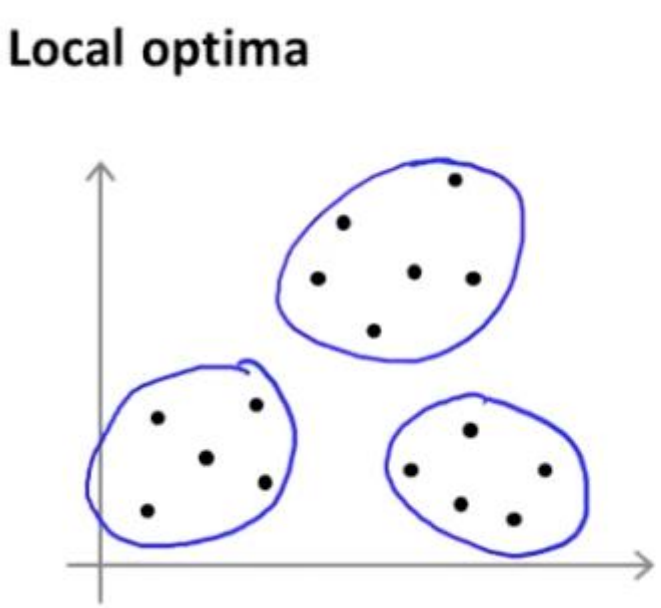
Randomly pick K training examples.

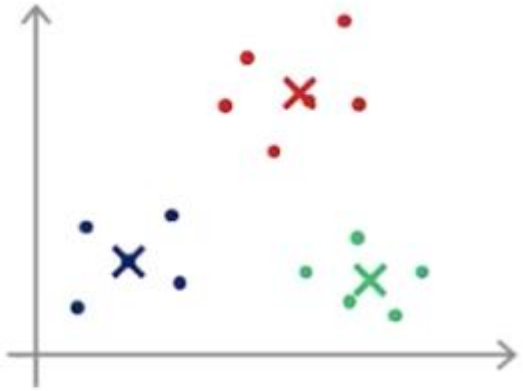
Set $\mu_1, \ldots, \mu_K$ equal to these K examples.



K-means Clustering Algorithm

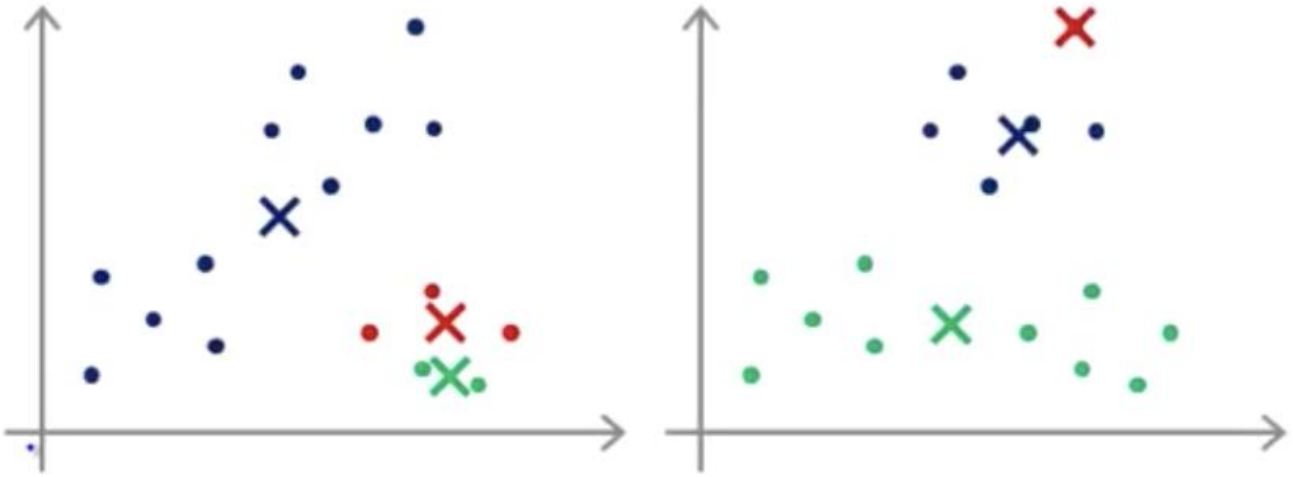
How to initialize cluster centroids?





K-means Clustering Algorithm

How to initialize cluster centroids?



if K-means getting stuck in local optima?

If you want to increase the odds of K-means finding the best possible clustering,

what we can do? try multiple, random initializations?

So, instead of just initializing K-means **once** and hopping that that works, what we can do is, initialize K-means **lots of times** and run K-means lots of times, and use that to try to make sure we get as good a solution, as good a local or global optima as possible.

Random initialization

For i = 1 to 100 {

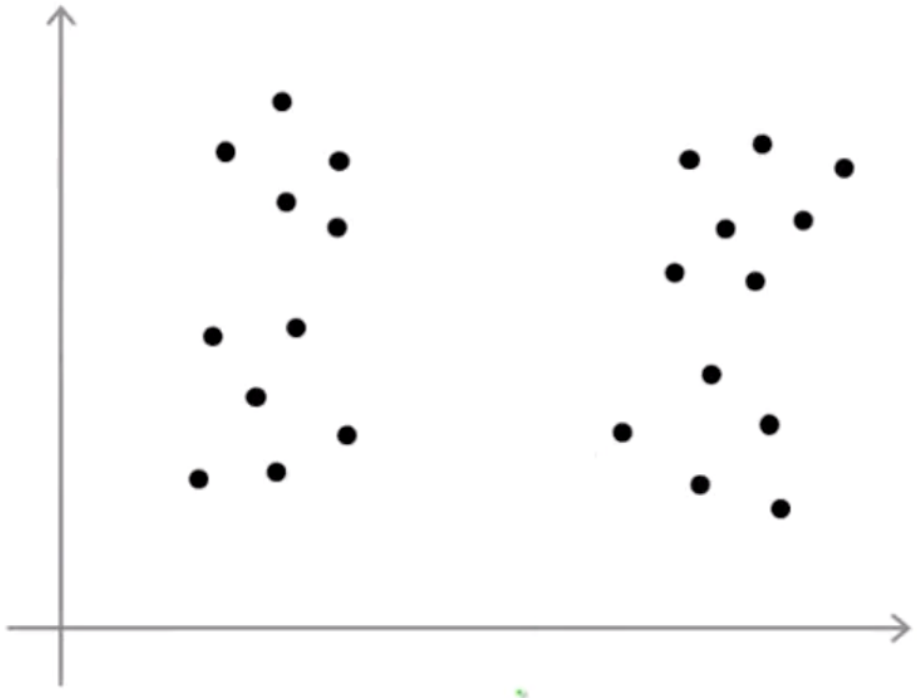
Randomly initialize K-means.
Run K-means. Get $c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K$.
Compute cost function (distortion)
$J(c^{(1)}, \ldots, c^{(m)}, \mu_1, \ldots, \mu_K)$
}

Using this algorithm, we will get 100 different clustering

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

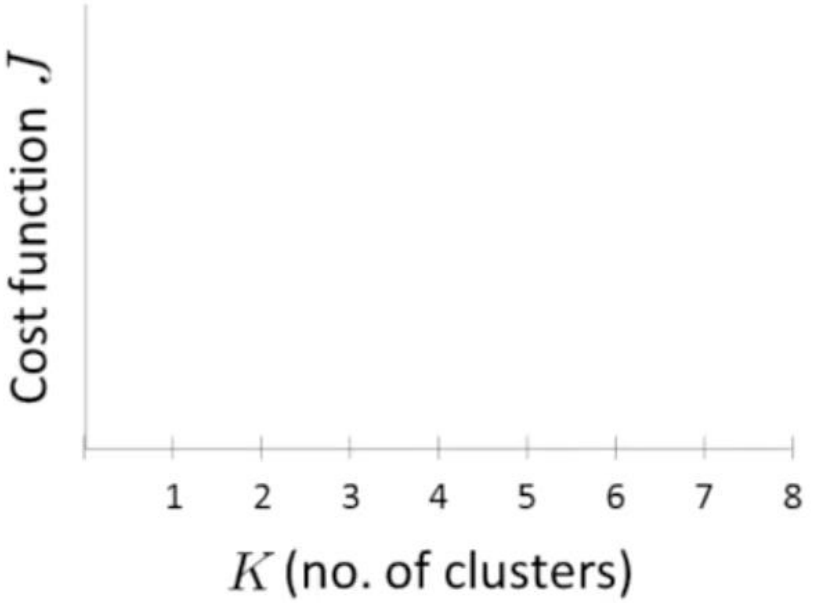
How to choose number of clusters?

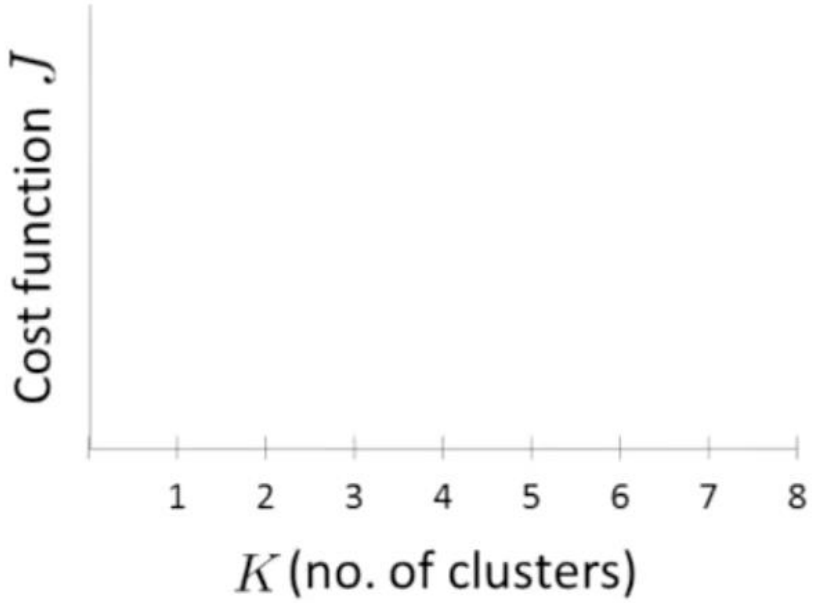
What is the right value of K?



How to choose number of clusters?

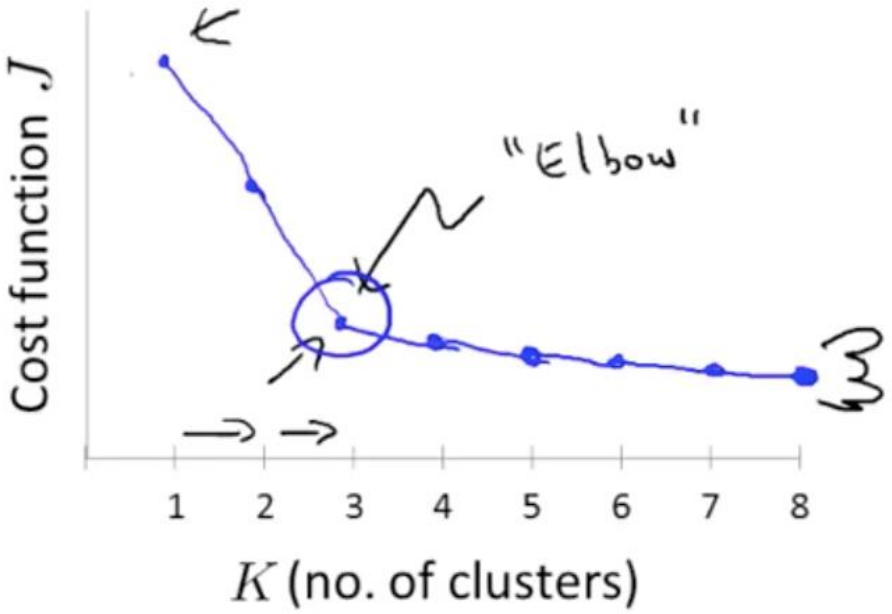
Elbow method:

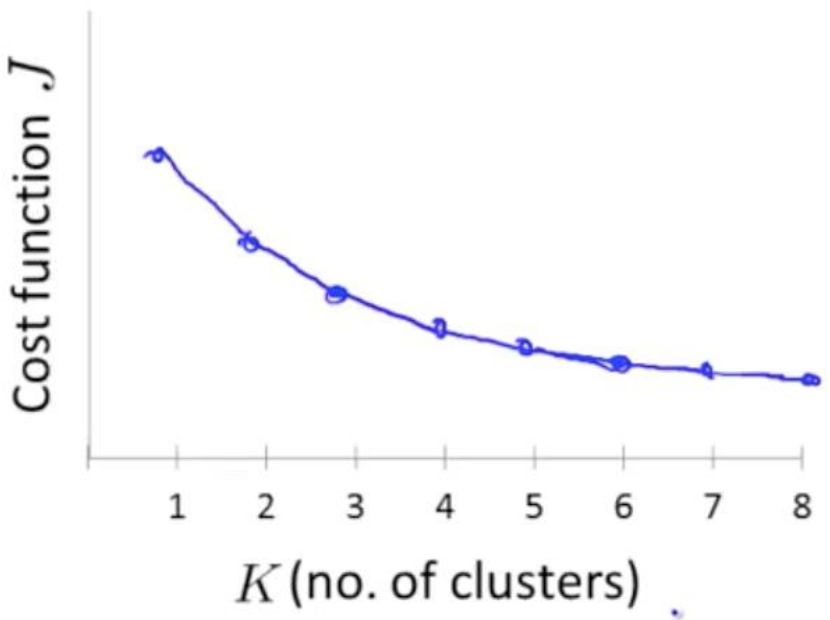




How to choose number of clusters?

Elbow method:



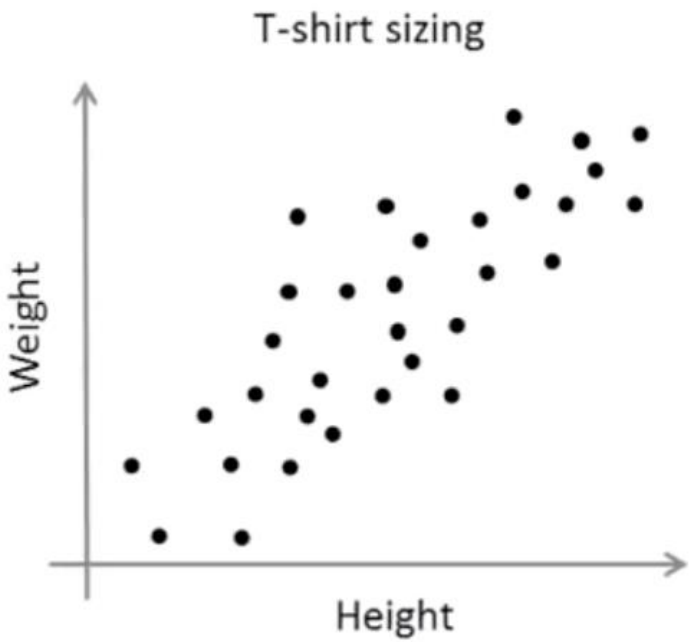


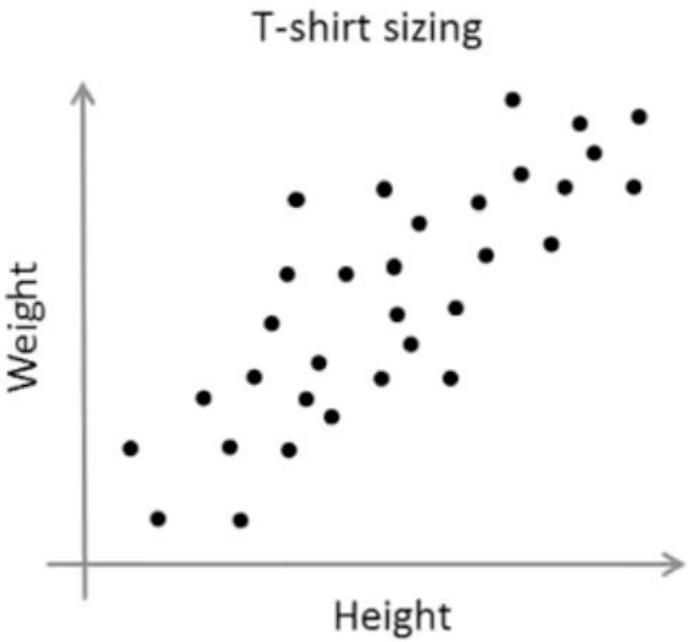
How to choose number of clusters?

Choosing the value of K

Sometimes, you're 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.





How to choose number of clusters?

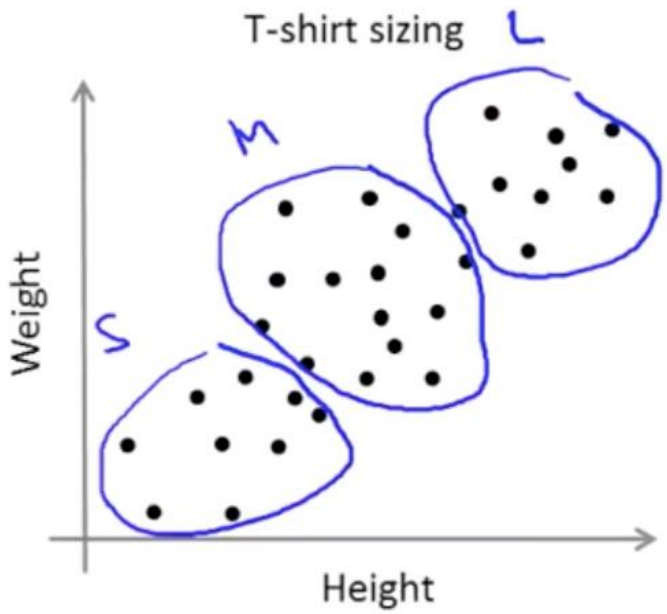
Choosing the value of K

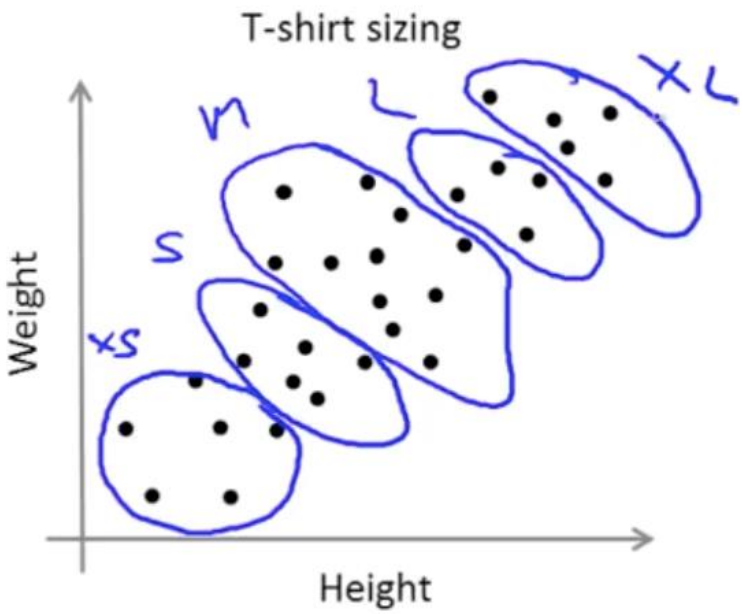
Sometimes, you're running K-means to get clusters to use for some later/downstream purpose. Evaluate K-means based on a metric for how well it performs for that later purpose.

k=3 S, M, L

k=5 XS, S, M, L, XL

E.g.





Drawbacks: K-means

- *Sensitivity to initial configuration.* Since the basic algorithms are local search heuristics and K-means cost function is non-convex, it is very sensitive to the initial configuration and the obtained partition is often only suboptimal (not the globally best partition).
- *Lack of robustness.* As the sample mean and variance are very sensitive estimate against outliers. So-called breakdown point is zero, which means that one gross errors may distort the estimate completely. The obvious consequent is that the k-means problem formulation is highly non-robust as well.
- *Unknown number of clusters.* Since the algorithm is a kind "flat" or "non-hierarchical" method [32], it does not provide any information about the number of clusters.
- *Empty clusters.* The Forgy's batch version may lead to empty clusters on unsuccessful initialization.
- *Order-dependency.* The MacQueen's basic and converging variants are sensitive to the order in which the points are relocated. This is not the case for the batch versions.
- *Only spherical clusters.* K-means presumes the symmetric Gaussian shape for cluster density functions. From this it follows that a large amount of clean data is usually needed for successful clustering.
- *Handling of nominal values.* The sample mean is not defined for nominal values.

Distance Measures

Similarity and Dissimilarity

- **Similarity**
  - Numerical measure of how alike two data objects are.
  - Is higher when objects are more alike.
  - Often falls in the range [0,1]
- **Dissimilarity**
  - Numerical measure of how different are two data objects
  - Lower when objects are more alike
  - Minimum dissimilarity is often 0
  - Upper limit varies

Euclidean Distance

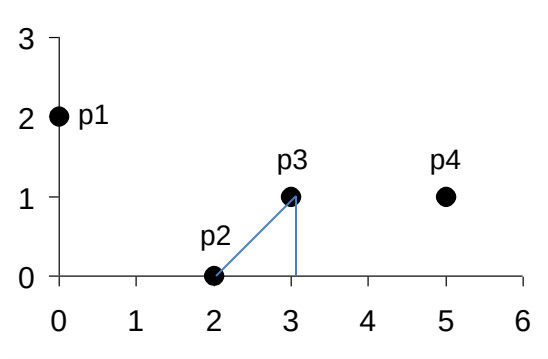
- Euclidean Distance

$$dist = \sqrt{\sum_{k=1}^{n} (p_k - q_k)^2}$$

Where n is the number of dimensions (attributes) and p_k and q_k are, respectively, the kth attributes (components) or data objects p and q.

- Standardization is necessary, if scales differ.

Euclidean Distance



point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

Distance Matrix