SUS: Machine Learning: Lab5

Eyad Kannout

Clustering Techniques:

Clustering is a Machine Learning technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group.

Clustering is a method of **unsupervised learning** and is a common technique for statistical data analysis used in many fields. In Data Science, we can use clustering analysis to gain some valuable insights from our data by seeing what groups the data points fall into when we apply a clustering algorithm.

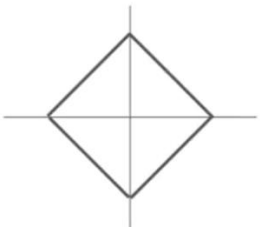
Distance Metric Rules:

All spaces for which we can perform a clustering should have a distance measure, giving a distance between any two points in the space.

An example is the Euclidean space, where points are vectors of real numbers and the length of the vector is the number of dimensions of that space. The components of the vector are commonly called coordinates of the represented points. The common **Euclidean distance** (square root of the sums of the squares of the differences between the coordinates of the points in each dimension) serves for all Euclidean spaces (some other are the Manhattan distance).

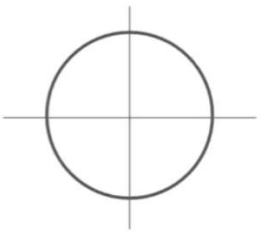
L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$



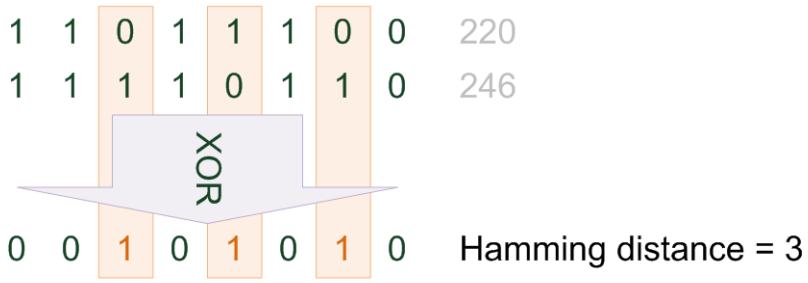
L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

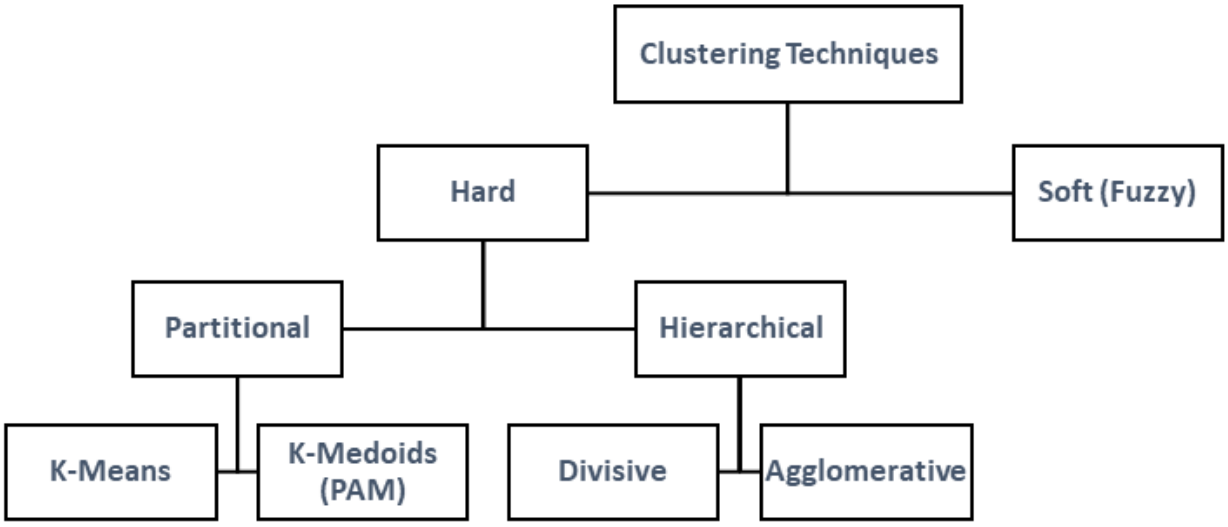


There are also other distance measures for non-Euclidean spaces. These include the **Jaccard distance**, **cosine distance**, **Hamming distance**, **edit distance**.

$$\cos(u_i, u_k) = \frac{\sum_{j=1}^{m} v_{ij} v_{kj}}{\sqrt{\sum_{j=1}^{m} v_{ij}^2 \sum_{j=1}^{m} v_{kj}^2}}$$



Clustering strategies



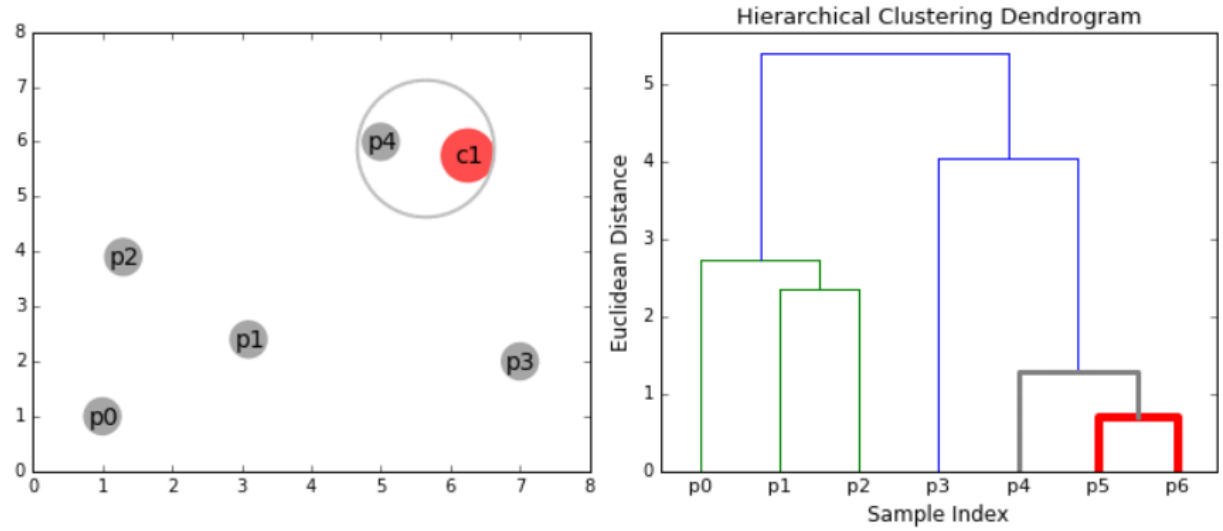
Agglomerative Hierarchical Clustering (Hierarchical grouping):

We start with one-element clusters, each cluster has one element, and then combine the clusters the most similar until we meet the criterion of stop. The stop criteria are, for example, the number of clusters.

We begin with every point in its own cluster. As time goes on, larger clusters will be constructed by combining two smaller clusters, and we have to decide in advance:

- How the clusters will be represented?
- How we will choose which two clusters to merge?
- When we will stop combining clusters?

Hierarchical clustering algorithms actually fall into two categories: **top-down** and **bottom-up**



Advantages:

- Hierarchical clustering does not require us to specify the number of clusters and we can even select which number of clusters looks best since we are building a tree.
- A particularly good use case of hierarchical clustering methods is when the underlying data has a hierarchical structure and you want to recover the hierarchy; other clustering algorithms can't do this.

Disadvantages:

- The cost of lower efficiency, as it has a time complexity of $O(n^3)$, unlike the linear complexity of K-Means and GMM.

****

k-means (k-centroids) and k-medoids:

K-means is a classical partitioning technique of clustering that clusters the data set of n objects into k clusters with k known a priori.

Given an initial set of k means (centroids) m1(1),…,mk(1), the algorithm proceeds by alternating between two steps:

- Assignment step: Assign each observation to the cluster with the closest mean
- Update step: Calculate the new means to be the centroid of the observations in the cluster.

The result of the algorithm depends on the initial choice of center points, so there is a risk that the algorithm can go to the local minimum not to the global minimum.

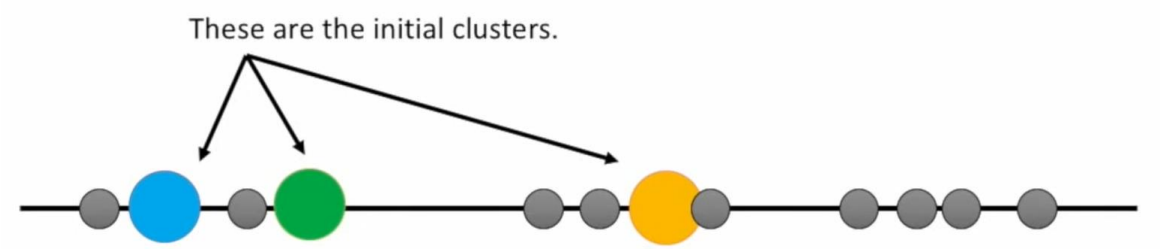
k-medoid clustering:

The most common realization of k-medoid clustering is the Partitioning around Medoids (PAM) algorithm and is as follows:

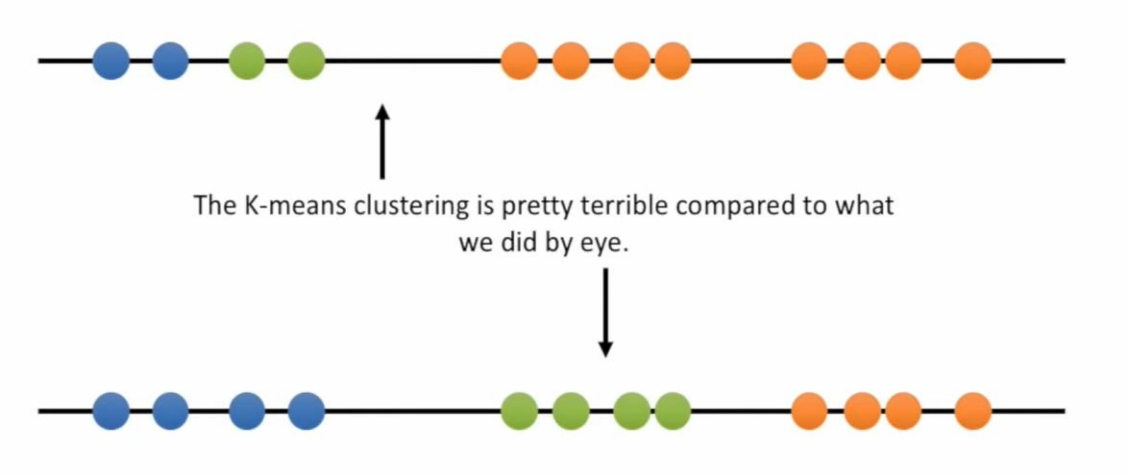
- Initialize: randomly select k of the n data points as the medoids
- Assignment step: Associate each data point to the closest medoid.
- Update step: For each medoid m and each data point o associated to m swap m and o and compute the total cost of the configuration. Select the medoid o with the lowest cost of the configuration.
- Repeat alternating steps 2 and 3 until there is no change in the assignments.

Stopping Criteria:
The algorithm is deemed to have converged when the assignments no longer change.

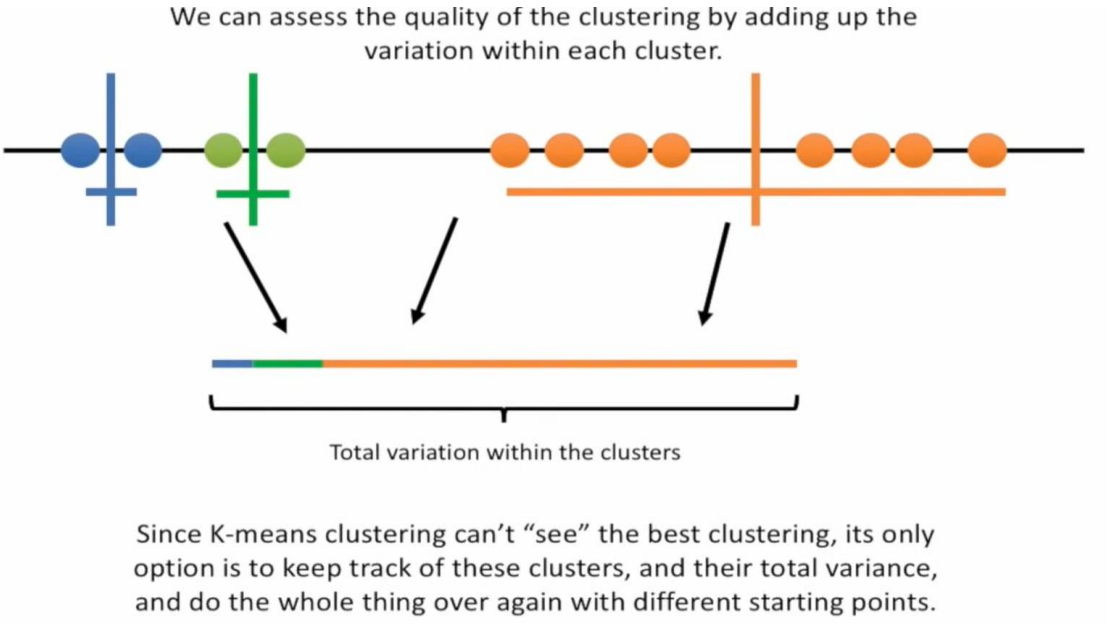
How the initial choice of clusters' centers impacts the final results?
Let's assume we select below colored point as clusters' centers



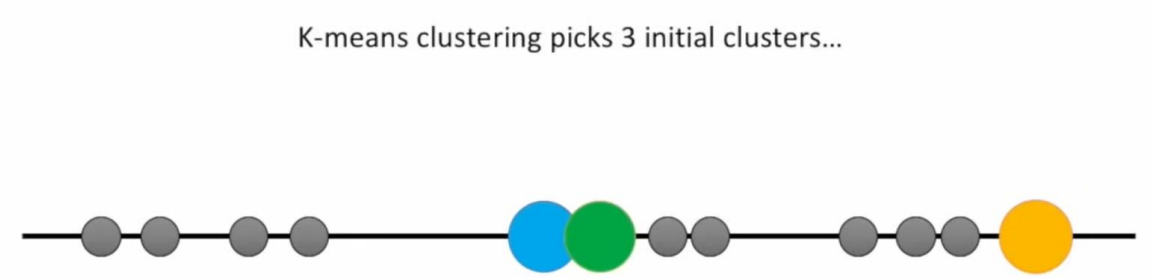
After applying KMeans, we find that the final results is not optimal:



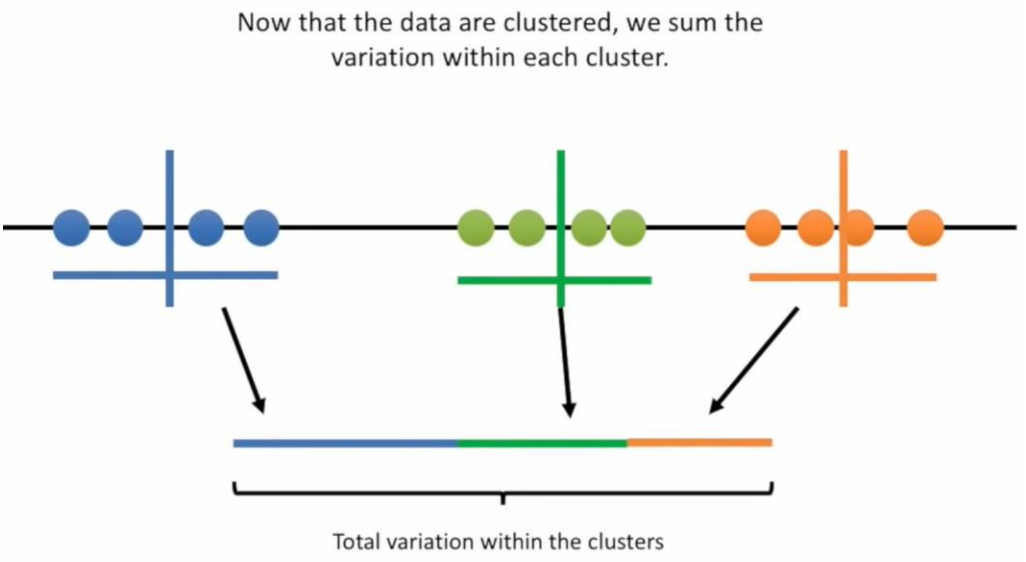
How can we assess the quality of clustering?

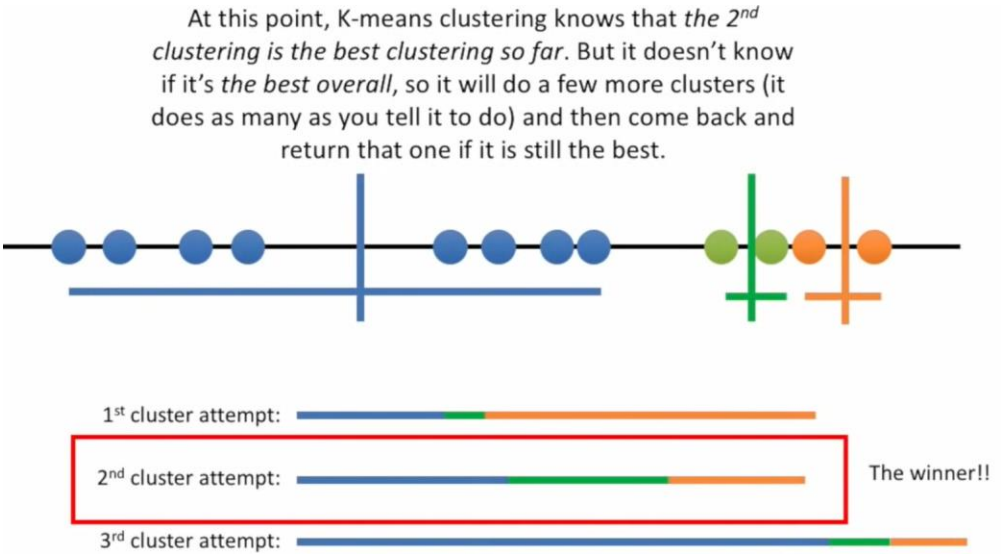


Let's take other initial clusters:

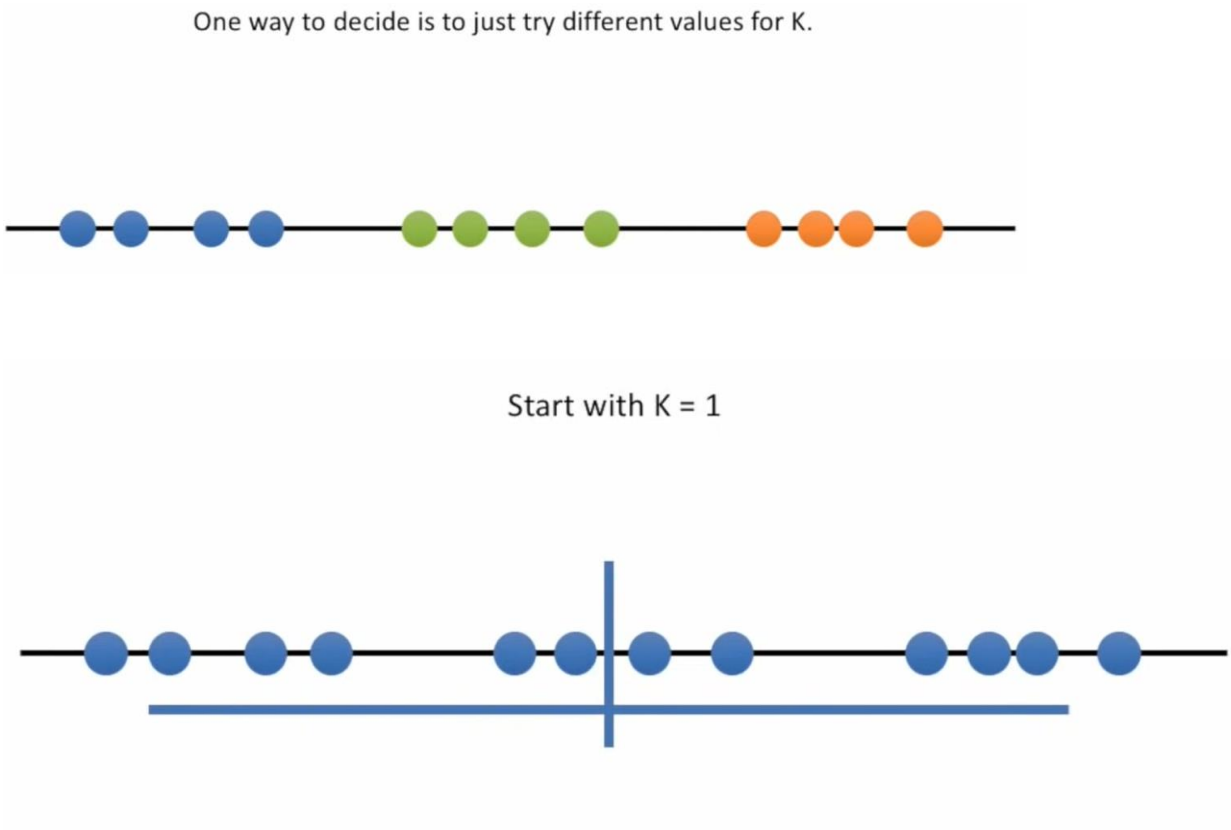


The results:



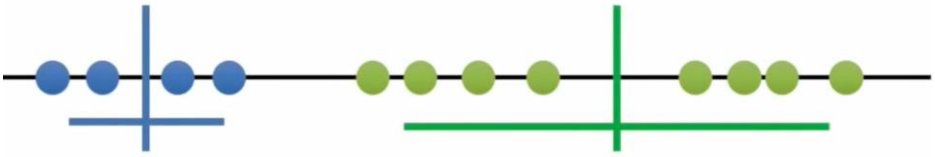


How do you figure out what value to use for "K"?



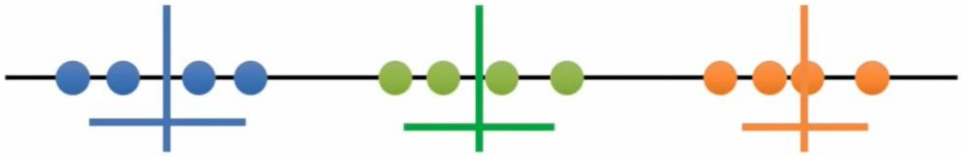
K = 1 is the worst case scenario. We can quantify its "badness" with the total variation.

Now try K = 2



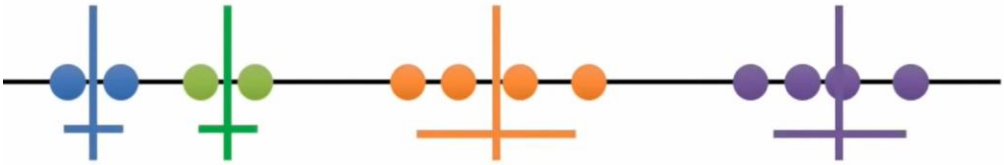
K = 2 is better, and we can quantify how much better by comparing the total variation within the 2 clusters to K = 1

Now try K = 3

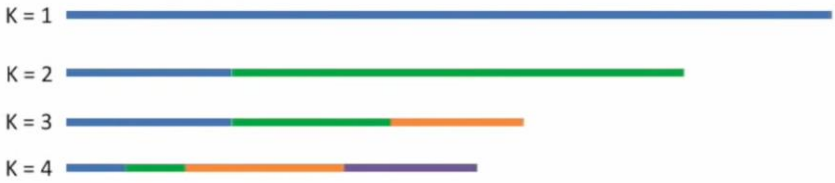


K = 3 is even better! We can quantify how much better by comparing the total variation within the 3 clusters to K = 2

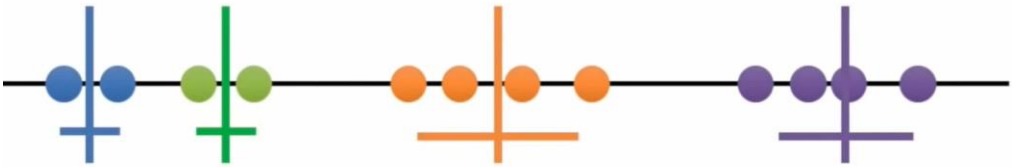
Now try K = 4



The total variation within each cluster is less than when K=3

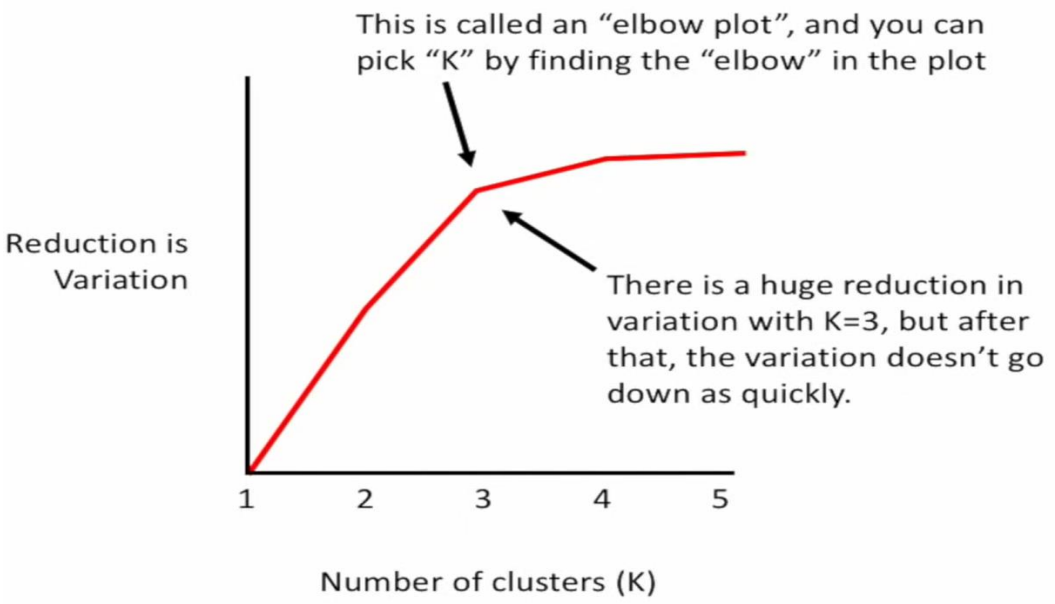


Now try K = 4



The total variation within each cluster is less than when K=3

Each time we add a new cluster, the total variation within each cluster is smaller than before. And when there is only one point per cluster, the variation = 0.



****

Algorithm: Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM):

We assume our data is sampled from K different sources (probability distributions). The expectation maximization (EM) algorithm allows us to discover the parameters of these distributions, and figure out which point comes from each source at the same time.

Types of clustering methods:

1. Hard clustering: Clusters don't overlap. Elements either belongs to the cluster or not.
2. Soft clustering: Clusters may overlap. Elements may belong to more than one cluster with different degree of belief.

Remember:

Gaussian functions are often used to represent the probability density function of a normally distributed random variable with expected value(mean) μ = b and variance σ2 = c2. In this case, the Gaussian is of the form:

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

σ: standard deviation (square root of variance)
μ: mean

Gaussian Mixture Models (GMMs) give us more flexibility than K-Means.

- With GMMs we assume that the data points are Gaussian distributed; this is a less restrictive assumption than saying they are circular by using the mean.
- That way, we have two parameters to describe the shape of the clusters: the mean and the standard deviation!

The steps:

- We begin by selecting the number of clusters (like K-Means does) and randomly initializing the Gaussian distribution parameters for each cluster.
- Given these Gaussian distributions for each cluster, compute the probability that each data point belongs to a particular cluster. The closer a point is to the Gaussian's center, the more likely it belongs to that cluster.

$$P(b \mid x_i) = \frac{P(x_i \mid b)P(b)}{P(x_i \mid b)P(b) + P(x_i \mid a)P(a)}$$

P(xi|b): probability that xi came from b gaussian distribution, here we use Gaussian function to calculate it:

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

P(b): will be described later

- Based on these probabilities, we compute, or estimate, a new set of parameters for the Gaussian distributions such that we maximize the probabilities of data points within the clusters. We compute these new parameters using a weighted sum of the data point positions, where the weights are the probabilities of the data point belonging in that particular cluster.
- The weights (confidences) here are the posterior probabilities. The main difference between this and k-means is that in k-means the probabilities are 0 or 1.

$$\mu_b = \frac{b_1 x_1 + b_2 x_2 + \ldots + b_n x_{n_2}}{b_1 + b_2 + \ldots + b_n}$$

where: b1: is the probability of the point x1 belongs to cluster b (distribution b).

We do the same for the variance as following:

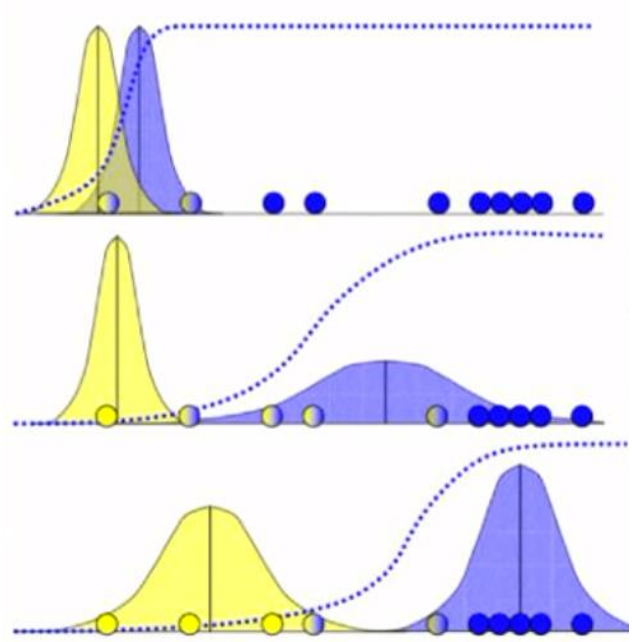
$$\sigma_b^2 = \frac{b_1 (x_1 - \mu_1)^2 + \ldots + b_n (x_n - \mu_n)^2}{b_1 + b_2 + \ldots + b_n}$$

And for all clusters:

$$\mu_a = \frac{a_1 x_1 + a_2 x_2 + \ldots + a_n x_{n_2}}{a_1 + a_2 + \ldots + a_n}$$

$$\sigma_a^2 = \frac{a_1 (x_1 - \mu_1)^2 + \ldots + a_n (x_n - \mu_n)^2}{a_1 + a_2 + \ldots + a_n}$$

After updating the gaussian parameters and after a few iterations they will fit more the points assigned to them as follows:



- Also, you could estimate the priors, if they are not uniform distributed (fixed), as follows: Priors tell you what proportion of points is the blue distribution describing value

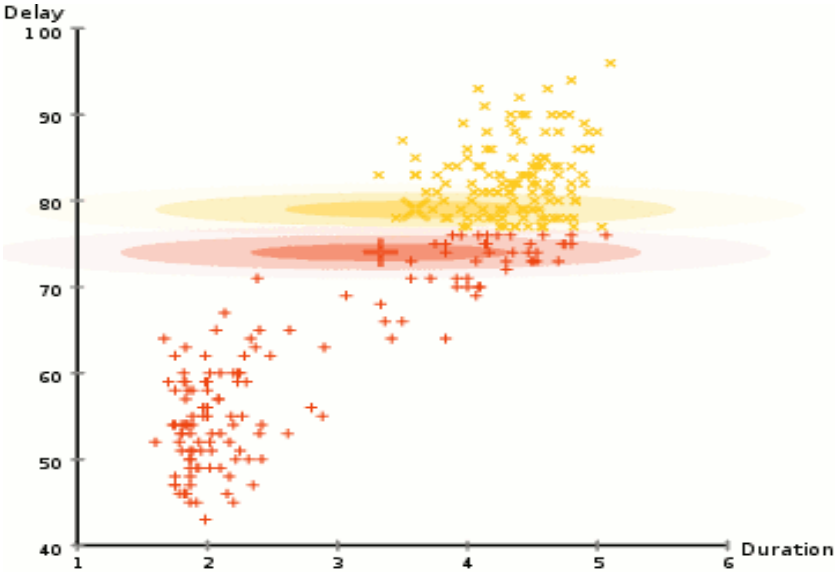
could also estimate priors:

$$P(b) = (b_1 + b_2 + \dots b_n) / n$$
$$P(a) = 1 - P(b)$$

where: b1: is the probability of the point x1 belongs to cluster b (distribution b).

- Steps 2 and 3 are repeated iteratively until convergence, where the distributions don't change much from iteration to iteration.

Summary of EM:



Task:

Implement the EM algorithm for a pair of one-dimensional normal distributions and cluster it with the series [0; 0; 1; 1; 2; 3; 4; 6; 9]. Assume that

μA = 0; σA = 1; τA = 1/2

μB = 0; σB = 10; τB = 1/2.

Here: tau represents prior probability in Bayes theorem and it can be estimated as follows:

could also estimate priors:

$$P(b) = (b_1 + b_2 + \dots b_n) / n$$
$$P(a) = 1 - P(b)$$