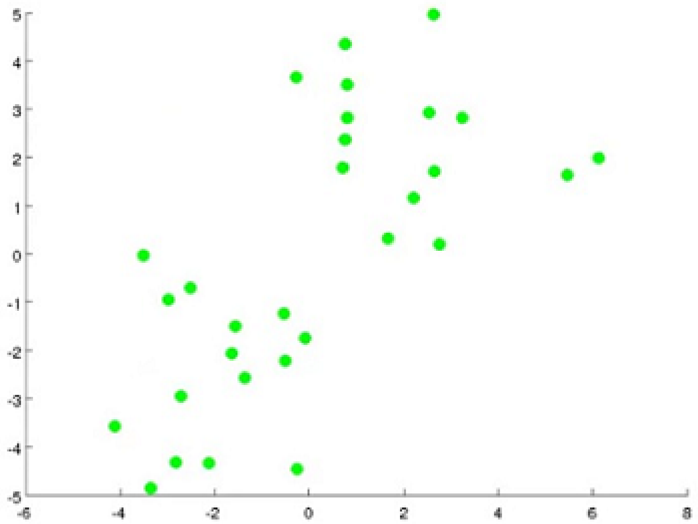
K-Means Algorithm

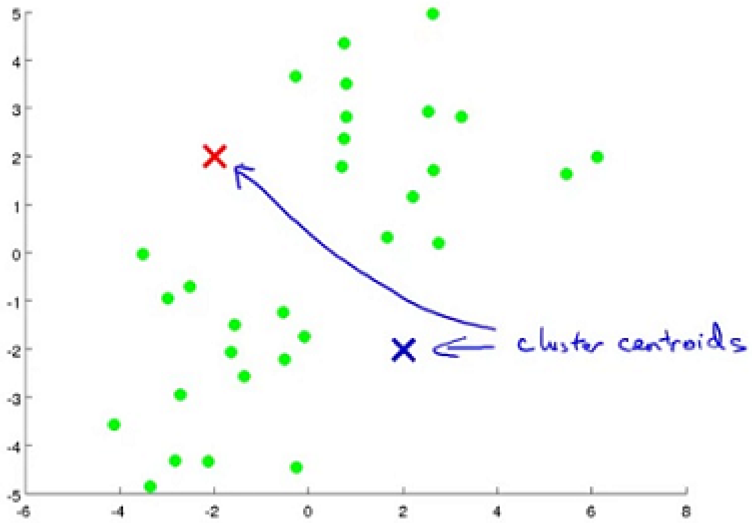
In the clustering problem we are given an unlabeled data set and we would like to have an algorithm **automatically group the data** into coherent subsets or into coherent clusters for us. The K Means algorithm is by far the most popular, by far the most widely used clustering algorithm.

The K means clustering algorithm is best illustrated in pictures, let's say we want to take an unlabeled data:

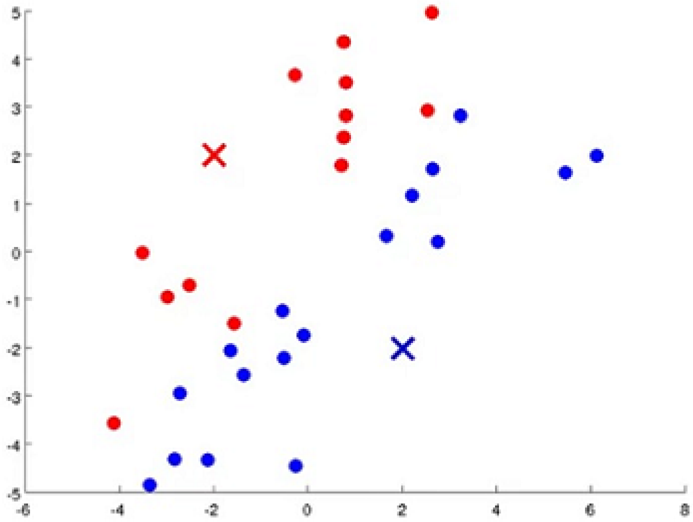


K-Means Algorithm Overview

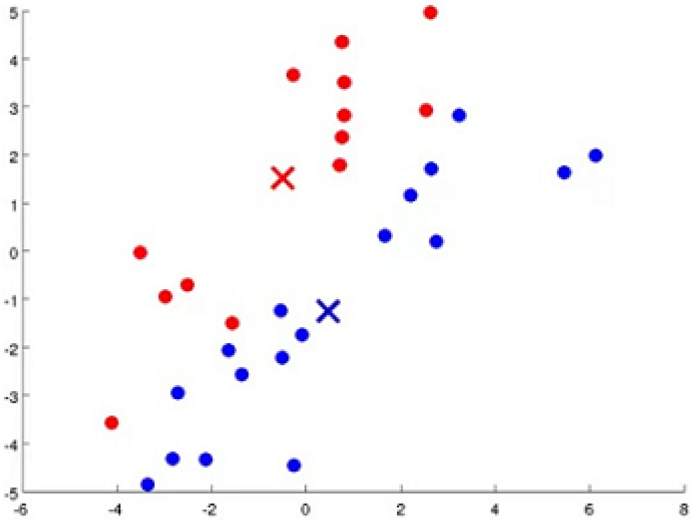
K-Means is an iterative algorithm and it does two things. First is a cluster assignment step, and second is a move centroid step:



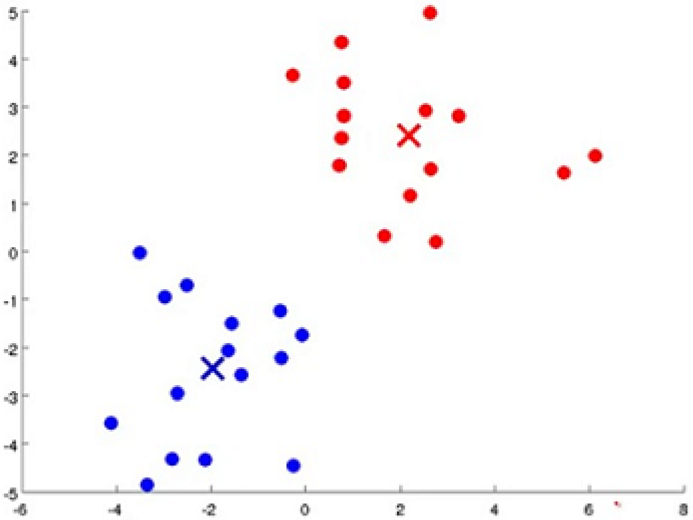
- 1. The first step (cluster assignment step) is to randomly initialize two points, called the **cluster centroids**.
    - Randomly allocate two points as the cluster centroids
        - Have as many cluster centroids as clusters we want to do (K cluster centroids, in fact).
        - In our example we just have two clusters
    - Go through each example and depending on if it's closer to the red or blue centroid assign each point to one of the two clusters
    - To demonstrate this, we've gone through the data and "colour" each point red or blue



- 2. The second step is **the move centroid step**.
  - Take each centroid and move to the average of the correspondingly assigned data-points



We're going to repeat the 1^{st} and 2^{nd} step until convergence.



In fact if we keep running additional iterations of K means from there the cluster centroids will not change any further and the colours of the points will not change any further.

And so, at this point K means has converged and it's done a pretty good job finding the two clusters in the data.

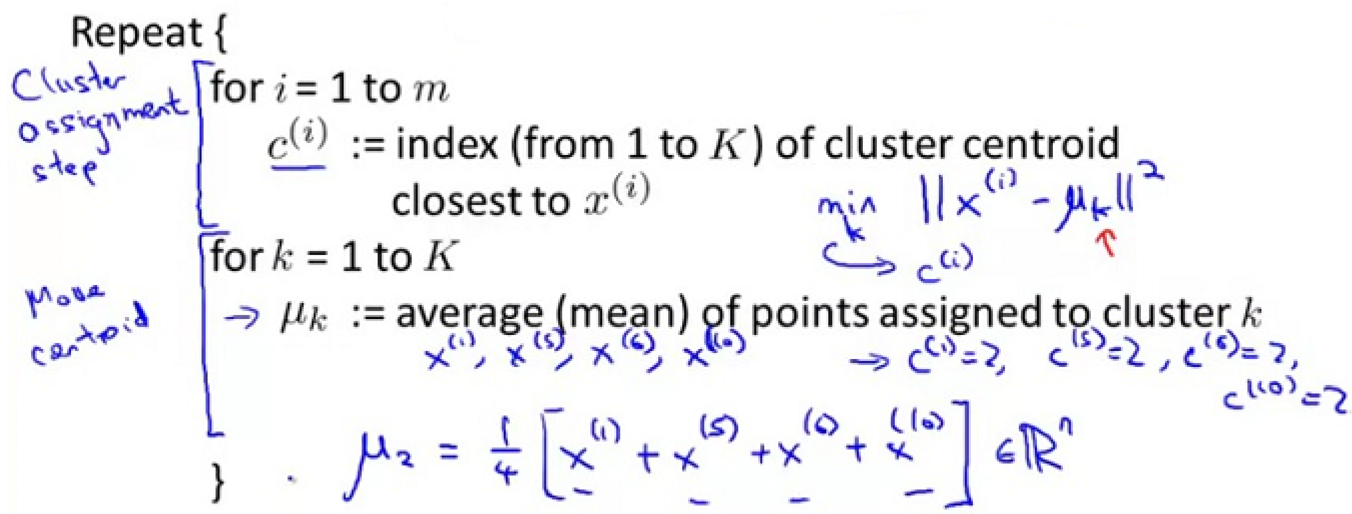
More formally definition:

Input:

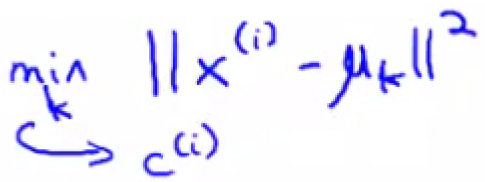
- K (number of clusters)
- Training set $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

$x^{(i)} \in \mathbb{R}^n$ (drop $x_0 = 1$ convention)

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



- **For loop 1 - Cluster assignement step**
  - This inner loop repeatedly sets the $c^{(i)}$ variable to be the index of the closes variable of cluster centroid closes to $x^{(i)}$.
  - i.e. take i^{th} example, measure squared distance to each cluster centroid $||x^{(2)} - \mu_k||^2$, assign $c^{(i)}$ to the cluster closest.
    - Uppercase K is going to be used to denote the total number of cluster centroids
    - We're going to use lowercase k to index into our different cluster centroids.



- **For loop 2 - Move centroid**
  - Loops over each centroid calculate the average mean based on all the points associated with each centroid from $c^{(i)}$
  - i.e. let's say that one of our cluster centroids μ_2, has training examples: $x^{(1)}, x^{(5)}, x^{(6)}, x^{(10)}$
    - What this means is: $c^{(1)} = 2, c^{(5)} = 2, c^{(6)} = 2, c^{(10)} = 2$
    - Then move centroid step, what we're going to do is just compute the average of the training examples:
      - $\mu_2 = \frac{1}{4}[x^{(1)} + x^{(5)} + x^{(6)} + x^{(10)}] \in \mathbb{R}^n$
  - What if there's a centroid with no data:
    - Remove that centroid, so end up with $K - 1$ classes
    - Sometimes if we really need K clusters, then the other thing we can do if we have a cluster centroid with no points assigned to it, is we can just randomly reinitialize that cluster centroid.
    - It's more common to just eliminate a cluster if somewhere during K means it with no points assigned to that cluster centroid.

Video Question: Suppose you run k-means and after the algorithm converges, you have:
$c^{(1)} = 3, c^{(2)} = 3, c^{(3)} = 5, \ldots$

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

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

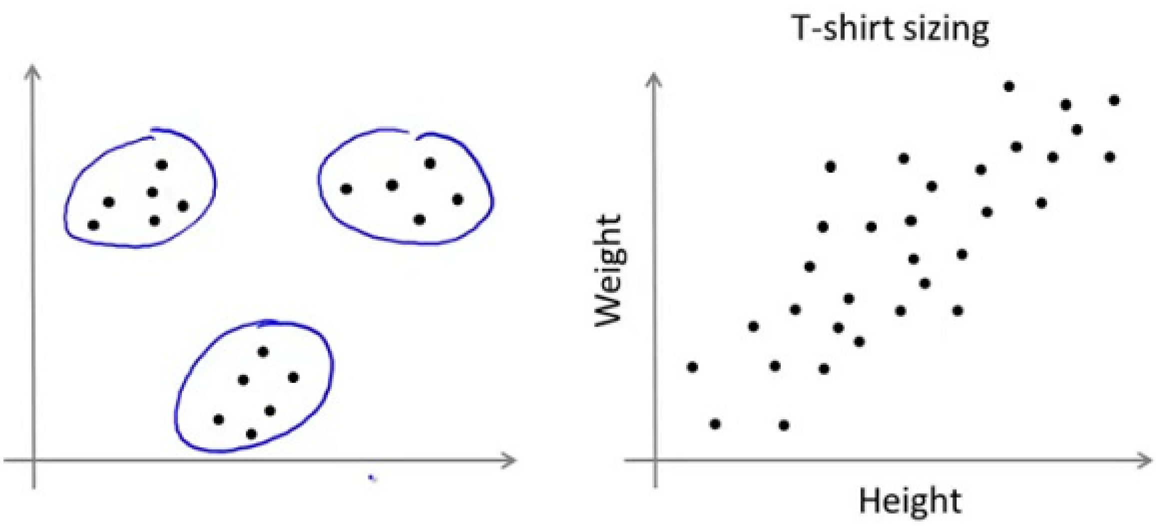
The first and second training examples $x^{(1)}$ and $x^{(2)}$ have been assigned to the same cluster.

- The second and third training examples have been assigned to the same cluster.

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

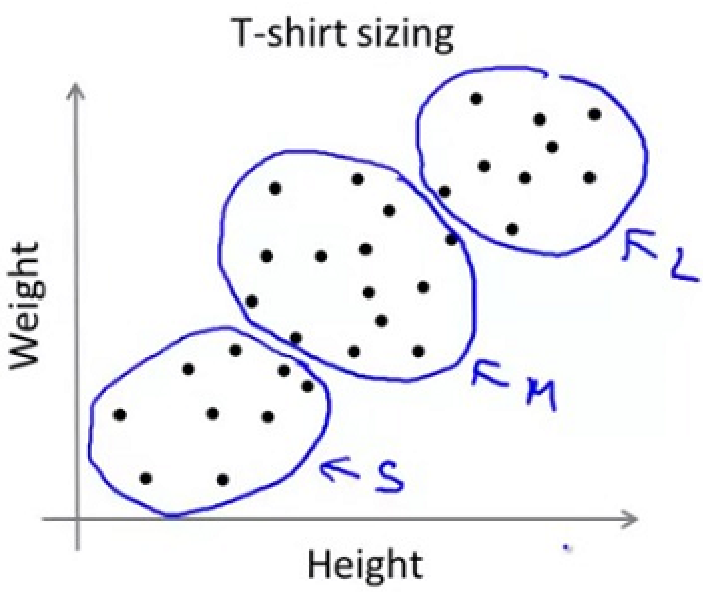
K-means for non-separated clusters

So far we've been picturing K-Means and applying it to data sets where we have three pretty well separated clusters, and we'd like an algorithm to find maybe the 3 clusters for us. But it turns out that very often K-Means is also applied to data sets where there may not be several very well separated clusters. E.g. T-shirt sizing:



Say we want to have three sizes (S, M, L) how big do we make these?

- One way would be to run K-means on this data
  - One way would be to run K-means on this data
  - May do the following:



So, even though the data, before hand it didn't seem like we had 3 well separated clusters, K-Means will kind of separate out the data into multiple clusters for us.

- Look at first population of people
  - Try and design a small T-shirt which fits the 1^{st} population
  - And so on for the other two
- This is an example of market segmentation
  - Build products which suit the needs of our subpopulations