K-Means Clustering

Clustering is one of the most common EDA technique used to get an intuition about the structure of the data.

It can be defined as the task of identifying sub-groups / clusters in the data such that data points in the same cluster are very similar while data points in different clusters are very different.

Similarity measure → euclidean distance, correlation based distance.

clustering → based on features, based on samples.

Note: clustering is an unsupervised technique since we don't have the ground truth to compare its output.

We only want to try to investigate the structure of the data by grouping the data points into distinct subgroups / clusters.

K-Means algo. is an iterative algo. that tries to partition the dataset into (K) pre-defined distinct non-overlapping clusters where each data point belongs to only one group.

It tries to make the intra-cluster data points as similar as possible while also keeping the clusters as 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 in that cluster) is at the minimum.

Note: The less variation we have within clusters the more homogeneous / similar the data points are within the same cluster.

Scanned with CamScanner

Algo:

① specify beforehand the value of Ⓚ.

② initialize centroids by shuffling the dataset and then randomly selecting k-data points for the centroids without replacement.

③ keep iterating until there is no change to the centroids i.e. assignment of data points to clusters isn't changing.

④ Compute the sum of the squared error b/w data points and all centroids.

⑤ Assign each data point to the closest cluster (centroid)

⑥ Compute the centroids for the clusters by taking the average of all the data points belonging to that cluster.

The approach used by k-means algo. to solve the problem is known as "Expectation - Maximization".

E → assigning data points to the closest cluster.

M → computing centroid of each cluster.

---

objective function:

$$J = \sum_{i=1}^{m} \sum_{k=1}^{K} w_{ik} \cdot \| x^i - \mu_k \|^2$$

μ_k → centroid of cluster of x^i.

$\Rightarrow w_{ik} = 1$ (if point x^i belongs to cluster k)

else $= 0$

Note:

① Given K-means is iterative in nature and there is random initialization of centroids at the start of the algo, diff. initializations may lead to diff. clusters since K-mean algo may get stuck in a local optimum and may not converge to global optimum.

② Assignment of examples isn't changing is the same thing as no change in within-cluster variation.

Applications:

→ Market segmentation
→ document clustering
→ Image segmentation
→ Image compression.

→ Cluster then predict: Diff. models could be built for diff. clusters if we believe there is a wide variation in the behaviors of diff. subgroups.

Ex: clustering patients into diff. clusters and build a model for each cluster to predict the prob. of the risk of having heart attack.

Evaluation methods:

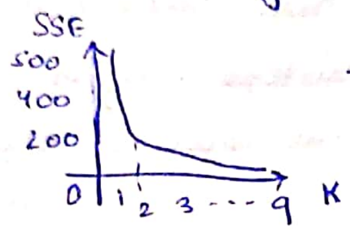
→ No ground truth values.
→ Since K is an input and isn't learnt from the data, there is no right answer in terms of no. of clusters.

⇒ Elbow method.
⇒ Silhouette analysis.

Elbow Method:

Elbow method gives us an idea on what a good (K) no. of clusters would be based on SSE between data points and their assigned cluster's centroid! (sum of sq.)

We pick 'K' at the spot where SSE starts to flatten out and start forming an elbow.



→ The graph shows that $k=2$ is not a bad choice. Still it is hard to figure out a good no. of clusters because the curve is monotonically decreasing.

→ It may not show any elbow or any point where curve starts flattening out.

Silhouette Analysis:

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

For each sample:

① Compute avg. distance from all data points in the same cluster → a^i

② Compute avg distance from all data point in the closest cluster → b^i

③ Compute: $c = \dfrac{b^i - a^i}{\max(a^i, b^i)}$ → $[-1, 1]$

if $c = 0$ (sample is very close to neighbouring clusters)

if $c = 1$ (sample is far away from neigh. clusters)

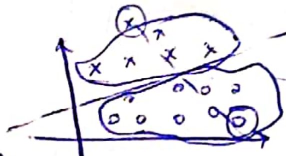
if $c = -1$ (sample is assigned to wrong cluster)

So, we want the coeff. to be as big as possible and close to 1. to have good no. of clusters.

→ K-means does a very good job when the clusters have kind of spherical shapes.

→ K means give more weight to bigger clusters.

Since K-means tries to minimize the within-cluster variation, it gives more weight to bigger clusters than smaller ones. In other words, data points in smaller clusters may be left away from the centroid in order to focus more on the larger cluster.

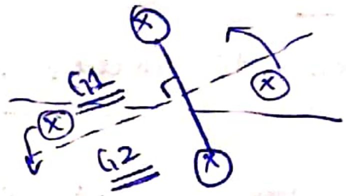


Distance calculation

↙ ↘

Euclidean — Manhattan

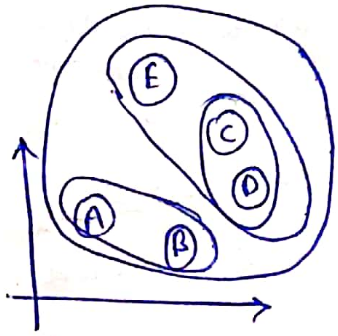
$\boxed{K=2}$



$$WCSS = \sum_{i=1}^{n} (c_i - x_i)^2$$

↓

within cluster Sum of squares.



Hierarchical clustering:

↙ ↘

Agglomerative

start with individual and keep on forming clusters bigg

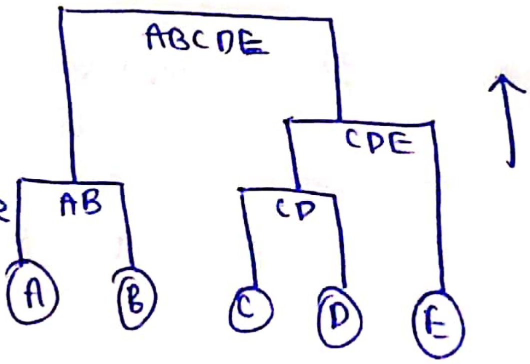
small to big

[bottom to top].

Divisive

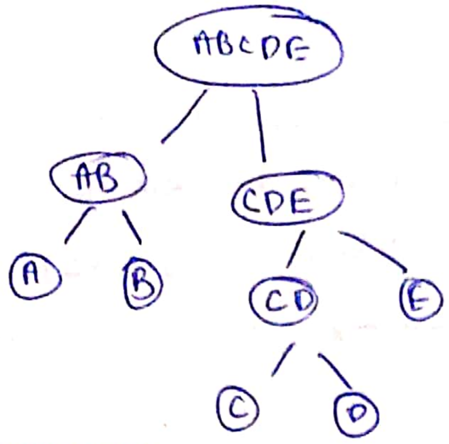
opp. to agglomerative

[top to bottom]



[Dendogram]

divisive



Agglomerative Hierar. Clustering:

In this technique initially each data point is considered as an individual cluster.

At each iteration, similar clusters merge with other clusters until 1 or K clusters are formed.

Algo: (Agglomerative)

1. Compute proximity matrix
2. Let each data point be a cluster.
3. Repeat: merge the two closest clusters and update the proximity matrix.

Key operation is the computation of the proximity of two clusters

Note: The hierarchical clustering Technique can be visualized using a dendrogram.

A dendrogram is a tree like structure that records the sequences of merges or splits.

Divisive Hierarchical Clustering:

Opp. of Agglomerative.

How to calculate the similarity between 2 clusters.

→ Approaches:

- → min
- → max
- → group avg.
- → distance between centroids
- → ward's method

① **MIN:** also known as single-linkage algo

similarity of two clusters = min. (similarity between points P_i and P_j : $P_i \in C_1$, $P_j \in C_2$)

——— x ——— x ——— x ———