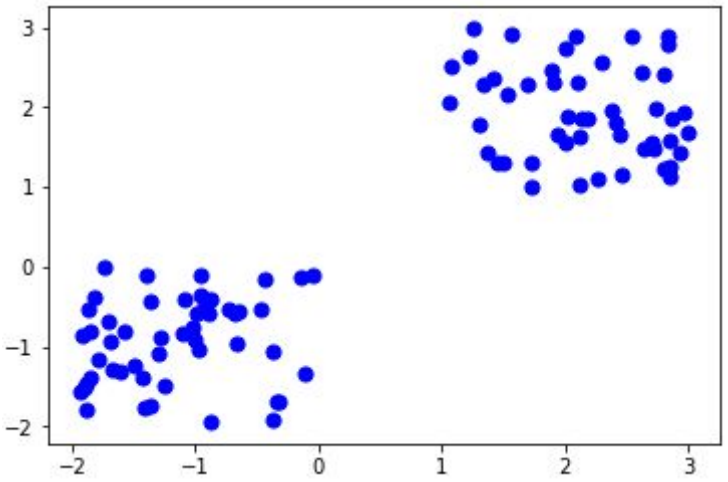
K-means Clustering

Clustering is one of the most widely used exploratory data analysis techniques to get an intuition about the structure or pattern of the data. It can be defined as the task of identifying subgroups in the data such that data points in the same subgroup (cluster) are very similar while data points in different clusters are very different. In other words, we try to find homogeneous subgroups within the data such that data points in each cluster are as similar as possible according to a similarity measure(distance) that suits the application like euclidean-based distance or correlation-based distance.



Unlike supervised learning, clustering is considered an unsupervised learning method because we don't have the ground truth or target label to compare the output of the clustering algorithm to the true target labels to evaluate its performance. We only want to try to investigate the structure of the data by grouping the data points into distinct subgroups.

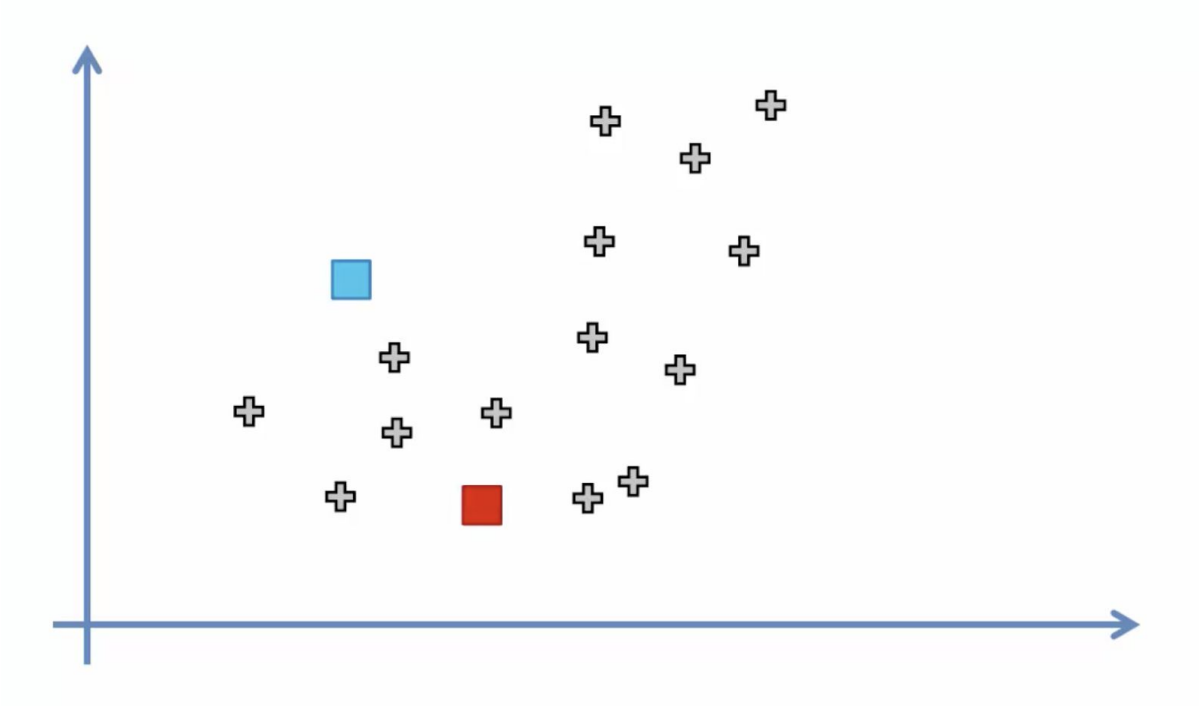
Some use cases of Clustering are:
1)It is used in market segmentation in which we try to find the customers who are having similar to each other which can be in terms of behaviors or attributes
2)Image segmentation/compression in which we try to group similar regions together
3)Document clustering based on topics, etc.

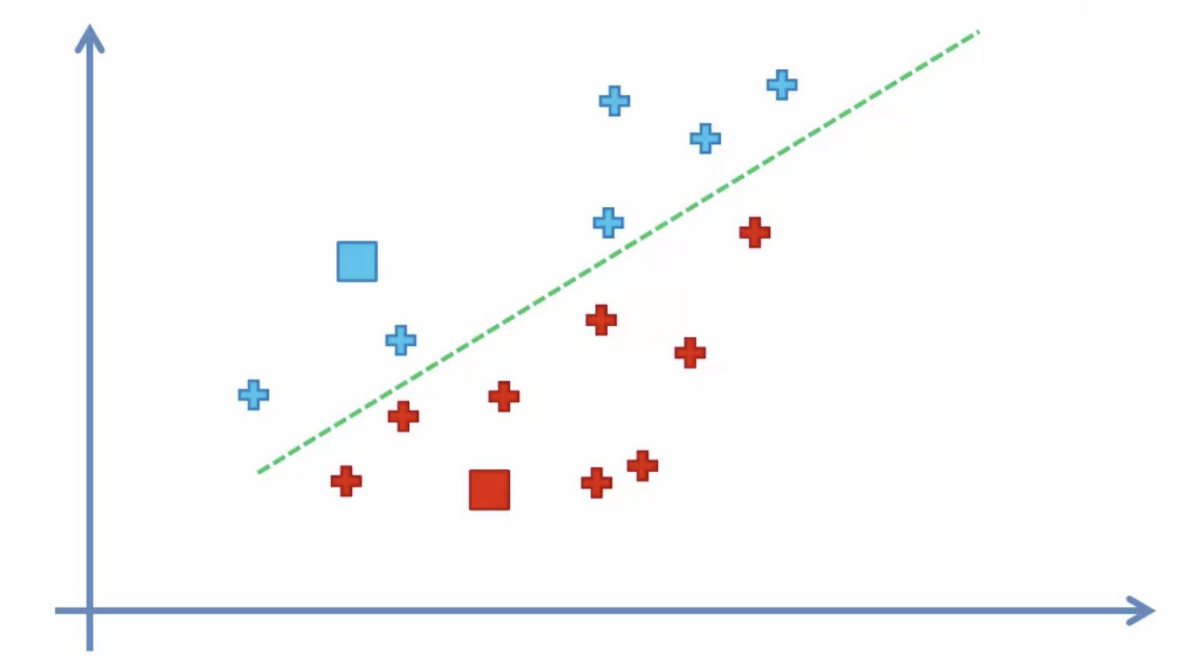
Kmeans algorithm is an iterative algorithm that tries to partition the dataset into *K*pre-defined distinct nonoverlapping subgroups(clusters) where each data point belongs to only one group. It tries to make the inter-cluster data points as similar as possible while also keeping the clusters as different (far) as possible. It assigns data points to a cluster such that the sum of the squared distance between the data points and the cluster's centroid (arithmetic mean of all the data points that belong to that cluster) is at the minimum. The less variation we have within clusters, the more homogeneous (similar) the data points are within the same cluster.

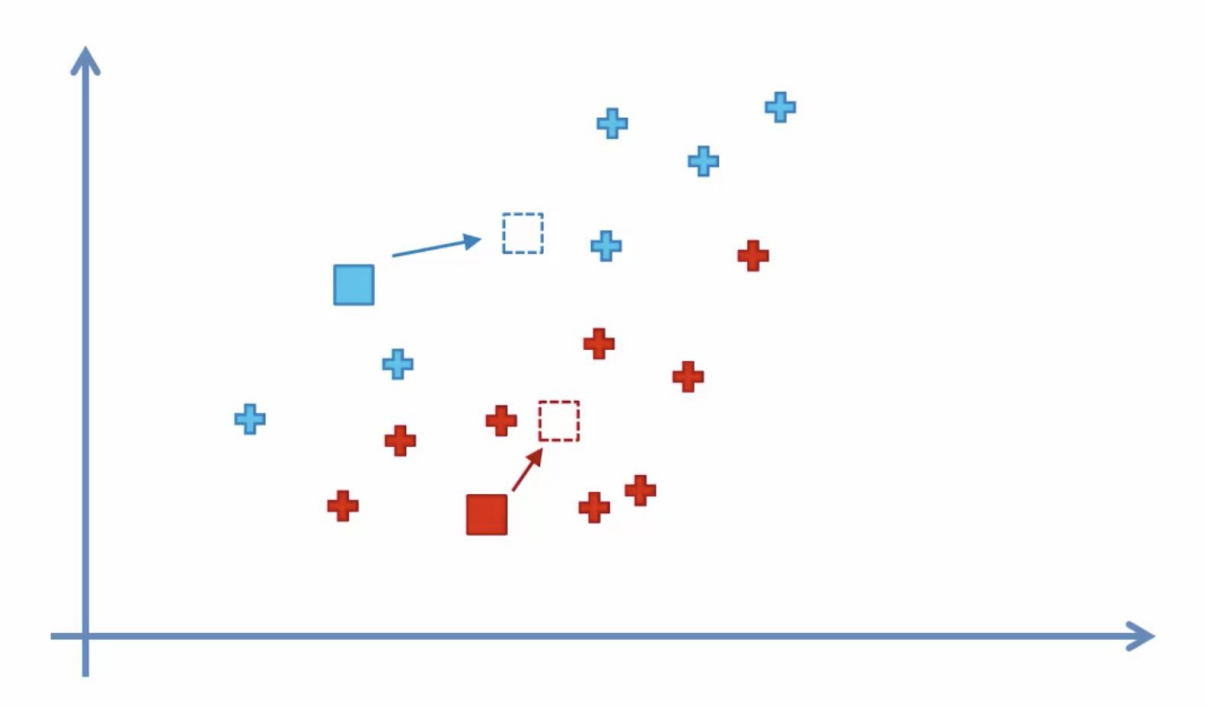
Steps involved in K-means clustering:

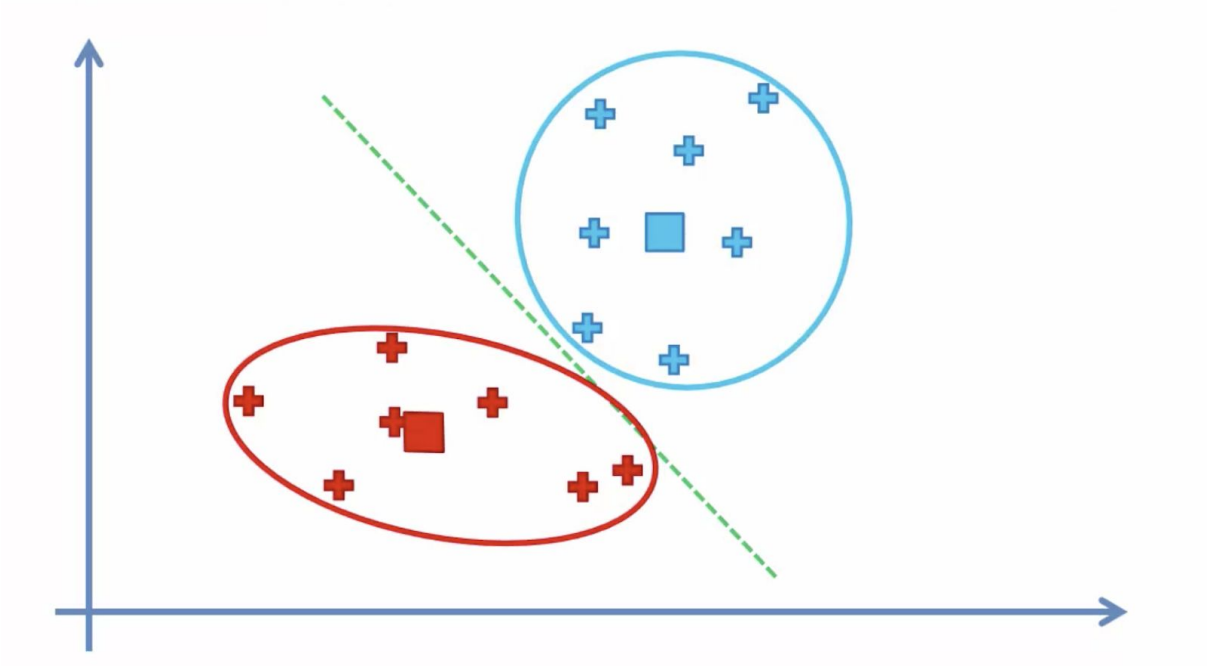
1. Specify the number of clusters *K*.
    a. It may be possible that for your application you have some value in mind.
    b. But if you don't have much clue then maybe you can plot the data and maybe look the plot and chose the value of K accordingly.
2. Then select K random points from the dataset which will be assigned as initial centroids.

3. Now for all the data points in the dataset, find the distance between the data point and all the centroids.
4. Assign each data point to the closest cluster (centroid).
5. Then after doing this for all data points compute the centroids for the clusters by taking the average of all data points that belong to each cluster. So the new calculated centroids will be considered for the next iteration.
6. Keep repeating steps 3 to 5 until there is not much change to the centroids.









Source:
https://towardsdatascience.com/machine-learning-algorithms-part-9-k-means-example-in-python-f2ad05ed5203