# Intelligent Systems

# Clustring Methods

Instructor : Prof.Hosseini

By: Hamidreza AliakhbariKhoie, Sahand Khoshdel, Narjes Noorzad

## TABLE OF CONTENTS

**ABSTRACT** ..... **3**
**A COMPREHENSIVE SURVEY OF CLUSTERING AGLGORITHMS (OUTLINE)** ..... **3**
**INTRODUCING COMMON CLUSTERING ALGORITHMS** ..... **15**
**TUNING ALGORITHM** ..... **25**
**ALGORITHM COMPARISON USING METRIC INDICATORS** ..... **#**
**CONCLUSIONS** ..... **65**

# Abstract

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups. In simple words, the aim is to segregate groups with similar traits and assign them into clusters.

## A Comprehensive Survey of Clustering Algorithms (Outline)

I. ABSTRACT

A. Clustering's Role in Data Analysis

1. Clustering as the basic composition of 'Data Analysis'
2. Increase in Clustering Tools along Information increase

B. General Analysis of Clustering Algorithms

1. Definition of Clustering (What's a Clustering Algorithm?)
2. Basic Elements of Clustering (Similarity Metrics & Evaluation Indicators)
3. Analyzing Algorithms from Traditional/Modern Perspective

II. INTRODUCTION

A. Clustering's Role

1. Most important question of unsupervised learning
2. Dealing with data partitioning in unknown area
3. Basis for further learning

B. Definition of Clustering

1. 2. Algorithm Output should be: (as much as possible…)

- Similar instances within clusters (low intra-distance)
- Different instances between clusters (high inter-distance)

C. Clustering Steps

1. Feature Extraction and Selection
   - Select most representative features (independent, less noisy)

2. Designing Algorithm
   - Design according to the characteristics of your problem

3. Result Evaluation
   - Evaluate and Compare using metrics
   - Judge validity of algorithm

4. Result Explanation
   - Practically explaining how and why data is clustered according to your algorithm

## III. DISTANCE AND SIMILARITY

A. Play an important role as the basis of constructing each clustering algorithm

B. Distance for Quantitative features, Similarity for Qualitative features

C. Typical Distance functions:

- Minkowski (p-norm) distance
- Standardized Euclidean Distance (scaled by the std of the cluster)
- Cosine Distance (based on dot product)
- Pearson Correlation Distance (1- Pearson Corr.)
- Mahalanobis Distance (based on Cluster's Covariance Matrix)

D. Typical Similarity functions:

- Jaccard Similarity (Ratio of Intersection to Union)
- Hamming Similarity (Info. Theory Metric, Opposite of Hamming Distance)
  - Minimum number of substitutions to change one datapoint to another
- Dichotomous Feature Similarity (For mixed data)

## IV. EVALUATION INDICATOR

A. Main purpose: Testing Algorithm Validity

B. 2 Types of indicators based on supervised/unsupervised learning:

1. Internal indicators (based on intra/inter distance)
2. External indicators (based on training labels imported with data)

C. Internal Evaluation indicators:

- takes internal data to test algorithms validity
- can't absolutely judge which algorithm is better
- 3 Typical ones:
  - Davies-Bouldin Indicator (based on ratio of distance to centroid to pairwise distances)
  - Dunn Indicator (based on ratio of inter-cluster distances to complete link intra-cluster distance)
  - Silhouette Coefficient (based on averaging distance in and between clusters)

D. External Evaluation indicators:

- Called as a gold standard for testing method
- Uses external data(labels) to test clustering validity
- Not completely correct
- 6 Common ones:
  - Rand indicator (Ratio of correct predictions to all)
    = (TP + TN) /all

- F indicator (Weighted Precision Recall Mix)
  $= \frac{(\beta^2+1).P.R}{\beta^2.P+R}$

- Jaccard indicator (Measuring Similarity of two sets)
  $= \frac{TP}{TP+FP+FN}$

- Fowlkes-Mallows indicator (Combining P & R)
  $= \sqrt{P.R}$

- Mutual Information (Information Clusters share)
  - Able to detect non-linear correlation

- Confusion Matrix
  - Demonstrating class confusions in multi-class datasets
  - Used to figure out difference from gold-standard (state of the art) clustering

## V. TRADITIONAL CLUSTERING ALGORITHMS

### A. Clustering Based on Partition

1. Common Algorithms:

- K-means (suitable for dense clusters)
  - Using iterative computation for updating centroids

- K-medoids (suitable for dense)
  - Medoid is the nearest point to clusters center

2. Other Algorithms:

- PAM
- CLARA
- CLARANS
- AP (Affinity Propagation) can be a partitioning algorithm

### B. Clustering based on Hierarchy

1. Basic Idea:

   - Construct hierarchical relationship among data for clustering

   - Based on merging/dividing clusters from individual data points to a single supercluster

2. Typical Algorithms:

   - BIRCH
     - Constructing full clustering tree (CF)
     - Each node representing a subcluster

   - CURE
     - Suitable for large-scale data
     - Uses random sampling- cluster samples separately, then integrate

   - ROCK
     - CURE's improved version
     - Deals with enumerated data(?)
     - Considering similarity from data around cluster

   - Chameleon
     - Dividing initial data to small clusters based on nearest neighbor graphs
     - Small clusters merge and become bigger based on Agglomerative fashion
     -

C. Clustering based on Data Distribution

1. Basic Idea:

   - If original data has been sampled from several distributions, data that coming from common distribution, belong to same cluster

2. Typical Algorithms:

   - DBCLASD
     - Assign nearest datapoint to a cluster to it expected distance distribution from inner datapoints are satisfied (?)

   - GMM

- Data Distribution is a Mixture of Gaussians
- Based on Maximizing ML/MAP measures for any data point to assign to a certain gaussian

D. Clustering based on Density

1. Basic Idea:

   - Data located in dense regions of data space, belong to the same cluster

2. Typical Algorithms:

- DBSCAN
  - Most well-known density-based algorithm
  - Assigning "Core, Border, Noise" tags to points based on density regions in a specific radius with specific threshold
- OPTICS
  - Improving DBSCAN's sensitivity to 2 initial parameters (min-points, radius)
- Mean-Shift
  - Locates local maximums of a density-based function by shifting the mean of a distribution repeatedly after estimation (?)

E. Clustering based on Graph Theory

1. Basic Idea:

   - Clustering is realized on a graph
   - Nodes represent datapoints
   - Edges represent relationships among data points

2. Typical Algorithms:

- CLICK
  - Carry out minimum weight division of the graph with iterations in order to generate clusters.

- MST-based clustering
  - Generates Minimum Spanning tree from the data graph as the key step to cluster analysis

F. Clustering based on Model

1. Basic Idea:

- Select a Particular Model for each cluster, find best fitting for the model

2. Typical Algorithms:

a. Based on Statistical Learning

- COWEB
  - Build a classification tree based on some heuristic criteria (?)
  - Perform hierarchical clustering along the tree, assuming independent attribute distributions

- GMM

b. Based on Neural Networks

- SOM
  - Build a mapping of dimension reduction from high dimensional input space to low dimensional output space, assuming input data contains topology (?)

- ART
  - An incremental algorithm, generating a new neuron for the network, dynamically to match a new pattern and create a new cluster when current neurons are not enough

## VI. MODERN CLUSTERING ALGORITHMS

A. Clustering based on Spectral Graph Theory

1. Basic Idea:

- Transform Clustering Problem to Graph Partitioning Problem

- Objects represent vertices
- Similarity among objects represent weighted edge

- Key idea in graph partitioning is finding a method to:

    - Decrease weight of connection between different groups

    - Increase weight of connections among the edges within a group

2. Typical Algorithms:

- Recursive Spectral Algorithms

    - SM
        - Usually used for image segmentation

        - Minimize Normalized Cut by heuristic methods based on eigenvectors

- Multiway Spectral Algorithms

    - NJW
        - Clustering in a feature space constructed by eigenvectors to the k largest eigenvalues of the Laplacian matrix

B. Clustering based on Affinity Propagation

1. Basic Idea and Procedure

- Regard all data points as potential cluster centers

- Affinity (Similarity) for each data point is the Negative Euclidean distance from other data points.

- The more the sum of affinity for a data point is, the more its likely to a be a cluster center

- Takes greedy strategy which maximizes the value of the global function of the clustering network during each iteration (?)

C. Clustering based on Density and Distance

1. Basic Idea:

- The algorithm is based on dividing data space to high- and low-density regions:

    - High local density: number od data points near the cluster center is high relative to a certain scope

    - Low local density: cluster center must be away from other center-candidate data points within the cluster (?)

2. Main Procedure

- Construct Decision Graphs based on:

    - Local Density of each datapoint

    - Shortest distance from each data point to other data point with higher local density

- Select cluster centers based on the decision graph

- Assign nearest cluster with higher local density to remaining data points

D. Clustering for Spatial Data

1. Definition:

- Spatial Data refers to data that:

    - Contains space and time dimensions at the same time

    - Is large in scale, high in speed and complex in information

2. Typical Algorithms:

- DBSCAN
- STING
- Wavecluster
    - Used for parallel processing
    - Perform Clustering in a new feature space after applying Wavelet Transform on original data
- CLARANS
    - Use CLARA for sampling and perform clustering using PAM

E. Clustering for Large-Scale Data

1. Definition:

- Data is considered as “Big Data” if it’s:
    - Large in Volume
    - Rich in Variety
    - High in Velocity
    - Doubt in Veracity

2. Basic Ideas:

    a. Sample Clustering

    b. Data merged Clustering

    c. Dimension Reduction Clustering

    d. Parallel Clustering

3. Typical Algorithms:

- K-means
- BIRCH
- CLARA
- CURE
- DBSCAN
- DENCULE
- Wavecluster
- FC (Fuzzy Clustering)

VII. Conclusion:

A. Review on the paper:

- Starts with defining clustering algorithms
- Lists commonly used Distance, Similarity functions and Evaluation indicators that underlie clustering process
- Introduces 9 categories of Traditional and 10 categories of modern algorithms, a total of 67 algorithms

B. Main purpose of the paper:

- Introducing basic and core idea (process) of each Clustering Algorithm mentioned in the categories.
- Specifying source of each algorithm
- Analyzing advantages and disadvantages of each one

<u>Answering Main Questions about the article (asked in the Project Description)</u>

All the questioned are answered in the “Article Outline” and “Algorithm Comparison using Metrics Indicator”.

## Compact Generalized Comparison of Clustering Algorithms

| Category | Typical algorithm | Complexity(Time) | Scalability | Large-scale data | High dimensional data | Shape of suitable data set | Sensitive to sequence of inputted data | Sensitive to outlier/ noise |
|---|---|---|---|---|---|---|---|---|
| Based on partition | K-means | Low O($knt$) | Middle | Yes | No | Convex | Highly | Highly |
| Based on Distribution | GMM | High O($n^2kt$) | High | No | No | Arbitrary | Highly | little |
| Based on model | | | | | | | | |
| Spectral clustering | SM | High(eigenvector+heuristics) | Middle | No | Yes | Arbitrary | little | little |
| | NJW | High(eigenvector) | Middle | No | Yes | Arbitrary | little | little |
| Based on affinity propagation | AP | High O($n^2logn$) | Low | No | No | Convex | Moderately | little |
| For spatial Data | DBSCAN | High O($n^2kt$) | High | No | No | Arbitrary | Highly | little |
| For large-scale data | | | | | | | | |
| Based on density | OPTICS | Middle O($n^2logn$) | Middle | Yes | No | Arbitrary | little | little |
| | Mean-shift | High (kernel) | Low | No | No | Arbitrary | little | little |

# Introducing Common Clustering Algorithms

## K-Means Method

The KMeans algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. This algorithm requires the number of clusters to be specified. It scales well to large number of samples and has been used across a large range of application areas in many different fields.

The k-means algorithm divides a set of N samples X into K disjoint clusters C, each described by the mean of the samples in the cluster. The means are commonly called the cluster "centroids".

$$Inertia = \sum_{i=0}^{n} \min_{\mu_j \in C} (\|x_i - \mu_j\|^2)$$

Inertia can be recognized as a measure of how internally coherent clusters are.

## Agglomerative Clustering

Hierarchical clustering is a general family of clustering algorithms that build nested clusters by merging or splitting them successively. This hierarchy of clusters is represented as a tree (or dendrogram). The root of the tree is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample. See the Wikipedia page for more details.

The Agglomerative Clustering performs a hierarchical clustering using a bottom up approach: each observation starts in its own cluster, and clusters are successively merged together.

## Linkage Type

- Single-Link: In single-link hierarchical clustering, we merge in each step the two clusters whose two closest members have the smallest distance (or: the two clusters with the smallest minimum pairwise distance).
- Complete-Link: In complete-link hierarchical clustering, we merge in each step the two clusters whose merger has the smallest diameter (or: the two clusters with the smallest maximum pairwise distance).
- Average-Link: Average-link clustering is a compromise between the sensitivity of complete-link clustering to outliers and the tendency of single-link clustering to form long chains that do not correspond to the intuitive notion of clusters as compact, spherical objects.
-

## Desired Number of Clusters

At the beginning we have same numbers of clusters as our data. In each iteration 2-closest clusters combine and make one new cluster. Based on Linkage type after certain number of iterations, number of clusters will be as small as Desired Number (which was given at first). If no number was entered as desired number, algorithm will continue its procedure until there be one single cluster. That means by adding desired number we play a same role as cutting the Dendrogram through a specific height of graph.

## Boosting Algorithm

Despite the fact that, algorithm requires big amount of time and computational complexity, we can lessen this complexity with making distance matrix at the beginning of algorithm and then use this distance matrix to calculate desired linkage. Single-Link method is more efficient for big datas than others. There is a huge gap between library and scratch performance of this algorithm because python is slow in computing big high-dimensional matrices.

### Mean Shift

Mean Shift clustering aims to discover blobs in a smooth density of samples. It is a centroid based algorithm, which works by updating candidates for centroids to be the mean of the points within a given region. These candidates are then filtered in a post-processing stage to eliminate near-duplicates to form the final set of centroids.

$$x_i^{t+1} = m(x_i^t)$$

$$m(x_i) = \frac{\sum_{x_j \epsilon N(x_i)} K(x_j - x_i) x_j}{\sum_{x_j \epsilon N(x_i)} K(x_j - x_i)}$$

The algorithm automatically sets the number of clusters, instead of relying on a parameter bandwidth, which dictates the size of the region to search through.

## Affinity Propagation

Affinity Propagation is a clustering algorithm based on sending messages between data points, including the information of how likely each datapoint is to be its neighbor's exemplars (members of the dataset that are representative of clusters)

In contrast to other traditional clustering methods, Affinity Propagation does not require to specify the number of clusters.

Each data point sends messages to all other points informing its targets of each target's relative attractiveness to the sender. Each target then responds to all senders with a reply informing each sender of its availability to associate with the sender, given the attractiveness of the messages that it has received from all other senders. Senders reply to the targets with messages informing each target of the target's revised relative attractiveness to the sender, given the availability messages it has received from all targets.

The message-passing procedure proceeds until a consensus is reached. Once the sender is associated with one of its targets, that target becomes the point's exemplar .All points with the same exemplar are placed in the same cluster.

## Parameter Tuning

The tuning parameter for affinity propagation is the diagonal values of the similarity matrix, that can be equal to the minimum of other matrix values, (leads to minimum clusters), or can be equal to the maximum of other values (leading to maximum clusters), or be equal to the median of the other non-diagonal values which results in a medium number of clusters

## Affinity Propagation Steps:

### Similarity Matrix:

The Similarity Matrix holds negation of Euclidean distance or an inverse function of it, demonstrating how close points are to each other.

### Responsibility Matrix:

We start off by constructing an availability matrix with all elements set to zero. Then, we calculate every cell in the responsibility matrix using the following formula:

$$r(i,k) \leftarrow s(i,k) - \max_{k' \neq k} \{a(i,k') + s(i,k')\}$$

where $i$ refers to the row and k the column of the associated matrix.

Responsibility Matrix shows how much a data point is likely to be member of another exemplar, the Availability Matrix has an opposite concept and shows how much a data point is available to be an exemplar for other points

$$a(i,k) \leftarrow \min\left(0, r(k,k) + \sum_{i' \notin \{i,k\}} \max(0, r(i',k))\right) \text{ for } i \neq k$$
$$a(k,k) \leftarrow \sum_{i' \neq k} \max(0, r(i',k)).$$

The last matrix is the Criterion matrix that is simply the sum of the responsibility and the availability matrix

$$c(i,k) \leftarrow r(i,k) + a(i,k)$$

## DBSCAN

## How it Works

· After defining initial parameters, a starting point is selected at random at it's neighborhood area is determined using radius eps. If there are at least minPts number of points in the neighborhood, the point is marked as core point and a cluster formation starts. If not, the point is marked as noise. Once a cluster formation starts (let's say cluster A), all the points within the neighborhood of initial point become a part of cluster A. If these new points are also core points, the points that are in the neighborhood of them are also added to cluster A. (*Note: A point that is marked as noise may be revisited and be part of a cluster.)*

· Next step is to randomly choose another point among the points that have not been visited in the previous steps and repeat steps

· This process is finished when all points are visited.

By applying these steps, DBSCAN algorithm is able to find high density regions and separate them from low density regions.

A cluster includes core points that are neighbors (i.e. reachable from one another) and all the border points of these core points. The required condition to form a cluster is to have at least one core point. Although very unlikely, we may have a cluster with only one core point and its border points.

· **Min-Ponts**: As a rule of thumb, a minimum *minPts* can be derived from the number of dimensions $D$ in the data set, as $m \geq D + 1$. The low value $minPts = 1$ does not make sense, as then every point on its own will already be a cluster.

· With $minPts \leq 2$, the result will be the same as of hierarchical clustring with the single link metric, with the dendrogram cut at height ε. Therefore, *minPts* must be chosen at least 3. However, larger values are usually better for data sets with noise and will yield more significant clusters. As a rule of thumb, $minPts = 2 \cdot dim$ can be used, but it may be necessary to choose larger values for very large data, for noisy data or for data that contains many duplicates.

· **ε**: The value for ε can then be chosen by using a k-distance graph, plotting the distance to the $k = minPts - 1$ nearest neighbor ordered from the largest to the smallest value. Good values of ε are where this plot shows an "elbow": if ε is chosen much too small, a large part of the data will not be clustered; whereas for a too high value of ε, clusters will merge and the majority of objects will be in the same cluster. In general, small values of ε are preferable.

· **Distance function**: The choice of distance function is tightly linked to the choice of ε, and has a major impact on the outcomes. Choosing Distance and Similarity Measueres are discussed in their own part.

*The main concept of DBSCAN algorithm is to locate regions of high density that are separated from one another by regions of low density.* So, how do we measure density of a region?

· Density at a point P: Number of points within a circle of Radius $(\epsilon)$ from point P.

· Dense Region: For each point in the cluster, the circle with radius ϵ contains at least minimum number of points (*Min-Pts*).

· The Epsilon neighborhood of a point P in the database $D$ is defined as (following the definition from Ester et.al.)

## OPTICS

The OPTICS algorithm shares many similarities with the DBSCAN algorithm, and can be considered a generalization of DBSCAN that relaxes the epsilon requirement from a single value to a value range. The key difference between DBSCAN and OPTICS is that the OPTICS algorithm builds a reachability graph, which assigns each sample both a reachability_ distance, and a spot within the cluster ordering_ attribute; these two attributes are assigned when the model is fitted, and are used to determine cluster membership

## Spectral Clustering

Spectral clustering is technique that reduces complex multidimensional datasets into clusters of similar data in rarer dimensions. The main idea is to cluster the all spectrum of unorganized data points into multiple groups based upon their uniqueness Spectral Clustering uses the connectivity approach to clustering, wherein communities of nodes (i.e. data points) that are connected or immediately next to each other are identified in a graph.

The nodes are then mapped to a low-dimensional space that can be easily segregated to form clusters.

Spectral Clustering uses information from the eigenvalues of special matrices (i.e. Affinity Matrix, Degree Matrix and Laplacian Matrix) derived from the graph or the data set.

Spectral clustering treats the data clustering as a graph partitioning problem without making any assumption on the form of the data clusters.

### *Similarity Graphs*

There are several popular constructions to transform a given set of data points with pairwise similarities $s_{ij}$ or pairwise distances $d_{ij}$into a graph. When constructing similarity graphs the goal is to model the local neighborhood relationships between the data points.

**THE ε-NEIGHBORHOOD GRAPH** : Here we connect all points whose pairwise distances are smaller than ε. As the distances between all connected points are roughly of the same scale (at most ε), weighting the edges would not incorporate more information about the data to the graph. Hence, the ε-neighborhood graph is usually considered as an unweighted graph.

**K-NEAREST NEIGHBOR GRAPHS** : Here the goal is to connect vertex $v_i$ with vertex $v_j$ if $v_j$ is among the k-nearest neighbors of $v_i$. However, this definition leads to a directed graph, as the neighborhood relationship is not symmetric. There are two ways of making this graph undirected. The first way is to simply ignore the directions of the edges, that is we connect $vi$ and $vj$ with an undirected edge if $v_i$ is among the k-nearest neighbors of $v_j$ or if $v_j$ is among the k-nearest neighbors of $v_i$. The resulting graph is what is usually called the k-nearest neighbor graph. The second choice is to connect vertices $v_i$and $v_j$ if both $v_i$is among the k-nearest neighbors of $v_j$and $v_j$is among the k-nearest neighbors of $v_i$.

**THE FULLY CONNECTED GRAPH** : Here we simply connect all points with positive similarity with each other, and we weight all edges by $s_{ij}$. As the graph should represent the local neighborhood relationships, this construction is only useful if the similarity function itself models local neighborhoods.

An example for such a similarity function is the Gaussian similarity function $s(x_i, x_j) = \exp(-\frac{(x_i - x_j)^2}{2\sigma^2})$, where the parameter σ controls the width of the neighborhoods. This parameter plays a similar role as the parameter ε in case of the ε-neighborhood graph.

## *Spectral Clustering Matrix Representation*

### Adjacency and Affinity Matrix (A)

The graph (or set of data points) can be represented as an Adjacency Matrix, where the row and column indices represent the nodes, and the entries represent the absence or presence of an edge between the nodes (i.e. if the entry in row 0 and column 1 is 1, it would indicate that node 0 is connected to node 1).

An Affinity Matrix is like an Adjacency Matrix, except the value for a pair of points expresses how similar those points are to each other. If pairs of points are very dissimilar then the affinity should be 0. If the points are identical, then the affinity might be 1. In this way, the affinity acts like the weights for the edges on our graph.

### Degree Matrix (D)

A Degree Matrix is a diagonal matrix, where the degree of a node (i.e. values) of the diagonal is given by the number of edges connected to it. We can also obtain the degree of the nodes by taking the sum of each row in the adjacency matrix.

### Laplacian Matrix (L)

This is another representation of the graph/data points, which attributes to the beautiful properties leveraged by Spectral Clustering. One such representation is obtained by subtracting the Adjacency Matrix from the Degree Matrix .The Laplacian's diagonal is the degree of our nodes, and the off diagonal is the negative edge weights. This is the representation we are after for performing spectral clustering.

**THE UNNORMALIZED GRAPH LAPLACIAN :** $L = D - W$

**THE NORMALIZED GRAPH LAPLACIANS:** $D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$

### Spectral Gap

The first non-zero eigenvalue is called the Spectral Gap. The Spectral Gap gives us some notion of the density of the graph.

### Fiedler Value

The second eigenvalue is called the Fiedler Value, and the corresponding vector is the Fiedler vector. Each value in the Fiedler vector gives us information as to which side of the decision boundary a particular node belongs to.

Using L, we find the first large gap between eigenvalues which generally indicates that the number of eigenvalues before this gap is equal to the number of clusters.

## UNNORMALIZED SPECTRAL CLUSTERING STEPS :

Input : *Similarity matrix* $S \in n \times n$, number $k$ of clusters to construct.

Construct a *similarity graph* by one of the ways described.

Let W be its weighted *adjacency matrix*.

Compute the *unnormalized Laplacian* L.

Compute the first $k$ eigenvectors $u_1, \dots, u_k$ of L.

Let $U \in n \times k$ be the matrix containing the vectors $u_1, \dots, u_k$ as columns.

For $i = 1, \dots, n$, let $y_i \in k$ be the vector corresponding to the $i$-th row of U.

Cluster the points with the k-means algorithm into clusters $C_1, \dots, C_k$.

Output: Clusters $A_1, \dots, A_k$ with $A_i = \{j | y_j \in C_i\}$.

### Gaussian Mixture Model

The Gaussian mixture model is simply a mix of Gaussian distributions. In this case, Gaussian means the multivariate normal distribution $N(\mu_j, \sum_j)$ and mixture means that several different gaussian distributions, all with different mean vectors $\mu_j$ and different covariance matrices $\sum_j$ are combined by taking the weighted sum of the probability density functions:

$$f_{GMM}(x) = \sum_{j=1}^{k} \phi_j \, f_{N(\mu_j, \sum_j)}(x) \quad subject\ to \sum_{j=1}^{k} \phi_j = 1$$

GMM is a multimodal distribution with on distinct bump per class. (Sometimes you get fewer than k distinct local maxima in the pdf, if the bumps are sufficiently close together or if the weight of one class is zero or nearly so, but in general you get k distinct bumps.)

To fit a GMM model to a particular dataset, we attempt to find the maximum likelihood estimate of the parameters:

$$\theta = \{\mu_1 , \Sigma_1 , \dots , \mu_k , \Sigma_k\}$$

Because the $n \times m$ example matrix $X$ is assumed to be a realization of n $i.i.d.$ samples from $f_{GM} \quad (x)$, we can write down our likelihood function as:

$$\mathcal{L}\,(\theta\,; X) = P(X;\theta) = \prod_{i=1}^{n}\sum_{j=1}^{k} P(C_i = j\,)\,P(X_i|C_i = j\,)$$

We know that $X_i$ has a multivariate normal distribution with parameters determined by the class, so the conditional probability can be written down pretty much directly from the definition:

$$P(\mathbf{X}_i|C_i = j) = \frac{1}{\sqrt{(2\pi)^k|\Sigma_j|}}\exp\left(-\frac{(\mathbf{X}_i - \boldsymbol{\mu}_j)^T\Sigma_j^{-1}(\mathbf{X}_i - \boldsymbol{\mu}_j)}{2}\right)$$

We know that the unconditional probability is given by

$$P(C_i = j) = \phi_j$$

So using Bayes' theorem:

$$P(C_i = j\,|X_i) = \frac{P(C_i = j\,)P(X_i|C_i = j\,)}{P(X_i)} = \frac{\phi_j\,P(X_i|C_i = j\,)}{\sum_{l=1}^{k} P(X_i|C_i = l\,)}$$

3 above equations when taken together, constitute the complete likelihood function.

However, these equations have a problem - they depend on the unknown random variable $C_i$. This variable tells us which class each $X_i$ was drawn from and makes it much easier to reason about the distribution, but we don't actually know what $C_i$ is for any $i$. This is called a *latent random variable* and its presence in our model causes a kind of causality dilemma .

If we knew $\mu_j$ and $\Sigma_j$ for $j = (1,2,\ldots,k)$ then we could make a guess about what $C_i$ is by looking at which $\mu_j$ is closest to $X_i$. If we knew Ci, we could estimate $\mu_j$ and $\Sigma_j$ by simply taking the mean and covariance over all Xi where $C_i = j$ .

The solution to our dilemma is an iterative algorithm called the expectation-maximization algorithm, or EM algorithm for short. The EM algorithm is actually a meta-algorithm: a very general strategy that can be used to fit many different types of latent variable models.

The EM algorithm requires us to introduce a pseudo-parameter to model the unknown latent variables $C_i$ . Because $C_i$ can take on $k$ discrete values, this new parameter will be a $n \times k$ matrix where each element $w_{ij}$ is an estimate of $P(C_i = j|X_i;\ \theta)$.

Each element of this matrix represents the probability that the i-th data point came from cluster j. This pseudo-parameter is only used when fitting the model and will be discarded afterwards; in that sense it is not a true parameter of the model.

The EM algorithm then proceeds iteratively, with each iteration being divided into two steps: the E-step and the M-step.

In the E-step, we use our current best knowledge of the centers and shapes of each cluster to update our estimates of which data point came from which class. Concretely, we hold $\mu_j$ and $\Sigma_j$ fixed and update $w_{ij}$ and $\phi$ .

$$w_{ij} = \frac{P(X_i|K=j)}{P(K_i)} = \frac{P(X_i|K=j)}{\sum_{l=1}^{k} P(X_i|K=l)}$$

$$w_{ij} = \frac{f_{\mathcal{N}(\mu_i,\Sigma_i)}(\mathbf{X}_i)}{\sum_{l=1}^{k} f_{\mathcal{N}(\mu_l,\Sigma_l)}(\mathbf{X}_i)}$$

The probability of each class $\phi$ can then be estimated by averaging over all examples in the training set:

$$\phi_j = \sum_{i=1}^{n} w_{ij}$$

In the M-step, we use our current best knowledge of which class each point belongs to update and improve our estimates for the center and shape of each cluster. Concretely, we use $w_{ij}$ as sample weights when updating $\mu_j$ and $\Sigma_j$ by taking weighted averages over $X$.

$$\boldsymbol{\mu}_j = \frac{1}{n} \sum_{i=1}^{n} w_{ij} \mathbf{X}_i$$

$$\Sigma_j = \frac{1}{n} \sum_{i=1}^{n} w_{ij} (\mathbf{X}_i - \boldsymbol{\mu}_j)(\mathbf{X}_i - \boldsymbol{\mu}_j)^T$$

This process is guaranteed to converge a (local) maximum likelihood because of the ratchet principle: at each step, likelihood can only increase and never decrease. This can be viewed as a type of coordinate ascent.

## Tunning Algorithms

The silhouette Method is also a method to find the optimal number of clusters and interpretation and validation of consistency within clusters of data. The silhouette method computes silhouette coefficients of each point that measure how much a point is similar to its own cluster compared to other clusters. by providing a succinct graphical representation of how well each object has been classified.

Compute silhouette coefficients for each of point, and average it out for all the samples to get the silhouette score.

The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The value of the silhouette ranges between [1, -1], where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters. If most objects have a high value, then the clustering configuration is appropriate. If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

## Algorithm Comparison using Metrics Indicator

### Separation Index

The basic idea is that data points should be as similar as possible inside a cluster and as different as possible among 2 different clusters. Inter Distance and Intra Distance are defined as followed:

$$d(S_i, S_j) = \min_{x,y} \{d(x, y | x \epsilon S_i, y \epsilon S_j)\}$$

$$d(S_l, S_l) = \max_{x,y} \{d(x, y | x, y \epsilon S_l)\}$$

Separation index is based on the ratio of the inter cluster distance to intra cluster distance, and then minimizing it across all the possible pairs.

$$SI = \min_{j} \left\{ \min_{i(i \neq j)} \left\{ \frac{d(S_i, S_j)}{\max_l d(S_l, S_l)} \right\} \right\}$$

### Fisher's Discrimination Index

The basic idea, like Separation, is maximizing a function of the mentioned ratio, but by using covariance matrices of within and between cluster data points, the between variability is calculated among centroids and the total centroid and the within variability is simply the sum of covariance matrices of the data points in a cluster.

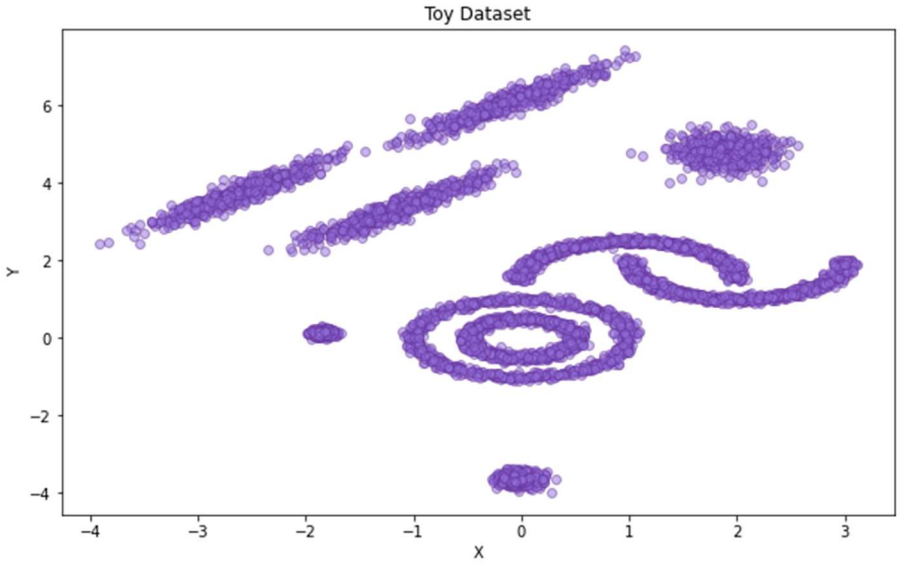
$$FDI = trace\left(S_W^{-1} S_B\right)$$

$$S_W = \sum_{i=1}^{k} S_i \qquad S_B = \sum_{i=1}^{k} Q_i \left(\hat{\mu}_i - \hat{\mu}\right)\left(\hat{\mu}_i - \hat{\mu}\right)^T \qquad S_i = \sum_{q=1}^{Q_i} \left(x^q - \hat{\mu}_i\right)\left(x^q - \hat{\mu}_i\right)^T$$

$$\hat{\mu}_i = \sum_{x^q \epsilon S_i} x^q \qquad \mu = \sum_{q=1}^{Q} x^q$$

# Case study

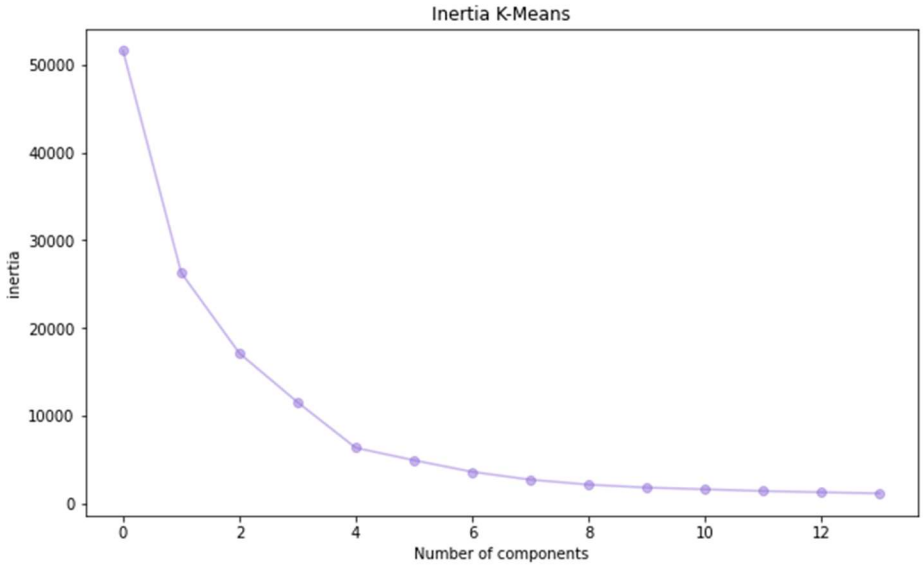
## Toy data set :



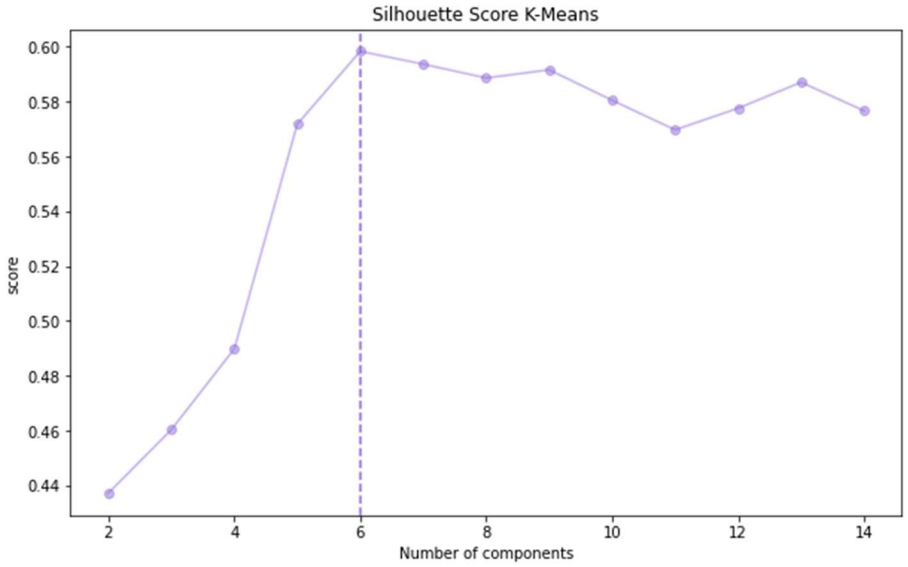

## K-means

**Tuning hyper parameters :**

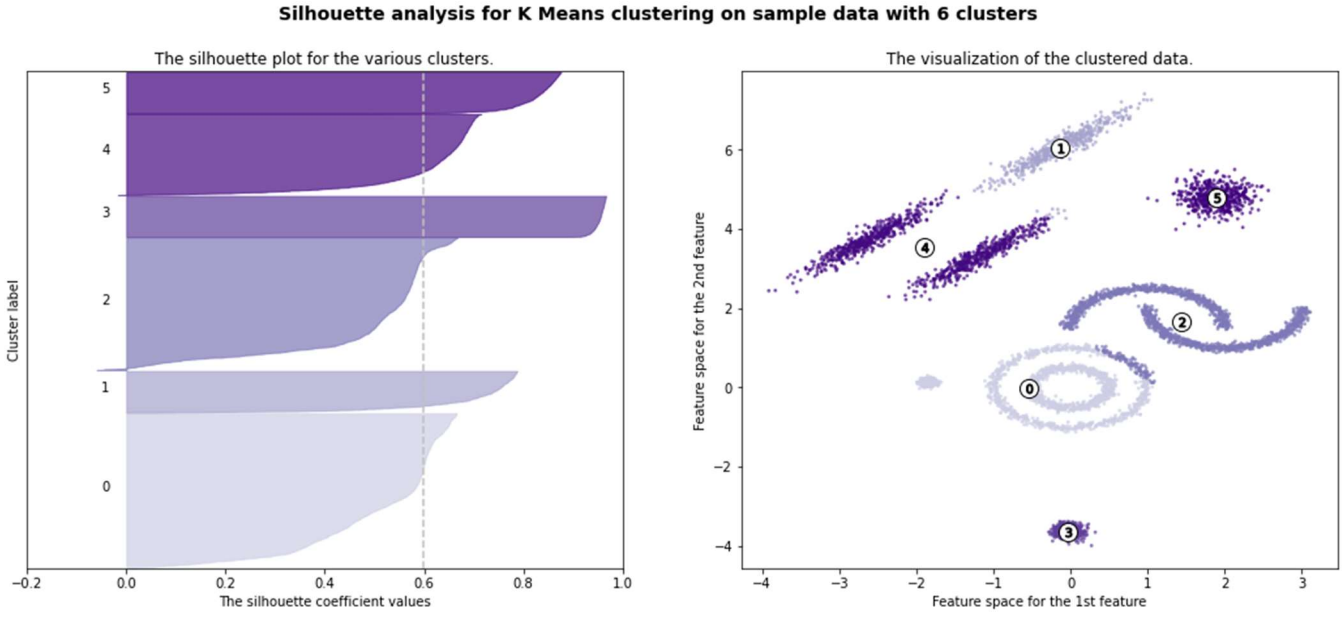
**ELBOW METHOD :** There isn't an elbow

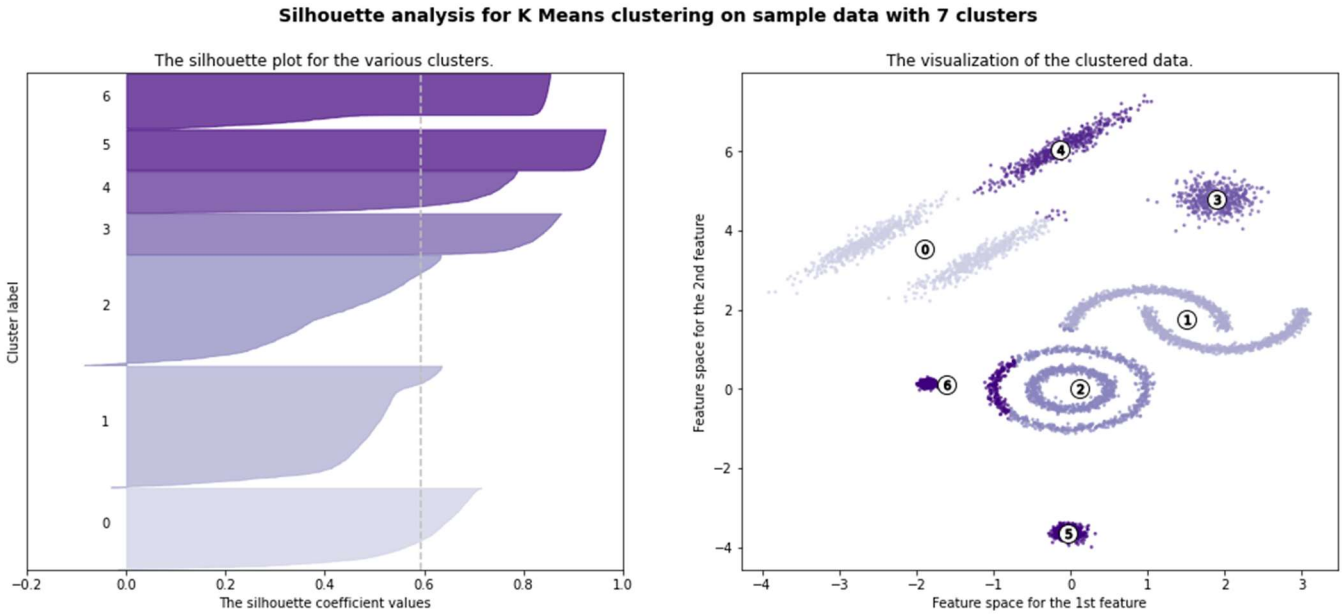


## SILHOUETTE SCORE :

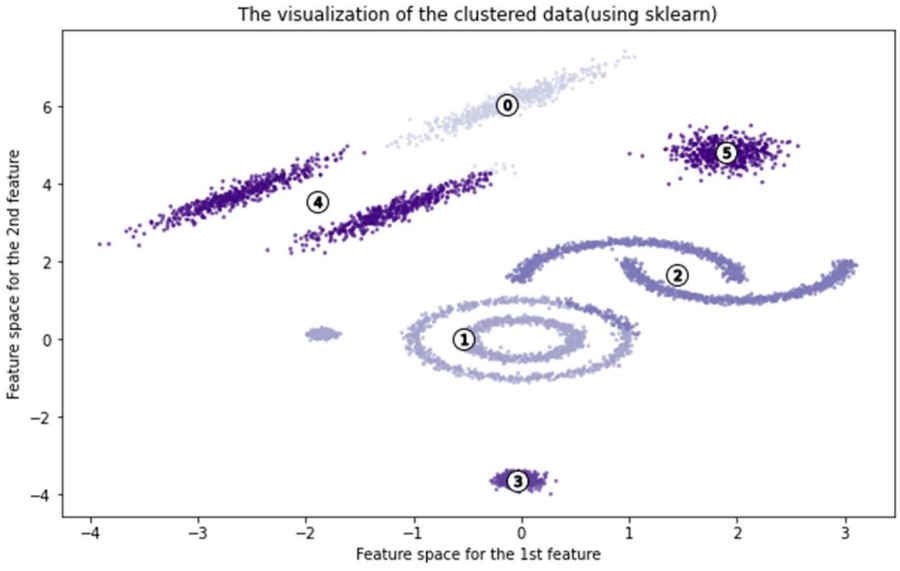



## SILHOUETTE ANALYSIS :





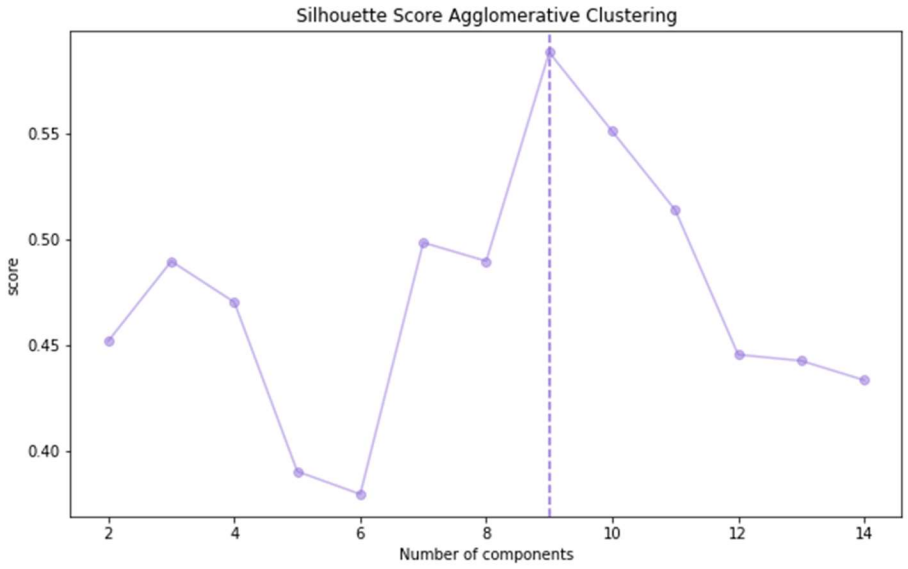

Clustered data :



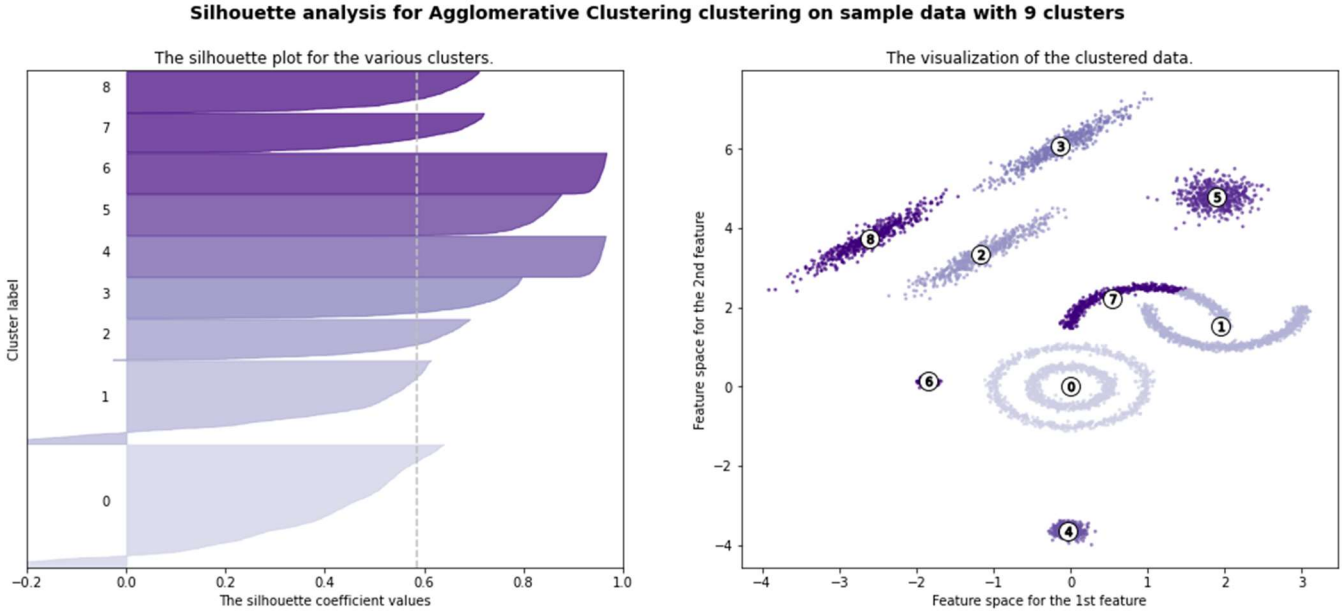
## Agglomerative Clustering

Tuning hyper parameters :

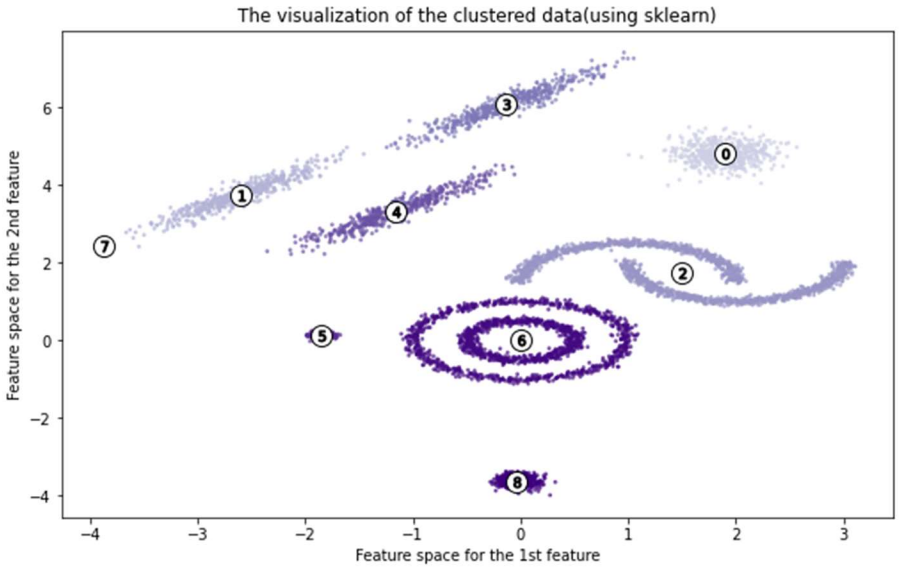
SILHOUETTE SCORE :

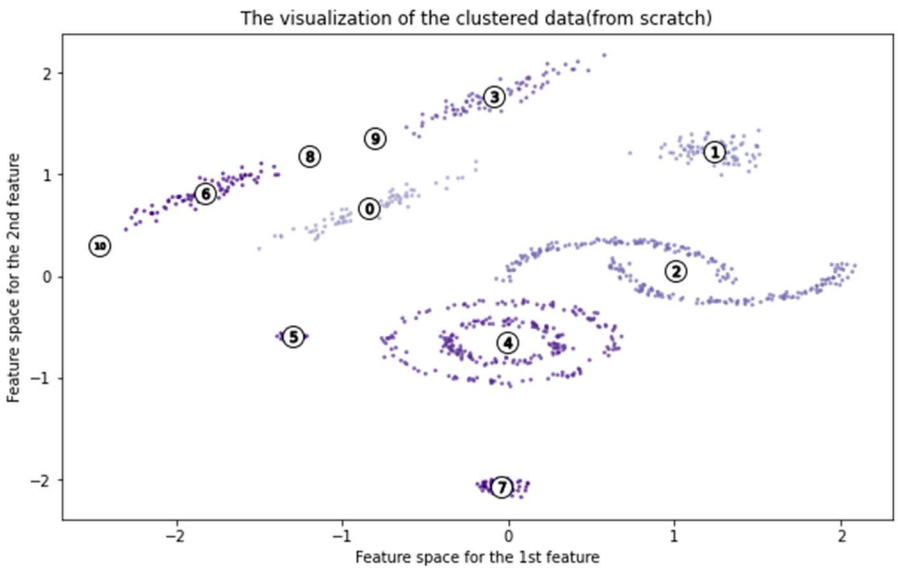



SILHOUETTE ANALYSIS :




Clustered data :





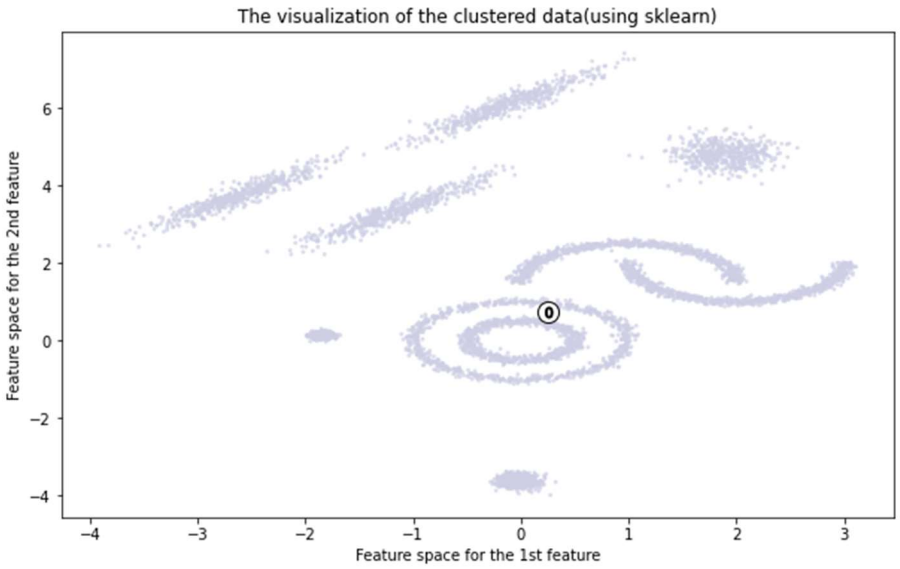

Due to the very high clustering time of the data, it was clustered using samples

## Mean Shift

Tuning hyper parameters :

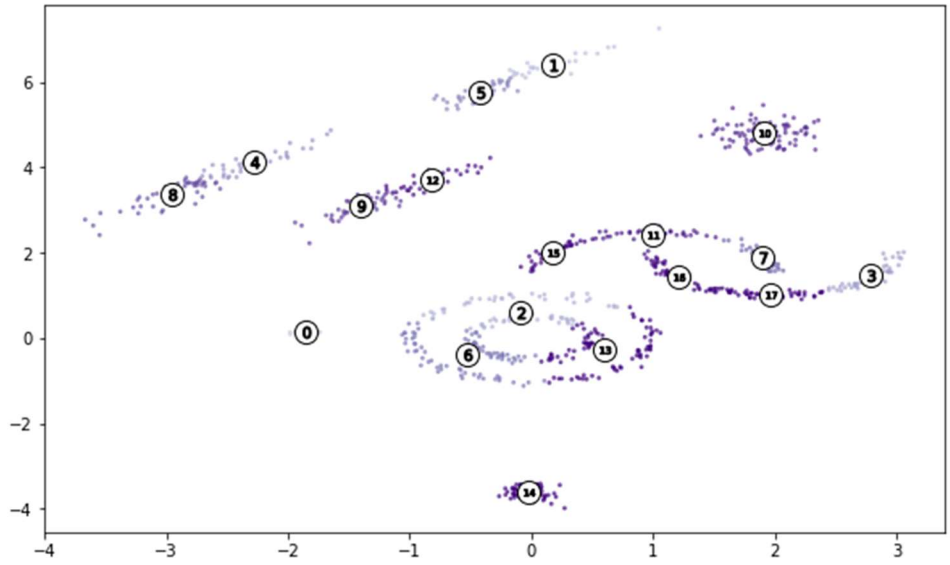
SKLEARN.CLUSTER.ESTIMATE_BANDWIDT: 2.6

Clustered data :



## Affinity Clustering

Clustered data :



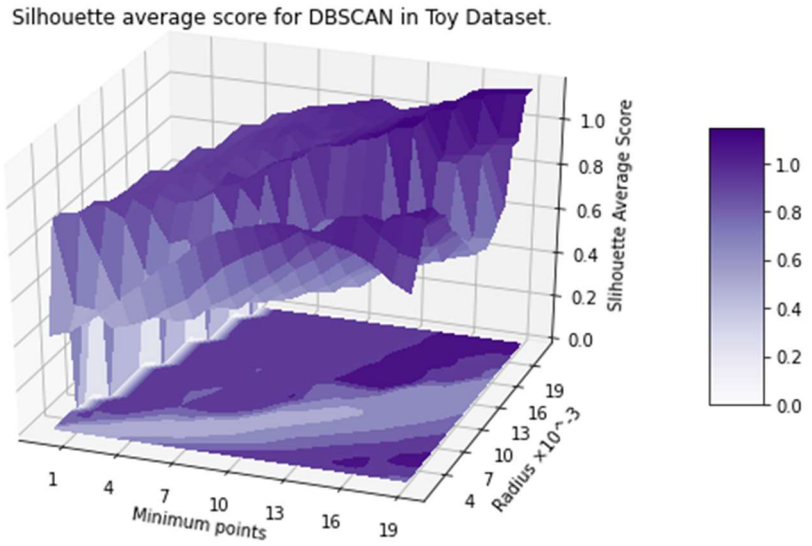

Due to the very high clustering time of the data, it was clustered using samples

And 0.8 damping factor !

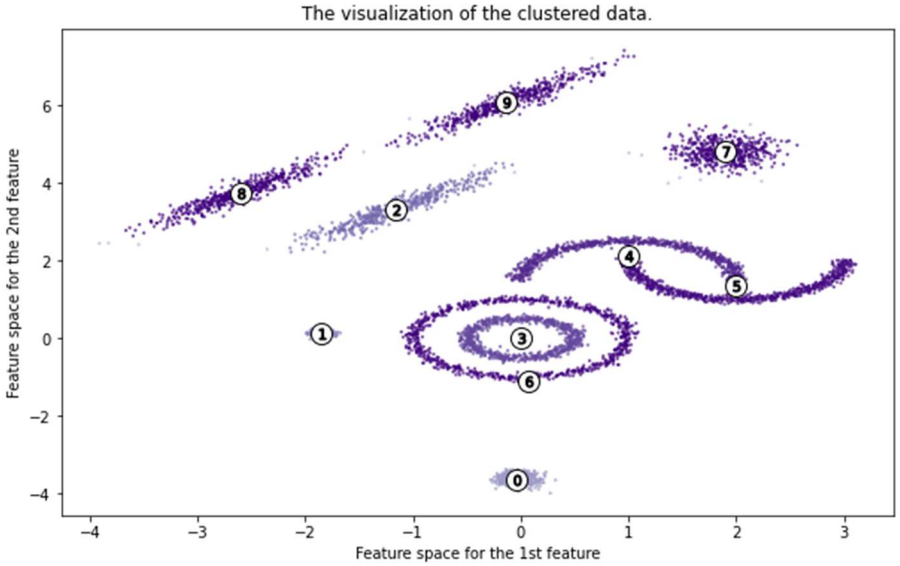
## DBSCAN

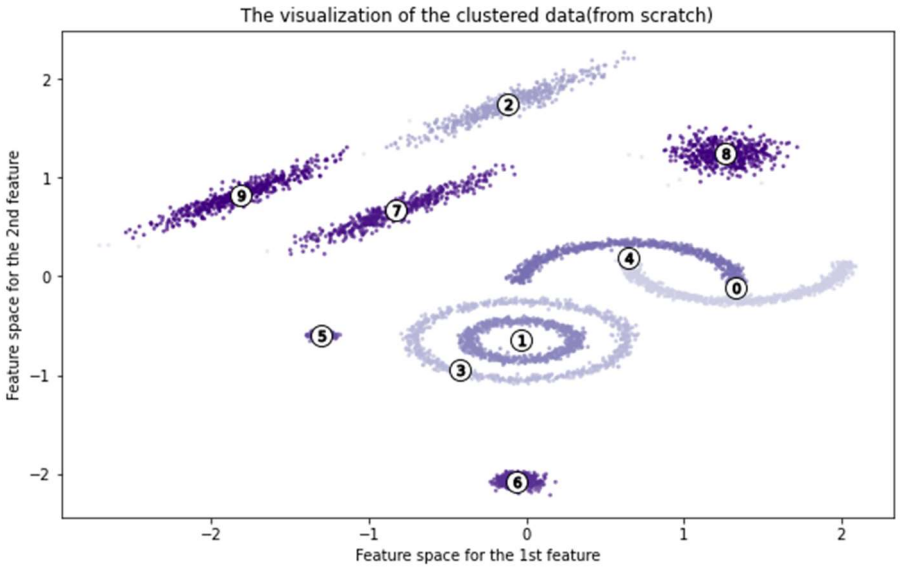
Tuning hyper parameters :

**SILHOUETTE SCORE :**



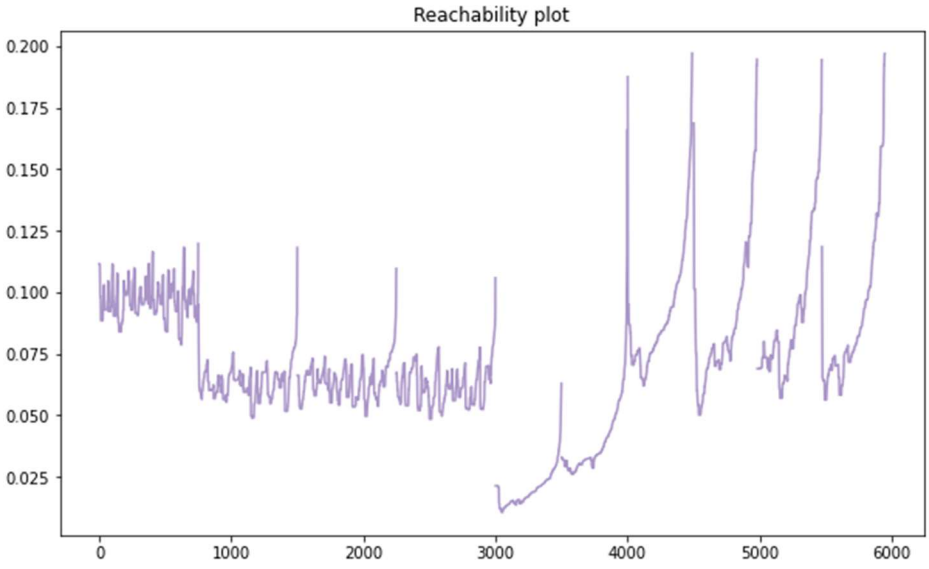

Clustered data :



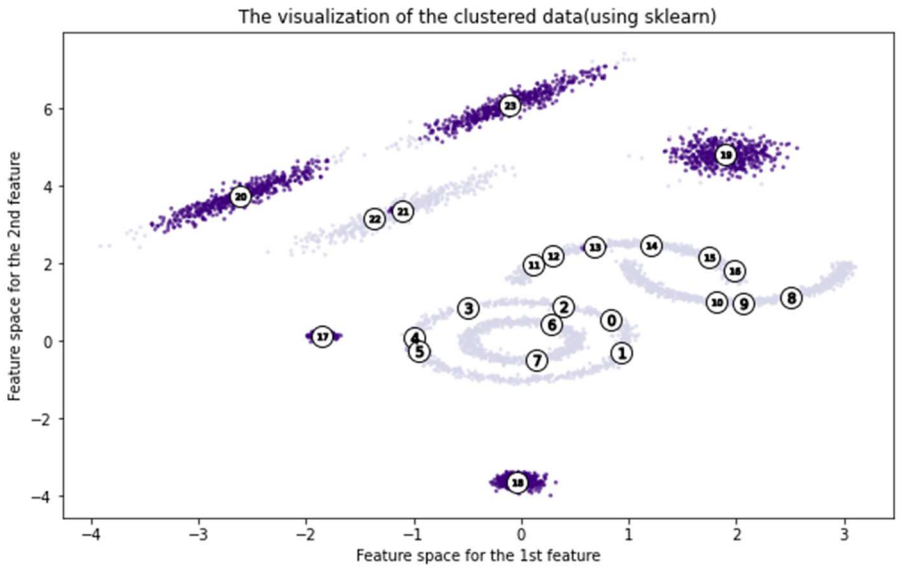

The visualization of the clustered data(from scratch)
Feature space for the 2nd feature
Feature space for the 1st feature
2
1
0
−1
−2
−2
−1
0
1
2
9
2
8
7
4
0
5
1
3
6

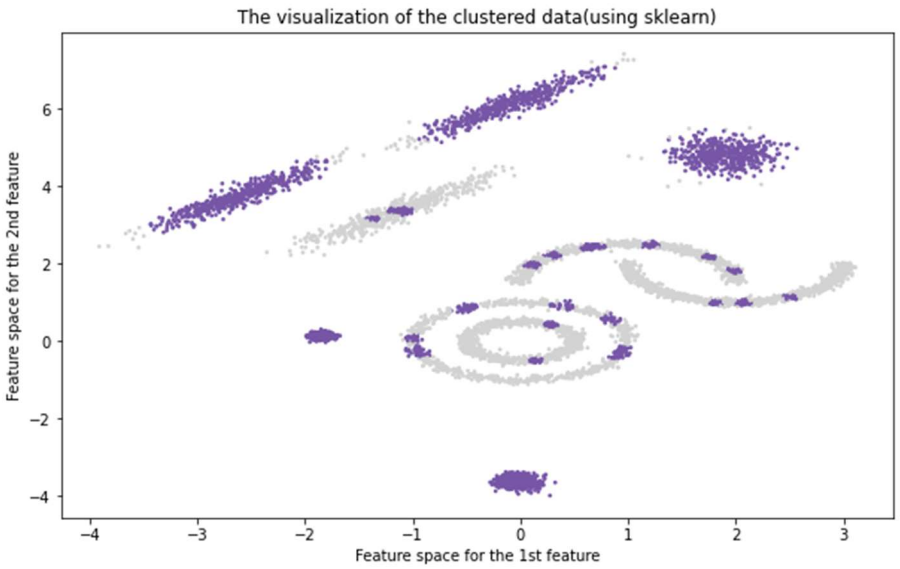
## OPTICS

Reachability Plot :




Clustered data :



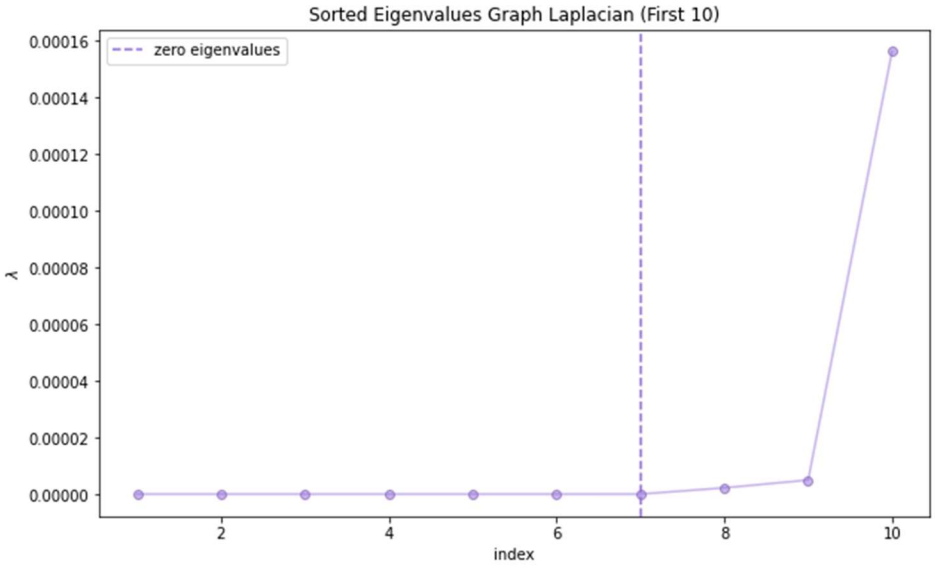



This type of clustering detects noise and  labels it with -1, which is shown in a different color in the figure above.

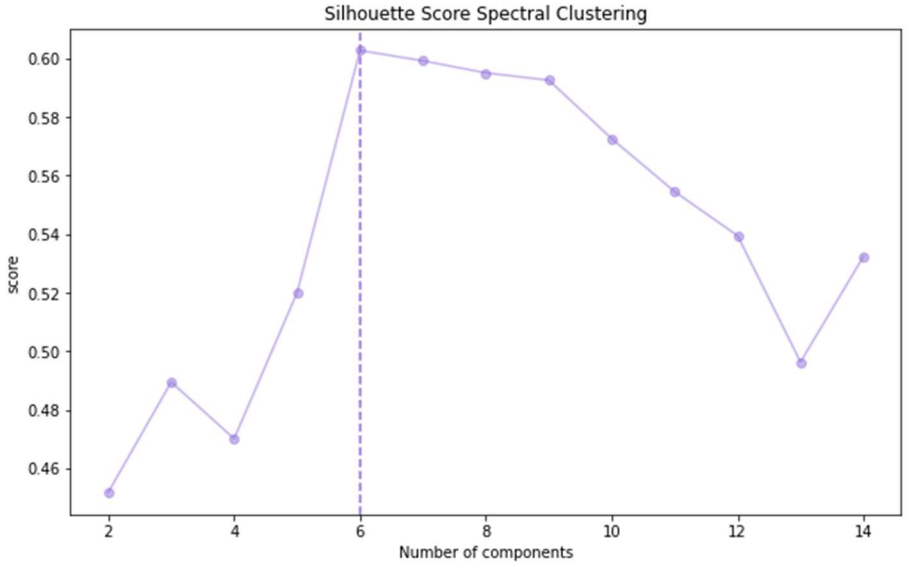
```
Estimated no. of clusters: 25
Estimated no. of noise points: 2903
```

Tuning hyper parameters :

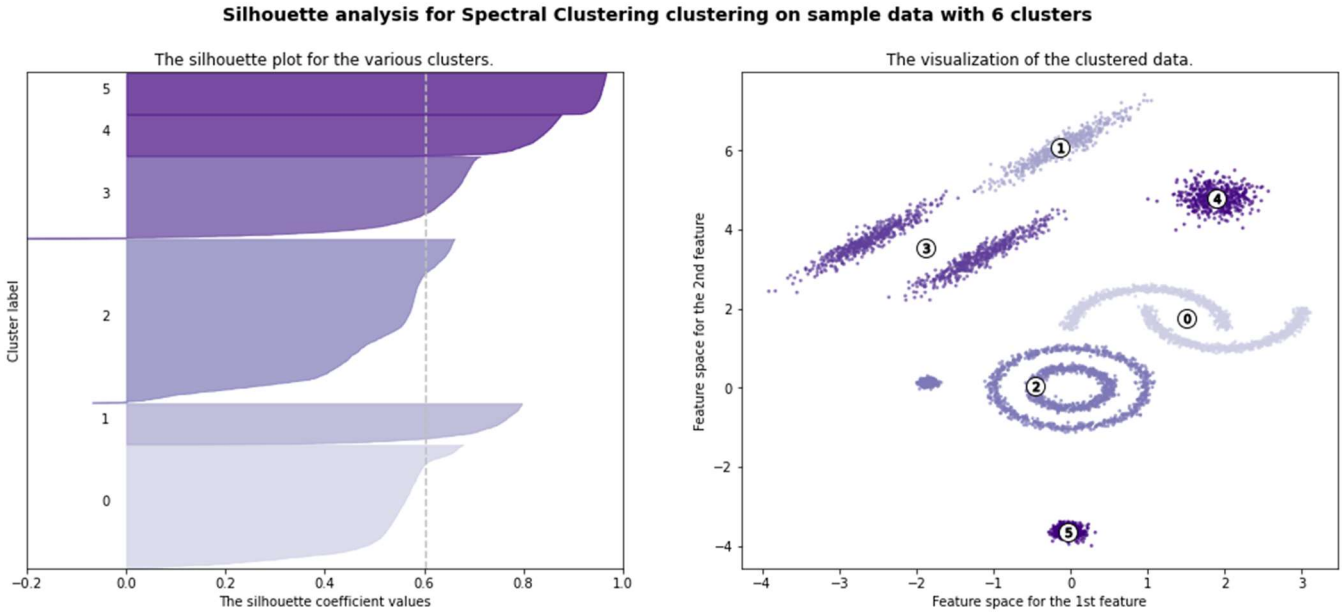
**SPECTRAL GAP:**

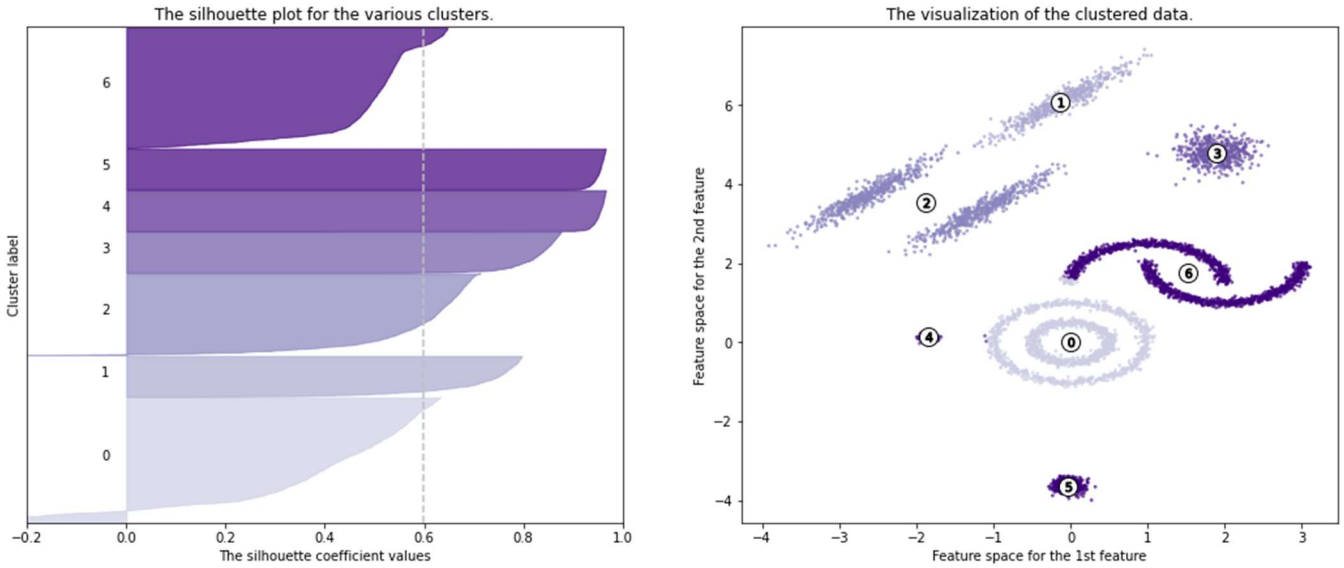



**SILHOUETTE SCORE :**



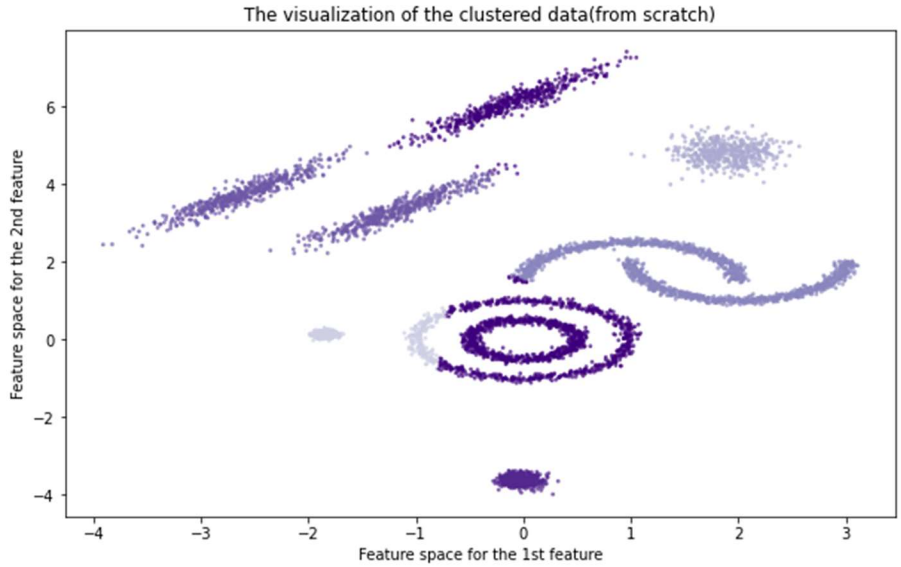
## SILHOUETTE ANALYSIS :

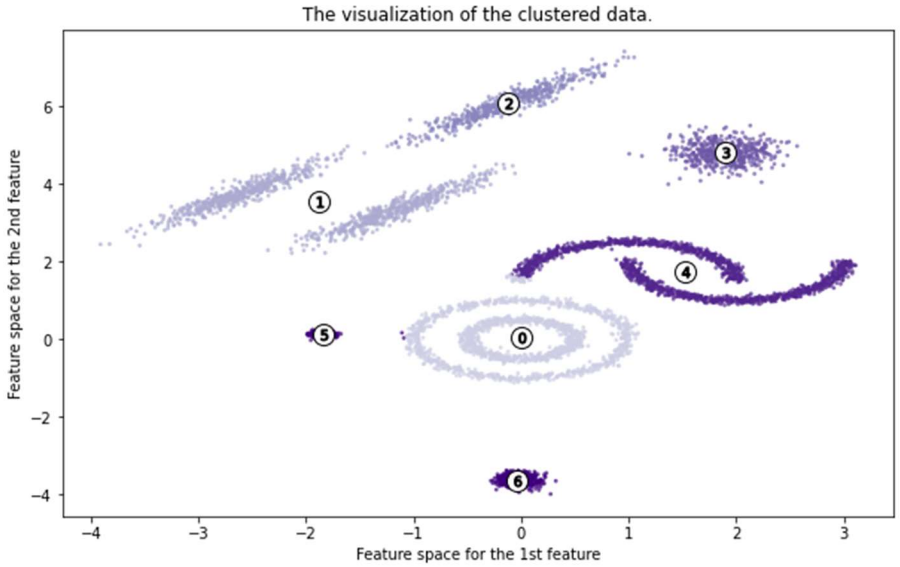


Clustered data :

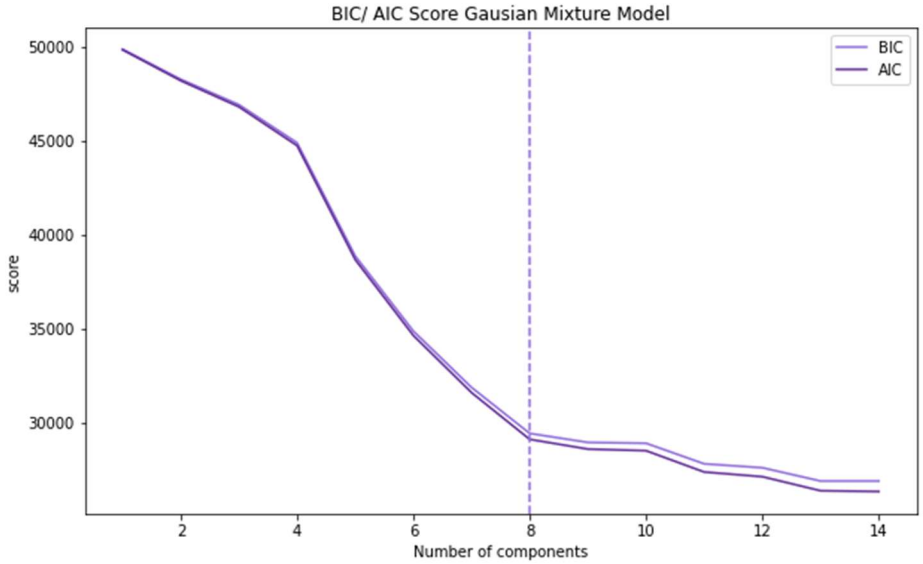
K NEAREST NEIGHBORHOOD METHOD , K = 5



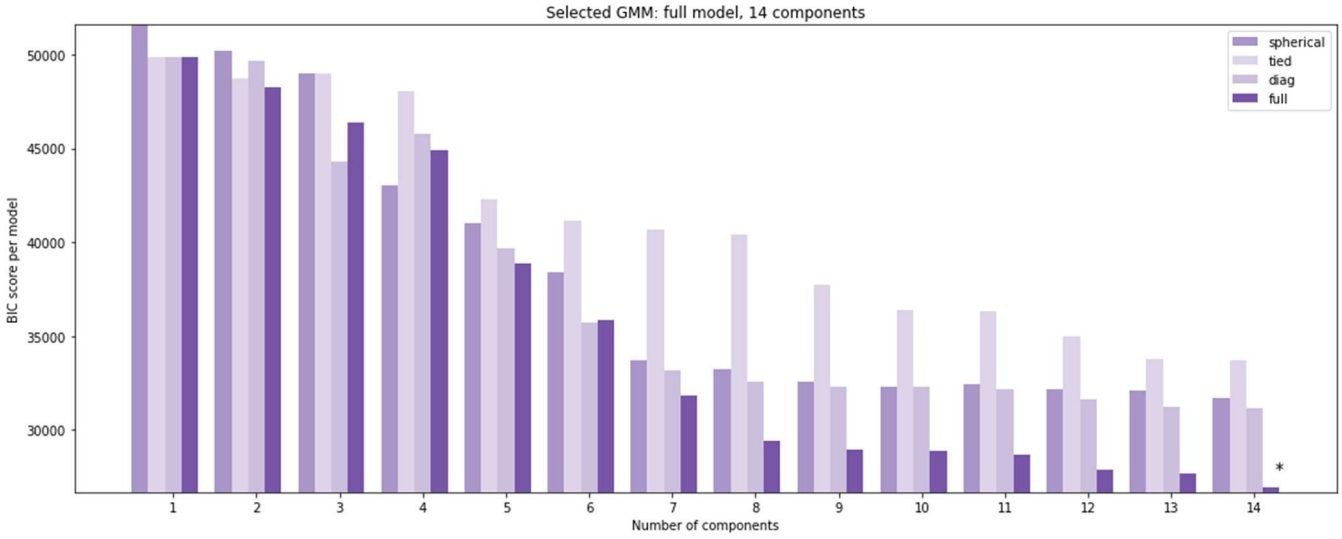



Tuning hyper parameters :

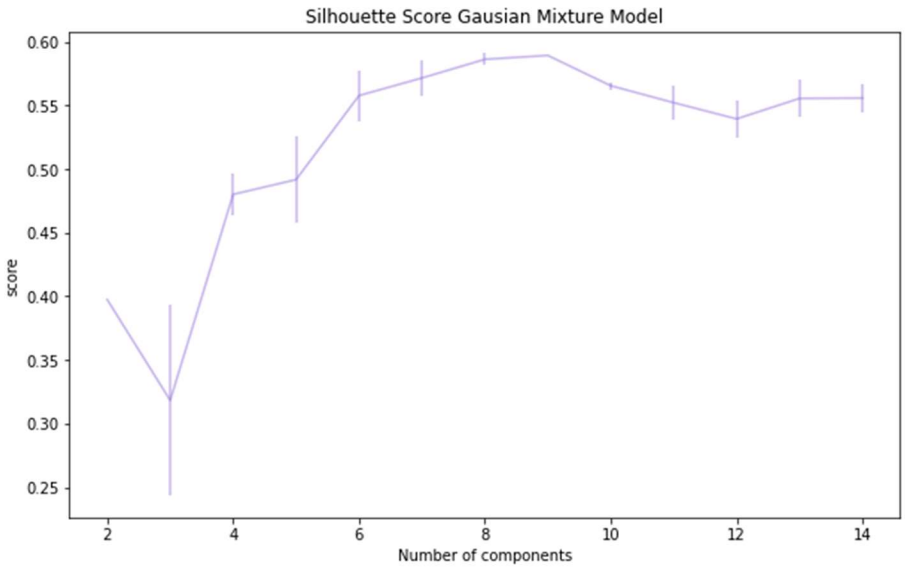
**BIC AND AIC:**



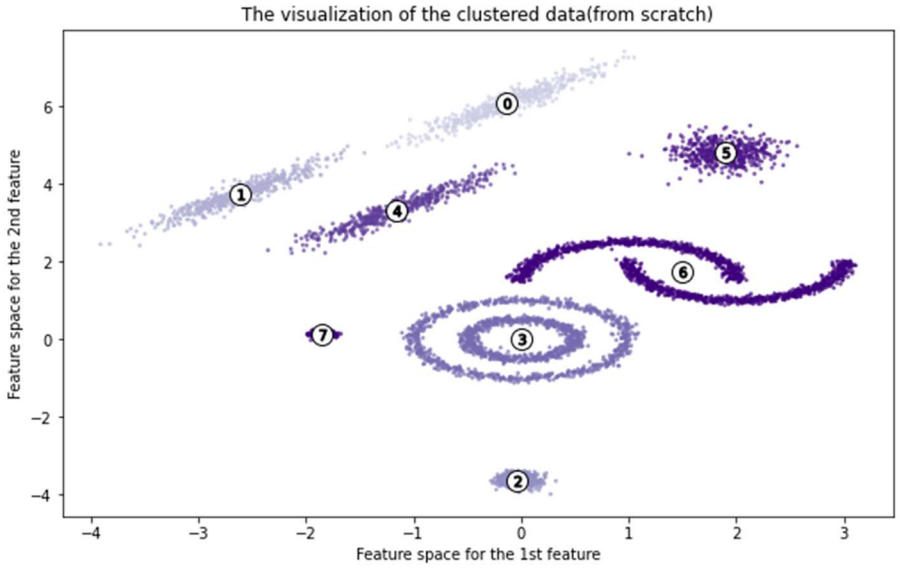

**BIC WITH DIFFERENT COVARIANCE TYPE :**

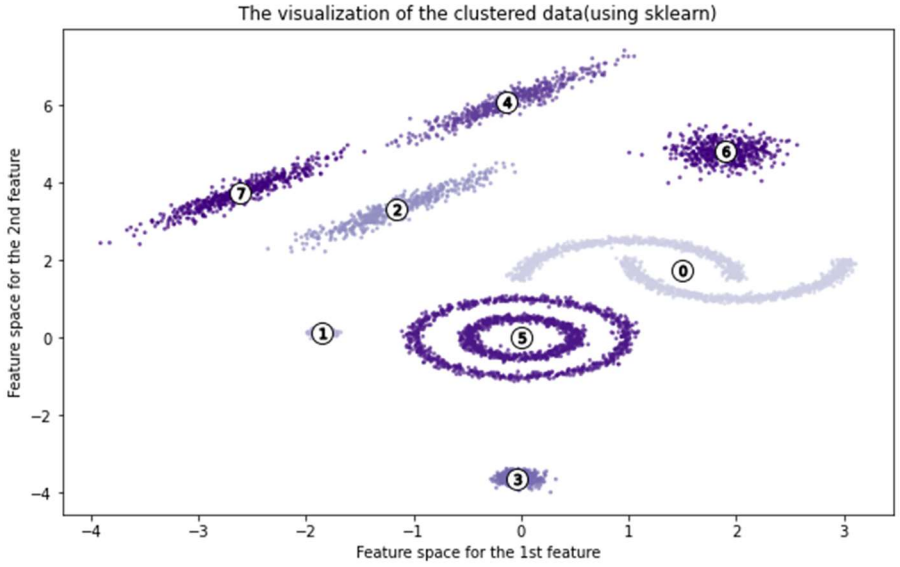


## SILHOUETTE SCORE WITH ERROR BAR :

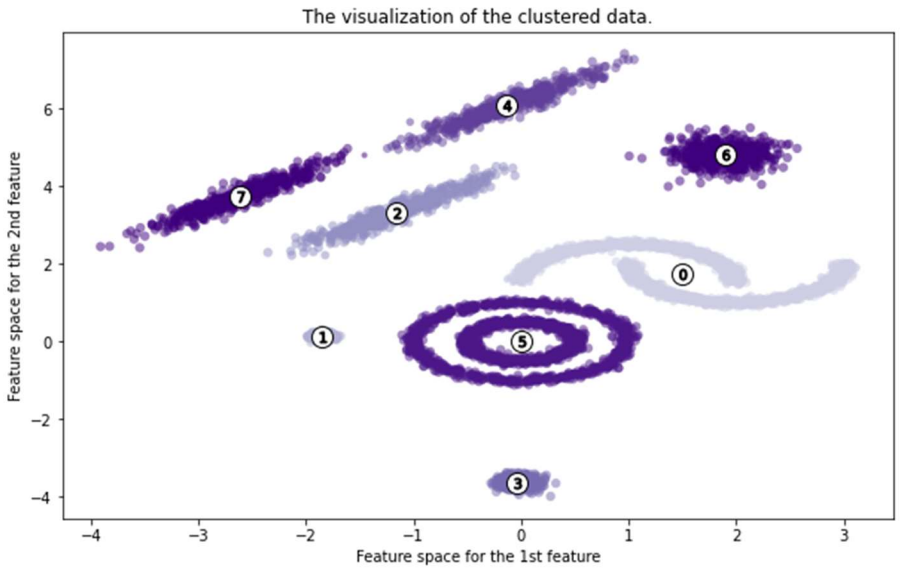


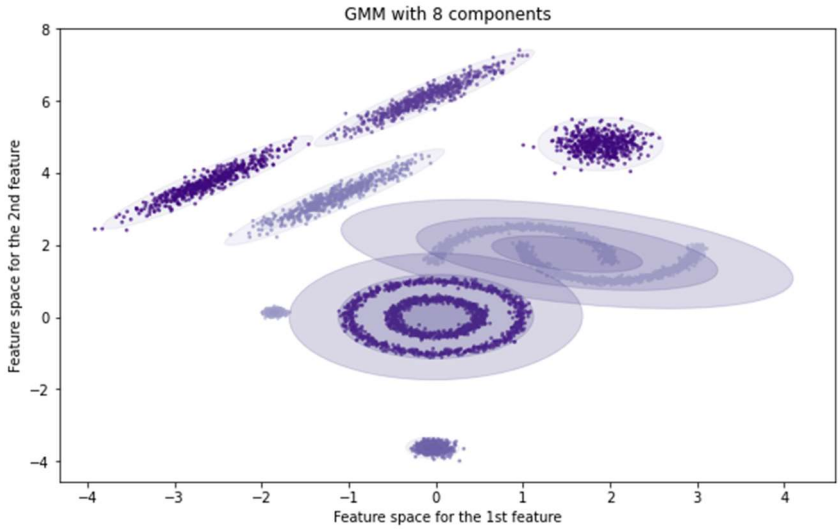

Clustered data :



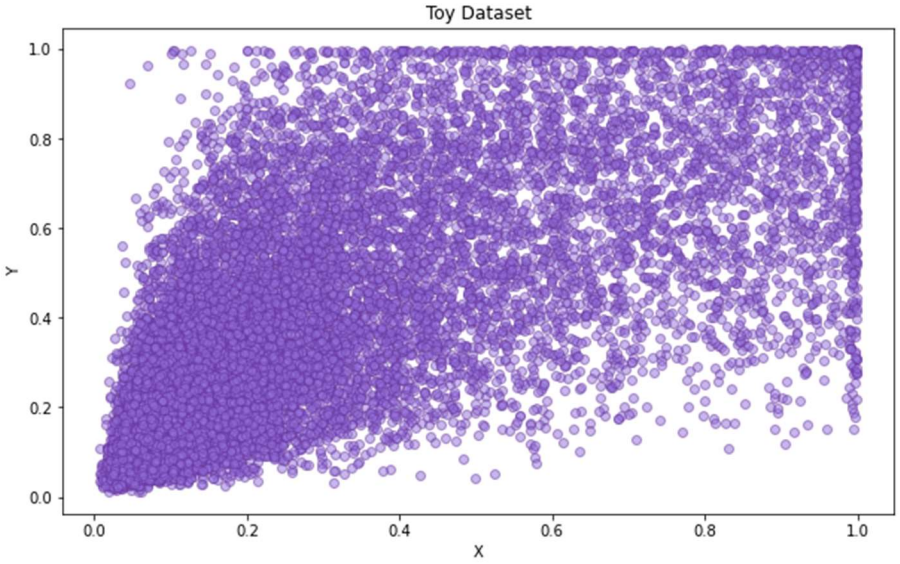



## EXAGGERATED CLUSTERED DATA :




GMM with 8 components
Feature space for the 2nd feature
Feature space for the 1st feature
8
6
4
2
0
−2
−4
−4
−3
−2
−1
0
1
2
3
4

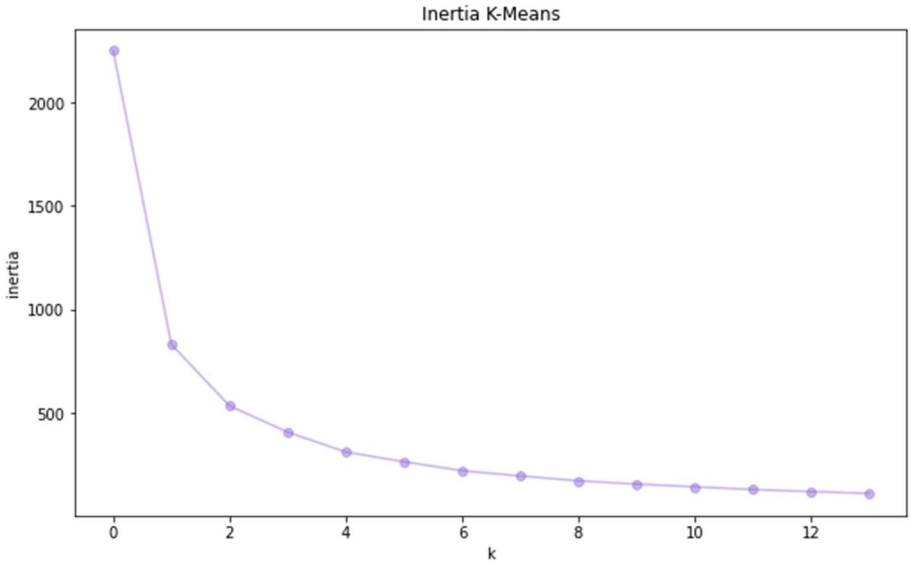
BBox data set :



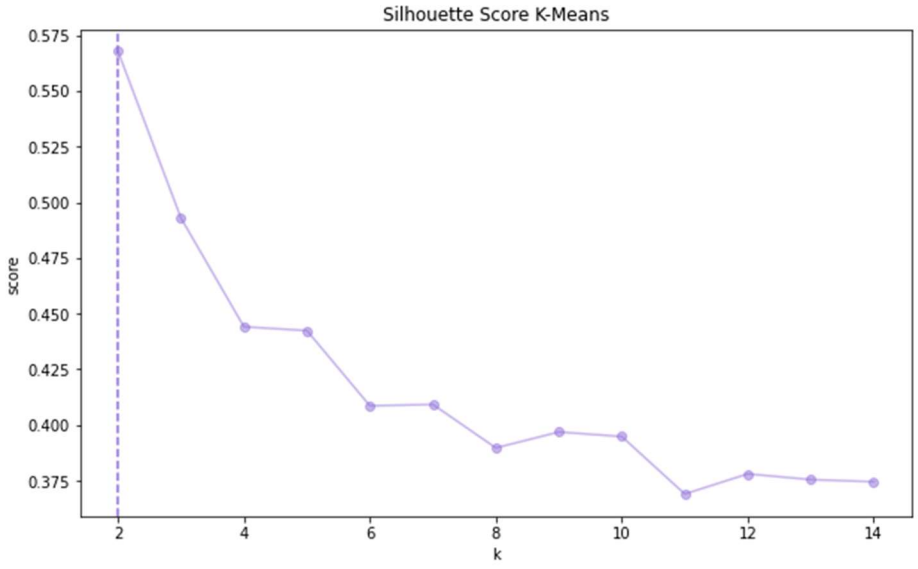
## K-means

Tuning hyper parameters :

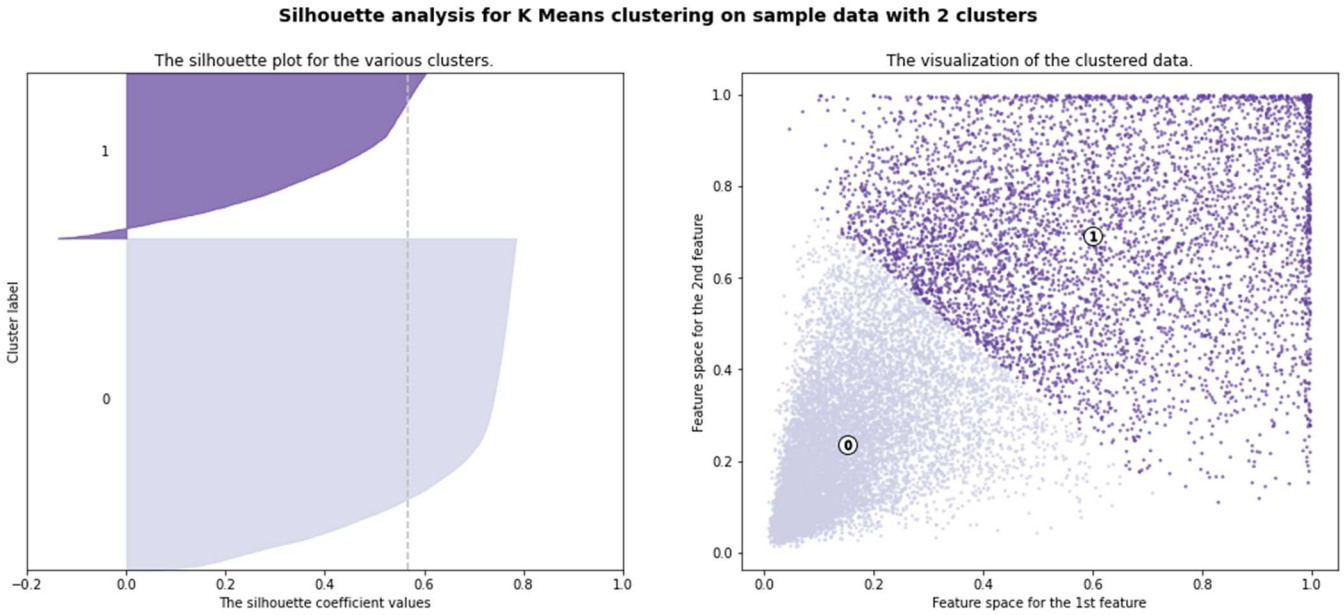
**ELBOW METHOD :** There isn't an elbow

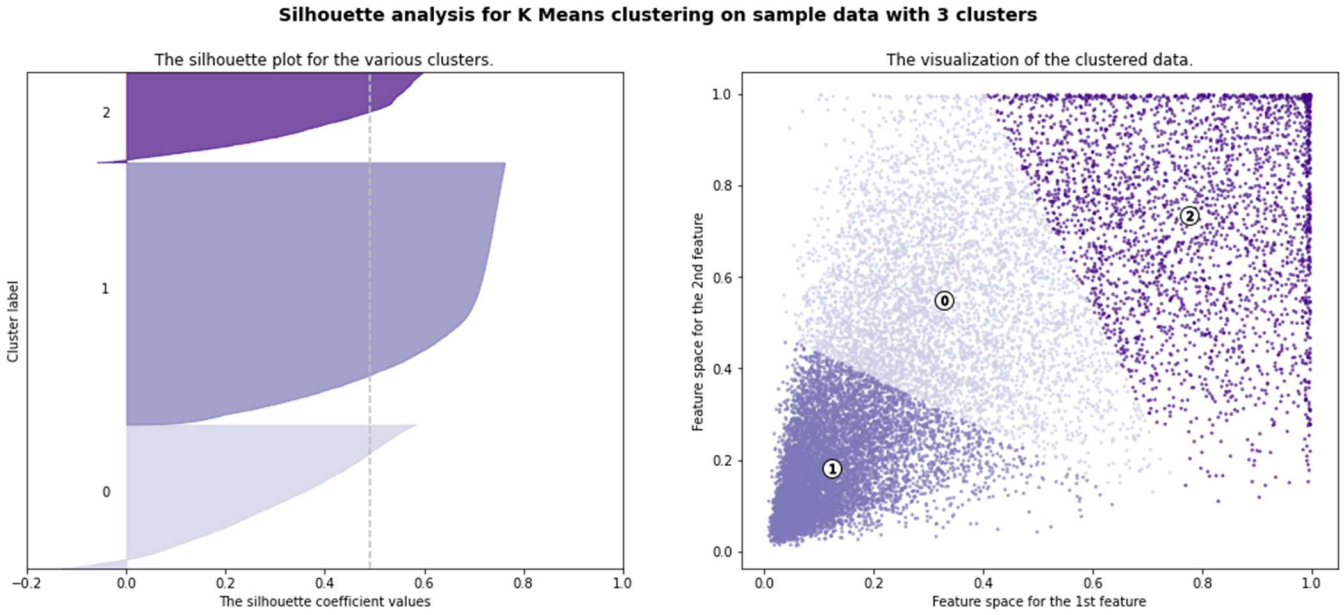



**SILHOUETTE SCORE :**

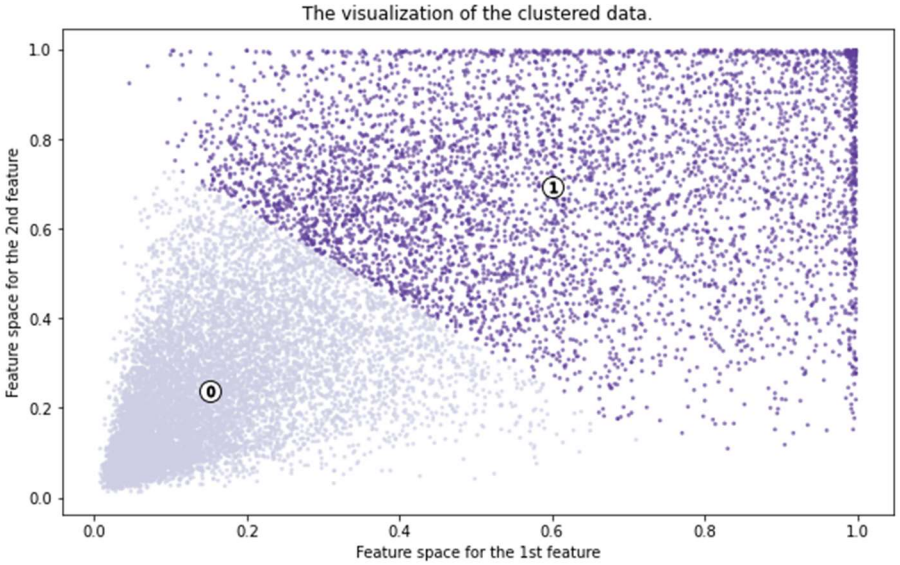


## SILHOUETTE ANALYSIS :



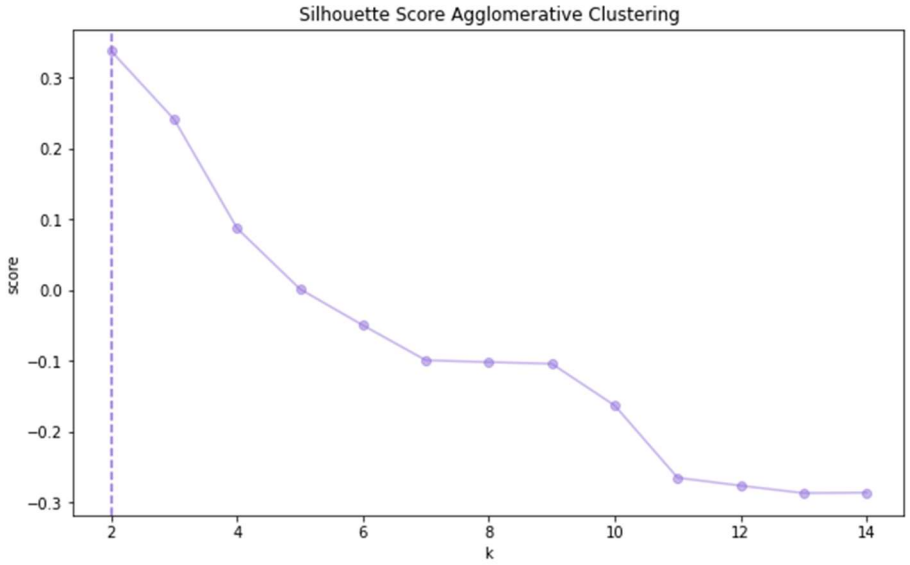
Clustered data :



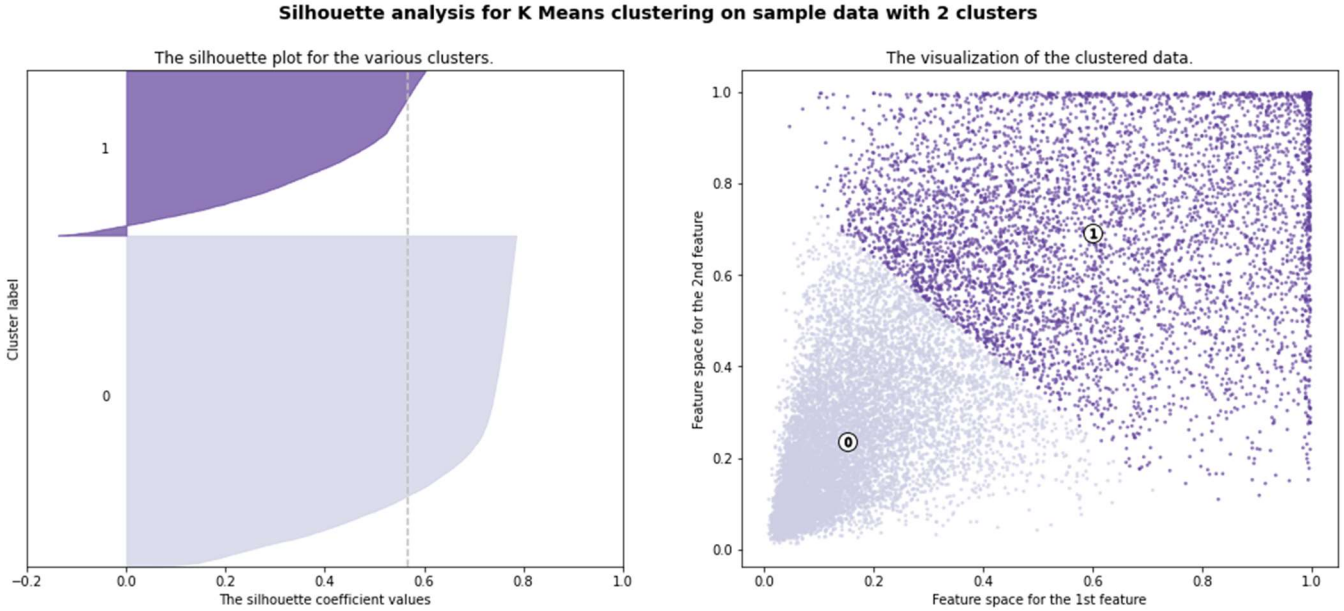
## Agglomerative Clustering

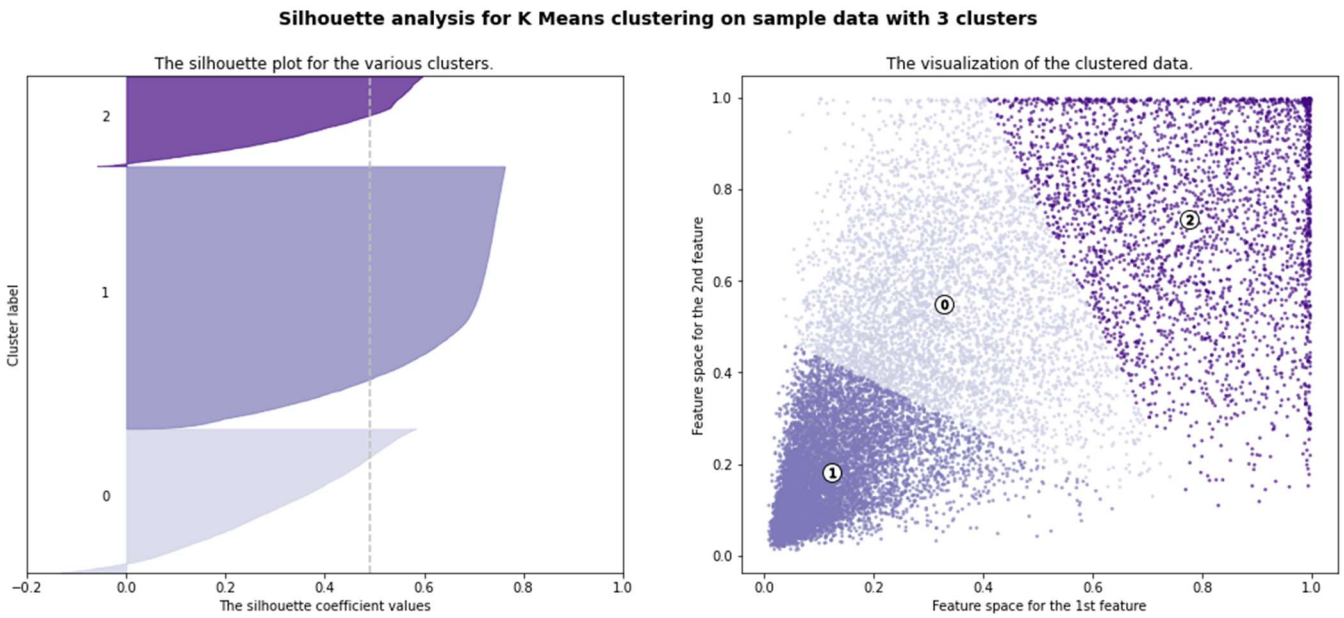
Tuning hyper parameters :

SILHOUETTE SCORE :

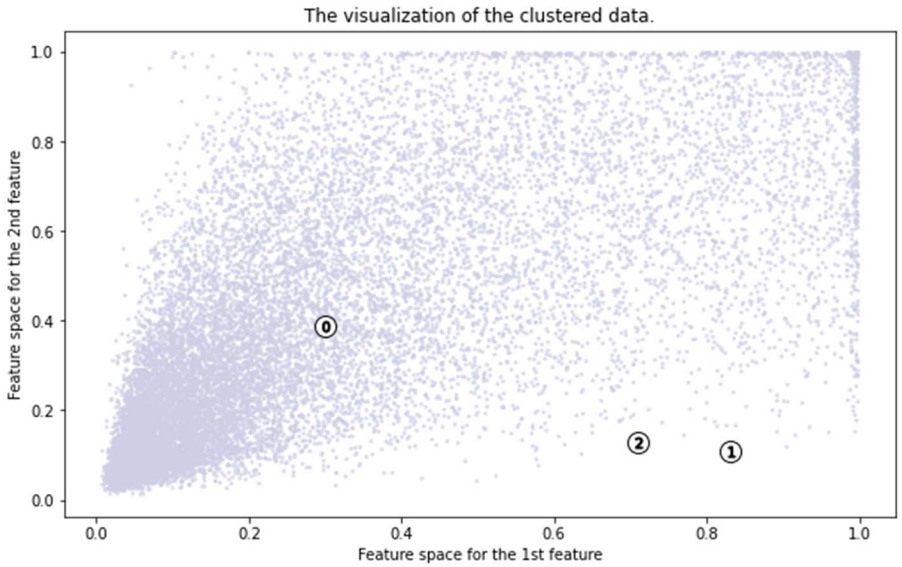


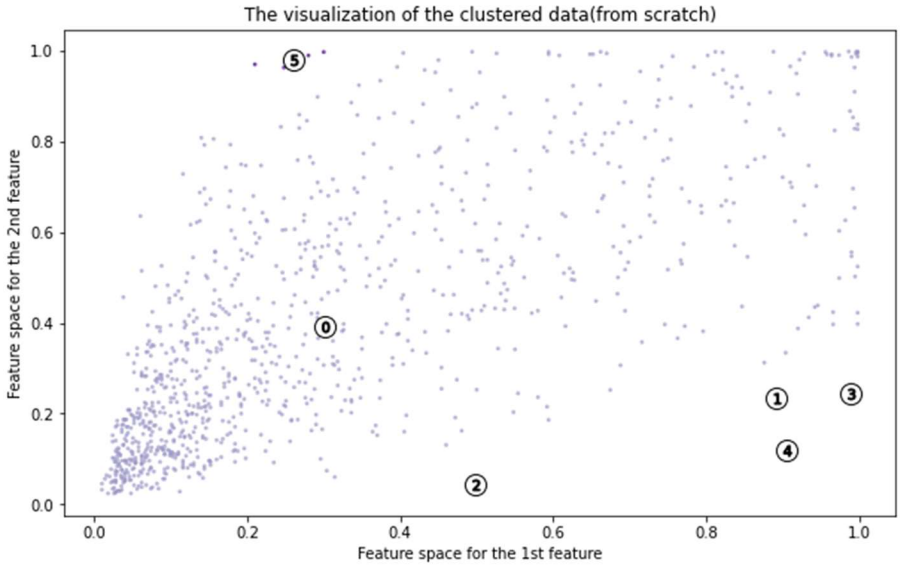

SILHOUETTE ANALYSIS :



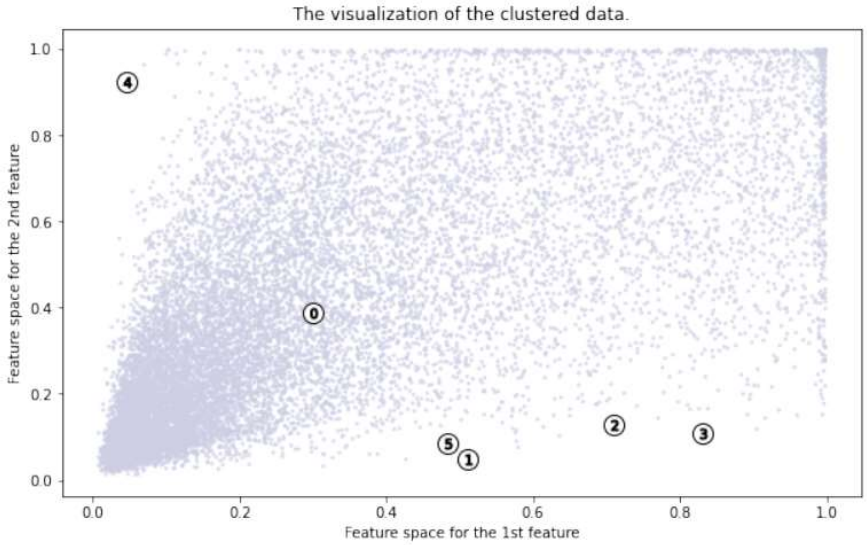



Clustered data :






Due to the very high clustering time of the data, it was clustered using samples

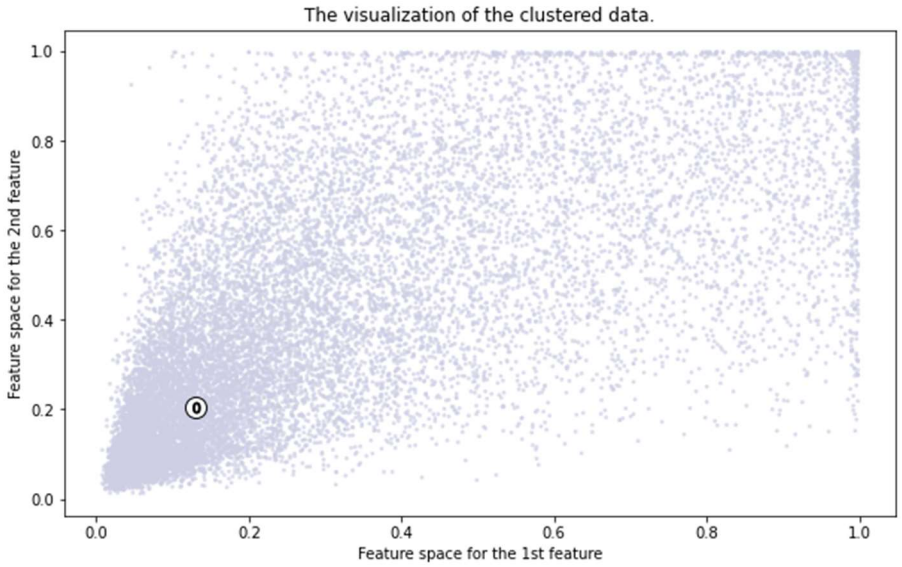


## Mean Shift

Tuning hyper parameters :

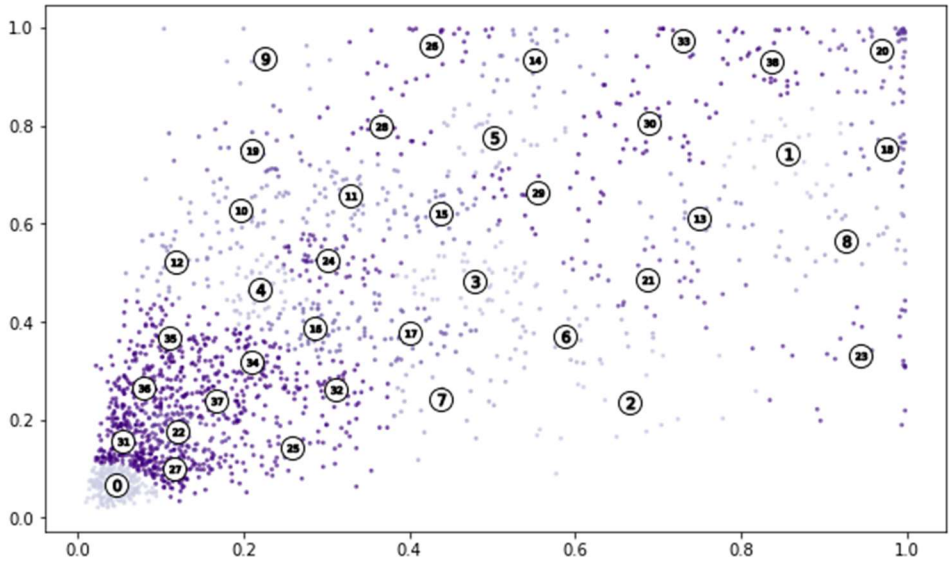
SKLEARN.CLUSTER.ESTIMATE_BANDWIDT: 0.27

Clustered data :




## Affinity Clustering

Clustered data :

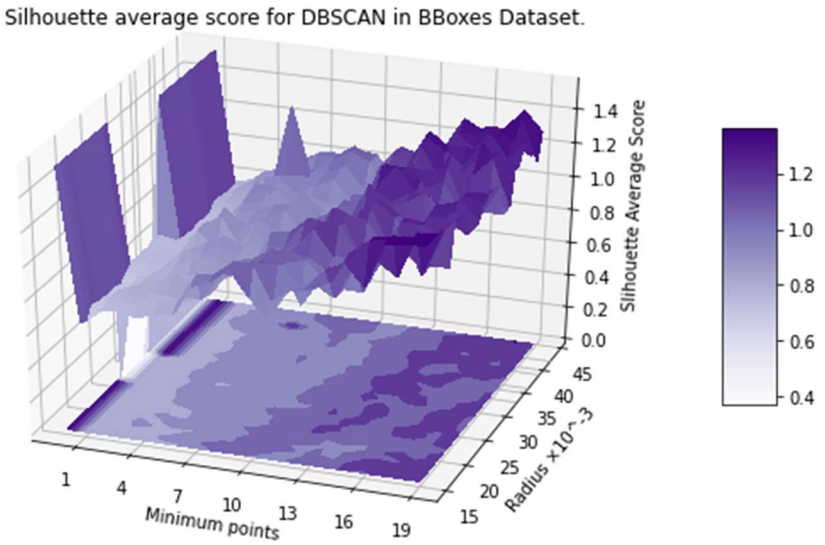


Due to the very high clustering time of the data, it was clustered using samples

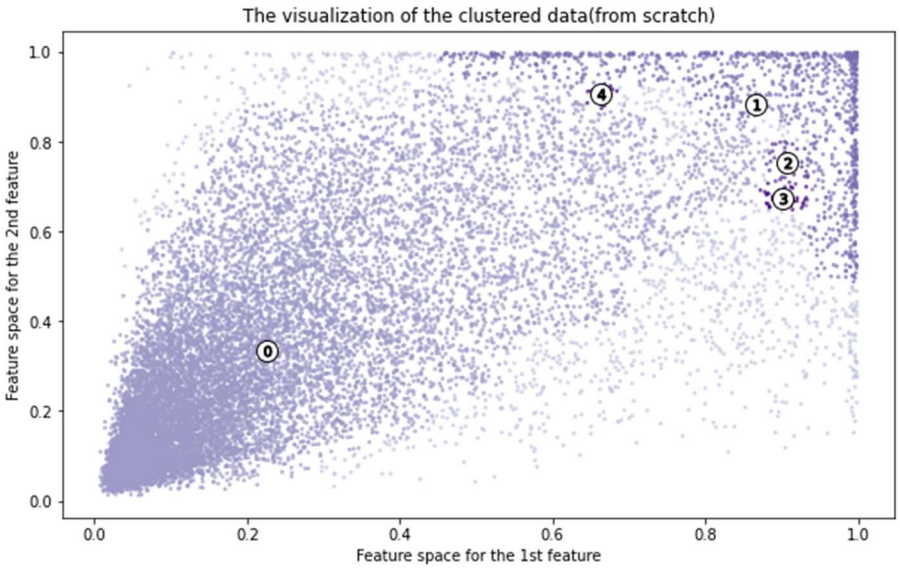
## DBSCAN

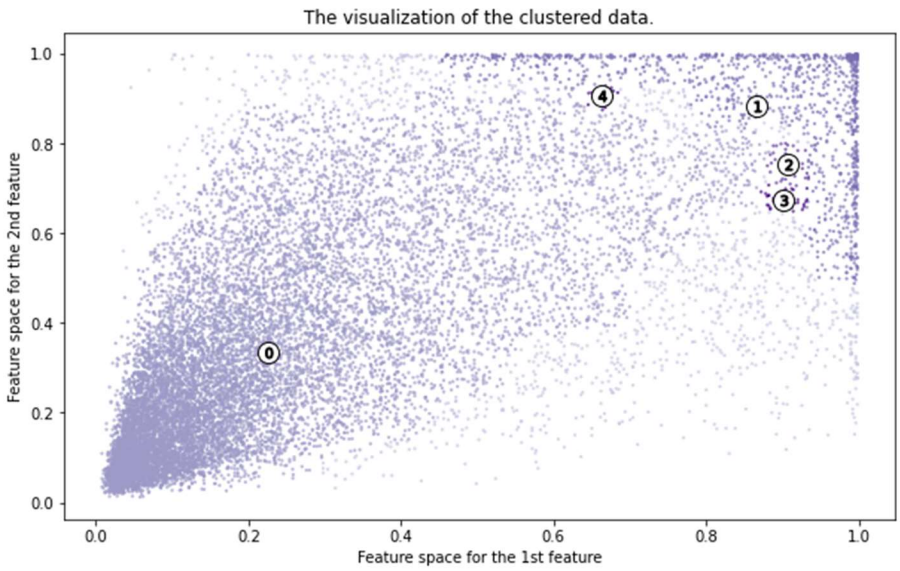
Tuning hyper parameters :

**SILHOUETTE SCORE :**



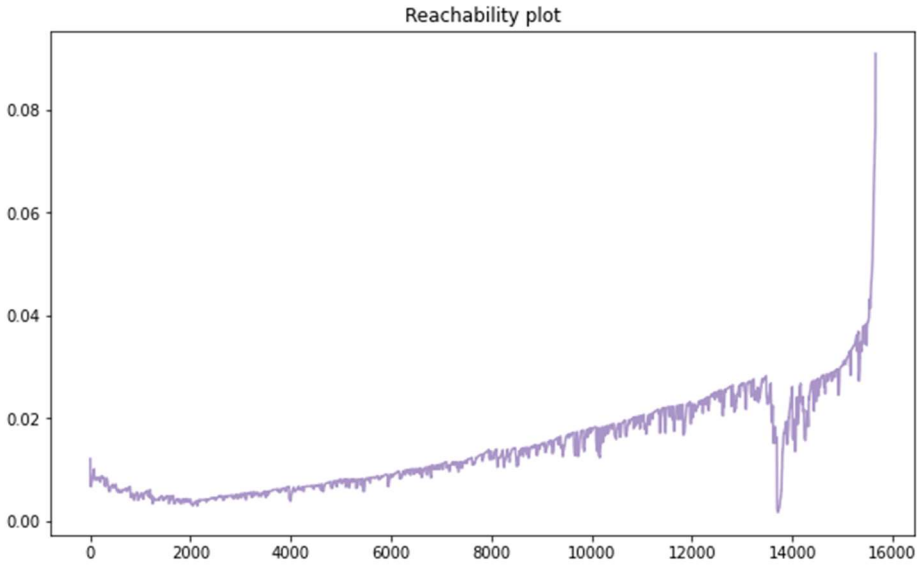

Clustered data :



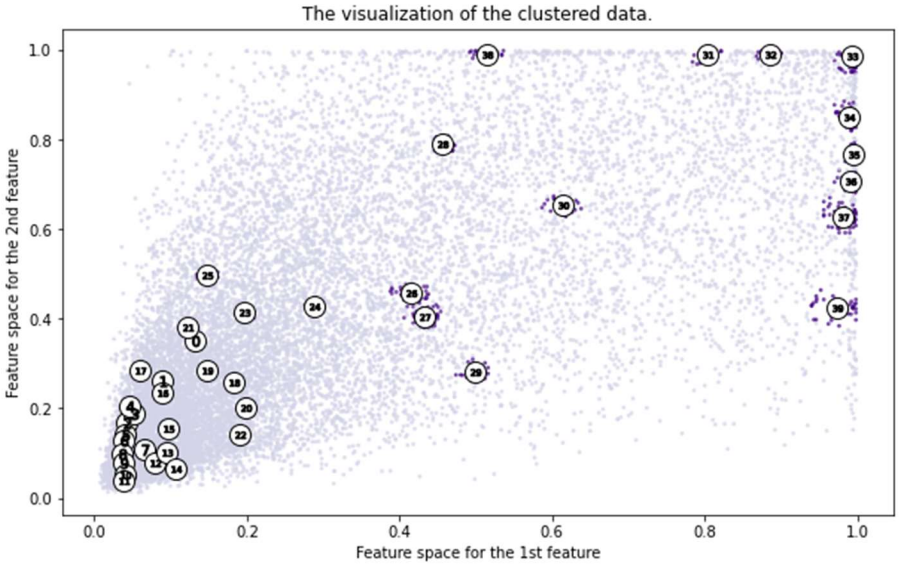

The visualization of the clustered data.
Feature space for the 2nd feature
Feature space for the 1st feature
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4

## OPTICS

Reachability plot :




Clustered data :



This type of clustering detects noise and  labels it with -1, which is shown in a different color in the figure above.

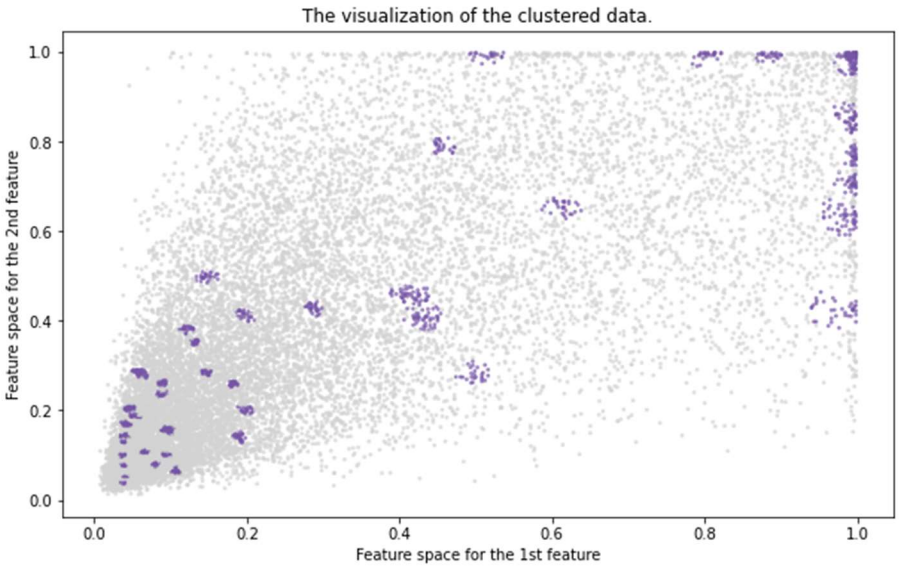
```
Estimated no. of clusters: 41
Estimated no. of noise points: 14386
```

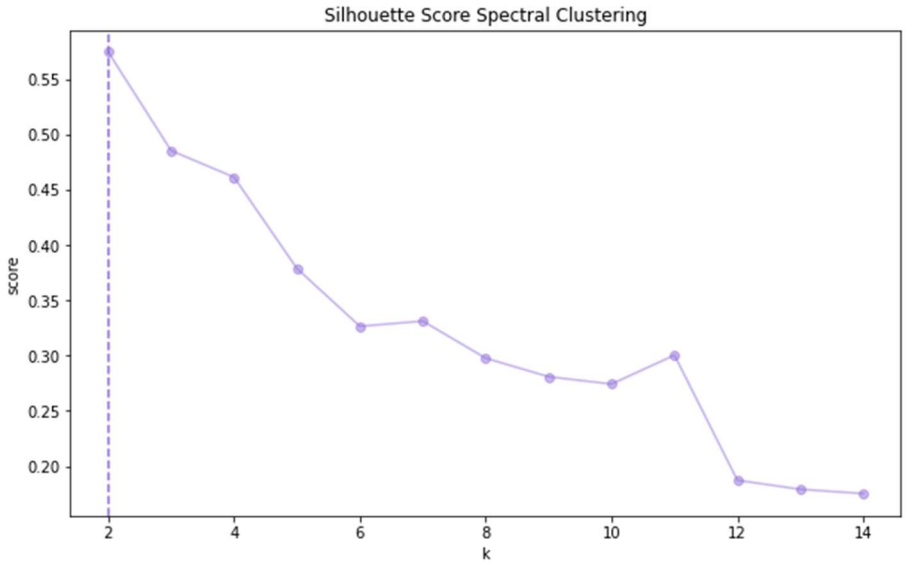
## Spectral Clustering

Tuning hyper parameters :

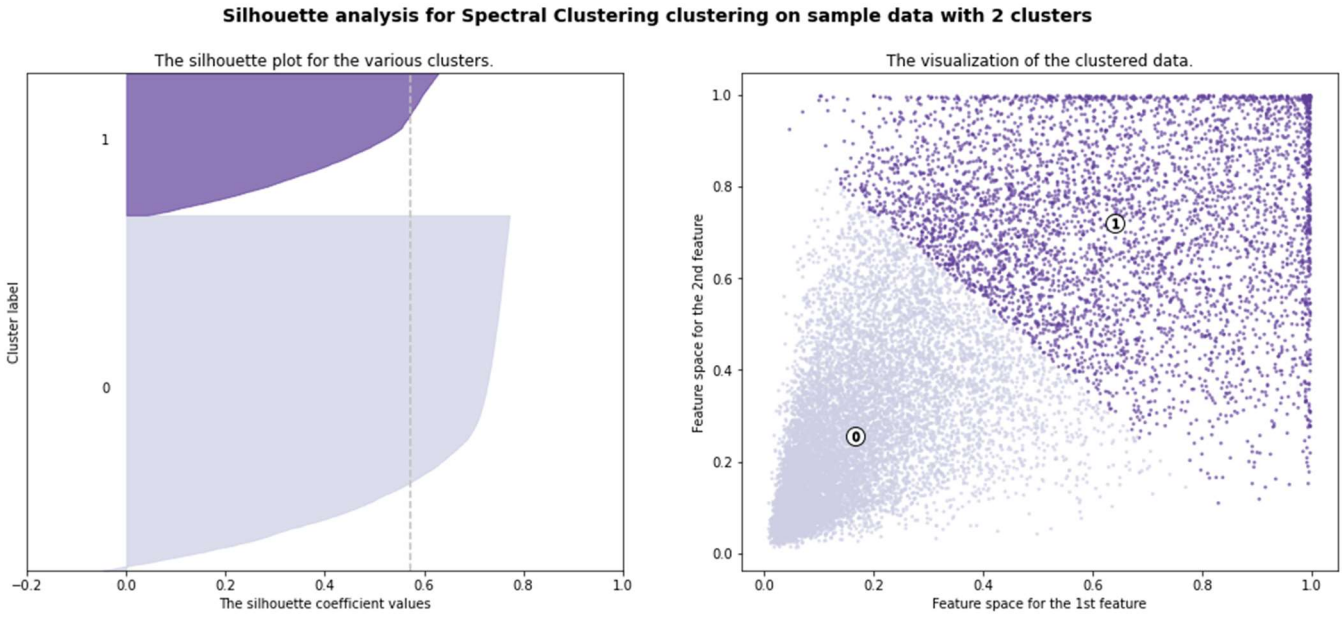
**SPECTRAL GAP:**

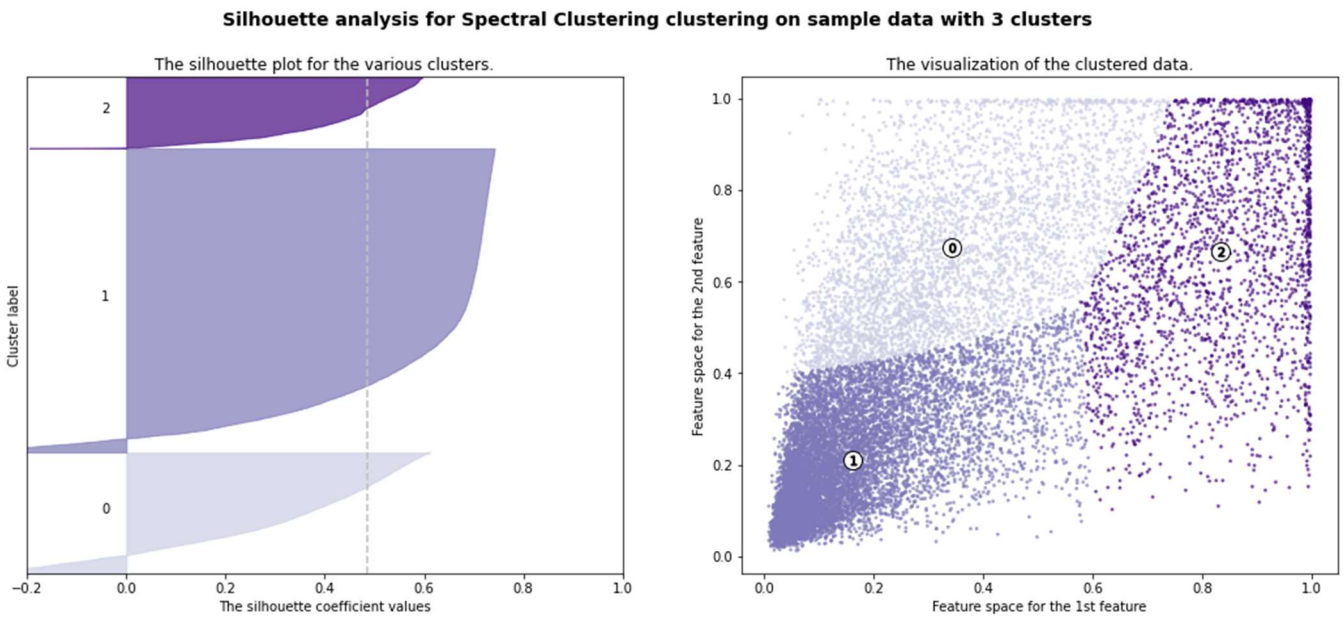
**SILHOUETTE SCORE :**





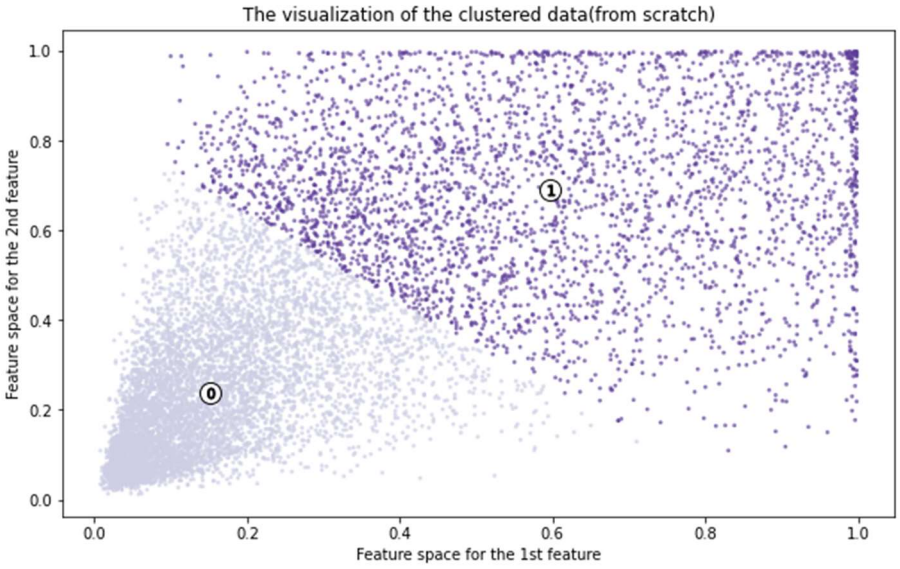

## SILHOUETTE ANALYSIS :



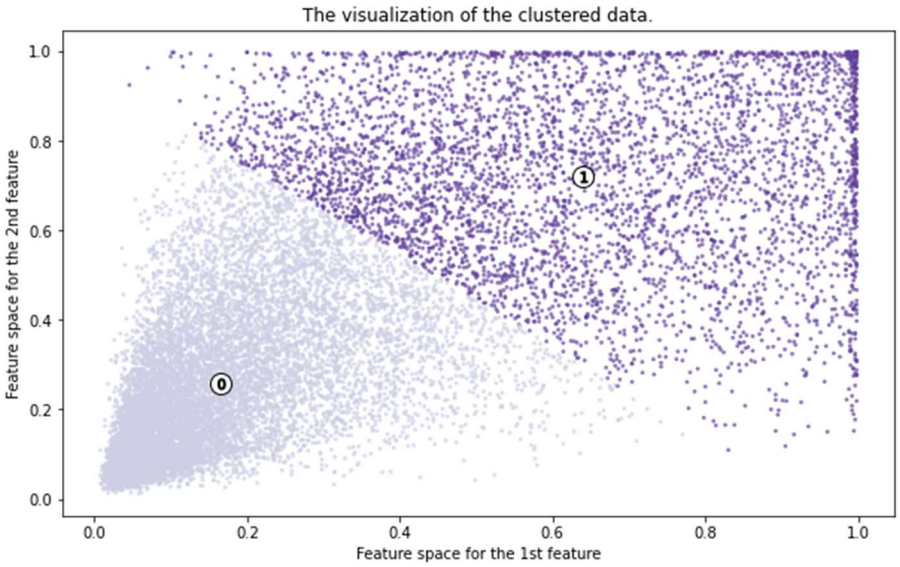



Clustered data :

K NEAREST NEIGHBORHOOD METHOD , K = 5




Due to the very high clustering time of the data, it was clustered using samples

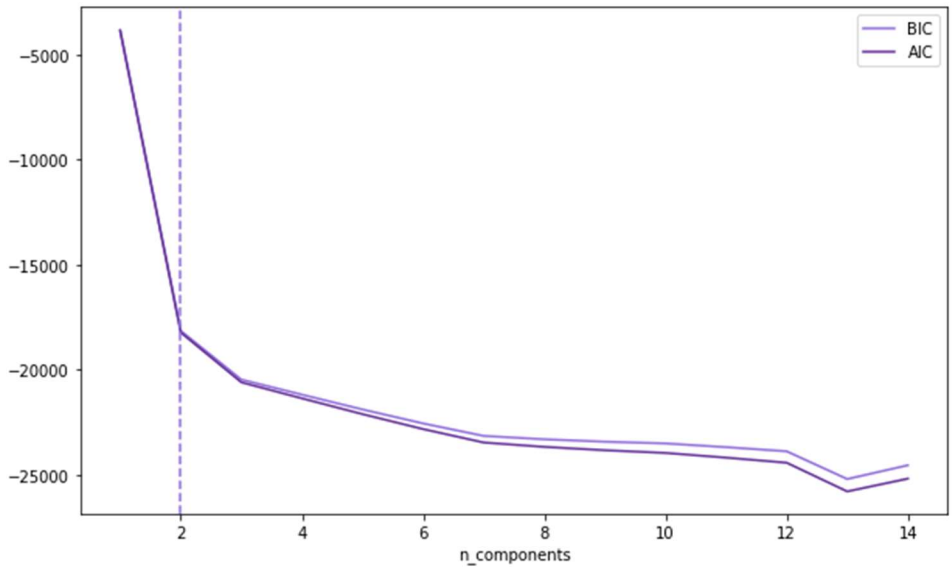



Due to the very high clustering time of the data, it was clustered using samples

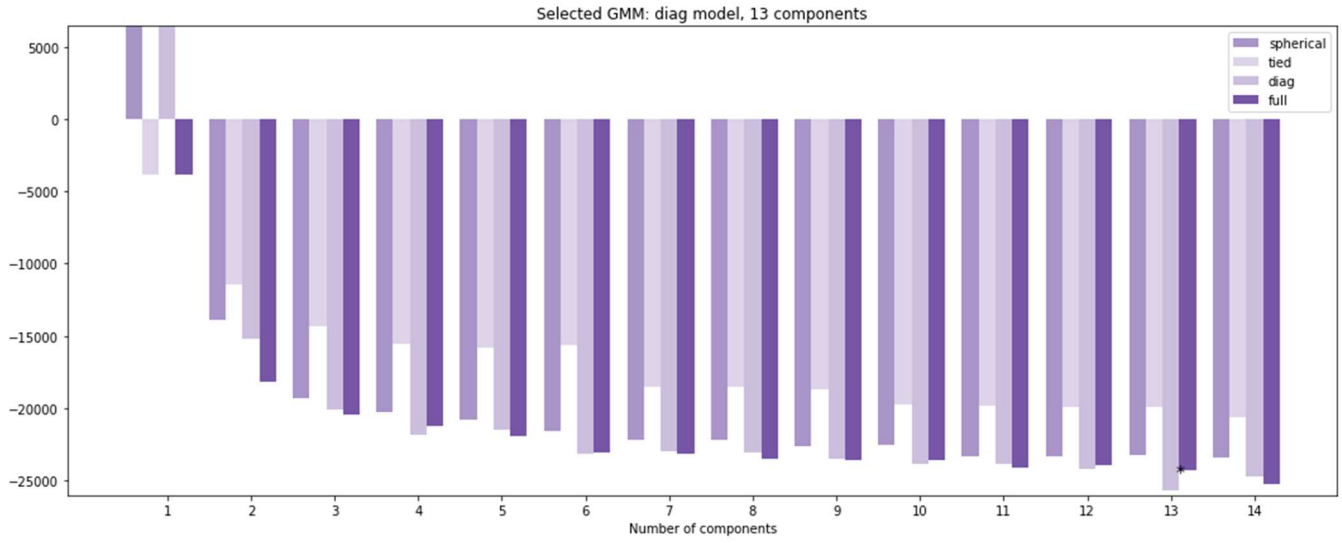
## Gaussian Mixture Model

Tuning hyper parameters :

**BIC AND AIC:**



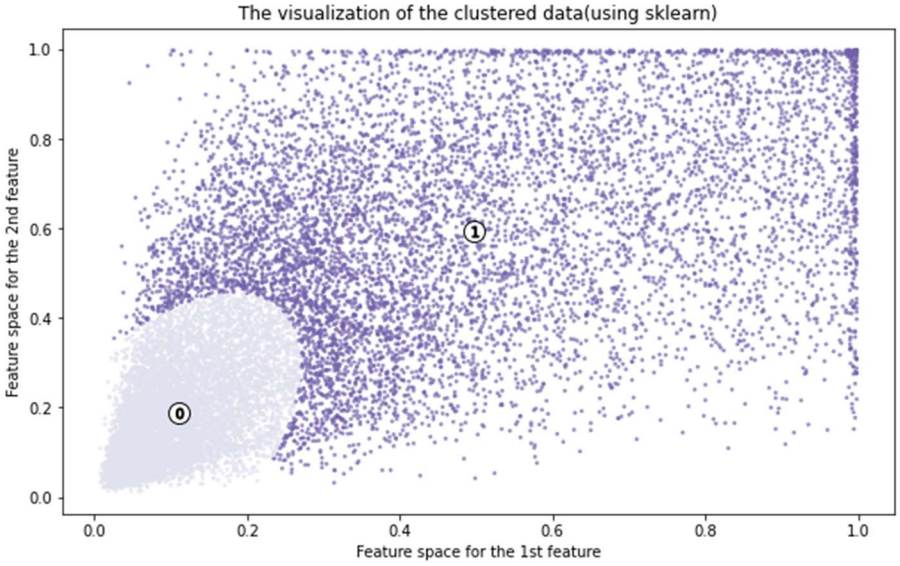

**BIC WITH DIFFERENT COVARIANCE TYPE :**

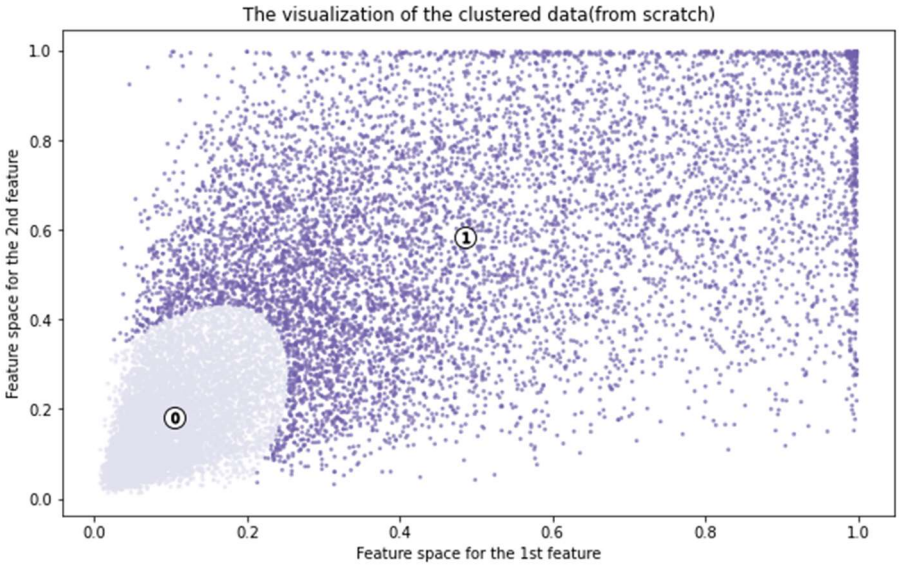


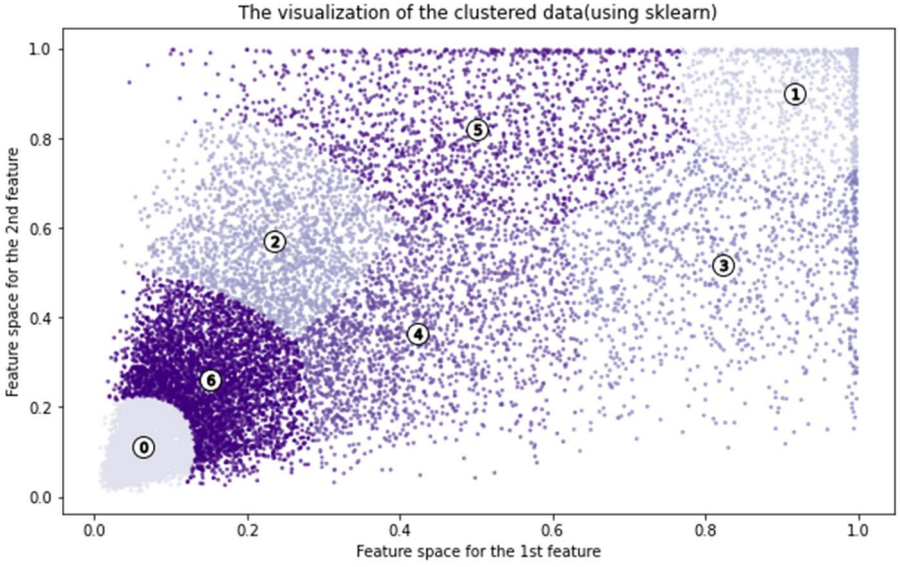
## SILHOUETTE SCORE WITH ERROR BAR :

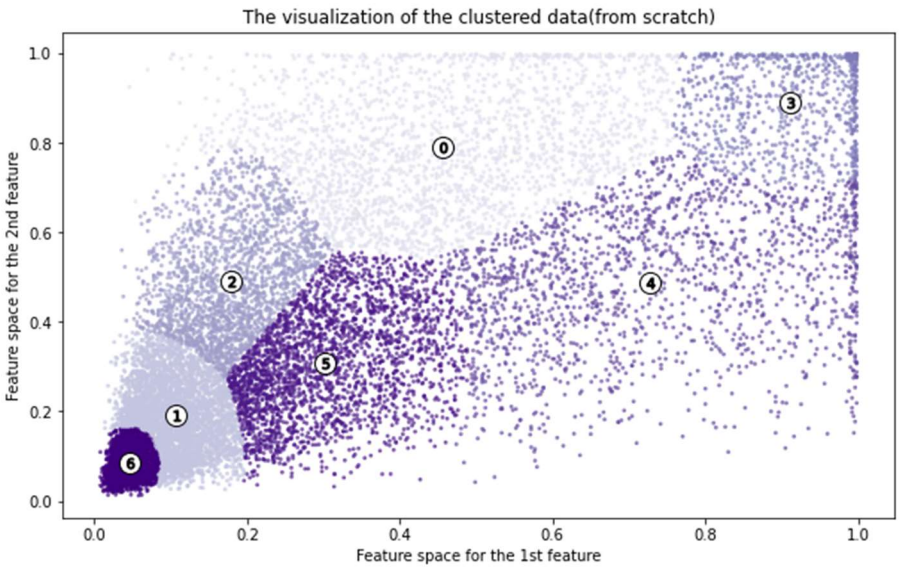

As can be seen from the error-plot, the lowest amount of stochasticity occurs in GMM when clustering with component size of either 2, 3, 5 or 7.

Clustered data :



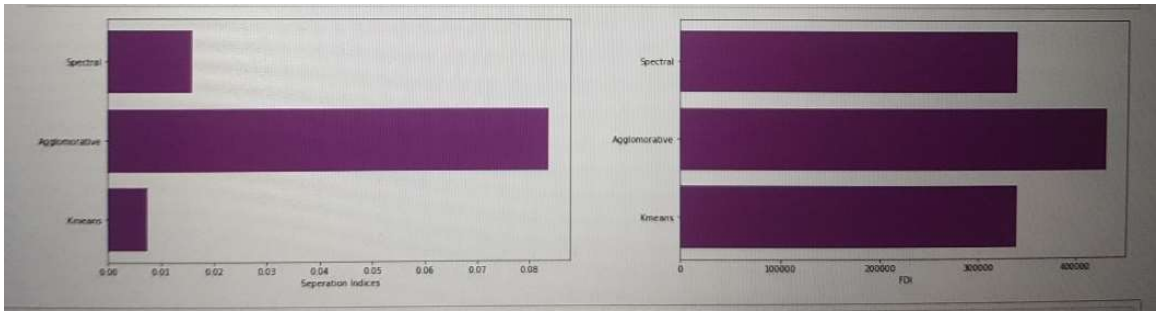

The visualization of the clustered data(from scratch)
Feature space for the 2nd feature
Feature space for the 1st feature
0.0
0.2
0.4
0.6
0.8
1.0
0
1


The visualization of the clustered data(using sklearn)
Feature space for the 2nd feature
Feature space for the 1st feature
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6




## Final Comparison of Tuned Algorithms using Metric Indicators

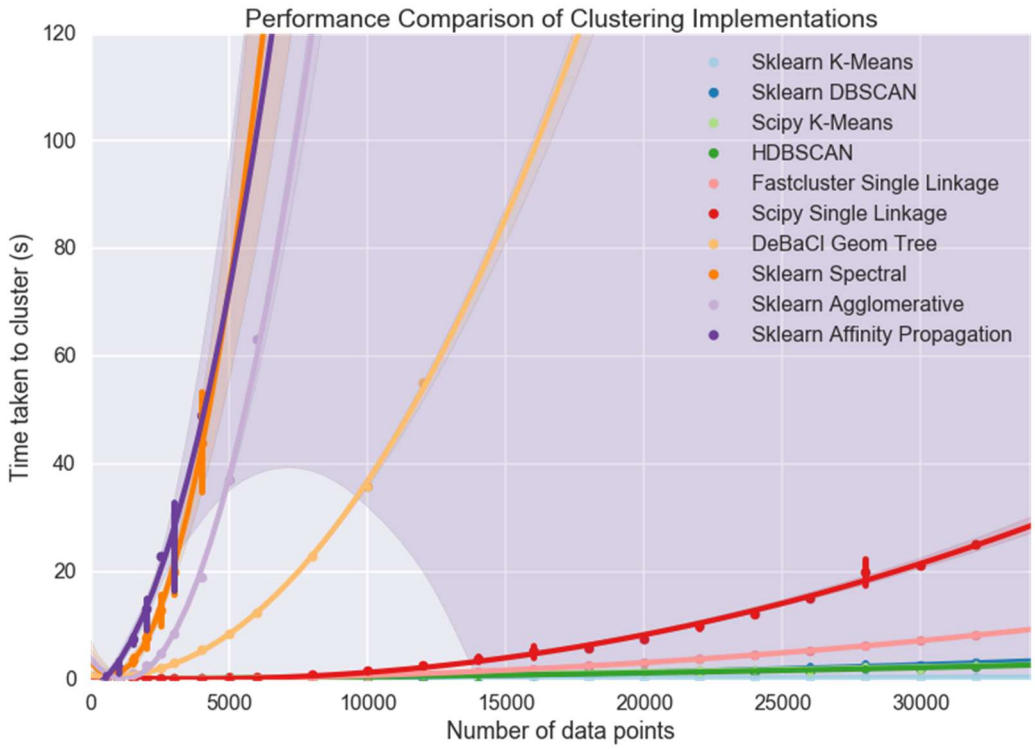
We have shown the separation index and fisher metrics associated to our results using horizontal bar chart. As Singularity is a very common issue in "Within Covariance" Matrix, we used a small epsilon number that results in non-singular Matrix, but increases fisher index value by a tremendous scale. (epsilon = $\varepsilon = 10^{-5}(Data\ Range)$)




Sorry for poor and incomplete data demonstration. We will show the complete results in the power-point presentations, hopefully.

# Conclusion

- Some algorithms that had high time/computation complexity implemented for 'Toy data set' needed sampling methods to reduce data size




- Some Algorithms are very sensitive to initial parameters, we used silhouette indicator to tune each algorithm that was also mentioned in internal evaluation indicators in the paper.

- As, B-boxes Dataset is high dimensional data used for the YOLO algorithm, different dimension reduction techniques, can yield different feature representations and therefore different clustering results. We followed the formulas in the Projects Instructions

- As GMM algorithm is a stochastic clustering algorithm, the initial parameter should randomly be chosen, but we tuned it to control the sensitivity

# References


https://www.analyticsvidhya.com/blog/2020/09/how-dbscan-clustering-works/

https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/#:~:text=Agglomerative%20clustering%20works%20in%20a,new%20bigger%20cluster%20(nodes).&text=Divisive%20clustering%20is%20good%20at%20identifying%20large%20clusters.

https://towardsdatascience.com/dbscan-clustering-explained-97556a2ad556

https://www.datanovia.com/en/lessons/dbscan-density-based-clustering-essentials/

https://scikit-learn.org/stable/modules/clustering.html

https://hdbscan.readthedocs.io/en/latest/performance_and_scalability.html

https://towardsdatascience.com/spectral-graph-clustering-and-optimal-number-of-clusters-estimation-32704189afbe

https://scikit-learn.org/stable/auto_examples/cluster/plot_optics.html

https://towardsdatascience.com/how-to-code-gaussian-mixture-models-from-scratch-in-python-9e7975df5252

https://towardsdatascience.com/math-and-intuition-behind-affinity-propagation-4ec5feae5b23

https://www.adeveloperdiary.com/data-science/machine-learning/linear-discriminant-analysis-from-theory-to-code/