2.1. K-mean Clustering

What is clustering problem?

The dictionary meaning of clustering is grouping. In data science, too, clustering is an unsupervised learning technique that helps in grouping our data points.

Clustering refers to a very broad set of techniques for finding subgroups, or clusters, in a data set. When we cluster the observations of a data set, we seek to partition them into distinct groups so that the observations within each group are quite similar to each other, while observations in different groups are quite different from each other.

Clustering looks to find homogeneous subgroups among the observations. In other words, the data points in the same subgroup (cluster) are very similar while data points in different clusters are very different.

Clustering is considered an unsupervised learning method since we don't have the ground truth to compare the output of the clustering algorithm to the true 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.

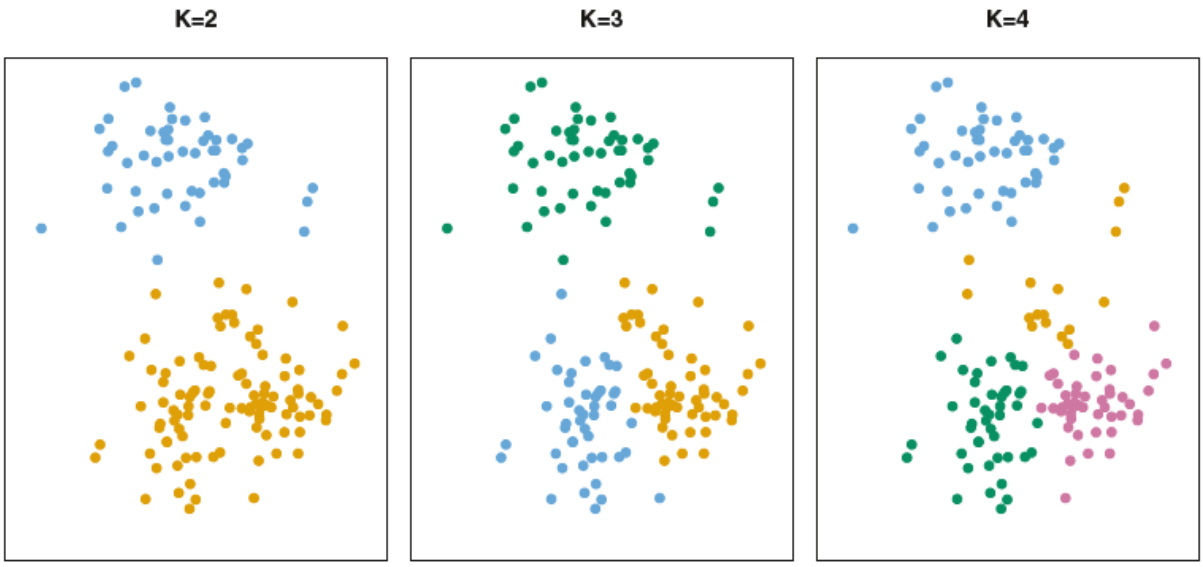
K-Means Clustering

K-means clustering is a simple and elegant iterative algorithm for partitioning a data set into K distinct, non-overlapping clusters. Within the universe of clustering techniques, K-means is probably one of the mostly known and frequently used.

It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as different (far) as possible.

The key objective of a k-means algorithm is to organize data into clusters such that there is high intra-cluster similarity and low inter-cluster similarity. An item will only belong to one cluster, not several.

To perform K-means clustering, we must first specify the desired number of clusters K; then the K-means algorithm will assign each observation to exactly one of the K clusters. The figure below shows the results obtained from performing K-means clustering on a simulated example consisting of 150 observations in two dimensions, using three different values of K.

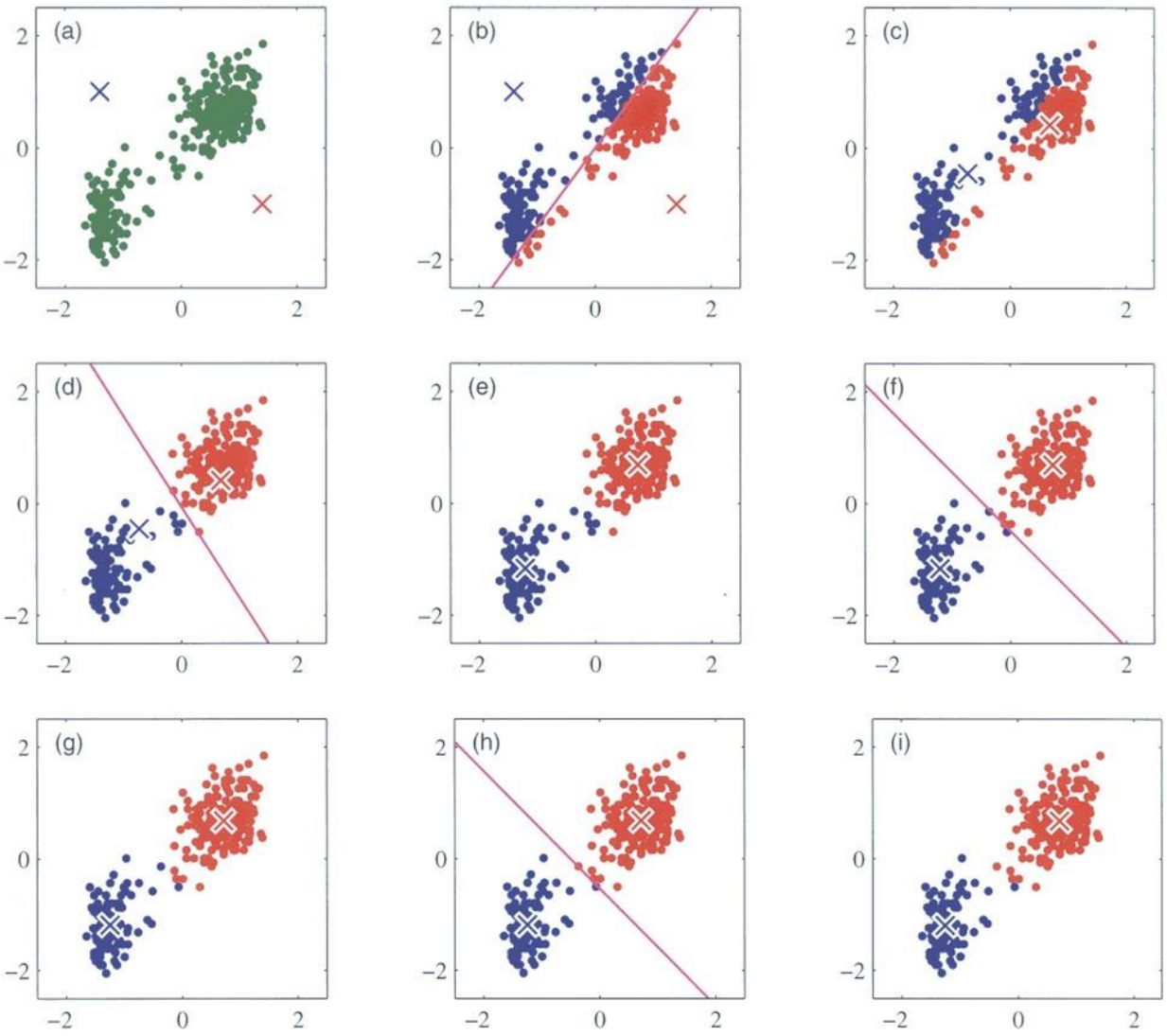


K-means clustering is one of the most common clustering techniques. In K-means clustering, the algorithm attempts to group observations into K groups, with each group having roughly equal variance. The number of groups, K, is specified by the user as a hyperparameter.

Specifically, in K-means we:

1. Specify number of clusters K.
2. K cluster "center" points or centroids are initialized at random locations.
3. For each observation:
    a. The distance between each observation and the K center points is calculated.
    b. The observation is assigned to the cluster of the nearest center point.
4. The center points are moved to the means (i.e., centers) of their respective clusters.
5. Steps 2 and 3 are repeated until no observation changes in cluster membership.

Example:



(a) In initialization, K = 2 (represented with crosses) initial points (centroids) are randomly selected from data set. Currently we are using two dimensional Euclidian space, but similar process can be performed in n-dimensional data set, only difference is the distance function to determine positions between each object in n- dimensional space.

(b) In next step, whole data is assigned into clusters depending on which cluster center is nearest.

(c) After division, each cluster center is re-computed to be the mean of the points assigned to the corresponding cluster

(d) - (h) shows iterative improvement of cluster centers. Algorithm slowly converges into state where cluster means are stable and are no more changing. These are chosen as representative, and clusters are selected accordingly.

The objective of K-mean clustering is to assign 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.

Important things to note in here:

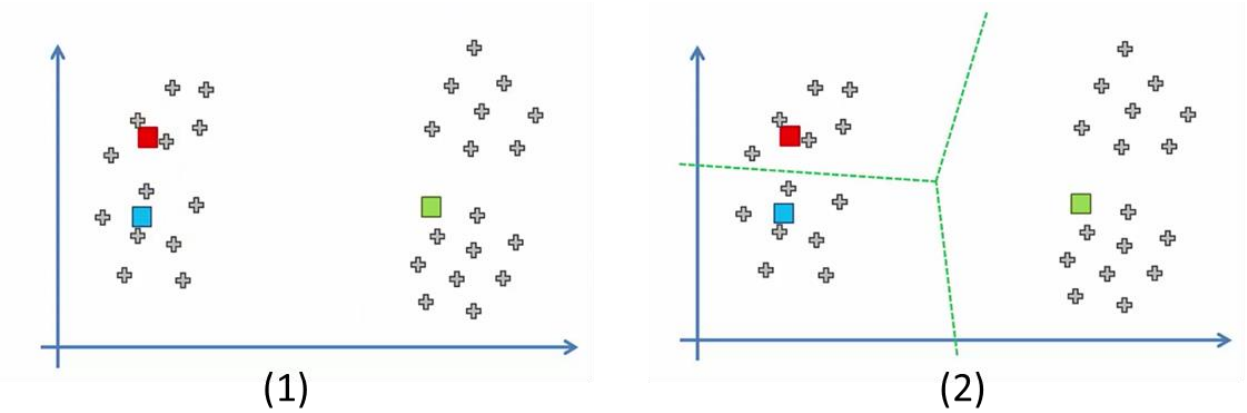
- ✓ Since clustering algorithms including K-means use distance-based measurements to determine the similarity between data points, it's recommended to standardize the data to have a mean of zero and a standard deviation of one since almost always the features in any dataset would have different units of measurements such as age vs income.

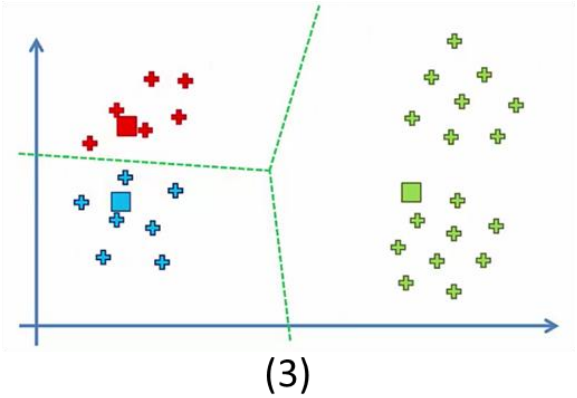
- ✓ Given K-means iterative nature and the random initialization of centroids at the start of the algorithm, different initializations may lead to different clusters. Therefore, it's recommended to run the algorithm using different initializations of centroids and pick the results of the run that that yielded the lower sum of squared distance.

Method of initialization:

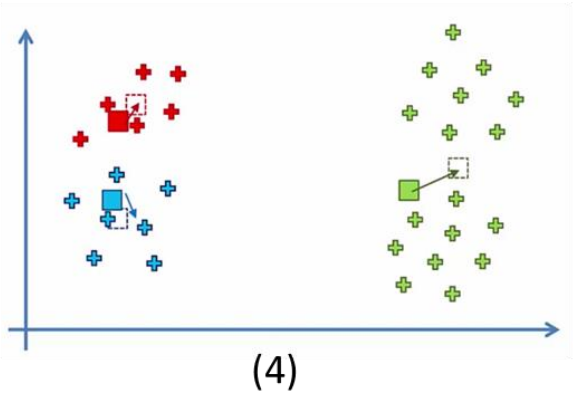
The first cluster center is chosen uniformly at random from the data points that are being clustered, after which each subsequent cluster center is chosen from the remaining data points with probability proportional to its squared distance from the point's closest existing cluster center.

Well, this is also potentially problematic because we might get different clusters every time. In some cases, if the initialization of clusters is not appropriate, K-Means can result in arbitrarily bad clusters. For example, look at the figures below:

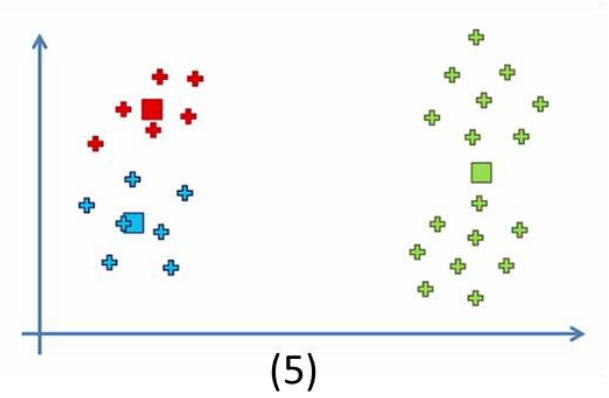




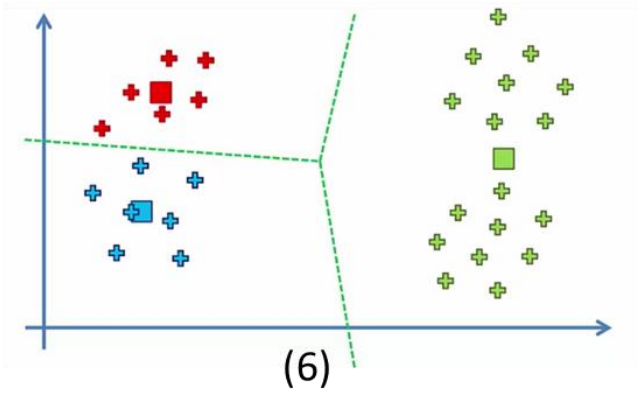
(3)



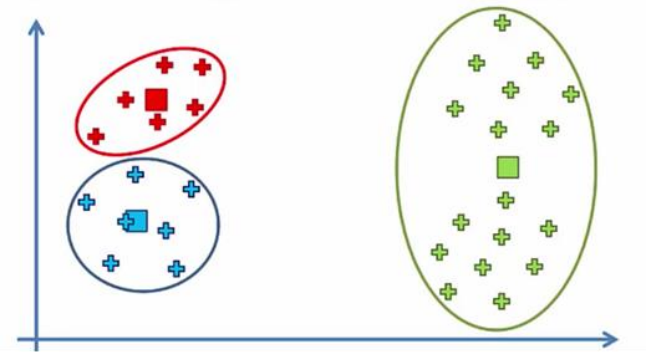
(4)



(5)



(6)



(7)

So, to solve this problem of random initialization, there is an algorithm called K-Means++ that can be used to choose the initial values, or the initial cluster centroids, for K-Means. It specifies a procedure to initialize the cluster centers before moving forward with the standard k-means clustering algorithm.

'random' : choose `n_clusters` observations (rows) at random from data for the initial centroids.

'kmeans++' : selects initial cluster centers for k-mean clustering in a smart way to speed up convergence.

Evaluation Methods: How to choose the K ?

Contrary to supervised learning where we have the ground truth to evaluate the model's performance, clustering analysis doesn't have a solid evaluation metric that we can use to evaluate the outcome of different clustering algorithms.

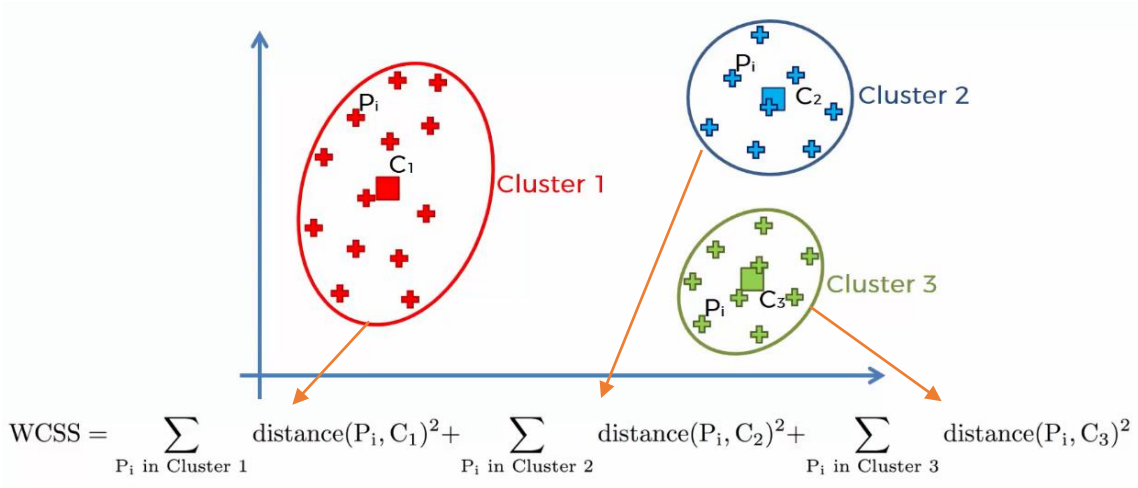
Moreover, since K-means requires K as an input and doesn't learn it from data, there is no right answer in terms of the number of clusters that we should have in any problem. Sometimes domain knowledge and intuition may help but usually that is not the case. In the cluster-predict methodology, we can evaluate how well the models are performing based on different K clusters since clusters are used in the downstream modeling

Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. WCSS stands for *Within Cluster Sum of Squares*, which defines the total variations within a cluster. It is also called as Inertia.

Just like the name suggests, WCSS is the summation of each clusters distance between that specific clusters each points against the cluster centroid. The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$\text{WCSS}= \sum_{P_i \text{ in Cluster1}} \text{distance}(P_i\, C_1)^2 + \sum_{P_i \text{ in Cluster2}} \text{distance}(P_i\, C_2)^2 + \sum_{P_i \text{ in CLuster3}} \text{distance}(P_i\, C_3)^2$$



$$\text{WCSS} = \sum_{P_i \text{ in Cluster 1}} \text{distance}(P_i, C_1)^2 + \sum_{P_i \text{ in Cluster 2}} \text{distance}(P_i, C_2)^2 + \sum_{P_i \text{ in Cluster 3}} \text{distance}(P_i, C_3)^2$$

In the above formula of WCSS,

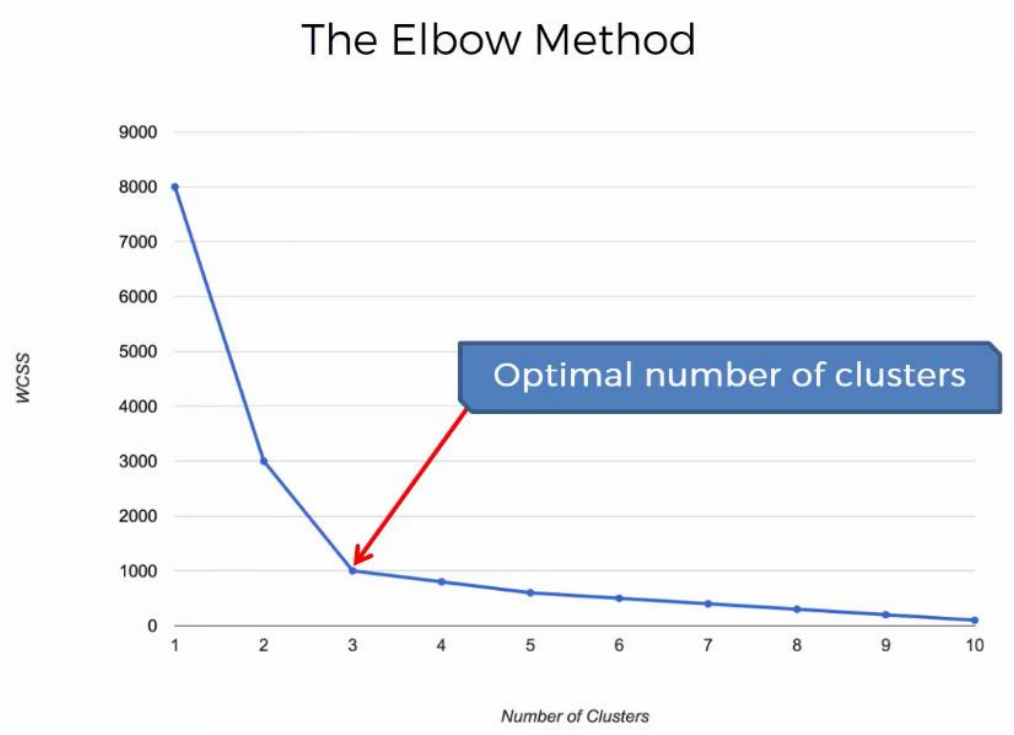
$\sum P_{i \text{ in Cluster 1}}$ distance $(P_i\, C_1)^2$ is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

1. It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
2. For each value of K, calculates the WCSS value.
3. Plots a curve between calculated WCSS values and the number of clusters K.
4. The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



We pick K at the spot where WCSS starts to flatten out and forming an elbow. So when we observe the above image, we can clearly see that when x - axis is on 3, graph clearly has an elbow. Finding the optimal number of clusters using the elbow of the graph is called as the Elbow method. Which says,

The cluster value where this decrease in inertia value becomes constant can be chosen as the right cluster value for our data.

Silhouette Analysis

Silhouette analysis can be used to determine the degree of separation between clusters.

For each sample:

1. Compute the average distance from all data points in the same cluster (a_i).
2. Compute the average distance from all data points in the closest cluster (b_i).
3. Compute the coefficient using:

$$\frac{b_i - a_i}{\max(a_i, b_i)}$$

The coefficient can take values in the interval [-1, 1].

- ✓ If it is 0 –> the sample is very close to the neighboring clusters.
- ✓ If it is 1 –> the sample is far away from the neighboring clusters.
- ✓ If it is -1 –> the sample is assigned to the wrong clusters.

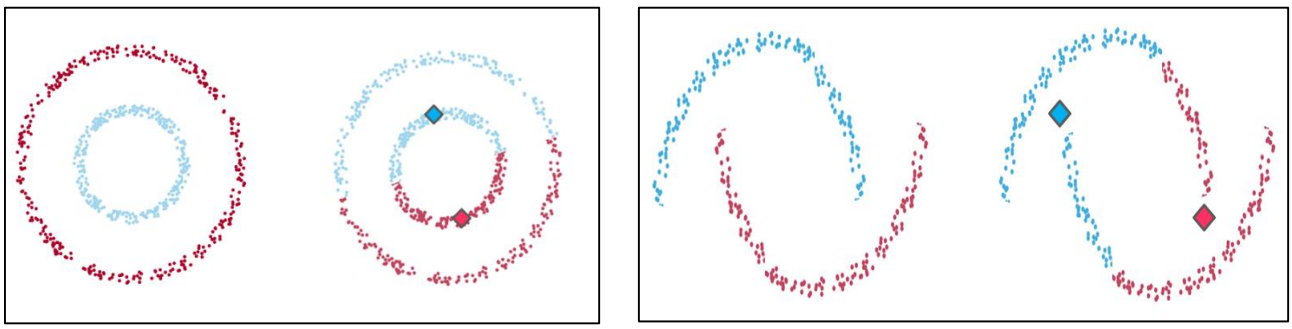
Therefore, *we want the coefficients to be as big as possible and close to 1 to have a good clusters*.

The bottom line is: Good `n_clusters` will have a well above 0.5 silhouette average score as well as all of the clusters have higher than the average score.

Limitations of K-means:

1. The first one is that you need to define the number of clusters, and this decision can seriously affect the results. Also, as the location of the initial centroids is random, results may not be comparable and show lack of consistency.

2. Additionally, it assumes that data points in each cluster are modeled as located within a sphere around that cluster centroid (spherical limitation), but when this condition (or any of the previous ones) is violated, the algorithm can behave in non-intuitive ways. Following are some examples, where K-mean algorithms fails to identify the intuitive clustering.



On the left-hand side the intuitive clustering of the data, with a clear separation between two groups of data points (in the shape of one small ring surrounded by a larger one). On the right-hand side, the

same data points clustered by K-means algorithm (with a K value of 2), where each centroid is represented with a diamond shape. As you see, the algorithm fails to identify the intuitive clustering.

3. If there is heavy overlapping between clusters, K-means doesn't have an intrinsic measure for uncertainty for the examples belong to the overlapping region in order to determine for which cluster to assign each data point.

4. Kmeans gives more weight to the bigger clusters.

Applications

Clustering is used in a wide variety of applications, including:

- For customer segmentation: you can cluster your customers based on their purchases, their activity on your website, and so on. This is useful to understand who your customers are and what they need, so you can adapt your products and marketing campaigns to each segment. For example, this can be useful in recommender systems to suggest content that other users in the same cluster enjoyed.

- For data analysis: when analyzing a new dataset, it is often useful to first discover clusters of similar instances, as it is often easier to analyze clusters separately.

- As a dimensionality reduction technique: once a dataset has been clustered, it is usually possible to measure each instance's affinity with each cluster (affinity is any measure of how well an instance fits into a cluster). Each instance's feature vector x can then be replaced with the vector of its cluster affinities. If there are k clusters, then this vector is k dimensional. This is typically much lower dimensional than the original feature vector, but it can preserve enough information for further processing.

- For search engines: for example, some search engines let you search for images that are similar to a reference image. To build such a system, you would first apply a clustering algorithm to all the images in your database: similar images would end up in the same cluster. Then when a user provides a reference image, all you need to do is to find this image's cluster using the trained clustering model, and you can then simply return all the images from this cluster.

- To segment an image: by clustering pixels according to their color, then replacing each pixel's color with the mean color of its cluster, it is possible to reduce the number of different colors in the image considerably. This technique is used in many object detection and tracking systems, as it makes it easier to detect the contour of each object.