Beyond K-Means: What Really makes a cluster

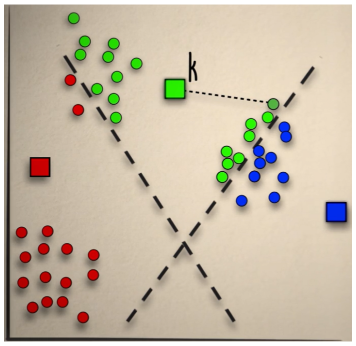
We already know about the clustering problem and the particular algorithm for coming up with the solution to it, which is K-Means algorithm or LLOYD's algorithm.
Now what happens if the output of the K-Means algorithm is not what we expected or wanted?
So, in this article we will discuss how to troubleshoot K-Means and ways to make K-Means clustering better.

- The first thing to do is to make sure we are getting the best possible output from the K-Means algorithm.
- We know the K-Means algorithm eventually stops and there are only finitely many possible clusterings. And the K-Means algorithm moves to a new clustering only if the new clustering decreases the K-Means objective function.
- But just because the algorithm will stop eventually, it does not mean the algorithm will stop quickly.

In practice, the K-Means algorithm tends to be quite fast. But we may find that, in a particular dataset the algorithm is taking a while to run. So, what to do in such a situation?

Various speed ups methods have been proposed in such situations.

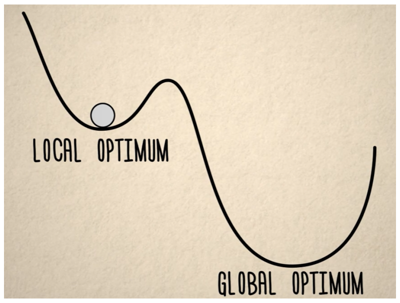
- In the K-Means algorithm we need to calculate the distance from every data point to every cluster center. But, we don't actually need to check every possible cluster center to decide which is the closest cluster for a given data point. Here we can easily observe that the data point is closest to the cluster center in green (Distance is shown in dotted line).



This observation relies on the fact known as the **Triangle Inequality**, that lets us ignore cluster centers that are relatively far away from a given data point in our calculation.

Another related issue is that when the K-Means algorithm stops, it does not necessarily stop at a minimum value of the K-Means objective, which is Global Dissimilarity. When the K-Means algorithm stops at a value that is not the minimum value of the K-Means objective function, then we call it Local Optimum.

But it would be better if we could reach the true minimum value, that is the **Global Optimum**. But that is a very difficult problem which we can't solve in general.



One option is to try to get closer to the global optimum. We can try to do this, by running the K-Means algorithm multiple times for multiple random initialization.

Then we finally take the configuration of cluster centers and cluster assignments with the lowest objective function value, that is the least global dissimilarity.

This process does require a more total computation time, but these multiple runs can be performed completely in parallel, which makes the process less complex.

Another option to get better output from our algorithm is to use **smarter initialization**, like in **K-Means ++**. K-means++ is designed to improve the centroid initialization for k-means. The basic idea is that the initial centroid should be far away from each other. The algorithm starts by randomly choosing a centroid c_0 from all data points. For centroid c_i, the probability of a data point x to be chosen as a centroid is proportional to the squares of the distance of x to its nearest centroid. In this way, k-means++ always tries to select centroids that are far away from the existing centroids, which leads to significant improvement over k-means.This also has the potential to reduce total running time.

Now suppose somehow we had access to the global optimum, that could know the actual set of cluster centers and assignment of data points to clusters that minimize the K-Means Objective function but we could still be dissatisfied with our result. This is a topic which we will see in later articles.The idea here is, in this case the issue is not our K-Means algorithm but the K-Means problem itself.