# Clustering

Clustering is one of the most common exploratory data analysis technique used to get an intuition about the structure of the data. It can 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 such as Euclidean-based distance or correlation-based distance. The decision of which similarity measure to use is application-specific.

Clustering analysis can done based on features where we try to find subgroups of samples based on features or based on samples where we try to find subgroups of features based on samples. Clustering used in market segmentation; where we try to fined customers that are similar to each other whether in terms of behaviors or attributes, image segmentation/compression; where we try to group similar regions together, document clustering based on topics, etc.

Unlike supervised learning, clustering is considered an unsupervised learning method since we do not 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 Algorithm

K-Means clustering is one of the simplest and popular unsupervised machine learning algorithms. K-Means algorithm is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping 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.

**The way K-Means algorithm works is as follows:**

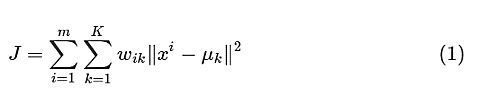
1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids without replacement. This involves following steps

* Compute the sum of the squared distance between data points and all centroids.
* Assign each data point to the closest cluster (centroid).

1. Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.
2. Keep iterating step 3 & 4 until there is no change to the centroids. i.e assignment of data points to clusters isn’t changing.

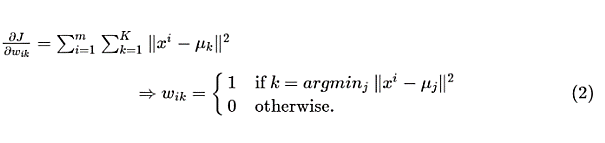
The approach kmeans follows to solve the problem is called Expectation-Maximization. The E-step is assigning the data points to the closest cluster. The M-step is computing the centroid of each cluster. Below is a break down of how we can solve it mathematically (feel free to skip it).

The objective function is:

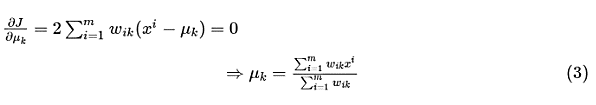


where wik=1 for data point xi if it belongs to cluster k; otherwise, wik=0. Also, μk is the centroid of xi’s cluster.

It’s a minimization problem of two parts. We first minimize J w.r.t. wik and treat μk fixed. Then we minimize J w.r.t. μk and treat wik fixed. Technically speaking, we differentiate J w.r.t. wik first and update cluster assignments. Then we differentiate J w.r.t. μk and recompute the centroids after the cluster assignments from previous step. Therefore,



In other words, assign the data point xi to the closest cluster judged by its sum of squared distance from cluster’s centroid.



Which translates to recomputing the centroid of each cluster to reflect the new assignments.

Few things to note 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 since K-Means algorithm may stuck in a local optimum and may not converge to global optimum. 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.
* Assignment of examples isn’t changing is the same thing as no change in within-cluster variation:



## ****Drawback of standard K-means algorithm:****

One disadvantage of the K-means algorithm is that it is sensitive to the initialization of the centroids or the mean points. Therefore, if a centroid is initialized to be a “far-off” point, it might just end up with no points associated with it and at the same time more than one clusters might end up linked with a single centroid. Similarly, more than one centroids might be initialized into the same cluster resulting in poor clustering.   
A poor initialization of centroids resulted in poor clustering.

|  |  |
| --- | --- |
|  |  |

## K-mean++

To overcome the above-mentioned drawback we use K-means++. This algorithm ensures a smarter initialization of the centroids and improves the quality of the clustering. Apart from initialization, the rest of the algorithm is the same as the standard K-means algorithm. That is K-means++ is the standard K-means algorithm coupled with a smarter initialization of the centroids.

### Steps involved

* Randomly select the first centroid from the data points.
* For each data, point compute its distance from the nearest, previously chosen centroid.
* Select the next centroid from the data points such that the probability of choosing a point as centroid is directly proportional to its distance from the nearest, previously chosen centroid. (i.e. the point having maximum distance from the nearest centroid is most likely to be selected next as a centroid)
* Repeat steps 2 and 3 until k centroids have been sampled

### Intuition

By following the above procedure for initialization, we pick up centroids, which are far away from one another. This increases the chances of initially picking up centroids that lie in different clusters. In addition, since centroids are picked up from the data points, each centroid has some data points associated with it at the end.

## Evaluation Methods

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 kmeans 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
* Silhouette analysis

### Elbow Method

Elbow method gives us an idea on what a good k number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters’ centroids. We pick k at the spot where SSE starts to flatten out and forming an elbow. We’ll use the geyser dataset and evaluate SSE for different values of k and see where the curve might form an elbow and flatten out.

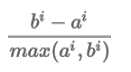
|  |  |
| --- | --- |
|  |  |

The graph above shows that k=2 is not a bad choice. Sometimes it’s still hard to figure out a good number of clusters to use because the curve is monotonically decreasing and may not show any elbow or has an obvious point where the curve starts flattening out.

### Silhouette Analysis

Silhouette analysis can used to determine the degree of separation between clusters. For each sample:

* Compute the average distance from all data points in the same cluster (ai).
* Compute the average distance from all data points in the closest cluster (bi).
* Compute the coefficient:



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

* If it is 0 –> the sample is very close to the neighboring clusters.
* It it is 1 –> the sample is far away from the neighboring clusters.
* It 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. We’ll use here geyser dataset again because it’s cheaper to run the silhouette analysis and it is actually obvious that there is most likely only two groups of data points.

## Takeaways

* Scale/standardize the data when applying K-Means algorithm.
* Elbow method in selecting number of clusters does not usually work because the error function is monotonically decreasing for all k’s.
* K-Means gives more weight to the bigger clusters.
* K-Means assumes spherical shapes of clusters (with radius equal to the distance between the centroid and the furthest data point) and does not work well when clusters are in different shapes such as elliptical clusters.
* If there is 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.
* K-Means may still cluster the data even if it cannot clustered such as data that comes from uniform distributions.

# Hierarchical Clustering

Hierarchical clustering, also known as hierarchical cluster analysis, is an algorithm that groups similar objects into groups called clusters. The endpoint is a set of clusters, where each cluster is distinct from each other cluster, and the objects within each cluster are broadly similar to each other.

Hierarchical clustering involves creating clusters that have a predetermined ordering from top to bottom. For example, all files and folders on the hard disks are organized in a hierarchy. There are two types of hierarchical clustering, Divisive and Agglomerative.

## Agglomerative Hierarchical clustering

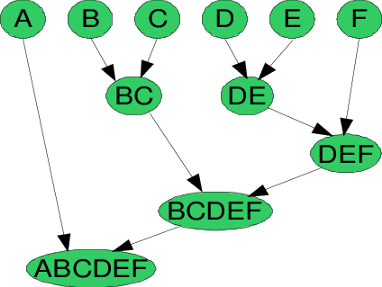
In this technique, initially each data point is considered as an individual cluster. At each iteration, the similar clusters merge with other clusters until one cluster or K clusters are formed.

The basic algorithm of Agglomerative is straightforward.

* Compute the proximity matrix
* Let each data point be a cluster
* Repeat: Merge the two closest clusters and update the proximity matrix
* Until only a single cluster remains
* Key operation is the computation of the proximity of two clusters

To understand better let us see a pictorial representation of the Agglomerative Hierarchical clustering Technique. Let’s say we have six data points {A,B,C,D,E,F}.

* Step- 1: In the initial step, we calculate the proximity of individual points and consider all the six data points as individual clusters as shown in the image below.

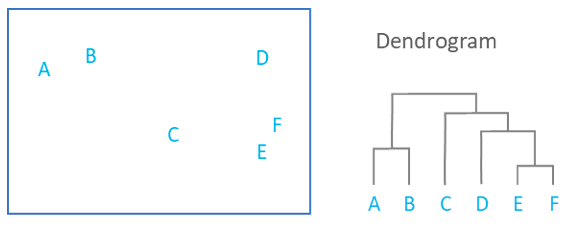


* Step- 2: In step two, similar clusters are merged together and formed as a single cluster. Let’s consider B,C, and D,E are similar clusters that are merged in step two. Now, we’re left with four clusters which are A, BC, DE, F.
* Step- 3: We again calculate the proximity of new clusters and merge the similar clusters to form new clusters A, BC, DEF.
* Step- 4: Calculate the proximity of the new clusters. The clusters DEF and BC are similar and merged together to form a new cluster. We’re now left with two clusters A, BCDEF.
* Step- 5: Finally, all the clusters are merged together and form a single cluster.

The Hierarchical clustering Technique can visualize using a Dendrogram. A Dendrogram is a tree-like diagram that records the sequences of merges or splits.

### Dendrogram

A dendrogram is a diagram that shows the hierarchical relationship between objects. It is most commonly created as an output from hierarchical clustering. The main use of a dendrogram is to work out the best way to allocate objects to clusters. The dendrogram below shows the hierarchical clustering of six observations shown to on the scatterplot to the left. (Dendrogram is often miswritten as Dendogram.)

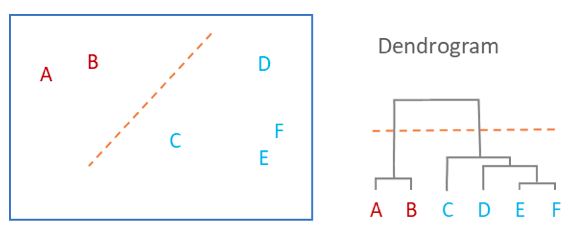


The key to interpreting a dendrogram is to focus on the height at which any two objects are joined together. In the example above, we can see that E and F are most similar, as the height of the link that joins them together is the smallest. The next two most similar objects are A and B.

In the dendrogram above, the height of the dendrogram indicates the order in which the clusters were joined. A more informative dendrogram can be created where the heights reflect the distance between the clusters as is shown below. In this case, the dendrogram shows us that the big difference between clusters is between the cluster of A and B versus that of C, D, E, and F.

It is important to appreciate that the dendrogram is a summary of the distance matrix, and, as occurs with most summaries, information is lost. For example, the dendrogram suggests that C and D are much closer to each other than is C to B, but the original data (shown in the scatterplot), shows us that this is not true. To use some jargon, a dendrogram is only accurate when data satisfies the ultrametric tree inequality, and this is unlikely for any real-world data.

The consequence of the information loss is that the Dendrograms are most accurate at the bottom, showing which items are very similar. Observations are allocated to clusters by drawing a horizontal line through the dendrogram. Observations that are joined together below the line are in clusters. In the example below, we have two clusters, one that combines A and B, and a second combining C, D, E, and F.



Dendrograms cannot tell you how many clusters you should have. A common mistake people make when reading Dendrograms is to assume that the shape of the Dendrogram gives a clue as to how many clusters exist. In the example above, the (incorrect) interpretation is that the Dendrogram shows that there are two clusters, as the distance between the clusters (the vertical segments of the Dendrogram) are highest between two and three clusters.

Such an interpretation is justified only when the Ultrametric tree inequality holds, which, as mentioned above, is very rare. In general, it is a mistake to use Dendrograms as a tool for determining the number of clusters in data. Where there is an obviously “correct” number of clusters this will often be evident in a Dendrogram. However, Dendrograms often suggest a correct number of clusters when there is no real evidence to support the conclusion.

## Divisive Hierarchical clustering

Since the Divisive Hierarchical clustering Technique is not much used in the real world, Here is a brief of the Divisive Hierarchical clustering Technique.

In simple words, we can say that the Divisive Hierarchical clustering is exactly the opposite of the Agglomerative Hierarchical clustering. In Divisive Hierarchical clustering, we consider all the data points as a single cluster and in each iteration, we separate the data points from the cluster which are not similar. Each data point which is separated is considered as an individual cluster. In the end, we’ll be left with n clusters. As we’re dividing the single clusters into n clusters, it is named as Divisive Hierarchical clustering.

## Measures of distance (similarity)

In the example above, the distance between two clusters has been computed based on length of the straight line drawn from one cluster to another. This is commonly referred to as the Euclidean distance. Many other distance metrics have been developed.

The choice of distance metric should be made based on theoretical concerns from the domain of study. That is, a distance metric needs to define similarity in a way that is sensible for the field of study. For example, if clustering crime sites in a city, city block distance may be appropriate (or, better yet, the time taken to travel between each location). Where there is no theoretical justification for an alternative, the Euclidean should generally be preferred, as it is usually the appropriate measure of distance in the physical world. There are certain approaches which are used to calculate the similarity between two clusters:

* MIN
* MAX
* Group Average
* Distance Between Centroids
* Ward’s Method

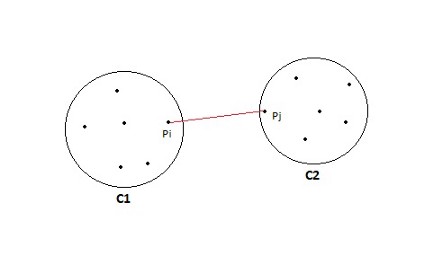
#### MIN

Also known as single linkage algorithm can be defined as the similarity of two clusters C1 and C2 is equal to the minimum of the similarity between points Pi and Pj such that Pi belongs to C1 and Pj belongs to C2.

Mathematically this can be written as,

Sim(C1,C2) = Min Sim(Pi,Pj) such that Pi ∈ C1 & Pj ∈ C2

In simple words, pick the two closest points such that one point lies in cluster one and the other point lies in cluster 2 and take their similarity and declare it as the similarity between two clusters.



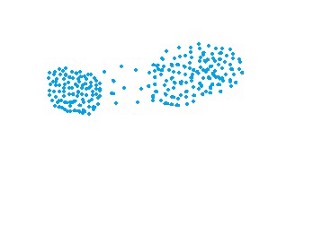
##### Pros of MIN

This approach can separate non-elliptical shapes as long as the gap between two clusters is not small.

|  |  |
| --- | --- |
| https://miro.medium.com/max/442/1*C-BNL_MDzJ6uXKgvhBy0Pg.jpeg | https://miro.medium.com/max/330/1*dcukVWU3Ny2VhhUug5J0rg.jpeg |

##### Cons of MIN

MIN approach cannot separate clusters properly if there is noise between clusters.



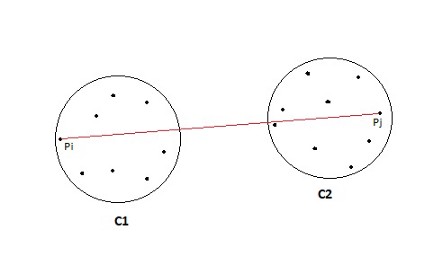
#### MAX

Also known as the complete linkage algorithm, this is exactly opposite to the **MIN** approach. The similarity of two clusters C1 and C2 is equal to the **maximum** of the similarity between points Pi and Pj such that Pi belongs to C1 and Pj belongs to C2.

Mathematically this can be written as,

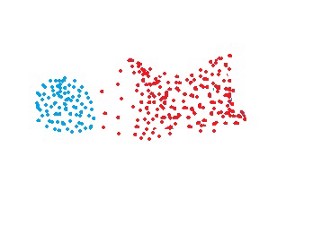
Sim(C1,C2) = Max Sim(Pi,Pj) such that Pi ∈ C1 & Pj ∈ C2

In simple words, pick the two farthest points such that one point lies in cluster one and the other point lies in cluster 2 and take their similarity and declare it as the similarity between two clusters.



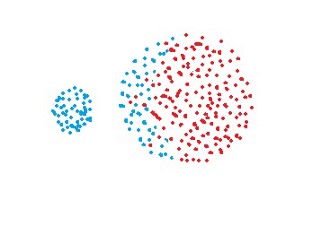
##### Pros of MAX

MAX approach does well in separating clusters if there is noise between clusters.



##### Cons of Max

* Max approach is biased towards globular clusters.
* Max approach tends to break large clusters.



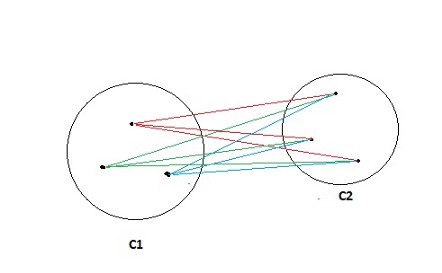
#### Group Average

Take all the pairs of points and compute their similarities and calculate the average of the similarities.

Mathematically this can be written as,

sim(C1,C2) = [∑](https://en.wiktionary.org/wiki/%E2%88%91) sim(Pi, Pj)/|C1|\*|C2|

where, Pi ∈ C1 & Pj ∈ C2



##### Pros of Group Average

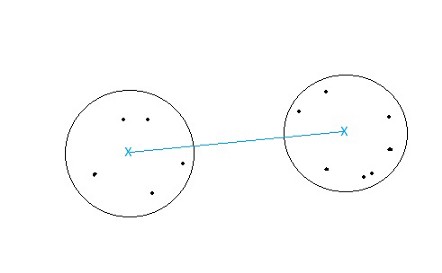
The group Average approach does well in separating clusters if there is noise between clusters.

##### Cons of Group Average

The group Average approach is biased towards globular clusters.

Distance between centroids

Compute the centroids of two clusters C1 & C2 and take the similarity between the two centroids as the similarity between two clusters. This is a less popular technique in the real world.



#### Ward’s Method

This approach of calculating the similarity between two clusters is the same as Group Average except that Ward’s method calculates the sum of the square of the distances Pi and PJ.

Mathematically this can be written as,

sim(C1,C2) = [∑](https://en.wiktionary.org/wiki/%E2%88%91) (dist(Pi, Pj))²/|C1|\*|C2|

##### Pros of Ward’s method

Ward’s method approach also does well in separating clusters if there is noise between clusters.

##### Cons of Ward’s method

Ward’s method approach is also biased towards globular clusters.

#### Space complexity

The space required for the Hierarchical clustering Technique is very high when the number of data points are high as we need to store the similarity matrix in the RAM. The space complexity is the order of the square of n.

Space complexity = O(n²) where n is the number of data points.

#### Time complexity

Since we’ve to perform n iterations and in each iteration, we need to update the similarity matrix and restore the matrix, the time complexity is also very high. The time complexity is the order of cube of n.

Time complexity = O(n³) where n is the number of data points.

## Linkage Criteria

After selecting a distance metric, it is necessary to determine from where distance is computed. For example, it can be computed between the two most similar parts of a cluster (single-linkage), the two least similar bits of a cluster (complete-linkage), the center of the clusters (mean or average-linkage), or some other criterion. Many linkage criteria have been developed.

As with distance metrics, the choice of linkage criteria should be made based on theoretical considerations from the domain of application. A key theoretical issue is what causes variation. For example, in archeology, we expect variation to occur through innovation and natural resources, so working out if two groups of artifacts are similar may make sense based on identifying the most similar members of the cluster.

Where there are no clear theoretical justifications for choice of linkage criteria, Ward’s method is the sensible default. This method works out which observations to group based on reducing the sum of squared distances of each observation from the average observation in a cluster. This is often appropriate as this concept of distance matches the standard assumptions of how to compute differences between groups in statistics (e.g., ANOVA, MANOVA).

## Limitations of Hierarchical clustering Technique:

* There is no mathematical objective for Hierarchical clustering.
* All the approaches to calculate the similarity between clusters has its own disadvantages.
* High space and time complexity for Hierarchical clustering. Hence this clustering algorithm cannot be used when we have huge data.