Choosing the Number of Clusters

How to choose the number of clusters, or how to choose the value of the parameter K.

- Not a great way to do this automatically.
- The most common way of choosing the number of clusters, is still choosing it manually by looking at visualizations or by looking at the output of the clustering algorithm or something else.
- The most common thing is actually to choose the number of clusters by hand.

What is the right value of K?

A large part of why it might not always be easy to choose the number of clusters is that it is often generally ambiguous:

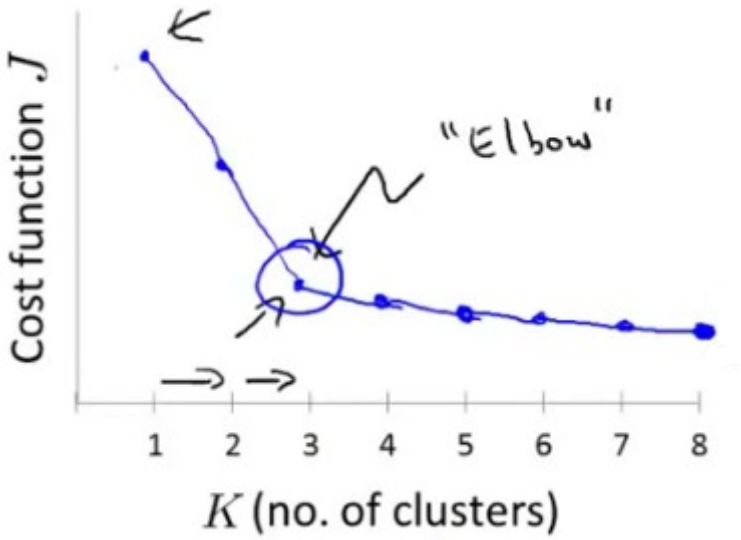
- e.g. how many clusters to choose in the data?, two clusters or four clusters?
- Not necessarily a correct answer

This is one of the things that makes it more difficult to have an automatic algorithm for choosing how many clusters to have.

Choosing the value of K - Elbow method:

- Vary K and compute cost function (the distortion J) at a range of K values.
- As K increases $J(\dots)$ minimum value should decrease
  - i.e. we decrease the granularity so centroids can better optimize

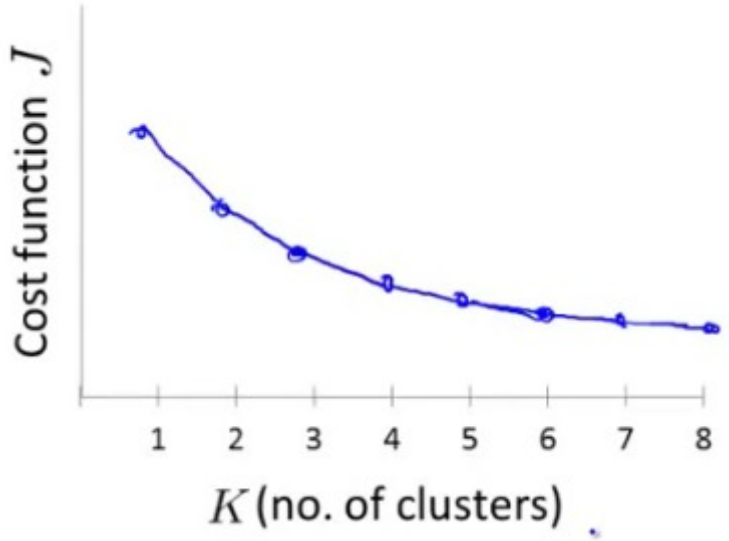
Plot K (no. of clusters) vs. J (cost function) and look for the "elbow" on the graph:



- Chose the "elbow" number of clusters
- If we get a nice plot this is a reasonable way of choosing K

Concretely the elbow method is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use.

It turns out the Elbow Method isn't used that often, and one reason is that, if we actually use this on a clustering problem, it turns out that fairly often we end up with a curve that looks much more ambiguous (Normally we don't get a a nice line - no clear elbow on curve, not really that helpful).



The summary of the Elbow Method is that is worth the shot but I wouldn't necessarily have a very high expectation of it working for any particular problem.

Video 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?

- This is mathematically impossible. There must be a bug in the code.
- The correct number of clusters is k = 3.

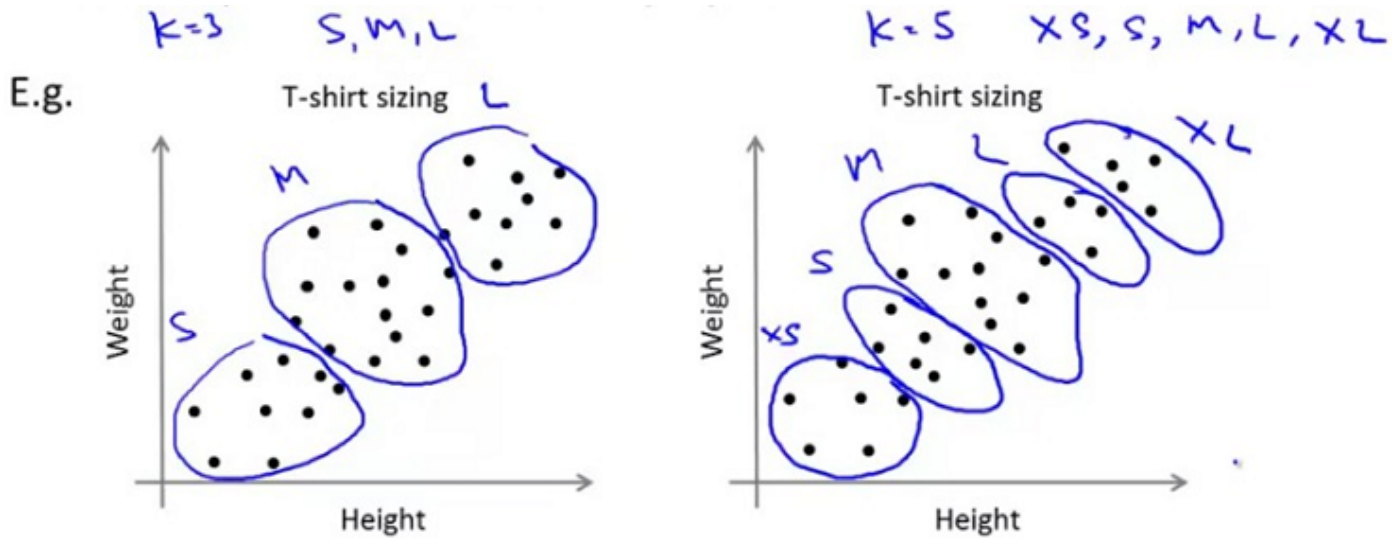
> In the run with k = 5, k-means got stuck in a bad local minimum. You should try re-running k-means with multiple random initializations.

- 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 than with k = 5.

Another method for choosing the value of K:

Sometimes, we'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.

- T-shirt size example:
  - If we have three sizes (S, M, L)
  - Or five sizes (XS, S, M, L, XL)
  - Run K means where K = 3 and K = 5



- This gives a way to chose the number of clusters
  - Could consider the cost of making extra sizes vs. how well distributed the products are
  - How important are those sizes though?
    - e.g. more sizes might make the customers happier
  - So applied problem may help guide the number of clusters