K-MEANS
LBG

BY :

Naman Anand (200101070)

Paide Ashish (200101072)

***k*-means clustering :**

- method of vector quantization, originally from signal processing, that aims to partition *n* observations into *k* clusters in which each observation belongs to a cluster with the nearest centroid , serving as a prototype of the cluster.
- **RESULT :** Voronoi diagram (PARTITION OF PLANE INTO REGIONS CLOSE TO EACH OF A GIVEN SET OF OBJECTS)

k-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular Euclidean distances, which would be the more difficult Weber problem: the mean optimizes squared errors, whereas only the geometric median minimizes Euclidean distances. For instance, better Euclidean solutions can be found using *k*-medians and *k*-medoids.

Minimize the within-cluster sum of squares (WCSS)

The Euclidean squared distance metric makes use of the same equation as the Euclidean distance metric, but it does not take the square root. Because of this, clustering can be performed at a faster pace with the Euclidean Squared Distance Metric than it can be carried out with the regular Euclidean distance.
Even if you replace the Euclidean distance with the Euclidean squared distance metric, the output of Jarvis-Patrick clustering and of K-Means clustering will not be affected. But if you do this, the output of hierarchical clustering will be very likely to change.

> Computationally difficult (NP HARD); BUT efficient heuristics converge quickly to a local optimum

- It is equivalent to minimizing the pairwise squared deviations of points in the same cluster.
- Since the total variance is constant, this is equivalent to maximizing the sum of squared deviations between points in *different* clusters (between-cluster sum of squares, BCSS).
- This deterministic relationship is also related to the LAW OF TOTAL VARIENCE in probability theory.

Naïve Kmeans :

1. Also called as Lloyd's Algorithm
2. Initial set of means -> Assign nearest centroid -> Update centroid : Until Assignments don't change
3. The algorithm is not guaranteed to find the optimum
4. Other Modifications : Spherical kmeans and k mediods

K medoids :

K- Medoids is more robust as compared to K-Means as in K-Medoids we find k as representative object to minimize the sum of dissimilarities of data objects whereas, K-Means used sum of squared Euclidean distances for data objects

The main disadvantages of K-Medoids algorithm are that **it is not suitable for clustering a non-spherical group of object**

K-medoids is better at scalability for the larger dataset and also due to it being more efficient than K-means. It was also found that K-medoids showed its superiority over k means in execution time and to reduce the noise since it employs the method of minimization of the sum of dissimilarities of datasets, because it relies on minimizing the distances between the non-medoids objects and the medoids briefly.

INITIALIZATION METHODS :

- **Forgy method** randomly chooses *k* observations from the dataset and uses these as the initial means.

  Some Iterations : Good Points, Bad points

  Indication of **Good starting point** : Starting points are already in the respective clusters and are hence close to the true centroids. k-Means is most likely to converge to the global optimum in a few iterations whereas poor points : may result in the algorithm reaching a local optimum with a poor solution

  Sol : If we run k-Means from different initial configurations which will yield the global optimum

- **Random Partition method** we randomly assign each point in the data to a random cluster ID. Then, we group the points by their cluster ID and take the average (per cluster ID) to yield the initial points. Random Partition method is known to yield initial points close to the mean of the Data. **k-Means is more likely to get stuck in Local Optima if initialized using this method.**
- **SO FORGY METHOD > RANDOM PARTITION METHOD**

STANDARD METHOD : KMEANS++

Idea : to choose Initial points as far as possible

Choose a random point from the data -> Choose the next point such that is more probable to lie at a large distance from the first point. We do so by sampling a point from a probability distribution that is proportional to the squared distance of a point from the first center -> The remaining points are generated by a probability distribution that is proportional to the squared distance of each point from its closest center. So, a point having a large distance from its closest center is more likely to be sampled. This method of generating initial points helps k-Means converge to the Global Minimum in just a few iterations. **This is the recommended method to generate initial points for k-Means Algorithm.**

Even though we got great initial points by running kmeans++, it is still recommended to run k-Means from different starting points. Even though k-Means is a relatively simple algorithm that makes a lot of assumptions, it is still very useful in a variety of settings due to its speed, scalability and ease of interpretation.

Lloyd's algorithm

- spends a lot of processing time computing the distances between each of the *k* cluster centers and the *n* data points.
- Since points usually stay in the same clusters after a few iterations, much of this work is unnecessary, making the naïve implementation very inefficient.
- Some implementations use caching and the triangle inequality in order to create bounds and accelerate Lloyd's algorithm

Complexity [edit]

Finding the optimal solution to the *k*-means clustering problem for observations in *d* dimensions is:

- NP-hard in general Euclidean space (of *d* dimensions) even for two clusters,[15][16]
- NP-hard for a general number of clusters *k* even in the plane,[17]
- if *k* and *d* (the dimension) are fixed, the problem can be exactly solved in time $O(n^{dk+1})$, where *n* is the number of entities to be clustered.[18]

Thus, a variety of heuristic algorithms such as Lloyd's algorithm given above are generally used.

The running time of Lloyd's algorithm (and most variants) is $O(nkdi)$,[9][19] where:

- *n* is the number of *d*-dimensional vectors (to be clustered)
- *k* the number of clusters
- *i* the number of iterations needed until convergence.

CLUSTERING EVALUATION :

Three important factors by which clustering can be evaluated are

*(a) Clustering tendency (b) Number of clusters, **k** (c) Clustering quality*

Hopkin test for clustering Tendency :
A statistical test for spatial randomness of a variable, can be used to measure the probability of data points generated by uniform data distribution. If H>0.5, null hypothesis can be rejected and it is very much likely that data contains clusters. If H is more close to 0, then data set doesn't have clustering tendency.

No of Clusters :
If *k* is too high, each point will broadly start representing a cluster and if *k* is too low, then data points are incorrectly clustered. Finding the optimal number of clusters leads to granularity in clustering. There are two major approaches to find optimal number of clusters:
(1) Domain knowledge : // like iris -> k = 3
(2) Data driven approach

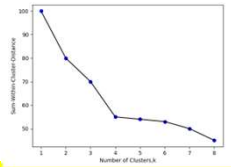
Data Driven Approach :

Empirical Method:-

A simple empirical method of finding number of clusters is Square root of N/2 where N is total number of data points, so that each cluster contains square root of 2 * N

Elbow method:-

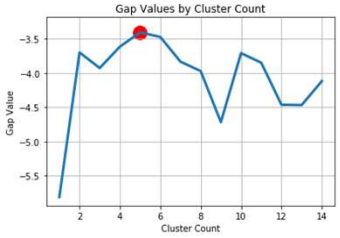
Within-cluster variance is a measure of compactness of the cluster. Lower the value of within cluster variance, higher the compactness of cluster formed . Sum of within-cluster variance, *W*, is calculated for clustering analyses done with different values of k. *W* is a cumulative measure how good the points are clustered in the analysis. Plotting the *k* values and their corresponding sum of within-cluster variance helps in finding the number of clusters.



Statistical approach:-

Similar to Elbow method, sum of within-cluster (intra-cluster) variance is calculated for different values of k. Then Random data points from reference null distribution are generated and Sum of within-cluster variance is calculated for the clustering done for different values of k.

In Simpler words, Sum-of-within-Cluster variance of original data set for different values of k to Sum-of-within-cluster variance of reference data set (null reference data set of uniform distribution) of corresponding values of k is compared to find the ideal *k* value where 'deviation' or 'Gap' between two is highest. As Gap statistic quantifies this deviation, More the Gap statistic means more the deviation.



CLUSTERING QUALITY

- Ideal clustering is characterized by minimal intra cluster distance and maximal inter cluster distance.
- *Extrinsic Measures* which require ground truth labels. Examples are Adjusted Rand index, Fowlkes-Mallows scores, Mutual information based scores, Homogeneity, Completeness and V-measure.
- *Intrinsic Measures* that does not require ground truth labels. Some of the clustering performance measures are Silhouette Coefficient etc.
- EXTRINSIC MEASURES : reqr actual classification

1. RAND INDEX : **Rand index** is a function that measures the **similarity** of the two assignments GIVE THE TRUE VALUE , THE VALUE WE GET. IT WILL COMPARE AND TELL
   a : no of pairs that belong to same cluster and are in same cluster
   b : no of pairs that belong to diff cluster and are in diff cluster

- a, the number of pairs of elements that are in the same set in C and in the same set in K
- b, the number of pairs of elements that are in different sets in C and in different sets in K

The unadjusted Rand index is then given by:

$$\mathrm{RI} = \frac{a+b}{C_2^{n_{samples}}}$$

- INTRINSIC MEASURES :

1. SILHOUTTE COEFFICIENT : Higher Silhouette Coefficient score relates to a model with better defined clusters.

a: The mean distance between a sample and all other points in the same class.
b: The mean distance between a sample and all other points in the *next nearest cluster*.

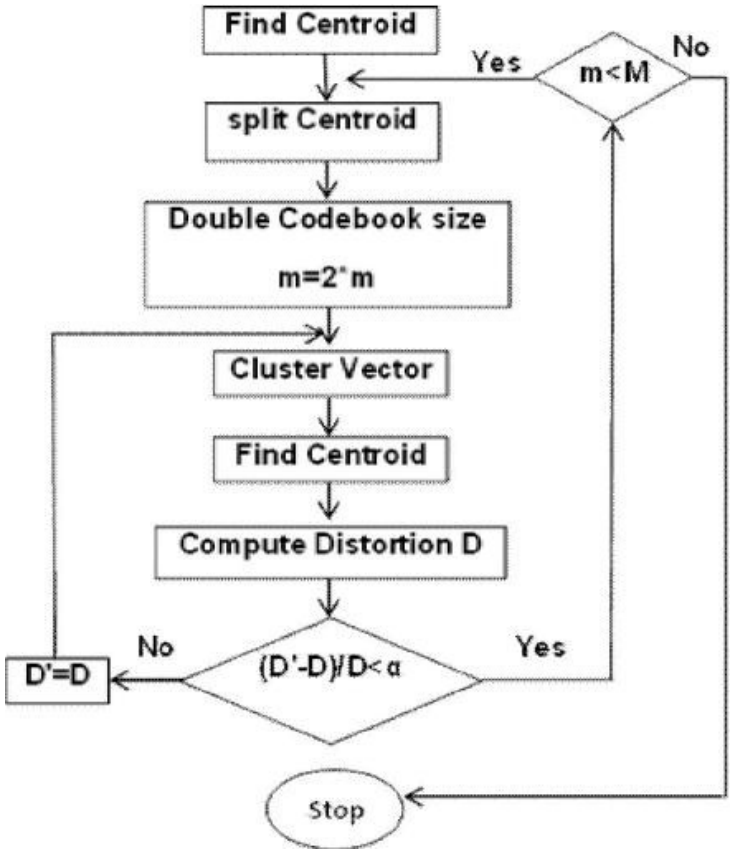
$$s = \frac{b-a}{max(a,b)}$$

•The score is bounded between -1 for incorrect clustering and +1 for highly dense clustering. Scores around zero indicate overlapping clusters.
•The score is higher when clusters are dense and well separated, which relates to a standard concept of a cluster.
•The score -1 show incorrect clustering

Linde Buzo Gray (LBG) :

Used For Designing Codebook with minimum distortion and error.

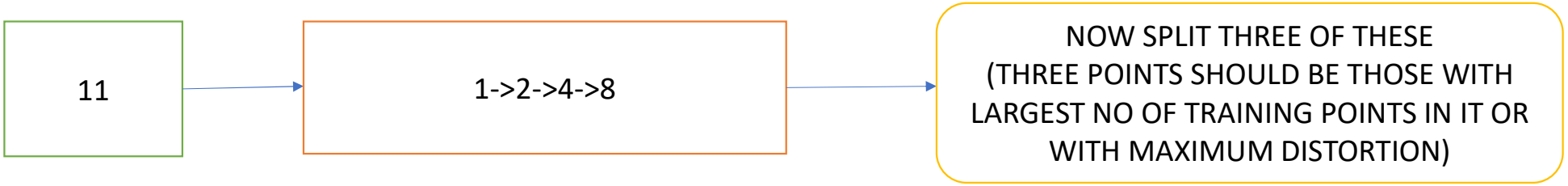
It is an iterative procedure and the basic idea is to divide the group of training vectors and use it to find the most representative vector from one group



Linde, Buzo, and Gray described a technique in their original paper [125] called the *splitting technique* for initializing the design algorithm. In this technique, we begin by designing a vector quantizer with a single output point; in other words, a codebook of size one, or a one-level vector quantizer. With a one-element codebook, the quantization region is the entire input space, and the output point is the average value of the entire training set. From this output point, the initial codebook for a two-level vector quantizer can be obtained by including the output point for the one-level quantizer and a second output point obtained by adding a fixed perturbation vector ϵ. We then use the LBG algorithm to obtain the two-level vector quantizer. Once the algorithm has converged, the two codebook vectors are used to obtain the initial codebook of a four-level vector quantizer. This initial four-level codebook consists of the two codebook vectors from the final codebook of the two-level vector quantizer and another two vectors obtained by adding ϵ to the two codebook vectors. The LBG algorithm can then be used until this four-level quantizer converges. In this manner we keep doubling the number of levels until we reach the desired number of levels. By including the final codebook of the previous stage at each "splitting," we guarantee that the codebook after splitting will be at least as good as the codebook prior to splitting.

VERY IMPORTANT

If the desired number of levels is not a power of two, then in the last step, instead of generating two initial points from each of the output points of the vector quantizer designed previously, we can perturb as many vectors as necessary to obtain the desired number of vectors. For example, if we needed an eleven-level vector quantizer, we would generate a one-level vector quantizer first, then a two-level, then a four-level, and then an eight-level vector quantizer. At this stage, we would perturb only three of the eight vectors to get the eleven initial output points of the eleven-level vector quantizer. The three points should be those with the largest number of training set vectors, or the largest distortion.



generate different vector quantizers. A good approach to codebook design is to initialize the codebook randomly several times, and pick the one that generates the least distortion in the training set from the resulting quantizers.

In 1989, Equitz [127] introduced a method for generating the initial codebook called the *pairwise nearest neighbor* (PNN) algorithm. In the PNN algorithm, we start with as many clusters as there are training vectors and end with the initial codebook. At each stage, we combine the two closest vectors into a single cluster and replace the two vectors by their mean. The idea is to merge those clusters that would result in the smallest increase in distortion. Equitz showed that when we combine two clusters C_i and C_j, the increase in distortion is

$$\frac{n_i n_j}{n_i + n_j} \| Y_i - Y_j \|^2, \tag{10.5}$$

where n_i is the number of elements in the cluster C_i, and Y_i is the corresponding output point. In the PNN algorithm, we combine clusters that cause the smallest increase in the distortion.

As the size of the training set increases, this procedure becomes progressively more time-consuming. In order to avoid this cost, we can use a fast PNN algorithm that does not attempt to find the absolute smallest cost at each step

We have looked at four different ways of initializing the LBG algorithm. Each has its own advantages and drawbacks. The PNN initialization has been shown to result in better designs, producing a lower distortion for a given rate than the splitting approach [127]. However, the procedure for obtaining the initial codebook is much more involved and complex. We cannot make any general claims regarding the superiority of any one of these initialization techniques. Even the PNN approach cannot be proven to be optimal. In practice, if we are dealing with a wide variety of inputs, the effect of using different initialization techniques appears to be insignificant.

EMPTY CELL PROBLEM

- When we assign the inputs to the initial output points if no input point gets assigned to the output point. This is a problem because in order to update an output point, we need to take the average value of the input vectors.
- There is a danger that we will end up with an output point that is never used. A common approach to avoid this is to remove an output point that has no inputs associated with it, and replace it with a point from the quantization region with the most output points. This can be done by selecting a point at random from the region with the highest population of training vectors, or the highest associated distortion.
- A more systematic approach is to design a two-level quantizer for the training vectors in the most heavily populated quantization region. This approach is computationally expensive and provides no significant improvement over the simpler approach

Sources :

1> Wikipedia
2> Data compression book by Khalid Sayood
3> GFG
4>https://www.kdnuggets.com/2020/06/centroid-initialization-k-means-clustering.html#:~:text=k%2Dmeans%2B%2B%3A%20As%20spreading,probability%20proportional%20to%20the%20squared
AND MANY MORE….