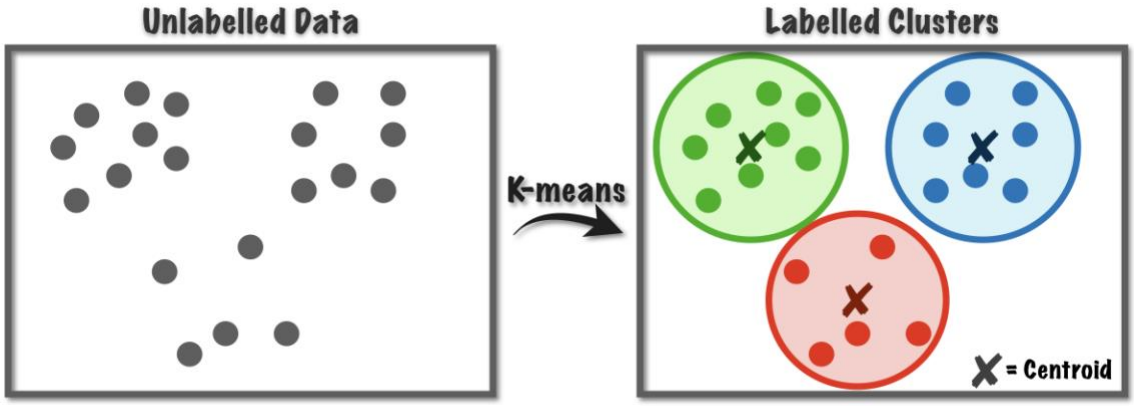
“this is just for reference because these are rough notes”

What is k-Means Clustering

This is an unsupervised learning algorithm, essentially meaning that the algorithm learns patterns from **untagged(unlabelled) data. This implies that you can train a model to create clusters on any given dataset without having to** initially label the data. (unsupervised)

The intuition behind the algorithm is to divide data points into different pre defined (K) clusters, where a datapoint in each cluster would only belong to that cluster. The cluster would consist of data which share similarities with one another, implying that data points in different clusters would be dissimilar to one another.



Algorithm

- Choose your value of K (elbow method and silhouette)
- Randomly select K data points to represent the cluster centroids
- Assign all other data points to its nearest cluster centroids
- Reposition the cluster centroid until it is the average of the points in the cluster
- Repeat steps 3 & 4 until there are no changes in each cluster

1. **Initialize K & Centroids**

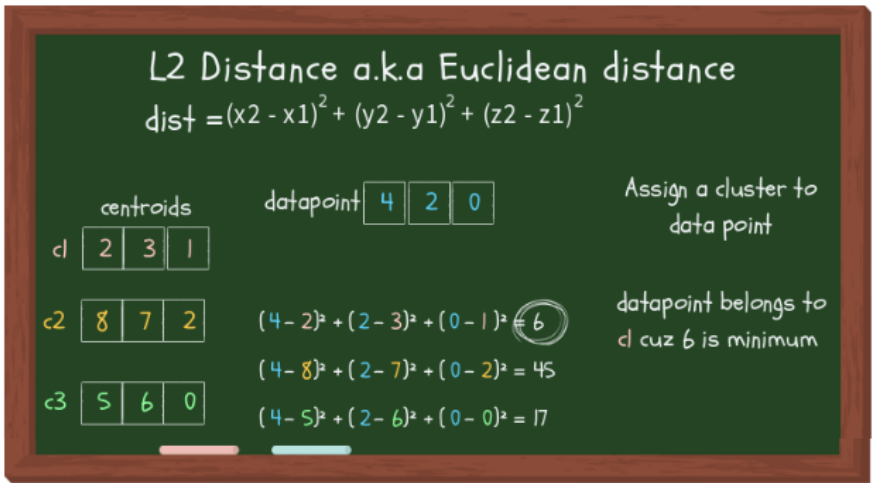
As a starting point, you tell your model how many clusters it should make. First the model picks up K, (let K = 3) datapoints from the dataset. These datapoints are called cluster centroids.

Now there are different ways you to initialize the centroids, you can either choose them at random — or sort the dataset, split it into K portions and pick one datapoint from each portion as a centriod.

2. **Assigning Clusters to datapoints**

From here on wards, the model performs calculations on it's own and assigns a cluster to each datapoint. Your model would calculate the distance between the datapoint & all the centroids, and will be assigned to the cluster with the nearest centroid. Again, there are different ways you can calculate this distance; all having their pros and cons. Usually we use the L2 distance.

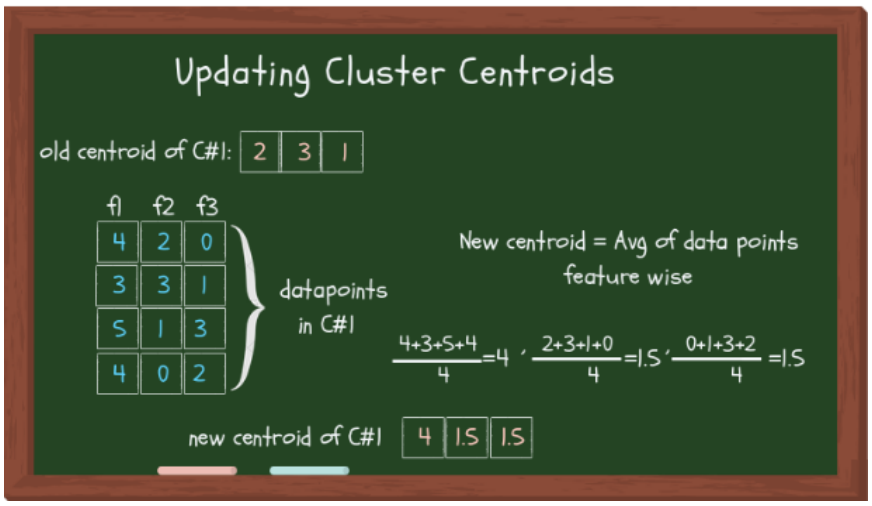
The picture below shows how to calculate the L2 distance between the centeroid and a datapoint. Every time a datapoint is assigned to a cluster the following steps are followed.



L2 or Euclidean distance

3. **Updating Centroids**

Because the initial centroids were chosen arbitrarily, your model the updates them with new cluster values. The new value might or might not occur in the dataset, in fact, it would be a coincidence if it does. This is because the updated cluster centorid is the average or the mean value of all the datapoints within that cluster.



Updating cluster centroids

Now if some other algo, like K-Mode, or K-Median was used, instead of taking the average value, mode and median would be taken respectively.

4. Stopping Criterion

Since step 2 and 3 would be performed iteratively, it would go on forever if we don't set a stopping criterion. The stopping criterion tells our algo when to stop updating the clusters. It is important to note that setting a stopping criterion would not necessarily return THE BEST clusters, but to make sure it returns reasonably good clusters, and more importantly at least return some clusters, we need to have a stopping criterion.

Like everything else, there are different ways to set the stopping criterion. You can even set multiple conditions that, if met, would stop the iteration and return the results. Some of the stopping conditions are:

The datapoints assigned to specific cluster remain the same (takes too much time)

Centroids remain the same (time consuming)

The distance of datapoints from their centroid is minimum (the thresh you've set)

Fixed number of iterations have reached (insufficient iterations → poor results, choose max iteration wisely)

Evaluating the cluster quality

The goal here isn't just to make clusters, but to make good, meaningful clusters. Quality clustering is when the datapoints within a cluster are close together, and afar from other clusters.

The two methods to measure the cluster quality are described below:

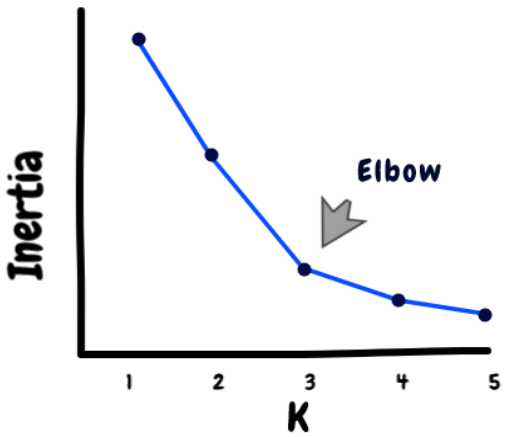
Inertia: Intuitively, inertia tells how far away the points within a cluster are. Therefore, a small of inertia is aimed for. The range of inertia's value starts from zero and goes up.

Silhouette score: Silhouette score tells how far away the datapoints in one cluster are, from the datapoints in another cluster. The range of silhouette score is from -1 to 1. Score should be closer to 1 than -1.

Choosing K (2 ways)

Elbow method :

- The elbow method uses the sum of squared distance (SSE) to choose an ideal value of k based on the distance between the data points and their assigned clusters. We would choose a value of k where the SSE begins to flatten out and we see an inflection point. When visualized this graph would look somewhat like an elbow, hence the name of the method.



Silhouette Analysis:

- Silhouette analysis can be used to determine the degree of separation between clusters. For each sample:

    - Compute the average distance from all data points in the same cluster (ai).
    - Compute the average distance from all data points in the closest cluster (bi).

    Compute the coefficient:

$$\frac{b^i - a^i}{max(a^i, b^i)}$$

    The coefficient can take values in the interval [-1, 1].
    - If it is 0 –> the sample is very close to the neighboring clusters.
    - It it is 1 –> the sample is far away from the neighboring clusters.
    - It it is -1 –> the sample is assigned to the wrong clusters.

    Therefore, we want the coefficients to be as big as possible and close to 1 to have a good clusters.

In theory, k-means++ is much nicer. It is a biased random sampling that prefers points that are farther from each other, and avoids close points. Random initialization may be unlucky and choose nearby centers.

So in theory, k-means++ should require fewer iterations and have a higher chance of finding the global optimum.

K-Means is an iterative clustering method which randomly assigns initial centroids and shifts them to minimize the sum of squares. One problem is that, because the centroids are initially random, a bad starting position could cause the algorithm to converge at a local optimum.(multiple centroids might be close to each other)

K-Means++ was designed to combat this - It chooses the initial centroids using a weighted method which makes it more likely that points further away will be chosen as the initial centroids. The idea is that while initialization is more complex and will take longer, the centroids will be more accurate and thus fewer iterations are needed, hence it will reduce overall time