| **Clustering Method using K-Means, Hierarchical and DBSCAN(Or Mall Customer Segmentation Using Unsupervised Learning)** | | | |
| --- | --- | --- | --- |
| Youtube | Blog | [Collab Notebook](https://colab.research.google.com/drive/1eFft2ELwhuerh4_jIS2_pzsEUbH2Z3Jz#scrollTo=-vfx7pG5kkAO) |  |
| [Video](https://www.youtube.com/watch?v=461Opp1TShk) | [Medium Article](https://nzlul.medium.com/clustering-method-using-k-means-hierarchical-and-dbscan-using-python-5ca5721bbfc3) |  |  |
| [video on k means ++](https://www.youtube.com/watch?v=5k-ngBquGBI) | [difference Between K means and K-means++](https://iq.opengenus.org/k-means-vs-k-means-p/) |  |  |
|  | [DB clustering](https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/) |  |  |
| Resume Points | 1. Segmented Customers into different clusters using K-Means, DBSCAN and Hierarchical Clustering 2. Obtained the optimal no of clusters as 5 using Elbow Method for K-means | | |

**Approach** which we followed in this project:

1. Import necessary libraries
2. Data analysis
3. Checked Null values: (There are no missing data. This simplifies the analysis but it is a very unlikely scenario in a real-life where analysts spend a significant amount of time cleaning their data before the core analysis is performed)
4. Then we performed EDA : This section contains a basic statistical investigation of a given database. It is a crucial point in any analysis as it allows for a better understanding of the underlying data. This part has two main sections: distributions and correlations.

### Distributions: In this distributions of numerical variables will be investigated in detail. Data will be stratified by gender - the only categorical variable.

Basically We draw histograms on the basis of Gender for :

1. For Age 2. For Annual Income 3. For Spending score

Observation from first plot :

[**Kolgomorov-Smirnov**](#g7r2nh345n04) **test p-value: 0.49**

The average age of male customers is lightly higher than female ones (39.8 versus 38.1). Distribution of male age is more uniform than females, where we can observe that the biggest age group is 30-35 years old. Kolgomorov-Smirnov test shows that the differences between these two groups are statistically insignificant. Means both data belongs to same distribution

Observation from second plot : **Kolgomorov-Smirnov test p-value: 0.78**

**For third plot :**

Kolgomorov-Smirnov test p-value: 0.29

Then we plot correlation plots in which we used pearson correlation and make following observations:

**Visualization of Age with Income:**

Observation: There are week negative correlations (<0.5) between age and spending score for both sex groups.

**Visualization of Income with spending**

Observation: There is a negligible correlation between annual income and spending score of customers for both sex groups.

After that we introduced three clustering techniques

This section of the notebook describes and demonstrates how to use three clustering algorithms:

* [K-Means](#2hu11iuwq2c4)
* [Density-Based Spatial Clustering of Applications with Noise (DBSCAN)](#os1ta88qywqz)
* Affinity Propagation.

As I tried to explain each algorithm in brief below and also mentioned sources of each in table So here i just briefly explain how each algorithms is working and how they are different from each other

1. In **K means** we are basically following three steps:

1. The first step involves the **random initialization of k** data points which are called means.
2. In this step we **cluster each data point to it's nearest mean** and after that we update the mean of the current clusters. mean: is the average of a group of values.
3. This cycle continues for a given number of repetitions and after that we have our final clusters.

Basically we have two methods to find K

1. In order to find an appropriate number of clusters, the elbow method will be used. In this method for this case, the inertia for a number of clusters between 2 and 10 will be calculated. The rule is to choose the number of clusters **where you see a kink or "an elbow" in the graph.**
2. Another way to choose the best number of clusters is to plot the silhuette score in a function of number of clusters.Silhouette score method indicates the best options would be 5 or 6 clusters.
3. Let's check for both cluster size=5 and cluster size = 6

Observation : when i applied K means cluster for cluster size =5 we obtained following results

Basically whole datapoints got divided into 5 clusters with following categories

* clients with **low** annual income and **high** spending score
* clients with **medium** annual income and **medium** spending score
* clients with **high** annual income and **low** spending score
* clients with **high** annual income and **high** spending score
* clients with **low** annual income and **low** spending score

There are no distinct groups is terms of customers age.

[**Silhouette score**](#pla94uta3lo2) **for K-means Clustering: 0.44428597560893024**

The biggest cluster is a cluster number 1 with 79 observations ("medium-medium" clients). There are two the smallest ones each containing 23 observations (cluster 3 "high-high" and cluster 0 "low-high" clients).

**Then we draw silhouette plot just to visualize silhouette score**

**Observation**: The silhouette plot will show the silhouette scores for each data point in the dataset. The x-axis represents the silhouette coefficient values, and the y-axis shows the data points sorted by their assigned cluster labels.

The width of each bar corresponds to the silhouette score of the data point.

The plot will help you assess the clustering quality and identify potential issues, such as overlapping clusters or poorly separated clusters.

Higher and wider bars indicate well-clustered data points with higher silhouette scores.

Same steps with cluster size=6

K-Means algorithm generated the following 6 clusters and obtained following results :

1. younger clients with \*\*medium\*\* annual and \*\*medium\*\* spending score
2. clients with \*\*high\*\* annual income and \*\*low\*\* spending score
3. younger clients with \*\*medium\*\* annual and \*\*medium\*\* spending score
4. clients with \*\*high\*\* annual income and \*\*high\*\* spending score
5. clients with \*\*low\*\* annual income and \*\*low\*\* spending score
6. clients with \*\*low\*\* annual income and \*\*high\*\* spending score

**2. DBSCAN**: This describes how DBSCAN (Density-Based Spatial Clustering of Applications with Noise) works and shows its implementation to the mall customers segmentation.

In DBSCAN there are two major hyperparameters:

* eps
* min\_samples

Because DBSCAN creates clusters itself based on those two parameters let's check the number of generated clusters.

In our code we checked each possible combination of eps value and min\_sample value and estimated possible number of clusters for each possible pairs of eps value and min\_sample by using[**Pivot table**](https://www.machinelearningplus.com/pandas/pandas-pivot-table-in-python/)

Then i plotted Correlation map for every possible pair of eps and min\_samples and to find their corresponding silhoutti score

Global maximum is 0.26 for eps=12.5 and min\_samples=4.

One way to find the eps value is based on the **k-distance graph.**

large value of eps means more number of datapoints will come in same cluster and hence not useful and small value of eps means big part of data points will be considered as outlier

It is difficult arbitrarily to say what values will work the best. Therefore, I will first create a matrix of investigated combinations.

Observation : DBSCAN created 5 clusters plus outliers cluster (-1). Sizes of clusters 0-4 vary significantly - some have only 4 or 8 observations. There are 18 outliers.

### Then we draw plots between Annual income and spending score

Silhouette Score for DBSCAN Clustering: 0.15189248068906694

### **Observation:** In your case, the silhouette score for DBSCAN is approximately 0.1519. This value is positive, indicating that the clustering has some level of separation between clusters, but it is relatively close to 0. It suggests that the clusters may be somewhat overlapping or not well-defined, which could mean that DBSCAN might not be finding clear boundaries between clusters in the data.

3. Hierarchical clustering: Silhouette Score for Agglomerative Clustering: **0.4401889024290193**

**Comparisons**: A table below shows sizes of created clusters. Please note that numbering of clusters is different in each method, e.g. cluster no.0 in K-Means is equivalent of cluster no.2 in DBSCAN and no.2 in Affinity Propagation.

**Insights of Result :**

# **K means insights:**

The negative cluster label (-1) is not used in K-Means since all data points are assigned to a cluster. K-Means has divided the data into six clusters, with varying numbers of data points in each cluster.

# **Insight: DBSCAN**

It has identified five clusters (labeled 0 to 4) and assigned 18 data points as noise points (labeled as -1). Noise points are data points that do not belong to any cluster and are considered outliers.

# **Insight: Agglomerative Clustering**

It has divided the data into five clusters (labeled 0 to 4).

From the above comparisons, it is clear that DBSCAN failed to generate reasonable clusters. It is most likely because DBCSAN tries to find clusters based on the density of points. If one of our clusters is less dense than others DBSCAN will produce suboptimal results by not recognising the least dense group as a cluster.

In turn, K-Means and Affinity Propagation algorithms created reasonable 6 clusters.

### 

### **Extra Theory about each mechanism**

### **K-Means**

The most well-known **partitional clustering algorithm** is K-Means.It gained great popularity because of its ease of implementation, simplicity and many empirical successes (e.g. in business, medicine and science).

There are **3 main steps in K-Means algorithm** (known also as Lloyd’s algorithm):

1. The first step involves the random initialization of **k** data points which are called means.
2. In this step we cluster each data point to it's nearest mean and after that we update the mean of the current clusters. **mean**: is the average of a group of values.
3. This cycle continues for a given number of repetitions and after that we have our final clusters.

**Some of the important points about K means clustering**

This is known as NP-hard problem, meaning this is a **greedy algorithm** and converges to the local minimum.

The computational cost of Lloyd’s K-Means algorithm is O(kn), where k is a number of clusters and n is a number of samples. This is not bad when compared with other clustering algorithms.

Despite converging usually to a local minimum, **K-means is relatively fast** and when groups are well isolated from each other it is likely that it converges to the global minimum.

Because the result of clusterisation depends on the initialisation criteria it is common to run the analysis for various initialisation points and choose the one with minimum resultant inertia. There are some improvements to the algorithm solving problem of the local minima. One, exemplary, improvement is to use enhanced Firefly Algorithms about which you can read [here](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fwww.sciencedirect.com%2Fscience%2Farticle%2Fpii%2FS1568494619305447&link_redirector=1).

In general, a user of the K-Means algorithm is required to define three main parameters:

1. **Initialisation criteria**In scikit-learn, a clever initialisation scheme is implemented**:** [**“k-means++”**](#z8lrhsiw27uz) proposed by Arthur and Vassilvitskii. It creates initial centroids generally distant from each other increasing probability of obtaining better results. There is also a possibility to use a random point’s generator. There are ongoing efforts to create the most efficient seeding method for K-Means algorithm, one of them is based on Independent Component Analysis and you can read more about it [here](https://colab.research.google.com/corgiredirector?site=http%3A%2F%2Fwww.ymd.nii.ac.jp%2Flab%2Fpublication%2Fconference%2F2010%2FIWI-Onoda-2010.pdf&link_redirector=1).
2. **Number of clusters** One of the simplest and the most popular one is the **elbow method** shown in this analysis. Additionaly a **silhouette score** will be used as well. There are also other, often advanced, options for choosing the optimal number of clusters (however, not used in this notebook and not implemented in sklearn), e.g.:

* Minimum Message Length (MML) - [more](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FMinimum_message_length&link_redirector=1)
* Minimum Description Length (MDL) - [more](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FMinimum_description_length&link_redirector=1)
* Bayes Information Criterion (BIC) - [more](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FBayesian_information_criterion&link_redirector=1)
* Akaike Information Criterion (AIC) - [more](https://colab.research.google.com/corgiredirector?site=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FAkaike_information_criterion&link_redirector=1)
* Dirichlet Process
* Gap statistics

1. **A distance metric (not required in scikit learn implementation)**There are various options to calculate the distance between points. The most popular one is simply the Euclidean metric and it is the one implemented in scikit-learn. It is often called **spherical k-means model.** It has a drawback that it finds spherical-like groups only and tends to become inflated in highly multi-dimensional analyses (“curse of dimensionality”). There are other options but not implemented in scikit-learn, e.g.:

* Mahalonobis distance (high computiational cost)
* Itakura-Saito distance
* L1 distance
* Cosine distance
* Bregman distance

**There are numerous ongoing researches and variations proposed to K-Means, e.g.**:

* K-Medoid where the centroid is defined as the most centrally located object)
* K-Median where the centroid is calculated using median instead of a mean,
* Fuzzy C-means model

**Some take-aways about K-Means:**

1. Euclidean distances are used
2. Number of clusters has to be defined for the algorithm
3. Centroid is calculated using mean distance to cluster members
4. Clusters are assumed isotropic and convex
5. Stochastic algorithm – results depend on the initialisation criteria
6. Creates groups of equal variance (minimises inertia)
7. Prone to the “curse of dimensionality”
8. Can be run in parallel – so it scales well

### Use case - customers segmentation

The very first step is to decide value of k which will be decided by using elbow method

## **Algorithm of K++:**

1. In the starting we have to select a random first centroid point from the given dataset.
2. Now for every instance say 'i' in the dataset calculate the distance say 'x' from 'i' to the closest, previously chosen centroid.
3. Select the following centroid from the dataset with the end goal that the likelihood of picking a point as centroid is corresponding to the distance from the closest, recently picked centroid.
4. Last 2 steps are repeated until you get k mean points.

**Silhouette Coefficient:**

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

1 means clusters are well apart from each other and clearly distinguished

0 means distance between clusters insignificant

-1 means clusters assigned in wrong way

Formula =(b-a)/max(a,b)

Where a is avg intra dis and b is avg inter dist

Best score is high silhoutti score means high inter cluster distance hence clusters are well separated

**DBSCAN** stands for Density-Based Spatial Clustering of Applications with Noise and is one of clustering algorithms implemented in scikit-learn library

As the name of paper suggests the core idea of DBSCAN is around concept of dense regions

.

The assumption is that **natural clusters are composed of densely located points**. This requires definition of “dense region”. To do these two parameters are required for DBSCAN algorithm.

* Eps, ε - distance
* MinPts – Minimum number of points within distance Eps

Optionally the distance metric can be specified by a user, but **usually Euclidean distance** is implemented (like in scikit learn)

.

**Some important terms:**

**Epsilon** : If a point is epsilon , it will be radius of specific circle

**Core Point**: A point is a core point if there are atleast minpoints no of points(including point itself) in its surrounding area with radius eps.

**Border point:** a point is a border point if it is reachable from a core point and there are less than minpts no of points within surrounding area

**Noise point/outlier:** if it is not a core point and not reachable from any core point

Advantages of this approach:

* it finds number of clusters itself, based on eps and MinPts parameters
* It it able to differentiate elongated clusters or clusters surrounded by other clusters in contrary to e.g. K-Means where clusters are always convex.
* It is also able to find points not fitting into any cluster – it detects outliers.

**The biggest drawback of DBSCAN:**

* High computational expense of average O(n log(n)) coming from a need to execute a neighbourhood query for each point.
* Poorly identifies clusters with various densities

There are various variations and extensions proposed by machine learning researchers to DBSCAN algorithm. For example

**The Kolmogorov-Smirnov test** (KS test) is a statistical test used to compare

1. A sample distribution to a theoretical probability distribution or
2. To compare two sample distributions.

**Aim: It assesses whether the sample data comes from a specific distribution or whether two samples come from the same distribution.**

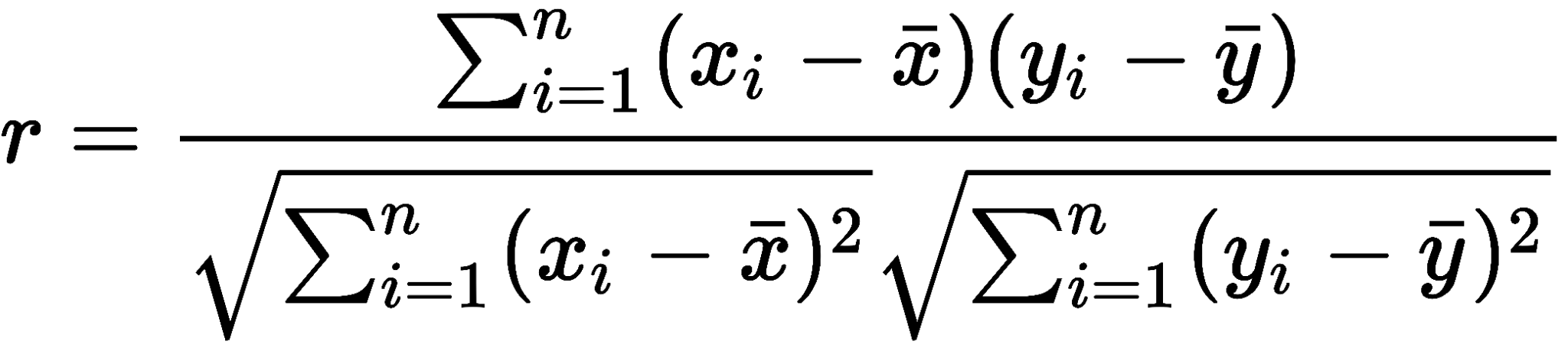
The **p-value** in the Kolmogorov-Smirnov test is a probability value that measures the **strength of evidence against the null hypothesis**.

Here's how the p-value is interpreted in the context of the KS test:

* 1. If the **p-value is large** (e.g., **greater than 0.05** or the chosen significance level), it suggests that there is not enough evidence to reject the null hypothesis. In other words, the sample data appears to be consistent with the theoretical distribution or the two samples likely come from the **same distribution.**
* 2. If the **p-value is small** (e.g., **less than or equal to 0.05** or the chosen significance level), it suggests that there is strong evidence to reject the null hypothesis. In this case, we conclude that the sample data significantly deviates from the theoretical distribution, or the two samples are likely drawn from **different distributions.**

the interpretation of the p-value is subject to the chosen significance level (alpha). Commonly used significance levels are 0.05 (5%) or 0.01 (1%), but they can vary depending on the specific analysis and domain.

**Pearson correlation coefficient**, often denoted as **"r**," is a statistical measure that quantifies the **strength and direction of the linear relationship between two continuous variables**. It is widely used in statistics and data analysis to assess the degree of association or correlation between two variables.



* The Pearson correlation coefficient takes values between -1 and 1:
* When "r" is close to +1, it indicates a strong positive linear correlation. This means that as one variable increases, the other variable also tends to increase in a linear fashion.
* When "r" is close to -1, it indicates a strong negative linear correlation. This means that as one variable increases, the other variable tends to decrease in a linear fashion.
* When "r" is close to 0, it indicates a weak or no linear correlation. This means that there is little or no linear relationship between the two variables.

The Pearson correlation coefficient is **sensitive to the scale of the data**, and it assumes that the relationship between the two variables is approximately linear.

If the relationship between the variables is non-linear, the correlation coefficient may not accurately capture the strength of the association.