1. A set of one-dimensional data points is given to you: 5, 10, 15, 20, 25, 30, 35. Assume that k = 2 and that the first set of random centroid is 15, 32, and that the second set is 12, 30.

a) Using the k-means method, create two clusters for each set of centroid described above.

b) For each set of centroid values, calculate the SSE.

**K-means Clustering:**

In K-means approach the data objects are classified based on their attributes or features into k number of clusters. The number of clusters i.e K is an input given by the user. K-means is one of the simplest unsupervised learning algorithms.

**K-means Algorithm:**

K- number of clusters

n- sample vectors x1,x2,....xn𝑥1,𝑥2,....𝑥𝑛

mi𝑚𝑖- The mean of vectors in cluster i

1. Assume k < n
2. Make initial guesses for the means m1,m2,...mk𝑚1,𝑚2,...𝑚𝑘
3. Until there are no changes in any mean
4. Use the estimated means to classify the samples into clusters.

for i = 1 to k

Replace mi𝑚𝑖 with the mean of all of the samples for cluster i

end\_for

end\_until

1. Following three steps are repeated until convergence:
2. Iterate till no object moves to a different group:
3. Find the centroid coordinate.
4. Find the distance of each object to the centroids
5. Based on minimum distance group the objects.

**K-Means Advantages :**

1. If variables are huge, then K-Means most of the times computationally faster than hierarchical clustering, if we keep k smalls.
2. K-Means produce tighter clusters than hierarchical clustering, especially if the clusters are globular.

**K-Means Disadvantages :**

1. Difficult to predict K-Value.
2. With global cluster, it didn't work well.
3. It does not work well with clusters (in the original data) of Different size and Different density.

**Numerical**

Given:

Dataset -{15, 15, 16, 19, 19, 20, 20, 21, 22, 28, 35, 40, 41, 42, 43, 44, 60, 61, 65}

Solution:

Assume K=2

So, two clusters are formed (Users point of view) i.e

Cluster1𝐶𝑙𝑢𝑠𝑡𝑒𝑟1 = 15, 16, 19, **20**, 21, 22, 28 ∴C1∴𝐶1 = 20 (mean value)

Cluster2𝐶𝑙𝑢𝑠𝑡𝑒𝑟2 = 35, 40, 41, 42, **43**, 44, 60, 61, 65 ∴C2∴𝐶2 = 43 (mean value)

Below table:

Calculate Cluster 1 and Cluster 2 values, by subtracting the datapoint from C1𝐶1 and C2𝐶2

i.e (20 - 15) = 5 and (43 - 15) = 28

then choose min value (5) and assign that cluster.

| **Data point** | **Cluster 1** | **Cluster 2** | **Cluster assign** |
| --- | --- | --- | --- |
| 15 | 5 | 28 | 1 |
| 15 | 5 | 28 | 1 |
| 16 | 4 | 27 | 1 |
| 19 | 1 | 24 | 1 |
| 19 | 1 | 24 | 1 |
| 20 | 0 | 23 | 1 |
| 20 | 0 | 23 | 1 |
| 21 | 1 | 22 | 1 |
| 22 | 2 | 21 | 1 |
| 28 | 8 | 15 | 1 |
| 35 | 15 | 8 | 2 |
| 40 | 20 | 3 | 2 |
| 41 | 21 | 2 | 2 |
| 42 | 22 | 1 | 2 |
| 43 | 23 | 0 | 2 |
| 44 | 24 | 1 | 2 |
| 60 | 40 | 17 | 2 |
| 61 | 41 | 18 | 2 |
| 65 | 45 | 22 | 2 |

New centroid is calculated by taking average value of clusters.

| **Old centroid** | **New centroid** |
| --- | --- |
| 20 | 19.5 |
| 43 | 47.88 |

| **Data point** | **Cluster 1** | **Cluster 2** | **Cluster assign** |
| --- | --- | --- | --- |
| 15 | 4.5 | 32.8 | 1 |
| 15 | 4.5 | 32.8 | 1 |
| 16 | 3.5 | 31.8 | 1 |
| 19 | 0.5 | 28.8 | 1 |
| 19 | 0.5 | 28.8 | 1 |
| 20 | 0.5 | 27.8 | 1 |
| 20 | 0.5 | 27.8 | 1 |
| 21 | 1.5 | 26.8 | 1 |
| 22 | 2.5 | 25.8 | 1 |
| 28 | 8.5 | 19.8 | 1 |
| 35 | 15.5 | 12.8 | 2 |
| 40 | 20.5 | 7.8 | 2 |
| 41 | 21.5 | 6.8 | 2 |
| 42 | 22.5 | 5.8 | 2 |
| 43 | 23.5 | 4.8 | 2 |
| 44 | 24.5 | 3.8 | 2 |
| 60 | 40.5 | 12.1 | 2 |
| 61 | 41.5 | 13.1 | 2 |
| 65 | 45.5 | 17.1 | 2 |

| **Old centroid** | **New centroid** |
| --- | --- |
| 19.5 | 19.5 |
| 47.88 | 47.88 |

Since, Old centriod is same as New centroid stop the iteration.

∴∴ Final answer is k1𝑘1 = {15,15,16,19,19,20,20,21,22,28}

k2𝑘2 = {35,40,41,42,43,44,60,61,65}

2. Describe how the Market Basket Research makes use of association analysis concepts.

Market basket analysis is a [data mining](https://searchsqlserver.techtarget.com/definition/data-mining) technique used by retailers to increase sales by better understanding customer purchasing patterns. It involves analyzing large data sets, such as purchase history, to reveal product groupings, as well as products that are likely to be purchased together.

The adoption of market basket analysis was aided by the advent of electronic point-of-sale (POS) systems. Compared to handwritten records kept by store owners, the digital records generated by POS systems made it easier for applications to process and [analyze large volumes of purchase data](https://www.techtarget.com/searchcustomerexperience/tip/6-ways-to-use-analytics-to-improve-customer-engagement).

Implementation of market basket analysis requires a background in statistics and [data science](https://www.techtarget.com/searchenterpriseai/definition/data-science), as well as some algorithmic computer programming skills. For those without the needed technical skills, commercial, off-the-shelf tools exist.

**Types of market basket analysis**

Retailers should understand the following types of market basket analysis:

* **Predictive market basket analysis.** This type considers items purchased in sequence to determine cross-sell.
* **Differential market basket analysis.** This type considers data across different stores, as well as purchases from different customer groups during different times of the day, month or year. If a rule holds in one dimension, such as store, time period or customer group, but does not hold in the others, analysts can determine the factors responsible for the exception. These insights can lead to new product offers that [drive higher sales](https://www.techtarget.com/searchcustomerexperience/tip/10-lead-scoring-best-practices-to-improve-sales-efficiency).

**Algorithms for market basket analysis**

In market basket analysis, [association rules](https://www.techtarget.com/searchbusinessanalytics/definition/association-rules-in-data-mining) are used to predict the likelihood of products being purchased together. Association rules count the frequency of items that occur together, seeking to find associations that occur far more often than expected.

Algorithms that use association rules include AIS, SETM and Apriori. The Apriori algorithm is commonly cited by data scientists in research articles about market basket analysis and is used to identify frequent items in the database, then evaluate their frequency as the datasets are expanded to larger sizes.

The [arules package for R](https://github.com/mhahsler/arules" \t "_blank) is an open source toolkit for association mining using the R programming language. This package supports the Apriori algorithm, along with the following other mining algorithms:

* arulesNBMiner
* Opusminer
* RKEEL
* RSarules

**Examples of market basket analysis**

Amazon's website uses a well-known example of market basket analysis. On a product page, Amazon presents users with related products, under the headings of "Frequently bought together" and "Customers who bought this item also bought."

Market basket analysis also applies to bricks-and-mortar stores. If analysis showed that magazine purchases often include the purchase of a bookmark, which could be considered an unexpected combination as the consumer did not purchase a book, then the bookstore might place a selection of bookmarks near the magazine rack.

**Benefits of market basket analysis**

Market basket analysis can increase sales and [customer satisfaction](https://www.techtarget.com/whatis/definition/customer-satisfaction-CSAT). Using data to determine that products are often purchased together, retailers can optimize product placement, offer special deals and create new product bundles to encourage further sales of these combinations.

These improvements can generate additional sales for the retailer, while making the shopping experience more productive and valuable for customers. By using market basket analysis, customers may feel a stronger sentiment or [brand loyalty](https://www.techtarget.com/searchcustomerexperience/feature/Customer-loyalty-vs-brand-loyalty-Whats-the-difference) toward the company.

3. Give an example of the Apriori algorithm for learning association rules.

The Apriori algorithm works by finding relationships among numerous items in a dataset. The method known as association rule mining makes this discovery.

For example, in a supermarket, a pattern emerges where people buy certain items together. Let’s assume that individuals might buy cold drinks and chips together to make the example more concrete. Similarly, it’s also found that customers also put notebooks and pens together in a purchase.

Through association rule mining, you, as a supermarket owner, can leverage identified relationships to boost sales. Strategies like packaging associated products together, placing them in close proximity, offering group discounts, and optimizing inventory management can lead to increased profits.

Association rule mining is complex in practice, involving thousands of items with numerous potential relationships. The challenge lies in sifting through this complexity to identify meaningful business connections, as not all relationships are relevant. Without proper techniques, the process can be akin to finding a needle in a haystack.

4. In hierarchical clustering, how is the distance between clusters measured? Explain how this metric is used to decide when to end the iteration.

# Hierarchical Clustering in Machine Learning

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as **hierarchical cluster analysis** or HCA.

In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the **dendrogram**.

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. **Agglomerative:** Agglomerative is a **bottom-up** approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
2. **Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a **top-down approach.**

### Why hierarchical clustering?

As we already have other [clustering](https://www.javatpoint.com/clustering-in-machine-learning) algorithms such as [**K-Means Clustering**](https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning), then why we need hierarchical clustering? So, as we have seen in the K-means clustering that there are some challenges with this algorithm, which are a predetermined number of clusters, and it always tries to create the clusters of the same size. To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

In this topic, we will discuss the Agglomerative Hierarchical clustering algorithm.

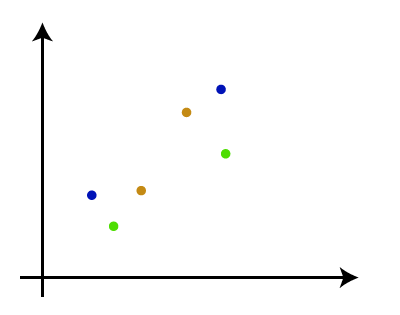
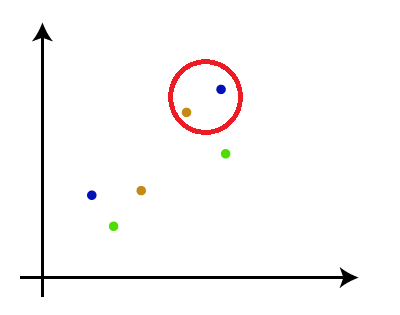
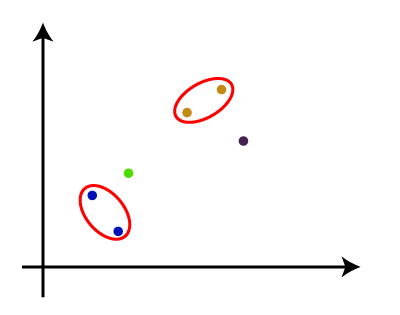
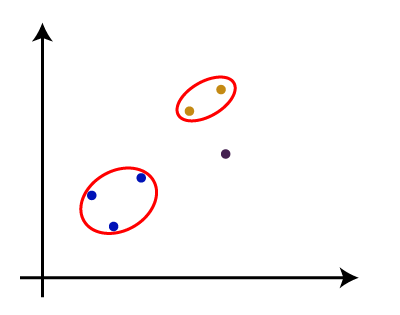
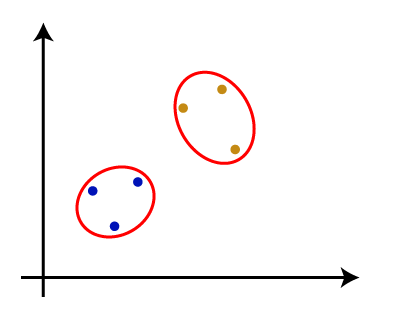
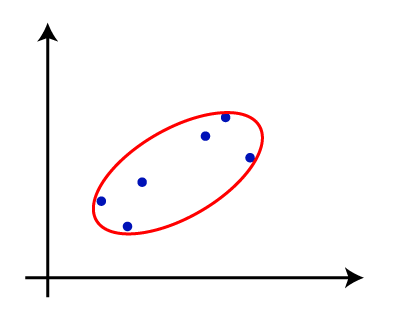
## Agglomerative Hierarchical clustering

The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the **bottom-up approach**. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram.

## How the Agglomerative Hierarchical clustering Work?

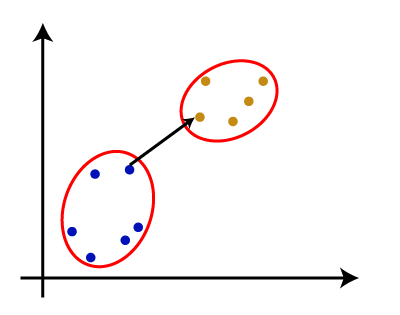
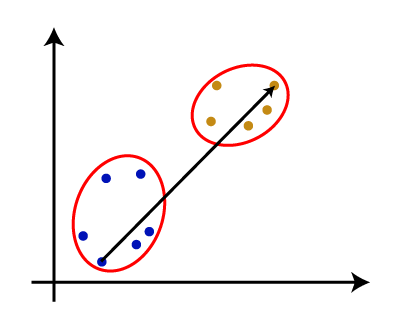
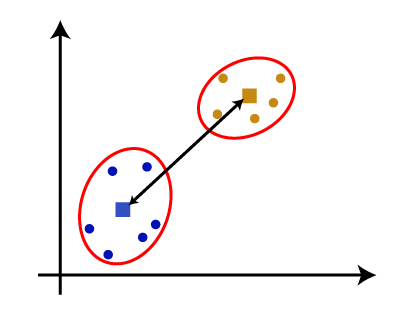
The working of the AHC algorithm can be explained using the below steps:

* **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.  
  
* **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.  
  
* **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters.  
  
* **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:  
    
    
  
* **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

#### Note: To better understand hierarchical clustering, it is advised to have a look on k-means clustering

### Measure for the distance between two clusters

As we have seen, the **closest distance** between the two clusters is crucial for the hierarchical clustering. There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called **Linkage methods**. Some of the popular linkage methods are given below:

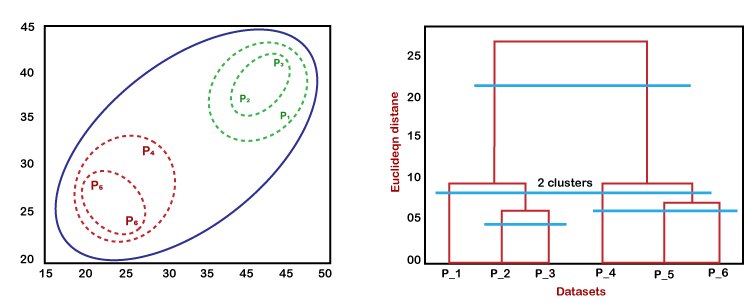
1. **Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:  
   
2. **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.  
   
3. **Average Linkage:** It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.
4. **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:  
   

From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

### Woking of Dendrogram in Hierarchical clustering

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

* As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The hight is decided according to the Euclidean distance between the data points.
* In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
* Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
* At last, the final dendrogram is created that combines all the data points together.

We can cut the dendrogram tree structure at any level as per our requirement.

## Python Implementation of Agglomerative Hierarchical Clustering

Now we will see the practical implementation of the agglomerative hierarchical clustering algorithm using Python. To implement this, we will use the same dataset problem that we have used in the previous topic of K-means clustering so that we can compare both concepts easily.

The dataset is containing the information of customers that have visited a mall for shopping. So, the mall owner wants to find some patterns or some particular behavior of his customers using the dataset information.

### Steps for implementation of AHC using Python:

The steps for implementation will be the same as the k-means clustering, except for some changes such as the method to find the number of clusters. Below are the steps:

1. **Data Pre-processing**
2. **Finding the optimal number of clusters using the Dendrogram**
3. **Training the hierarchical clustering model**
4. **Visualizing the clusters**

### Data Pre-processing Steps:

In this step, we will import the libraries and datasets for our model.

* **Importing the libraries**

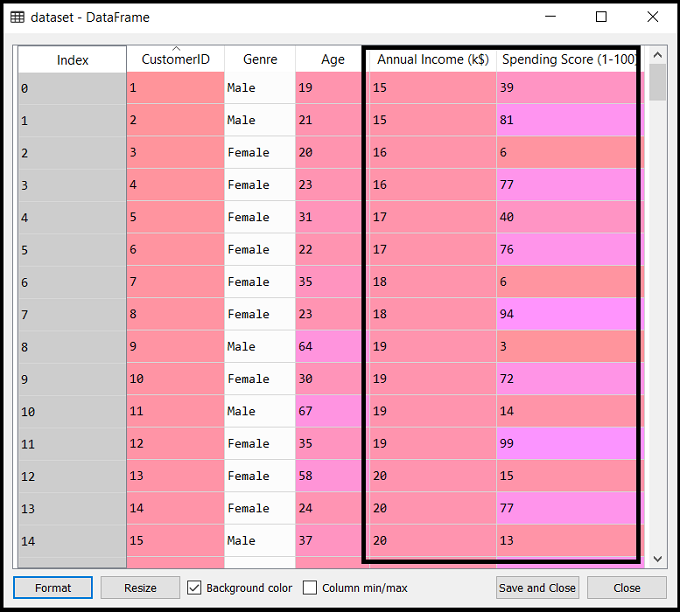
1. # Importing the libraries
2. **import** numpy as nm
3. **import** matplotlib.pyplot as mtp
4. **import** pandas as pd

The above lines of code are used to import the libraries to perform specific tasks, such as **[numpy](https://www.javatpoint.com/numpy-tutorial)** for the Mathematical operations, **[matplotlib](https://www.javatpoint.com/matplotlib)** for drawing the graphs or scatter plot, and [**pandas**](https://www.javatpoint.com/python-pandas) for importing the dataset.

* **Importing the dataset**

1. # Importing the dataset
2. dataset = pd.read\_csv('Mall\_Customers\_data.csv')

As discussed above, we have imported the same dataset of **Mall\_Customers\_data.csv,** as we did in k-means clustering. Consider the below output:



* **Extracting the matrix of features**

Here we will extract only the matrix of features as we don't have any further information about the dependent variable. Code is given below:

1. x = dataset.iloc[:, [3, 4]].values

Here we have extracted only 3 and 4 columns as we will use a 2D plot to see the clusters. So, we are considering the Annual income and spending score as the matrix of features.

### Step-2: Finding the optimal number of clusters using the Dendrogram

Now we will find the optimal number of clusters using the Dendrogram for our model. For this, we are going to use **scipy** library as it provides a function that will directly return the dendrogram for our code. Consider the below lines of code:

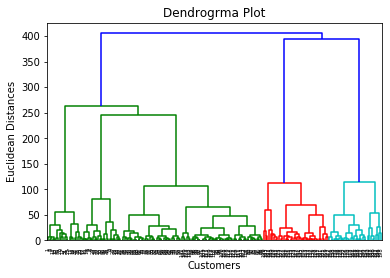
1. #Finding the optimal number of clusters using the dendrogram
2. **import** scipy.cluster.hierarchy as shc
3. dendro = shc.dendrogram(shc.linkage(x, method="ward"))
4. mtp.title("Dendrogrma Plot")
5. mtp.ylabel("Euclidean Distances")
6. mtp.xlabel("Customers")
7. mtp.show()

In the above lines of code, we have imported the **hierarchy** module of scipy library. This module provides us a method **shc.denrogram(),** which takes the **linkage()** as a parameter. The linkage function is used to define the distance between two clusters, so here we have passed the x(matrix of features), and method "**ward**," the popular method of linkage in hierarchical clustering.

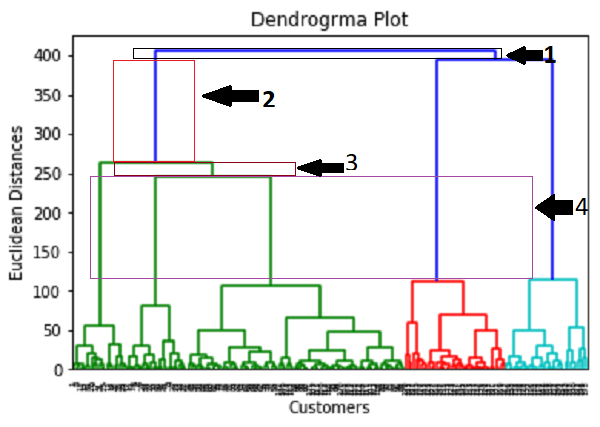
The remaining lines of code are to describe the labels for the dendrogram plot.

**Output:**

By executing the above lines of code, we will get the below output**:**



Using this Dendrogram, we will now determine the optimal number of clusters for our model. For this, we will find the **maximum vertical distance** that does not cut any horizontal bar. Consider the below diagram:



In the above diagram, we have shown the vertical distances that are not cutting their horizontal bars. As we can visualize, the 4th distance is looking the maximum, so according to this, **the number of clusters will be 5**(the vertical lines in this range). We can also take the 2nd number as it approximately equals the 4th distance, but we will consider the 5 clusters because the same we calculated in the K-means algorithm.

**So, the optimal number of clusters will be 5**, and we will train the model in the next step, using the same.

### Step-3: Training the hierarchical clustering model

As we know the required optimal number of clusters, we can now train our model. The code is given below:

1. #training the hierarchical model on dataset
2. from sklearn.cluster **import** AgglomerativeClustering
3. hc= AgglomerativeClustering(n\_clusters=5, affinity='euclidean', linkage='ward')
4. y\_pred= hc.fit\_predict(x)

In the above code, we have imported the **AgglomerativeClustering** class of cluster module of scikit learn library.

Then we have created the object of this class named as **hc.** The AgglomerativeClustering class takes the following parameters:

* **n\_clusters=5**: It defines the number of clusters, and we have taken here 5 because it is the optimal number of clusters.
* **affinity='euclidean'**: It is a metric used to compute the linkage.
* **linkage='ward'**: It defines the linkage criteria, here we have used the "ward" linkage. This method is the popular linkage method that we have already used for creating the Dendrogram. It reduces the variance in each cluster.

In the last line, we have created the dependent variable y\_pred to fit or train the model. It does train not only the model but also returns the clusters to which each data point belongs.

After executing the above lines of code, if we go through the variable explorer option in our Sypder IDE, we can check the y\_pred variable. We can compare the original dataset with the y\_pred variable. Consider the below image:



As we can see in the above image, the **y\_pred** shows the clusters value, which means the customer id 1 belongs to the 5th cluster (as indexing starts from 0, so 4 means 5th cluster), the customer id 2 belongs to 4th cluster, and so on.

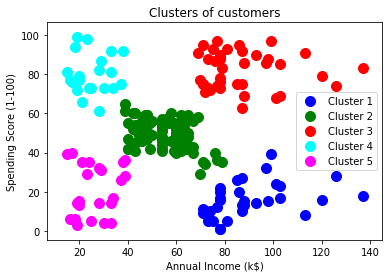
### Step-4: Visualizing the clusters

As we have trained our model successfully, now we can visualize the clusters corresponding to the dataset.

Here we will use the same lines of code as we did in k-means clustering, except one change. Here we will not plot the centroid that we did in k-means, because here we have used dendrogram to determine the optimal number of clusters. The code is given below:

1. #visulaizing the clusters
2. mtp.scatter(x[y\_pred == 0, 0], x[y\_pred == 0, 1], s = 100, c = 'blue', label = 'Cluster 1')
3. mtp.scatter(x[y\_pred == 1, 0], x[y\_pred == 1, 1], s = 100, c = 'green', label = 'Cluster 2')
4. mtp.scatter(x[y\_pred== 2, 0], x[y\_pred == 2, 1], s = 100, c = 'red', label = 'Cluster 3')
5. mtp.scatter(x[y\_pred == 3, 0], x[y\_pred == 3, 1], s = 100, c = 'cyan', label = 'Cluster 4')
6. mtp.scatter(x[y\_pred == 4, 0], x[y\_pred == 4, 1], s = 100, c = 'magenta', label = 'Cluster 5')
7. mtp.title('Clusters of customers')
8. mtp.xlabel('Annual Income (k$)')
9. mtp.ylabel('Spending Score (1-100)')
10. mtp.legend()
11. mtp.show()

Output: By executing the above lines of code, we will get the below output:



5. In the k-means algorithm, how do you recompute the cluster centroids?

This final method that we would like to examine is a non-hierarchical approach. This method was presented by MacQueen (1967) in the *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability.*

One advantage of this method is that we do not have to calculate the distance measures between all pairs of subjects. Therefore, this procedure seems much more efficient or practical when you have very large datasets.

Under this procedure, you need to pre-specify how many clusters to consider. The clusters in this procedure do not form a tree. There are two approaches to carrying out the *K*-Means procedure. The approaches vary as to how the procedure begins the partitioning. The first approach is to do this randomly, which is to start out with a random partitioning of subjects into groups and go from there. The alternative is to start with an additional set of starting points to form the centers of the clusters. The random nature of the first approach avoids bias.

Once this decision has been made, here is an overview of the process:

* **Step 1**: Partition the items into *K* initial clusters.
* **Step 2**: Scan through the list of *n* items, assigning each item to the cluster whose centroid (mean) is closest. Each time an item is reassigned, we recalculate the cluster mean or centroid for the cluster receiving that item and the cluster losing that item.
* **Step 3**: Repeat Step 2 over and over again until no more reassignments are made.

6. At the start of the clustering exercise, discuss one method for determining the required number of clusters.

# Elbow Method

It is the most popular method for determining the optimal number of clusters. The method is based on calculating the Within-Cluster-Sum of Squared Errors (WSS) for different number of clusters (k) and selecting the k for which change in WSS first starts to diminish.

The idea behind the elbow method is that the explained variation changes rapidly for a small number of clusters and then it slows down leading to an elbow formation in the curve. The elbow point is the number of clusters we can use for our clustering algorithm. Further details on this method can be found in this [paper](https://www.mdpi.com/2571-8800/2/2/16/pdf) by Chunhui Yuan and Haitao Yang.

We will be using the [YellowBrick](https://www.scikit-yb.org/en/latest/" \t "_blank) library which can implement the elbow method with few lines of code. It is a wrapper around Scikit-Learn and has some cool machine learning visualizations!

# Elbow Method for K means# Import ElbowVisualizer  
from yellowbrick.cluster import KElbowVisualizer  
model = KMeans()  
# k is range of number of clusters.  
visualizer = KElbowVisualizer(model, k=(2,30), timings= True)  
visualizer.fit(cluster\_df) # Fit data to visualizer  
visualizer.show() # Finalize and render figure

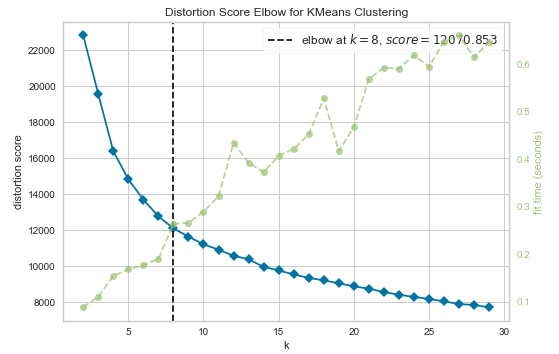


Fig 2: Elbow Method Results (Image by author)

The KElbowVisualizer function fits the KMeans model for a range of clusters values between 2 to 30. As shown in Figure 2, the elbow point is achieved with 8 clusters which is highlighted by the function itself. The function also informs us about how much time was needed to plot models for various numbers of clusters through the green line.

7. Discuss the k-means algorithm's advantages and disadvantages.

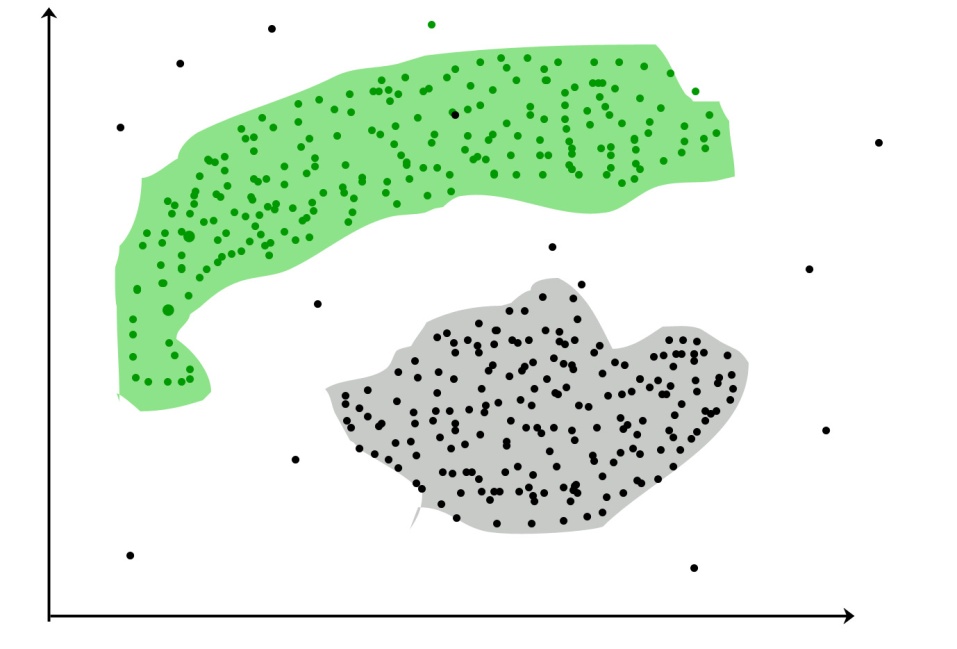
The K means clustering in machine learning offers several advantages, making it widely used in various applications:

* **Scalability:** K-means clustering is good for big data because it works fast and can handle lots of information without problems.
* **Simple and Easy to Implement:** Even if you don’t know much about machine learning. You can still use K-means because it’s simple and easy to understand.
* **Versatility:** K-means clustering works for different kinds of data, not just one type. So it’s useful for many different kinds of problems in data analysis.
* **Interpretable Results:** K-means clusters are easy to understand and can help us learn important things about how the data is organized.

While K-Means offers several advantages, it also has some limitations and disadvantages:

* **Sensitivity to Initial Centroids:** K means clustering in machine learning works best when we pick the starting points carefully. Because if we don’t, the groups might not be very good.
* **Determination of K:** Before using K-means clustering, we have to decide how many groups we want. Which can be tricky and might need some guessing or testing.
* **Sensitive to Outliers:** It can be thrown off by unusual data points. Because they can change where the center of each group ends up, affecting how the groups are made.
* **Assumes Spherical Clusters:** K-means thinks the groups are round and about the same size, but sometimes in real life. The groups might be different shapes or sizes, which can cause problems.

8. Draw a diagram to demonstrate the principle of clustering.



9. During your study, you discovered seven findings, which are listed in the data points below. Using the K-means algorithm, you want to build three clusters from these observations. The clusters C1, C2, and C3 have the following findings after the first iteration:

C1: (2,2), (4,4), (6,6); C2: (2,2), (4,4), (6,6); C3: (2,2), (4,4),

C2: (0,4), (4,0), (0,4), (0,4), (0,4), (0,4), (0,4), (0,4), (0,

C3: (5,5) and (9,9)

What would the cluster centroids be if you were to run a second iteration? What would this clustering's SSE be?

The new clusters after the second iteration will be c1: {(2,2), (4,4)} c2: {(0,4), (4,0), (5,5)} c3: {(6,6), (9,9)}.

**Explanation:**

From the above question,

They have given :

K-means clustering is an iterative algorithm that partitions observations into clusters. In each iteration, observations are reassigned to clusters based on their distance to the cluster centroid. The process is repeated until the clusters converge or the algorithm reaches the maximum number of iterations.

To calculate the centroids for the current clusters (c1, c2, c3).

Calculate the distances between the observations and the centroids.

Reassign the observations to clusters based on the distances calculated in step 2.

Repeat steps 1 to 3 until the clusters converge or the maximum number of iterations is reached.

The new clusters after the second iteration will be c1: {(2,2), (4,4)} c2: {(0,4), (4,0), (5,5)} c3: {(6,6), (9,9)}.

1. In a software project, the team is attempting to determine if software flaws discovered during testing are identical. Based on the text analytics of the defect details, they decided to build 5 clusters of related defects. Any new defect formed after the 5 clusters of defects have been identified must be listed as one of the forms identified by clustering. A simple diagram can be used to explain this process. Assume you have 20 defect data points that are clustered into 5 clusters and you used the k-means algorithm.

## Seven Principles of Software Testing

Before having an in-depth look at those two principles, let us briefly understand the seven principles of software testing.

***Let’s Explore!!***

### #1) Testing Shows the Presence of Defects

Every application or product is released into production after a sufficient amount of testing by different teams or passes through different phases like System Integration Testing, User Acceptance Testing, and Beta Testing etc.

So have you ever seen or heard from any of the testing team that they have tested the software fully and there is no defect in the software? Instead of that, every testing team confirms that the software meets all business requirements and it is functioning as per the needs of the end user.

In the software testing industry, no one will say that there is **no defect** in the software, which is quite true as testing cannot prove that the software is error-free or defect-free.

However, the objective of testing is to find more and more hidden defects using different techniques and methods. Testing can reveal undiscovered defects and if no defects are found then it does not mean that the software is defect free.

**Example 1**:

Consider a Banking application, this application is thoroughly tested and undergoes different phases of testing like SIT, UAT etc. and currently no defects are identified in the system.

However, there might be a possibility that in the production environment, the actual customer tries a functionality which is rarely used in the banking system and the testers overlooked that functionality, hence no defect was found till date or the code has never been touched by developers.

**Example 2:**

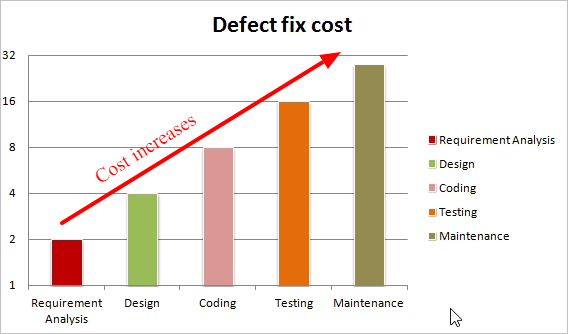
We have seen several advertisements for soaps, toothpaste, handwash or disinfectant sprays etc on television.

Consider a handwash advertisement which says on the television that 99% germs can be removed if that specific handwash is used. This clearly proves that the product is not 100% germ-free. Thus in our testing concept, we can say that no software is defect free.

### #2) Early Testing

Testers need to get involved at an early stage of the Software Development Life Cycle (SDLC). Thus the defects during the requirement analysis phase or any documentation defects can be identified. The cost involved in fixing such defects is very less when compared to those that are found during the later stages of testing.

**Consider the below image which shows how the cost of defect fixing gets increased as testing move towards the live production.**

[](https://www.softwaretestinghelp.com/wp-content/qa/uploads/2018/11/early-testing-defect-fixing-cost.png)

[image [*source*](http://www.softwaretestingtimes.com/)]

The above image shows that cost required for fixing a defect found during the Requirement Analysis is less and it goes on increasing as we move towards the Testing or the Maintenance phase.

Now the question is how early should the testing start?

Once the requirements are finalized, the testers need to involve for testing. Testing should be performed on requirement documents, specification or any other type of document so that if requirements are incorrectly defined then it can be fixed immediately rather than fixing them in the development phase.

### #3) Exhaustive Testing is Not Possible

It is not possible to test all the functionalities with all valid and invalid combinations of input data during actual testing. Instead of this approach, testing of a few combinations is considered based on priority using different techniques.

Exhaustive testing will take unlimited efforts and most of those efforts are ineffective. Also, the project timelines will not allow testing of so many number of combinations. Hence it is recommended to test input data using different methods like Equivalence Partitioning and Boundary Value Analysis.

**For Example**, If suppose we have an input field which accepts alphabets, special characters, and numbers from 0 to 1000 only. Imagine how many combinations would appear for testing, it is not possible to test all combinations for each input type.

The testing efforts required to test will be huge and it will also impact the project timeline and cost. Hence it is always said that exhaustive testing is practically not possible.

### #4) Testing is Context-Dependent

There are several domains available in the market like Banking, Insurance, Medical, Travel, Advertisement etc and each domain has a number of applications. Also for each domain, their applications have different requirements, functions, different testing purpose, risk, techniques etc.

Different domains are tested differently, thus testing is purely based on the context of the domain or application.

**For Example**, testing a banking application is different than testing any e-commerce or advertising application. The risk associated with each type of application is different, thus it is not effective to use the same method, technique, and testing type to test all types of application.

### #5) Defect Clustering

During testing, it may happen that most of the defects found are related to a small number of modules. There might be multiple reasons for this like the modules may be complex, coding related to such modules may be complicated etc.

This is the Pareto Principle of software testing where 80% of the problems are found in 20% of the modules. We will learn more about Defect clustering and Pareto Principle later in this article.

### #6) Pesticide Paradox

Pesticide Paradox principle says that if the same set of test cases are executed again and again over the period of time then these set of tests are not capable enough to identify new defects in the system.

In order to overcome this “Pesticide Paradox”, the set of test cases needs to be regularly reviewed and revised. If required a new set of test cases can be added and the existing test cases can be deleted if they are not able to find any more defects from the system.

### #7) Absence of Error

If the software is tested fully and if no defects are found before release, then we can say that the software is 99% defect free. But what if this software is tested against wrong requirements? In such cases, even finding defects and fixing them on time would not help as testing is performed on wrong requirements which are not as per needs of the end user.

**For Example,** suppose the application is related to an e-commerce site and the requirements against “Shopping Cart or Shopping Basket” functionality which is wrongly interpreted and tested. Here, even finding more defects does not help to move the application into the next phase or in the production environment.