**Data Mining and Machine Learning**

**Kamaljeet Kaur Sidhu**

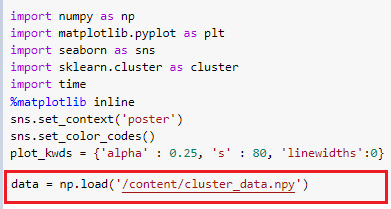
**Student ID- 19489382**

**Module code -7BUIS008W**

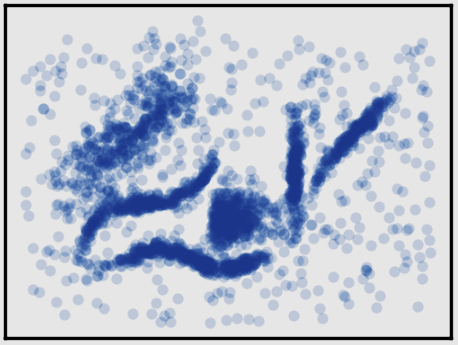
# Python environment for testing sklearn clustering algorithms

* **Load dataset**

To load the dataset and further perform clustering, there is a requirement to import some necessary libraries from sklearn packages. Moreover to have a visual representation of different clustering algorithms matplotlib needs to be imported/called.



* **Scatterplot of ‘*cluster\_data.npy*’**

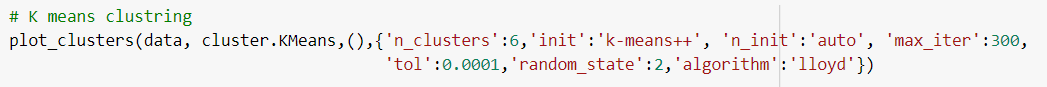


As per my analysis of the above displayed scatterplot, there are six clearly visible clusters. So I expect the different algorithms to detect six clusters as well.

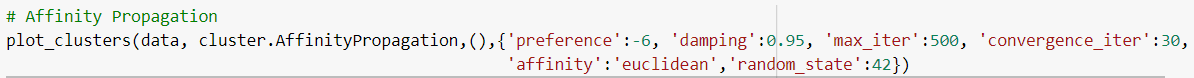
* **Establishment of *plot\_clusters(data, algorithm, args, kwds)* utility function:**

The parameters included in the plot\_cluster() function are considered important. As changes in the values effect the output of clustering algorithms.

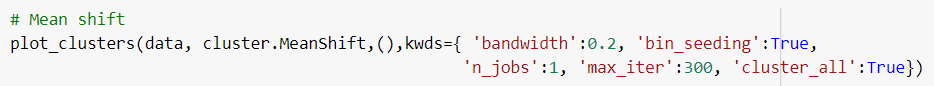
**K-Means:**



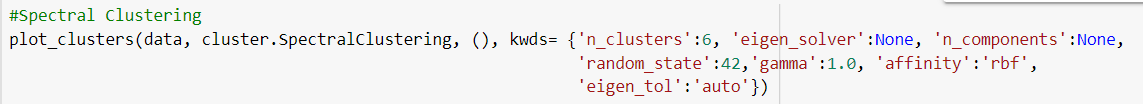
**Affinity Propagation:**



**Mean shift:**



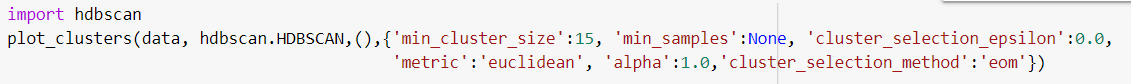
**Spectral Clustering:**



**Agglomerative Clustering:**



**HDBSCAN**



# 2.

* **K-Means**

|  |  |  |
| --- | --- | --- |
| **K-Means** | | |
| **Parameters** | **Possible values** | **Justification** |
| n\_clusters | 6 | I have detected six clusters from my naked eye. So I will pass the same number of clusters to clustering algorithm. |
| init | {‘k-means++’, ‘random’} | **k-means++** performs a sampling based on an empirical probability distribution of points and then selects one of them as an initial cluster centroid.  **Random** for initial centroids, it just choose the n\_clusters rows from data randomly. |
| n\_init | ‘auto’ or int | The number of times k-means algorithm will run with different centroids depends on the value initialized in init.  If init=’k-means++’ then the value of n\_init will be 1. However, if init=’random’ then the value of n\_init will be 10. |
| max\_iter | default=300 | For a single run, the maximum number of iterations of k-mean algorithm will be equal to the number specified to max\_iter.  The higher the value of max\_iter will be, it will guarantee that the entire feature space has explored. But most of the times it comes at the cost of diminishing returns. |
| tol | 0.0001 | The low value of tol ensures that the entire feature space has scnned. However the higher value of tol means that we are willing to tolerate a larger change of loss. |
| random\_state | 42 | The number passed to random\_state determines the number of ranom generations for centroid initialization. It can be any number. |
| algorithm | ‘lloyd’,’elkan’ | ‘lloyd’ is a default value of algorithm also it is the classical EM-style. However, for some datasets ‘elkan’ variation can be more accurate. |

***(SK LEARN K-Means, 2023)***

* **Affinity Propagation**

|  |  |  |
| --- | --- | --- |
| **Affinity Propagation** | | |
| **Parameters** | **Possible values** | **Justification** |
| preference | Range[-4,-6] | Affinity propagation will detect many exemplars (clusters) if the value passed for preference is high. However, if it’s low then it will detect small number of exemplars (clusters). An optimal choice for preference is minimum similarity. |
| damping | Range[0.5,1.0) | The current value is maintained relative to the incoming value when damping factor lies between 0.5 and 1. |
| max\_iter | default=300 | For a single run, the maximum number of iterations of k-mean algorithm will be equal to the number specified to max\_iter.  The higher the value of max\_iter will be, it will guarantee that the entire feature space has explored. But most of the times it comes at the cost of diminishing returns. |
| affinity | ‘euclidean’, ‘precomputed’ | ‘euclidean’ is a defult value of affinity and it considers the negative squared euciledan distance between points. |
| random\_state | 42 | The number passed to random\_state determines the number of random generations for centroid initialization. It can be any number. |
| **convergence\_iter** | Range[15,30] | Convergence\_iter referes to the count of iterations with no change in the count of estimated clusters that stops the convergence. |

***(SKLEARN AffinityPropagation, 2023)***

* **Mean\_Shift**

|  |  |  |
| --- | --- | --- |
| **Mean Shift** | | |
| **Parameters** | **Possible values** | **Justification** |
| Bandwidth | Range[0.1,0.9] | It is used to describe the region on the plot to form clusters. There is no need to specify the number of clusters as an algorithm manages to detect it automatically. |
| Bin\_seeding | ‘True’ | In order to increase the speed of algorithm, the value of bin\_seeding needs to be ‘True’. Because fewer seeds will be initialized when its ‘True’. |
| n\_jobs | 1 | To get results faster, parallel processing is used as it gives access to all the cores of the CPU while execution. |
| Max\_iter | default= 300 | For a single run, the maximum number of iterations of k-mean algorithm will be equal to the number specified to max\_iter.  The higher the value of max\_iter will be, it will guarantee that the entire feature space has explored. But most of the times it comes at the cost of diminishing returns. |
| Cluster\_all | ‘True’ | All points will be clustered, even those orphans that are out of the kernel’s range. |

***(SKLEARN MeanShift, 2023)***

* **Spectral Clustering**

|  |  |  |
| --- | --- | --- |
| **Spectral Clustering** | | |
| **Parameters** | **Possible values** | **Justification** |
| n\_clusters | 6 | I have detected six clusters from my naked eye. So I will pass the same number of clusters to clustering algorithm. |
| eigen\_solver | ‘arpack’, ‘lobpcg’, ‘amg’ | To use ‘amg’ as an eigen\_solver pyamg needs to be installed. It may leads to some instabilities when large or sparse problem is give but will provide faster results. If this is not the case, two other values can be used. By default it is set to ‘None’. |
| n\_components | int, default=None | n\_components refers to the count of eigenvactors to be used in the clustering. |
| random\_state | 42 | The number passed to random\_state determines the number of random generations for centroid initialization. It can be any number. |
| affinity | ‘rbf’ | A radial basis function cluster kernel is used to make the affinity matrix. By default it is set to ‘rbf’ only. |
| gamma | default=1.0 | It works as a kernel coefficient for most of the kernels including rbf. So the default value is passed to the kernel as affinity is set to ‘rbf’. |
| eigen\_tol | float, default=”auto” | The tolerance will depend on the value of eigen\_solver if eigen\_tol is set to ‘auto’.  eigen\_tol needs to be 0.0 if eigen\_solver is ‘arpack’. However, it needs to set to None when the value of eigen\_solver is either ‘lobpcg’ or ‘amg’. |

***(SKLEARN SpectralClustering, 2023)***

* Agglomerative Clustering

|  |  |  |
| --- | --- | --- |
| Agglomerative Clustering | | |
| **Parameters** | **Possible values** | **Justification** |
| n\_clusters | 6 | I have detected six clusters from my naked eye. So I will pass the same number of clusters to clustering algorithm. |
| **linkage** | ‘ward’, ‘complete’, ‘average’, ‘single’}, default=’ward’ | The distance to be used between set of rows/observations is determined by the linkage centroid. The algorithm will combine the pairs of clusters that minimize this centroid.  To minimize the variance of clusters being combined, ‘ward’ is an optimal value to set for linkage. |

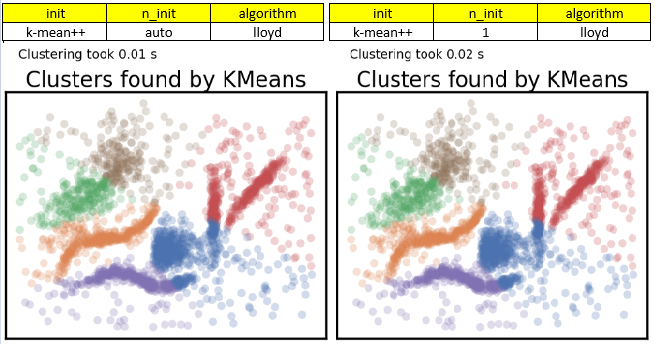
*(SKLEARN AgglomerativeClustering, 2023)*

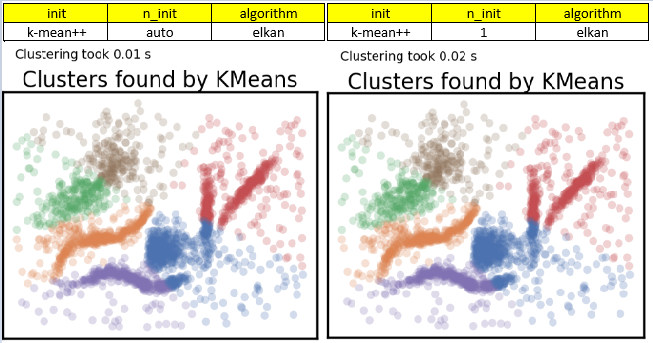
* HDBSCAN

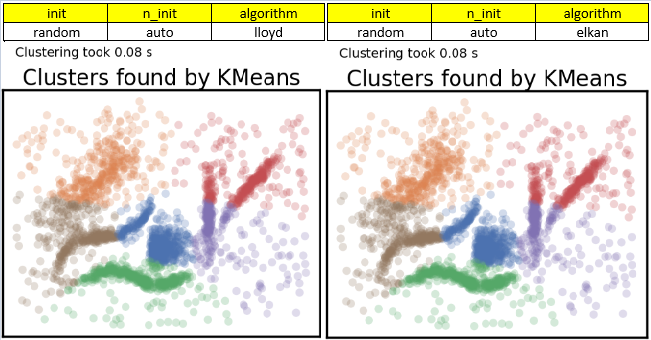
|  |  |  |
| --- | --- | --- |
| **HDBSCAN** | | |
| **Parameters** | **Possible values** | **Justification** |
| min\_cluster\_size | Range[15,30] | min\_cluster\_size is the primary parameter that effects the resulting of clustering algorithm. If the value of min\_cluster\_size is increased then the algorithm tries to combine the clusters and make it one where possible. Different set of values can be tried to analyse where the algorithm is capable to detect the desired number of clusters. |
| Min\_samples | None,1 | It has an integral effect on the output of clustering algorithm. The more conservative the clustering will be, when the value of min\_samples is larger. Also, more points will be declared as noise. |
| Cluster\_selection \_epsilon | 0.0 | Cluster\_selection\_epsilon ensures that the algorithm will not split up the clusters further, when the specified value of threshold is reached. The selection of Cluster\_selection\_epsilon value depends on the distance among data points. |
| alpha | default=1.0 | It is another important parameter to be considered for clustering. But in practice it is not good to play with this parameter. So the default value is used for resulting. |
| Cluster\_selection  \_method | ‘eom’,’leaf’ | It determines the selection process of flat clusters from the cluster tree hierarchy. ‘eom’ is the default value of this parameter but it’s not always the desirable approach for cluster selection. However, ‘leaf’ is a better option as it will select leaf nodes from tree, generating many small homogeneous clusters. |

***(SKLEARN parameter\_selection, 2023)***

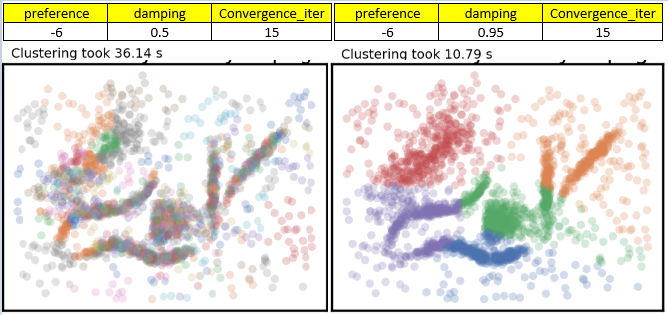
# 3. K-Means Clustering

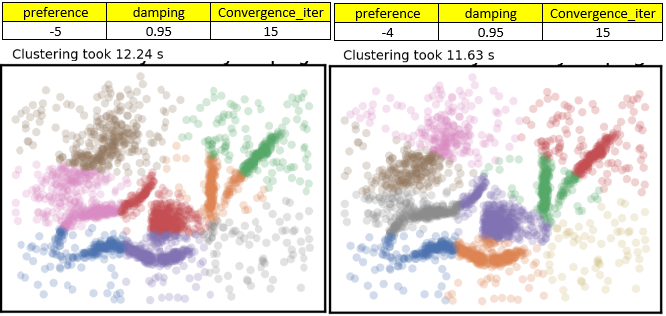




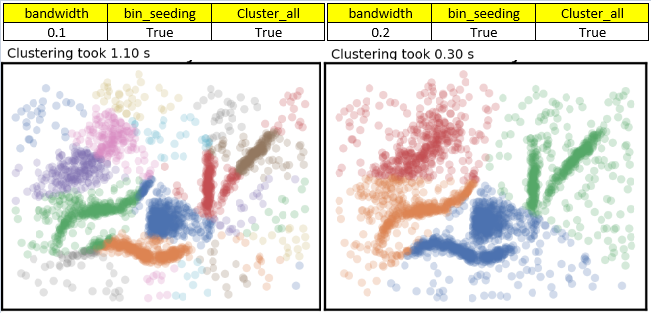


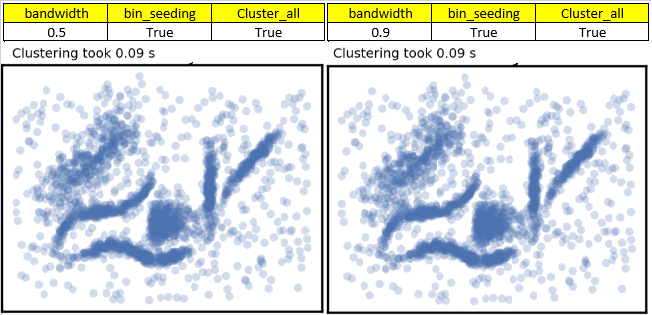
**Affinity Propagation**



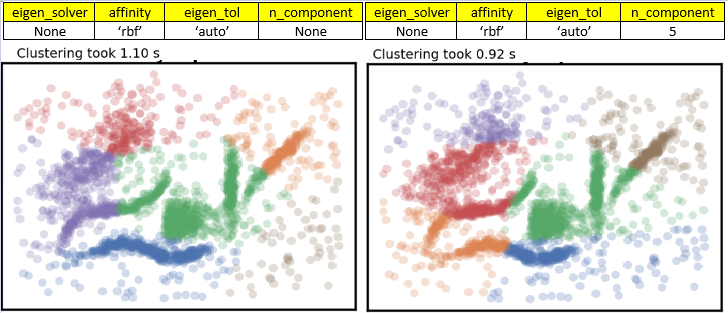


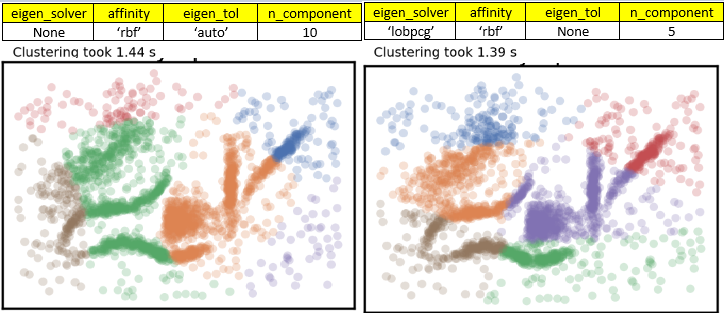
**Mean Shift**

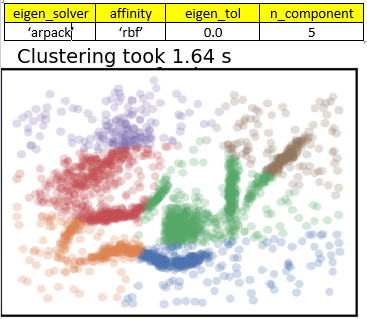




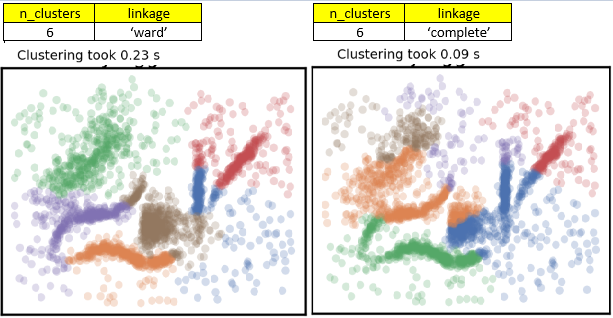
**Spectral Clustering**

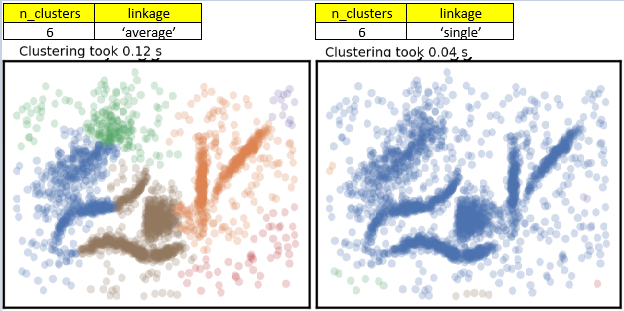




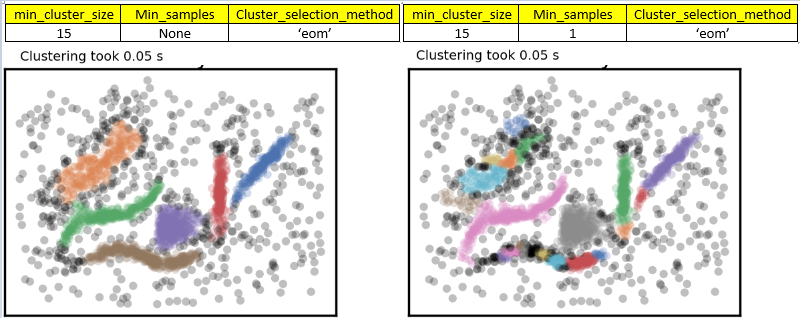


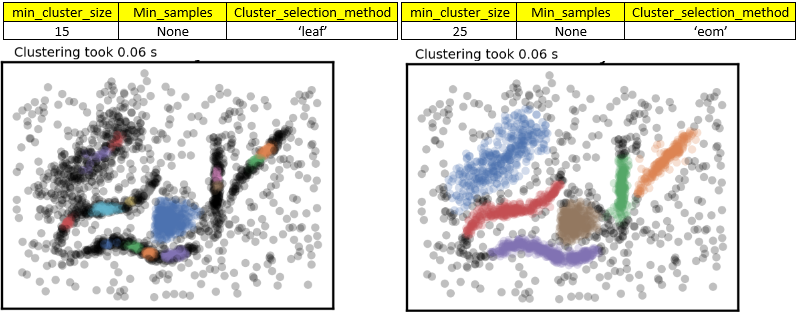
* Agglomerative Clustering

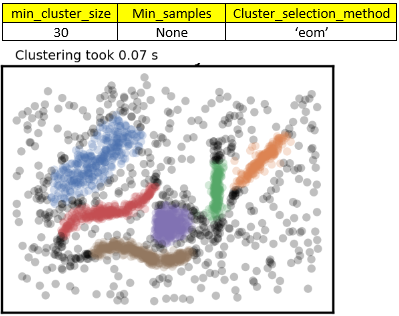




**HDBSCAN**







# 4.

# Technical Report

There are six different clustering algorithms used to perform the same task. This report includes the information about four different domains based on each algorithm, optimal values passed to the parameters of plot\_clusters() utility function for performing clustering, Intra cluster and Inter Cluster Distance of each algorithm with optimal values and the final scatter plot resulted by the implementation.

* **K-Means:**

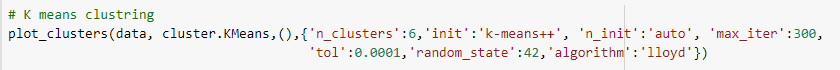
**Do not be wrong:** K-Means work randomly, as it just group the points into one cluster randomly rather than focusing on whether they belong to that group or not.

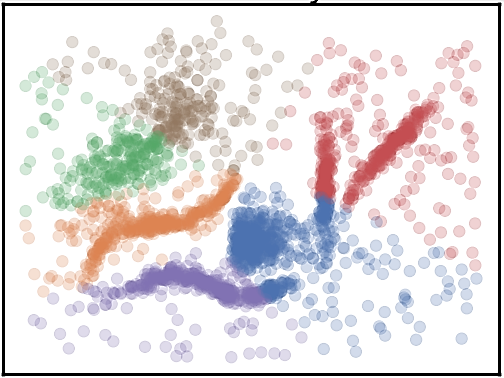
**Intuitive Parameters:** It works efficiently as I already know the number of clusters my dataset has and what I am expecting from the algorithm.

**Stability:** During multiple runs and checks based on different values of parameters, it has been noticed that, the algorithm remain stable.

**Performance:** It is noticed to be a good clustering algorithm after analysing different resulted plots. Moreover, with right selection of parameters and values its performance can be enhanced. As compared to some other clustering algorithms K-Means is more appropriate choice for clustering.

**I have selected the optimal values for the parameters after multiple trials by using different set of possible values.** Most of the parameters, which do not make much difference in the output are set to default values only. However, I have considered ***init, n\_init and algorithm*** as an important parameters that makes difference in the clustering after each run when values are changed.

 ***(SK LEARN, 2023) (SK LEARN K-Means, 2023)***



From the resulted scatterplot it can be analysed that in some regions Intra cluster distance is maximum for each cluster which is a good point to be considered. But the outliers which belong to the same cluster are far away from its centre or other relative points. However, intercluster distance is minimum. Resultantly, K-Means is not an option to select for clustering.

* **Affinity Propagation:**

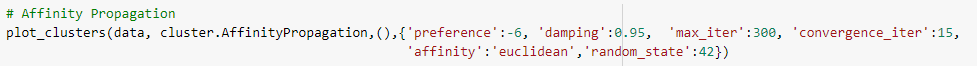
**Do not be wrong:** Affinity propagation just work the similar way as K-Means do. Like, it just perform clustering randomly rather than logically passing the points to closest cluster. Because it makes an assumption that clusters are globular.

**Intuitive Parameters:** The core parameters of this algorithm are preference and damping instead of number of clusters, as it is in K-Means. But it is a complex task to get the optimal value of preference. I tried multiple times for the same and got to know about (-6) where I found better results as compared to other values.

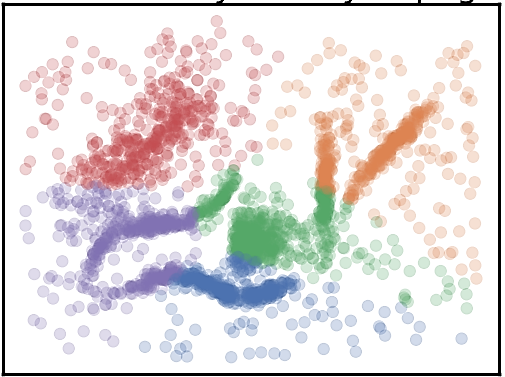
**Stability:** Affinity propagation works like the resulted output for clusters each time is the only possible option (as per the parameter values) and it cannot be improved by just running it multiple times.

**Performance:** It takes more time for displaying results as compared to K-Means. As it takes (11-37) seconds for execution, however K-Means just do the same work within (1-2) seconds.

The optimal values of the core parameters of this algorithm named ‘preference’ and ‘damping’ are selected after testing a set of possible values.



***(SKLEARN AffinityPropagation, 2023)***



It is visible from the plot that this algorithm is not capable to detect six clusters as I have detected from my necked eye.

As same as K-Means the Intra cluster distance is maximum and intercluster distance is minimum in some regions. So, affinity propagation is also not a good algorithm for this task.

* **Mean Shift:**

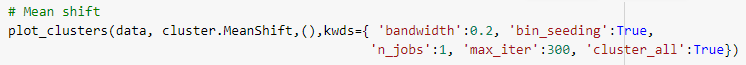
**Do not be wrong:** It has been analysed from various scatterplots of Mean shift clustering with different parameter values that it does not target to make every point a cluster. It clusters one point and leave the others even if it has a minor distance from the first. Resultantly, it can result less than the expected number of clusters. As, it happened in our case the number of clusters I detected are six, but algorithm is just capable to detect five only when the optimal value is passed to the core parameters.

**Intuitive Parameters:** The parameters of plot\_scatters() in Mean Shift make some sense, each time the value is changed.

**Stability:** The clustering output changes frequently, whenever the value of bandwidth is changed. It displays completely

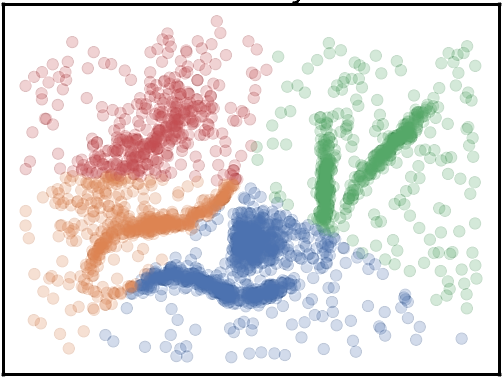
Different result even if there is a minor change in the bandwidth value.

**Performance:** It takes longer time to implement results.



***(SKLEARN MeanShift, 2023)***

‘Bandwidth’ has a great impact on the algorithms clustering performance. Most of the trial got failed as the algorithm is not capable to work with that specified bandwidth (0.1, 0.5 and 0.9). After many trials 0.2 is proven as optimal value for the algorithm.



Same as previous two algorithms the performance of intra cluster distance and inter cluster distance is same. So, this method as well is failed to perform desired clustering. Also, it is just capable to detect four clusters.

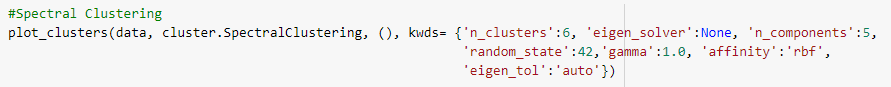
* **Spectral Clustering:**

**Do not be wrong:** Having some changes in values makes a difference but it gives the clustering output somewhat close to the desirable results.

**Intuitive Parameters:** As same as K-Means, the knowledge of the number of clusters a dataset has is required.

**Stability:** Almost same as of K-Means.

**Performance:** It implements the clustering slowly. As it takes approximately more than 1.0 seconds for execution.

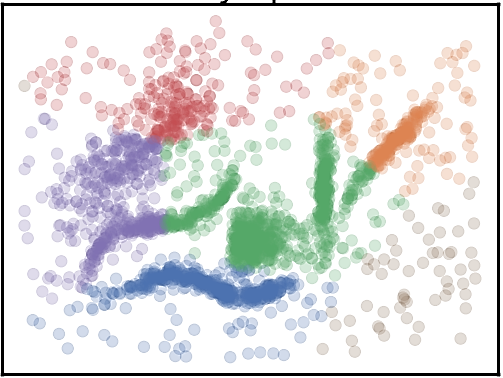


***(SKLEARN SpectralClustering, 2023)***

For ‘eigen\_solver’ and ‘eigen\_tol’ just default values are taken. Whatever the difference in the outcome is noticed is due to the value of n\_components.

From the scatter plot attached below it can be noticed that this algorithm is at least capable to detect six clusters. However, one of them in the bottom right corner is too small.

There is no improvement in the intra cluster and inter cluster distance of this algorithm as compared to the previous algorithms.



* Agglomerative Clustering

Do not be wrong: It is assumed that all the data available in the clusters is noise free.

**Intuitive Parameters:** Again same as K-Means the knowledge of number of clusters of dataset is required.

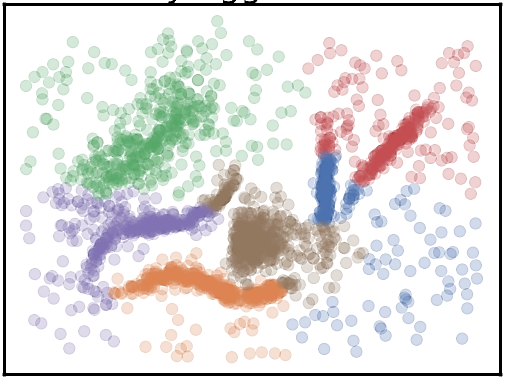
**Stability:** Agglomerative clustering has good stability as observed over the number of runs.

**Performance:** Performance can be good if its implementation is appropriate.



***(SKLEARN AgglomerativeClustering, 2023)***

The default value is assigned to the parameter ‘linkage’ and this the only value that is leading to good output as compared to other values for this parameter.



There is no improvement in the intra cluster and inter cluster distance of agglomerative clustering algorithm as compared to the previous algorithms.

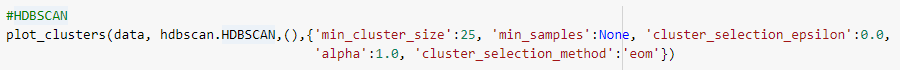
* **HDBSCAN:**

**Do not be wrong**: It is the strongest option as compared to others for ‘Do not be wrong’ segment.

**Intuitive Parameters:** It is not much sensitive to parameter values and some sensible default values can be used for this clustering algorithm.

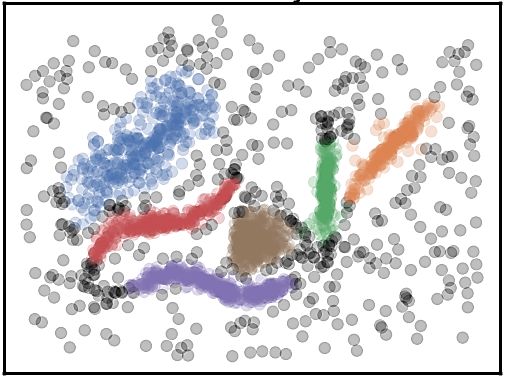
**Stability:** As analysed from different scatterplots, it remains stable over parameter values and multiple runs.

**Performance:** HDBSCAN is efficient in clustering. As, it has detected almost the exact number of clusters as I detected by myself. Also the boundaries’ of the clusters are somewhat similar to the normal clusters of the dataset.



***(SKLEARN parameter\_selection, 2023)***

The value assigned to ‘min\_cluster\_size’ is making a difference in the performance, however ‘Min\_samples’ is set to default value and optimal Cluster\_selection\_method is ‘eom’.



As compared to all other clustering methods HDBSCAN is the best clustering algorithm as it is capable to detect six clusters, even in the same pattern as I detected by myself.

Moreover, the Intra Cluster Distance is minimum for each cluster. While, Inter Cluster Distance is Maximum.

**Best Clustering Minimum Intra Cluster and Maximum Inter Cluster**

In this way HDBSCAN is the optimal algorithm for the task of clustering.

# References

*SK LEARN*. (2023, 04). Retrieved from SK LEARN: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

*SK LEARN K-Means*. (2023). Retrieved from SK LEARN: 7BUIS008W

*SKLEARN AffinityPropagation*. (2023, 4). Retrieved from SKLEARN AffinityPropagation: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html

*SKLEARN AgglomerativeClustering*. (2023, 4). Retrieved from SKLEARN AgglomerativeClustering: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

*SKLEARN MeanShift*. (2023, 4). Retrieved from SKLEARN MeanShift: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html

*SKLEARN parameter\_selection*. (2023, 4). Retrieved from SKLEARN parameter\_selection: https://hdbscan.readthedocs.io/en/latest/parameter\_selection.html

*SKLEARN SpectralClustering*. (2023, 4). Retrieved from SKLEARN SpectralClustering: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.SpectralClustering.html

*SKlLearn*. (n.d.). Retrieved from SKLearn: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html