**Machine Learning Algorithms(14) — K-Means Clustering and Hierarchical Clustering**

[[Kasun Dissanayake](https://kasunprageethdissanayake.medium.com/?source=post_page-----46acc005057d--------------------------------)](https://kasunprageethdissanayake.medium.com/?source=post_page-----46acc005057d--------------------------------)

[[Towards Dev](https://towardsdev.com/?source=post_page-----46acc005057d--------------------------------)](https://towardsdev.com/?source=post_page-----46acc005057d--------------------------------)

[Kasun Dissanayake](https://kasunprageethdissanayake.medium.com/?source=post_page-----46acc005057d--------------------------------)

·

[Follow](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F4a3ca0851dfe&operation=register&redirect=https%3A%2F%2Ftowardsdev.com%2Fmachine-learning-algorithms-14-k-means-clustering-and-hierarchical-clustering-46acc005057d&user=Kasun+Dissanayake&userId=4a3ca0851dfe&source=post_page-4a3ca0851dfe----46acc005057d---------------------post_header-----------)

Published in

[Towards Dev](https://towardsdev.com/?source=post_page-----46acc005057d--------------------------------)

·

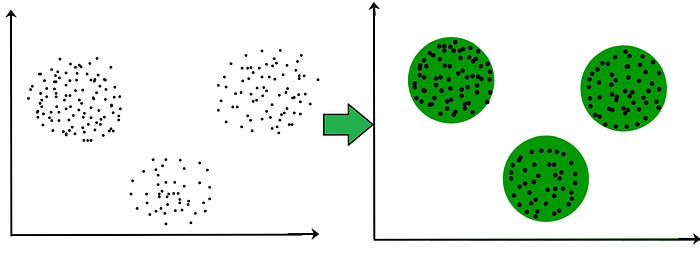
19 min read

·

4 days ago

458

1



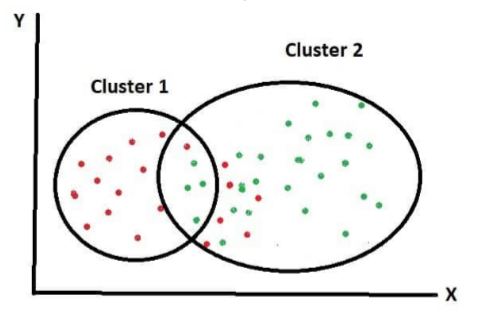
Inthis article, we are learning about K-means Clustering and Hierarchical Clustering. In Unsupervised ML, clustering is the most important thing that revolutionized machine learning. Clustering involves grouping similar data based on different features. There are various techniques for clustering, and one of them is known as **K-means clustering**. This technique involves plotting data points and grouping them into clusters based on their similarities.

**K-Means Clustering**

In unsupervised machine learning, clustering is a technique used to group similar data. The goal is to create clusters based on different data inputs, without any specific output in mind. One of the popular ways to cluster data is through K-means clustering.

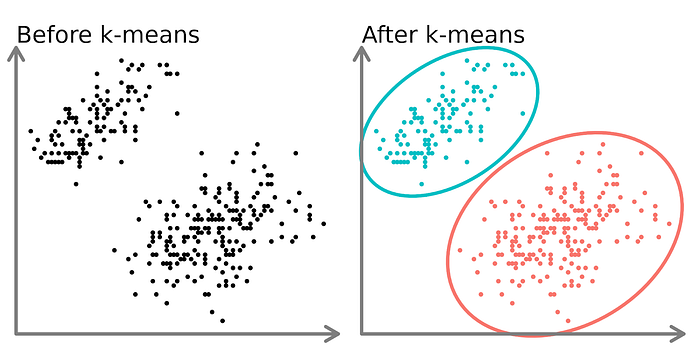


Let’s talk about simple data points. Imagine you have a set of data points with two features, **F1** and **F2**. We can plot the points in a two-dimensional space based on these features. Our objective is to group similar data points. From the clusters, let’s assume we can see that we have two groups. **The first group is made up of data points with similar characteristics, and the second group consists of data points with similar characteristics**. This is where clustering comes into play. We will explore why clustering is beneficial and delve into the mathematical intuition behind it. Additionally, we will examine the various uses of clustering.



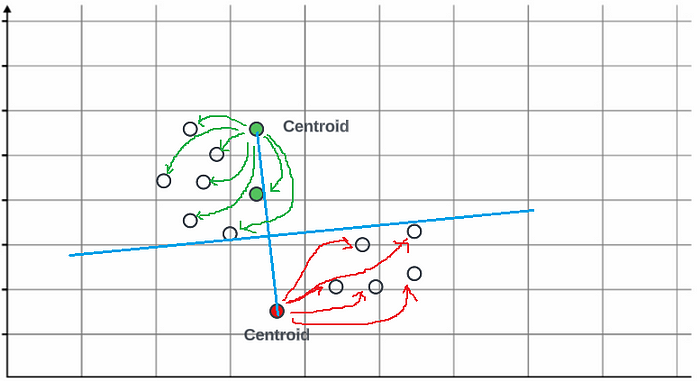
Groups made up of data points with similar characteristics

When creating a **custom ensemble model**, we usually start by clustering our data set. This involves applying a clustering algorithm to group our data into two or three separate clusters. Once we have our clusters, we can apply a **regression or classification algorithm** to each group to solve a specific output. To understand this process better, it’s important to focus on how the k-means clustering algorithm works. K means how many clusters you have. If we have two groups(clusters) then the **k will be 2**.



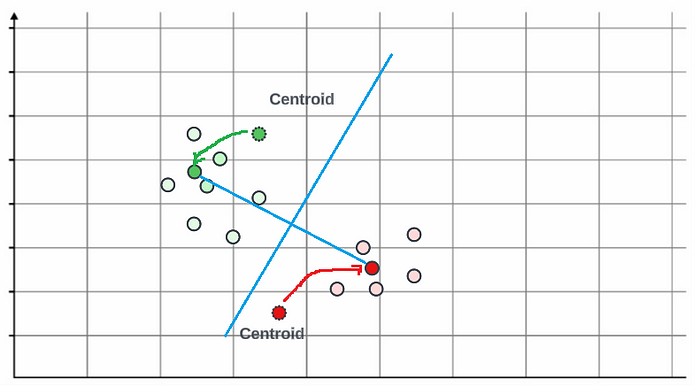
Each group has a **centroid**. If we have a data set like this in the picture, we can easily see that there are two separate groups because there are two dimension data. However, remember that real-world data sets will be much more complex due to **high dimensionality**. Then you will not see the data plotted. How do you conclude that only two groups are there? For that, we should follow some steps.

* First off, we need to **experiment with different K values**. We can achieve this by testing with various **centroids** until we find the appropriate **K value**. Assuming we have a particular data point, we can start by testing with K values of one, two, or three. In this case, let’s assume we start with **K = 2**. How did we arrive at this value? One crucial concept to consider is the **within-cluster sum of squares**. The within-cluster sum of squares is a measure of the variability of the observations within each cluster. In general, a cluster that has a small sum of squares is more compact than a cluster that has a large sum of squares. Clusters that have higher values exhibit greater variability of the observations within the cluster.
* In this case, we are working with two centroids. We will be initializing them randomly and marking them with different colors. After initializing the centroids, we need to**identify which points are closer to each centroid**. This can be done by calculating the distance between the points. To make it easier to visualize, we can draw a straight line connecting two centroids and a perpendicular line to the straight line. All the points that are closer to a particular centroid will be marked with the same color. Once we have identified all the points, we can calculate the summation of all the values using **Euclidean distance**. This will help us determine which centroid each point belongs to.



Points closer to the centroid are marked with the same color

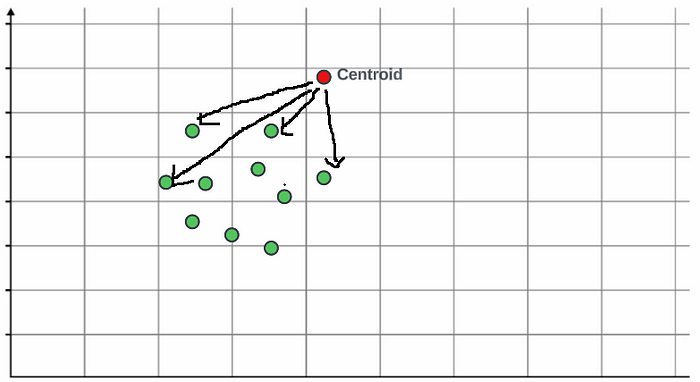
* In the third step, we will compute the **average of all the points** in the specific cluster(red or green). This is necessary because we need to update the centroids to reflect the new location of the points. Once we compute the average, the**centroid will move to a new location** which is somewhere in the center. We will calculate the distance and perpendicular line again, and you can see that all the points are now towards the centroid again and clustered from the right and the left side of the perpendicular line.



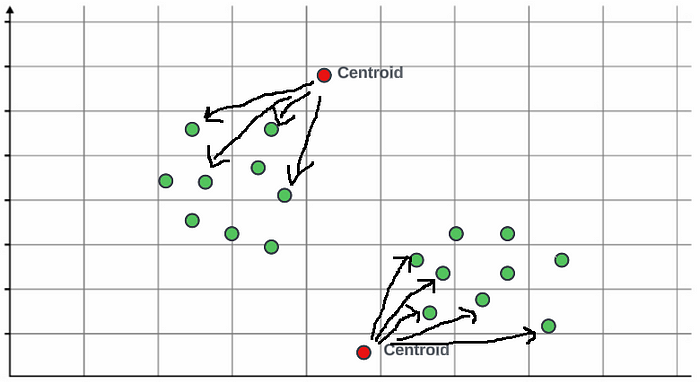
Update the centroid

**How do we decide the K value?**

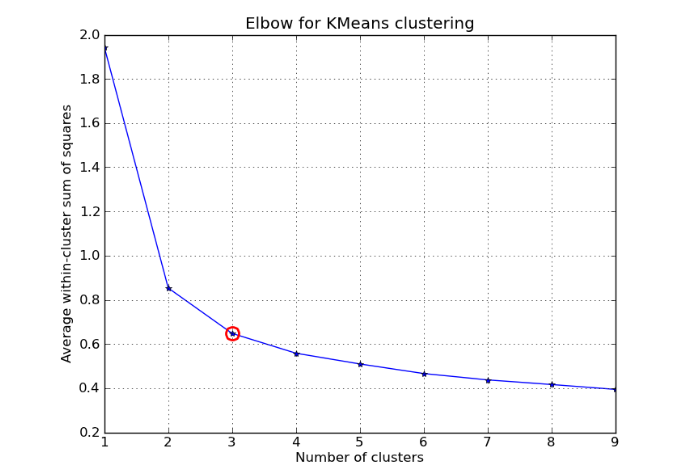
To determine the K value, we use a concept called the [**elbow method**](https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/#:~:text=The%20elbow%20method%20is%20a%20technique%20used%20in%20clustering%20analysis,Q2.), which helps us find the optimized K value. The elbow method involves constructing a graph between the **K value** and something called **WCSS**(**within-cluster sum of squares**), where we start with one centroid and iterate from **1 to 10**. For each iteration, we calculate the distance between each data point and the **centroid**.



In this example, the k value is 1 and the distance value will always be **greater than 0**. We can then observe the graph and determine the K value where the WCSS starts to level off, indicating the optimal number of clusters. If the**k value is 2 there will be 2 centroids.** You can calculate the WCSS value by taking the summation of all the distances with the centroid.



When you increase the k value you will be able to see that there will be abrupt changes in the WCSS value and this is called an elbow curve.

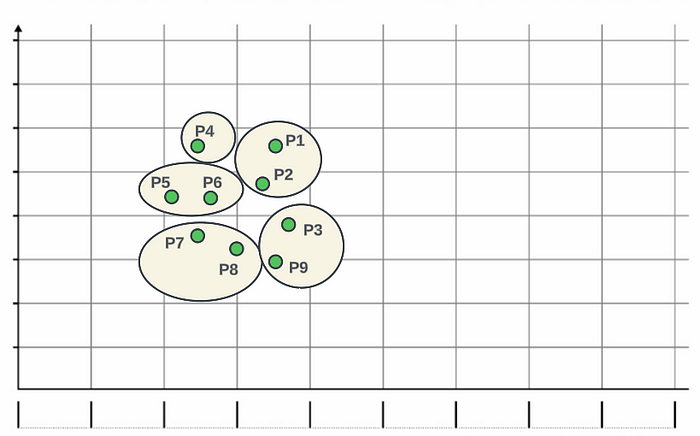


*This curve is called the Elbow Curve because it resembles an elbow, with an abrupt change at one point(red point) and then a straight line. It’s an essential tool in finding the K-value.*

For validation, we use the[**Silhouettes Score**](https://en.wikipedia.org/wiki/Silhouette_(clustering)). K-means clustering involves updating centroids and calculating distances based on that. As K increases, the distance becomes normal. We need to find the feasible K-value where the abrupt change happens. Once we find it, we take it as our K-value. In summary, if we want to find the cluster, we take the K-value and initialize K centroids. We update the centroids by computing the average and finding the distance. We continue this process until we get separate groups. To construct the Elbow Curve, we need to check different K-values and WCS values, which increases the model complexity.

**Hierarchical Clustering**

If you have data points P1, P2, P3, P4, P5, P6, P7, P8 and P9. Hierarchical clustering involves finding the nearest values and combining those points into one cluster. For instance, let’s say P1 and P2 are the closest points. We compute the distance between them and combine them into one group.



Data points can be combined into groups

Next, we find the next nearest points, and so on until we have combined all the nearest points into clusters. It’s a step-by-step process that connects the points based on their proximity. Assume you grouped all the points into 5 groups like this. Now you can see P4 group is near to the P5 — P6 group. So we can combine those 2 groups into one group. So we can combine the nearest groups into one group also.

(P4)GROUP + (P5, P6)GROUP

((P4) + (P5, P6))GROUP + (P1, P2)GROUP

(((P4) + (P5, P6)) + (P1, P2))GROUP + (P7 , P8)GROUP

((((P4) + (P5, P6)) + (P1, P2)) + (P7 , P8))GROUP + (P3, P9)GROUP

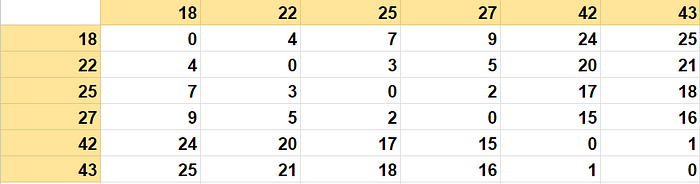
Let’s study more about Hierarchical Clustering using an example. To calculate the minimal distances you can use **Agglomerative Hierarchical Clustering using Dendogram.**

Considering the following set of 6 one-dimensional data points.

18, 22, 25, 42, 27, 43

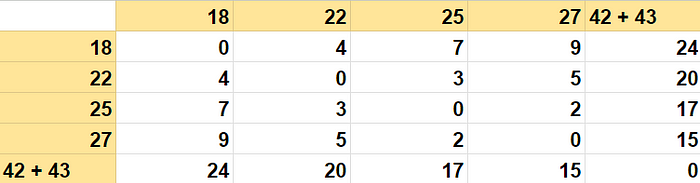
* Apply the **Agglomerative Hierarchical Clustering algorithm**to build the hierarchical clustering dendrogram**.**
* Merge the clusters using Minimal distance and update the proximity matrix accordingly.
* Clearly show the proximity matrix corresponding to each iteration of the algorithm.

In the first step, what we do is, we will calculate the distance between the given data points. First, we will write the data points on the horizontal and vertical axis. You can say that these are the data points given to us and the same thing we have written on these particular vertical lines. Also, now, we will try to calculate the distance between the data points; that is, the distance between 18 and 18 is **0**, and The distance between 18 and 22 is **4**. Similarly, the distance between 18 and 25 is **7**. The distance between 18 and 27 is **9**, the Distance between 18 and 42 is **24**, and the distance between 18 and 43 is **25**.



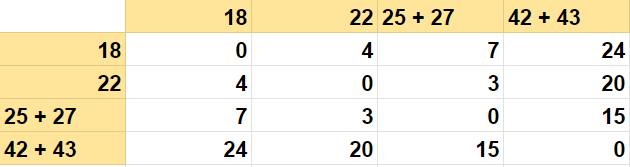
1st iteration

To determine the minimum distance between two data points, we analyzed a **distance matrix**. The distance between **42 and 43** was found to be **1**, which is the smallest distance in the matrix. We merged points 42 and 43 to form a cluster in the first step. After merging, we removed the row and column corresponding to point 43, and it was merged with 42. The resulting matrix is shown below.



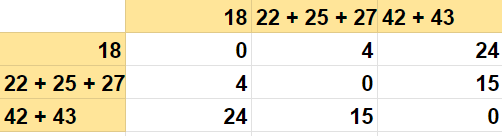
2nd iteration

In the second step, we attempt to find the minimum distance once again. Upon closer examination of the matrix, it appears that the minimum distance is between **27 and 25**. In such cases, we merge 27 with 25 to form a new cluster. The rows and columns corresponding to 27 will be removed, where 27 is merged into the 25th row and the column. This process results in a modified matrix like this. You may have a question about when merging 25,27 why is it 17 instead of 15? We take the minimum value possible for each step. So that the merging does not affect the next step of finding the minimum value in the whole set.



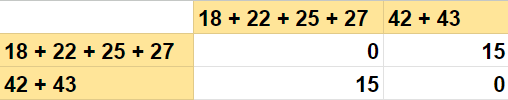
3rd iteration

Again to find the minimum distance, we need to look at the remaining data points in the matrix. In this case, the minimum distance is between 22 and 25 + 27, which is equal to **3**. As a result, we can merge 25 and 27 into 22. This will lead to the removal of the corresponding row and column. After merging 25,27, we will then merge 22, resulting in the final matrix.



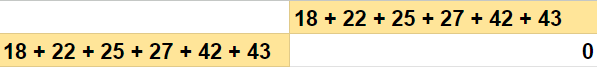
4th iteration

Now, from this one, if you look at the minimum distance, **4**is the minimum distance between 22 + 25 + 27 and 18, so we will merge it to 18. And then we remove this particular row and column here, so 18 is merged into 22 + 25 + 27. It looks something like this.



5th iteration

Now you can see 15 is the only distance here. This is the final cluster after combining everything. Or else you can combine 18 + 22 + 25 + 27 into 42 + 43 also and we will get a matrix like this.

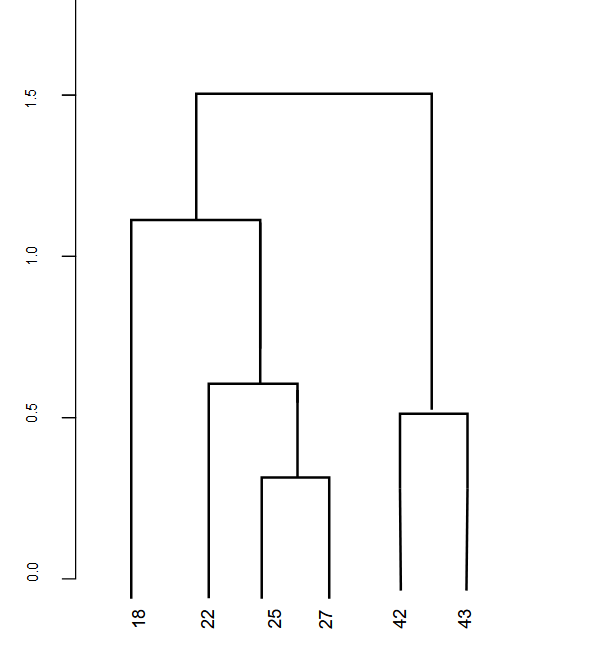


6th iteration

The final output dendrogram in this case is like this,

(42 + 43)   
(25 + 27)  
(22 + (25 + 27))  
(18 + (22 + (25 + 27)))  
(42 + 43) + (18 + (22 + (25 + 27)))

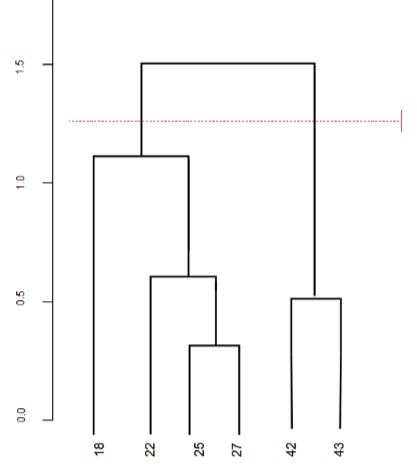
The same thing can be shown in a dendrogram like this.



dendrogram

Now, we can set a threshold distance and draw a horizontal line (*Generally, we try to set the threshold so that it cuts the tallest vertical line*).

**The more the distance of the vertical lines in the dendrogram, the more the distance between those clusters**. The number of clusters will be the number of vertical lines intersected by the line drawn using the threshold. In the above example, since the red dotted line intersects 2 vertical lines, we will have 2 clusters. 18 + 22 + 25 + 27 cluster and 42 + 43 cluster.



A horizontal line with a threshold distance

**What will take more time k-means or hierarchical clustering?**

The time taken by k-means clustering and hierarchical clustering can vary depending on several factors, including**the size of the dataset, the number of clusters, the dimensionality of the data, and the specific implementation of the algorithms**. However, in general, hierarchical clustering tends to be more computationally expensive compared to k-means clustering.

K-means clustering has a time complexity of**O(n \* k \* i \* d)**, where n is the number of data points, **k is the number of clusters**, **i is the number of iterations**, and **d is the number of dimensions**. K-means is an iterative algorithm that converges relatively quickly, especially if the dataset is well-behaved and the number of clusters is not excessively large.

On the other hand, hierarchical clustering has a time complexity of **O(n² \* log(n)) or O(n³)** depending on the specific algorithm used. Hierarchical clustering constructs a hierarchy of clusters by iteratively merging or splitting clusters based on a linkage criterion. The pairwise distance calculations required during the merging or splitting process can result in a higher computational cost, especially for large datasets.

In summary, if the number of clusters is reasonably small and the dataset is not too large, k-means clustering is generally faster compared to hierarchical clustering. However, for large datasets or when preserving the hierarchical structure is important, hierarchical clustering may be more suitable despite its higher computational cost.

**Validate Clustering Models**

Validating clustering models is an important step to assess their performance and reliability. Here are some common techniques to validate clustering models:

* **Within-cluster sum of squares** (**WCSS**): Compute the WCSS for different values of k and choose the value of k at the “elbow” point, where the decrease in WCSS becomes less significant. We learned this one earlier.
* **Silhouette coefficient**: Calculate the silhouette coefficient for each data point and compute the average across all points. Higher values indicate better clustering.
* **Gap statistic**: Compare the within-cluster dispersion of the data to a reference null distribution and select the value of k that maximizes the gap statistic.

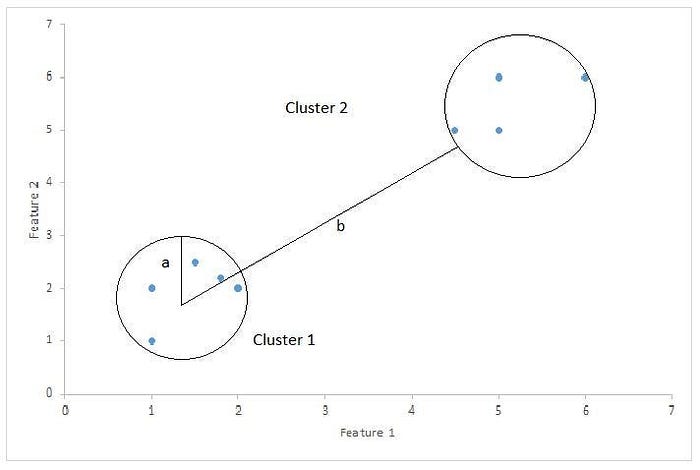
**Silhouette coefficient**

The Silhouette coefficient is a measure of how well each data point fits its assigned cluster while also being separated from other clusters. It quantifies the quality of clustering by considering both the compactness of the clusters and the separation between different clusters.

The Silhouette Coefficient or silhouette score is a metric used to calculate the goodness of a clustering technique. Its value ranges from -1 to 1.

* **A value close to 1** indicates that the data point is well-matched to its assigned cluster and is relatively far from neighboring clusters. This suggests a good clustering result.
* **A value close to 0** indicates that the data point is on or very close to the decision boundary between two neighboring clusters.
* **A value close to -1** indicates that the data point may have been assigned to the wrong cluster and is far from its neighboring cluster.

The Silhouette coefficient is computed using the following formula for a single data point:



s = (b - a) / max(a, b)  
  
  
a - the average distance between the data point and all other points within the same cluster (intra-cluster distance).  
b - the average distance between the data point and all points in the nearest neighboring cluster (inter-cluster distance).

To obtain the overall Silhouette coefficient for a clustering solution, you calculate the **average Silhouette coefficient** across all data points.

The Silhouette coefficient can be used to compare different clustering solutions or different values of k (the number of clusters). A higher average Silhouette coefficient indicates better clustering performance. **It is important to note that the Silhouette coefficient does not apply to all clustering algorithms and works best with methods that use distance-based measures, such as k-means clustering**.

When using the Silhouette coefficient, it is recommended to combine it with other validation methods to get a comprehensive understanding of the clustering results.

**The Difference Between K-Means and Hierarchical Clustering**

The main differences between K-Means and Hierarchical Clustering can be divided into several topics:

**Methodology**:

* K-means clustering: K-means is an iterative partitioning algorithm. It starts by randomly initializing k centroids and iteratively assigns data points to the nearest centroid, updates the centroids based on the assigned points, and repeats the process until convergence.
* Hierarchical clustering: Hierarchical clustering builds a hierarchy of clusters by iteratively merging or splitting clusters. It can be performed in two ways: agglomerative clustering (bottom-up) starts with each data point as a separate cluster and merges them iteratively based on their proximity, while divisive clustering (top-down) starts with all data points in a single cluster and splits them recursively.

**Number of Clusters:**

* K-means clustering: K-means requires the number of clusters (k) to be specified in advance. The algorithm aims to partition data into k clusters.
* Hierarchical clustering: Hierarchical clustering does not require the number of clusters to be predetermined. It creates a hierarchy of clusters, allowing for exploration of different levels of granularity.

**Cluster Shape:**

* K-means clustering: K-means assumes that the clusters are spherical and have similar sizes. It tries to minimize the within-cluster sum of squares, making it suitable for datasets with well-separated, globular clusters.
* Hierarchical clustering: Hierarchical clustering can handle clusters with arbitrary shapes. It is not limited to assuming specific cluster shapes, allowing for more flexibility in capturing complex cluster structures.

**Interpretability:**

* K-means clustering: K-means produces hard assignments, meaning each data point is assigned to only one cluster. The resulting clusters are easy to interpret and analyze.
* Hierarchical clustering: Hierarchical clustering creates a hierarchy of clusters, represented by a dendrogram. It provides a visual representation of cluster relationships and allows for exploration at different levels of granularity.

**Scalability**:

* K-means clustering: K-means is computationally efficient and can handle large datasets with a reasonable number of clusters. It is suitable for applications where efficiency is important.
* Hierarchical clustering: Hierarchical clustering can be computationally expensive, especially for large datasets. The time and memory requirements increase as the number of data points grows.

**Sensitivity to Initialization:**

* K-means clustering: K-means is sensitive to the initial placement of centroids. Different initializations can lead to different clustering results, and the algorithm may converge to sub-optimal solutions.
* Hierarchical clustering: Hierarchical clustering is less sensitive to initialization since it does not rely on random initialization. The clustering results are determined by the merging or splitting decisions based on the proximity of clusters.

**Pros and Cons of K-Means Clustering**

**Pros**:

* **Simplicity**: K-means clustering is relatively easy to understand and implement. It is a straightforward algorithm that iteratively assigns data points to clusters based on their proximity to centroids.
* **Scalability**: K-means clustering is computationally efficient and can handle large datasets with a reasonable number of clusters. It is suitable for applications where efficiency is important.
* **Interpretability**: The resulting clusters in K-means clustering can be easily interpreted since each data point is assigned to a single cluster. This makes it useful for exploratory data analysis and pattern recognition.
* **Versatility**: K-means clustering can work well with various types of data and distance metrics. It is not restricted to specific data types or assumptions about data distribution.

**Cons**:

* **Sensitivity to Initialization**: K-means clustering is sensitive to the initial placement of centroids. Different initializations can lead to different clustering results, and the algorithm may converge to sub-optimal solutions.
* **Fixed Number of Clusters**: K-means clustering requires the number of clusters (k) to be specified in advance. Determining the optimal number of clusters can be challenging and may require trial and error or additional validation techniques.
* **Sensitive to Outliers**: K-means clustering treats all data points equally and is sensitive to outliers. Outliers can significantly impact the cluster centroids and the resulting clusters.
* **Limited Cluster Shape Flexibility**: K-means clustering assumes that clusters are spherical and have similar sizes. It may struggle with complex cluster shapes or clusters with varying densities.
* **Lack of Robustness to Noise**: K-means clustering does not handle noise or outliers well. It tends to assign noisy data points to the nearest cluster, potentially affecting the overall cluster structure and interpretability.

One of the limitations of K-Means Clustering is that it assumes that the clusters are spherical and equally sized. This assumption may not hold for all datasets, leading to inaccurate results. Additionally, K-Means Clustering is not suitable for datasets with categorical variables, as it is a distance-based algorithm that requires numerical data.

On the other hand, K-Means Clustering can be useful in identifying patterns in customer segmentation, market research, and image segmentation. It can also be used in anomaly detection, where it can identify data points that do not belong to any cluster.

**Pros and Cons of Hierarchical Clustering**

**Pros**:

* **Hierarchical Structure**: Hierarchical clustering captures the hierarchical structure of the data by creating a tree-like structure called a dendrogram. This can provide insights into the relationships and subgroups within the data.
* **No Assumption of Number of Clusters**: Hierarchical clustering does not require the number of clusters to be specified in advance. The dendrogram allows you to explore different levels of granularity, enabling the identification of clusters at multiple scales.
* **Flexibility in Cluster Shape**: Hierarchical clustering can handle clusters with arbitrary shapes. It is not limited to assuming spherical clusters, allowing for more flexibility in capturing complex cluster structures.
* **Agglomerative and Divisive Approaches**: Hierarchical clustering offers both agglomerative and divisive approaches. Agglomerative clustering starts with individual data points as clusters and merges them iteratively, while divisive clustering starts with all data points in a single cluster and splits them recursively. This provides flexibility in the clustering process.
* **Visual Representation**: The dendrogram generated by hierarchical clustering can be visually appealing and informative. It allows for a visual exploration of the clustering structure and can aid in understanding the relationships among the clusters.

**Cons**:

* **Computational Complexity**: Hierarchical clustering can be computationally expensive, especially for large datasets. The time complexity is generally higher compared to algorithms like K-means clustering.
* **Memory Requirements**: Hierarchical clustering may require a significant amount of memory to store the distance matrix or linkage information, especially for large datasets. This can limit its applicability to datasets that do not fit into memory.
* **Lack of Scalabilit**y: Hierarchical clustering is not as scalable as some other clustering algorithms. As the number of data points increases, the computational cost and memory requirements also increase significantly.
* **Difficulty in Determining Number of Clusters**: While hierarchical clustering does not require specifying the number of clusters in advance, determining the appropriate number of clusters from the dendrogram can be subjective and challenging. It often requires visual inspection or additional techniques such as cutting the dendrogram or using clustering indices.
* **Sensitivity to Noise**: Hierarchical clustering can be sensitive to noise and outliers. Outliers can affect the merging or splitting decisions, leading to suboptimal clustering results.

Another advantage of Hierarchical Clustering is that it can handle missing data points effectively. It can also be used to identify outliers in the dataset, which can be useful in anomaly detection.

However, one of the major limitations of Hierarchical Clustering is that it is not suitable for large datasets. The algorithm can become computationally expensive and time-consuming when dealing with a large number of data points. Additionally, the results of Hierarchical Clustering can be highly dependent on the choice of distance metric used to measure the similarity between data points.

**When to Use K-Means Clustering Algorithm?**

* **Well-Separated Clusters**: K-means clustering performs well when the clusters in the dataset are well-separated and have distinct boundaries. The algorithm aims to minimize the within-cluster sum of squares, making it effective in identifying compact and spherical clusters.
* **Large Datasets**: K-means clustering is computationally efficient and can handle large datasets with a reasonable number of clusters. It scales well and is suitable for applications where efficiency is important.
* **Predefined Number of Clusters**: K-means requires the number of clusters (k) to be specified in advance. If you have prior knowledge or a specific requirement for the number of clusters, K-means can be a suitable choice.
* **Numeric Data**: K-means clustering works well with numeric data, as it relies on calculating distances between data points. It is suitable for continuous variables or attributes that can be represented numerically.
* **Quick Prototyping and Exploration**: Due to its simplicity and ease of implementation, K-means clustering is useful for quick prototyping and exploratory data analysis. It can provide initial insights into the structure of the data and help in forming hypotheses or making initial observations.
* **Initial Cluster Centers**: If you have some prior knowledge or reasonable estimates for the initial cluster centers, K-means can converge faster and produce more accurate results. This can be beneficial in cases where you have some understanding of the data or domain expertise.

**When to Use Hierarchical Clustering Algorithm?**

* **Exploration of Cluster Structures**: Hierarchical clustering provides a hierarchical structure of clusters through a dendrogram. This can be useful when you want to explore the relationships and subgroups within your data at different levels of granularity.
* **Unknown Number of Clusters**: Hierarchical clustering does not require the number of clusters to be predetermined. It allows you to explore different levels of the dendrogram and determine the number of clusters based on your data and analysis goals.
* **Arbitrary Cluster Shapes**: Hierarchical clustering can handle clusters with arbitrary shapes. It is not limited to assuming specific cluster shapes, allowing for more flexibility in capturing complex cluster structures.
* **Visual Representation**: The dendrogram generated by hierarchical clustering can be visually appealing and informative. It provides a visual representation of the clustering structure, making it easier to interpret and analyze the relationships among the clusters.
* **Proximity-based Analysis**: Hierarchical clustering is based on the concept of proximity or similarity between data points. If your data is best analyzed in terms of distance or similarity measures, hierarchical clustering can be a suitable choice.
* **Small to Medium-sized Datasets**: While hierarchical clustering can be computationally expensive for large datasets, it is well-suited for small to medium-sized datasets. It can handle datasets that fit into memory and does not require excessive computational resources.

Clustering algorithms are valuable tools in machine learning for grouping similar data objects. The selection of the most suitable clustering algorithm depends on various factors such as dataset size and dimensionality, the desired number of clusters, and available computational resources. Two commonly used algorithms are K-Means and Hierarchical Clustering, each with its strengths and weaknesses. By considering the pros and cons of these algorithms, their practical applications, and the influencing factors, you can confidently choose the optimal clustering algorithm for your specific task.

This is all about K-Means and Hierarchical Clustering. I hope you get a good idea about Cluturing Algorithms. See you in another article.

Thank You!

[K Means Clustering](https://medium.com/tag/k-means-clustering?source=post_page-----46acc005057d---------------k_means_clustering-----------------)

[Hierarchical Clustering](https://medium.com/tag/hierarchical-clustering?source=post_page-----46acc005057d---------------hierarchical_clustering-----------------)

[Machine Learning](https://medium.com/tag/machine-learning?source=post_page-----46acc005057d---------------machine_learning-----------------)

[Dendrogram](https://medium.com/tag/dendrogram?source=post_page-----46acc005057d---------------dendrogram-----------------)

[Silhouette Score](https://medium.com/tag/silhouette-score?source=post_page-----46acc005057d---------------silhouette_score-----------------)