ML701 Machine Learning: Group 2 - Inquiry into Strengths and Limitations of Clustering Methods - Final Report

Ghanim Alsuwaidi - 22010350
MBZUAI - Fall 2022 - November, 2022

Abstract

This project reviews the clustering output of various clustering methods on synthetic data with the goal of exploring the strengths and limitations of each. The report introduces the several clustering methods and their respective algorithms, followed by an overview of the synthetic data-sets, and proposes methods to estimate the necessary parameters for a given clustering method. Finally, the report concludes with visual representations of the clustering outputs, and comments on how the clustering outputs reflect the method's mechanisms.

1 Introduction

Clustering is a grouping technique employed on a set of observations in a given data-set in which more similar observations are classified as belonging in the same 'cluster' based on shared characteristics that sufficiently differentiate an entity from those in different clusters, as per the criterion of similarity or dissimilarity that a given clustering method employs.

This project is dedicated towards the study of the various clustering methods, specifically towards a given method's mechanism, and its eligibility or performance as compared to other methods as applied to various types of theoretical data.

Studying the mechanisms of clustering methods is a viable way to gain insights as to which clustering method is best employed upon a given data-set or towards a desired goal.

A particular challenge in facilitating this study will be to propose parameter selection methods to best represent the performance of the clustering methods. Furthermore, the 'accuracy' of a method cannot be endorsed if it is applied on unlabelled data, as it will cluster observations based on properties native to the data, irrespective of any nuance that could be present in real data.

The five clustering methods to be inspected are **K-Means** (with K-Means++ initialization [1]), **AGNES** (Agglomerative Nesting [2]), **BIRCH** (Balanced Iterative Reducing and Clustering using Hierarchies) [3]), **DB-SCAN** (Density Based Spatial Clustering of Applications with Noise [4]), and **GMM** (Gaussian Mixture Models [5]),

A flowchart of the finished code is available under section **'Progam Code Flowchart'**

2 Related Works

The project code-base is derived from the **Scikit-Learn 'Comparing different clustering algorithms on toy data-sets [6]'**. This document examines the output of multiple clustering methods applied unto several data-sets.

The code-base has been adapted to suit different clustering methods, and additional data-sets. Furthermore, an automatic parameter selection and performance inspection module has been implemented for the purpose of this project.

3 Methodology

3.1 K-Means

Initialize *'k'* cluster centroids randomly within the scope of the data, then:-

- **Step One:** Assign samples to nearest centroid based on a distance function.
- **Step Two:** Shift the position of the centroids to the mean of all member samples.

The two steps are repeated until convergence (no change in centroid position / cluster labels).

*'**K-Means++**'* initialization sets the furthest *'k'* quantity samples as initial cluster centroids instead, yielding a consistent result relative to regular initialization.

3.2 AGNES (Agglomerative Nesting)

Specify a dissimilarity (distance) and linkage metric, then:-

- **Step One:** Form a cluster between the two least dissimilar samples.
- **Step Two:** Compute the new dissimilarities between the samples.

The two steps are repeated until all observations are part of one cluster, with an accompanying dendrogram to show the clustering order.

This method may be customized by setting a limit on the number of clusters to produce, or to end clustering when a dissimilarity threshold is surpassed.

A brief description of the available linkage metrics are included in the appendix under section **'AGNES Linkage Functions'**. The linkage function employed in this project is 'Average' linkage.

3.3 BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)

This method performs clustering in a single scan of the data. It constructs a height-balanced tree designated as a **'Clustering Feature'** (CF) tree. A visual representation is available in the appendix under section **'Clustering Feature Tree Schema'**

A given node on this tree takes the form of [CFi, Childi], where Childi is is a pointer to the non-leaf node's i-th child node, and CFi is the CF of the sub-cluster represented by this child, and so non-leaf nodes represent each a cluster formed by all of the sub-clusters within their entries.

An individual 'CF' is a vector containing three values representing the member samples:-

- **N** : The number of observations in the cluster.
- **LS** : The linear sum of *N* samples.
- **SS** : The square sum of *N* samples.

Relevant features regarding the cluster may be inferred with these three values, namely, the **centroid** , the **radius**, and the **diameter** of the cluster.

The method requires the specification of two parameters *'B'* (branching factor) and *'T'* 'threshold'.

The following steps outline the algorithm of this clustering method:-

- **Step One:** Starting from the root node (alternatively make the root node of the CF tree with the first scanned observation), recursively descend down the CF tree by choosing the child node that is closest to the newest entity based on the dissimilarity metric specified.
- **Step Two:** Upon reaching a leaf node (as there are no more nodes to descend through), inspect the least dissimilar entry in the leaf node (e.g. Li), and:-

  - If the observation satisfies the T threshold, then Li 'absorbs' the entry and its CF vector is updated.
  - Else: If there is room on the leaf node for the observation (the limit set by B), then a new entry is made on the leaf node for this observation.
    * If there is room in the preceding node for the newly split non-leaf node, then the latter node is simply added to the former node.
    * Else the preceding node an additional level above must also be split.
- **Step Three:** Update the CF of all non-leaf nodes along the path to the leaf node.

The CF tree will be complete when these three steps have been performed on all observations in the data, and future additions will only be facilitated via the tree, instead of operating on the data-set itself, which is how **BIRCH** completes clustering in a single scan of the data-set.

3.4 DB-SCAN (Density Based Spatial Clustering of Applications with Noise)

Specify parameters *'epsilon'* (radius around a sample to check for) and *'minPts'*, then designate samples in the data as:-

- **Core:** if the quantity of neighboring samples within the range of *'epsilon'* are equal to or surpass the *'minPts'*.
- **Border:** if the sample borders a core point yet does not surpass the *'minPts'* threshold.
- As a **'Noise' (or Outlier) point otherwise.**

Border points are members of the same cluster as the nearest core points.

3.5 GMM (Gaussian Mixture Models)

Initialize *'k'* cluster centroids with a random **mean** and **covariance**, then:-

- **Expectation Step:** Calculate the probability of a sample belonging to each cluster (P(c|x)), and assign the sample to the most probable cluster label.
- **Maximization Step:** Compute the new **mean, covariance, and P(c) of each cluster.**

The two steps are repeated until convergence (no change in cluster labels / cluster parameters).

As **GMM** clustering assigns points to a cluster based on probabilities, the designations are considered to be 'soft', wherein a given point is only more likely to be a member of a specific cluster than another.

4 Experiments

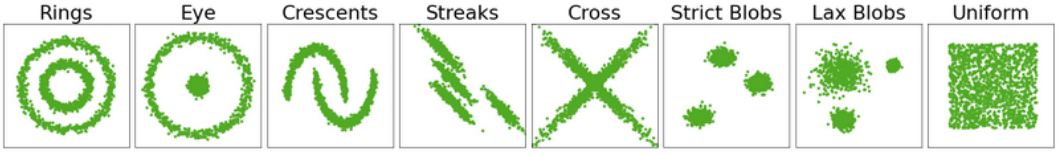
4.1 Data-sets

Towards the goal of comparing the performance of each clustering method, the test distributions must be sufficiently varied in shape, to properly accentuate the differences between each clustering method.

In regards to clustering, input data may be a graph-based representation of an image or a distribution of a predictor unto a response variable.

The scope of this project only encompasses two-dimensional data , yet uni-variate or three-dimensional clustering is possible. Furthermore, only euclidean space measurements are used to facilitate clustering.

The following eight different distribution types will be employed as test distributions:-



- **Rings:** Two rings where one is larger and contains the smaller ring. Its distinct feature is its continuous data such that only the individual rings form separate entities.
- **Eye:** A cluster of data encircled by an outer ring, forming an eye. Its distinct feature is the existence of a discernible central cluster as opposed to the 'Rings' set.
- **Crescents:** Two opposite curves where one edge of each curve is in close proximity to the other curve's center. The proximity of the curves and their shape itself are their distinct feature.
- **Streaks:** Four diagonally-warped distributions. The shape and noise surrounding the streaks are their distinct features.
- **Cross:** Two intersecting streaks. The overlapping structure is this set's distinct feature.
- **Strict Blobs:** A data-set of three strictly grouped clusters. Its simplicity is its distinct feature.
- **Lax Blobs:** Three distributions of varying standard deviations. Their varying densities impose a unique case for the methods.
- **Uniform:** A uniform distribution with no discernible cluster. Its uniformity may suggest the absence of clusters, but some methods may partition the distribution into clusters within.

4.2 Parameter Estimation

Automatic parameter estimation is implemented in several ways depending on the mechanism of a clustering method. This is especially the case when considering that the **DB-SCAN** method is not concerned with producing an output that sufficiently separates the clusters, as would be the case with other methods.

As such, it is necessary to implement parameter estimation methods tailored to a given method.

The parameter estimation methods used in this project are as follows:-

- **K-Means & BIRCH:** Parameter estimation applied for cluster quantity. Implemented by looping the method on iterating values of cluster quantity, and computing the Silhouette Score of the output. The cluster quantity yielding the highest score is selected.
- **AGNES & DB-SCAN:** Parameter estimation applied for dissimilarity and epsilon threshold respectively. Implemented via the use of k-nearest neighbours to identify a cut-off point where any further increase of the threshold requires the inclusion of outliers to the clusters. When plotting the threshold versus samples, a discernible 'knee' or 'elbow' is usually visible, which identifies this cut-off. The selection is implemented by selecting the point on the line with the most prominent curve.
- **GMM:** Parameter estimation applied for cluster quantity. Implemented by looping the method on iterating values of cluster quantity, and computing the **BIC** (Bayesian Information Criterion) of the output. The cluster quantity yielding the lowest score is selected.

5 Results

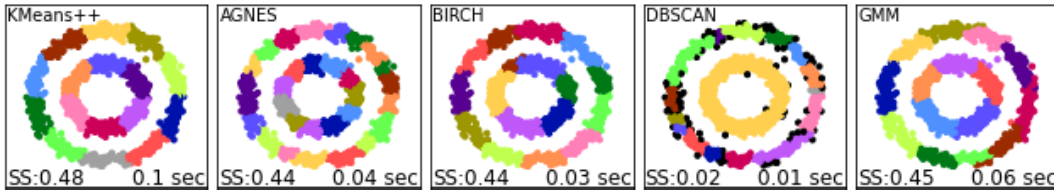
The following figures are the results of the chosen clustering methods applied on the test distributions introduced in the **'Experiments'** section. The columns represent the method employed in the order of **K-Means++**, **AGNES**, **BIRCH**, **DB-SCAN**, and **GMM**.

The values by the bottom-left corner represents the Silhouette score of the clustering outcome on the data-set, while the value by the lower-right represents the time taken for the clustering method to produce its output on the given distribution.

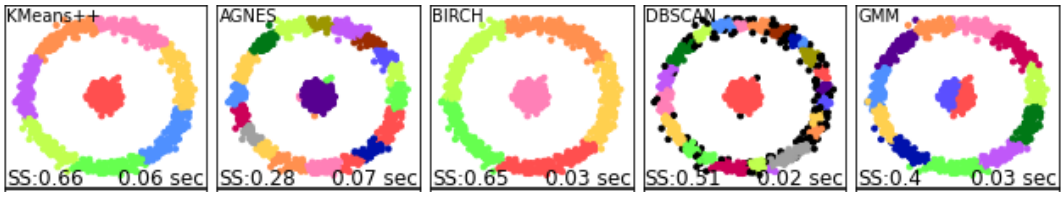
The color of an observation represents the cluster designation it is part of. However, the black observations are outliers that do not belong to any cluster. Outliers are currently only observed in the **DB-SCAN** clustering outputs.

According to the Silhouette metric, **K-Means** is the best performing clustering method overall, while the **DB-SCAN** is a consistent under-performer according to the same metric.

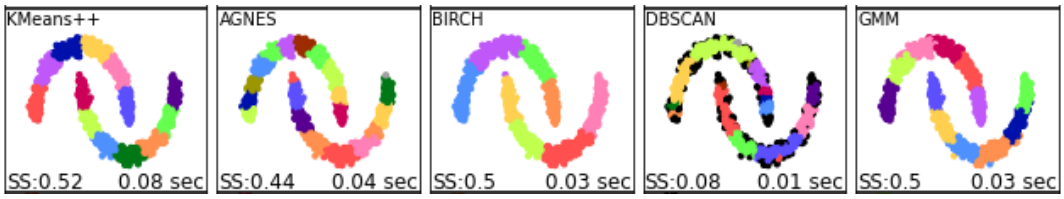
Timewise (yet subject to change by hardware or cluster parameters), **DB-SCAN** consistently produces clustering outputs in the least amount of time overall. On the contrary, **K-Means** takes the longest amount of time. Aside from the aforementioned algorithms, **BIRCH** is the second fastest algorithm, in reality completing the process in the same time for all distributions. **GMM** and **AGNES** are the third and fourth fastest, respectively.



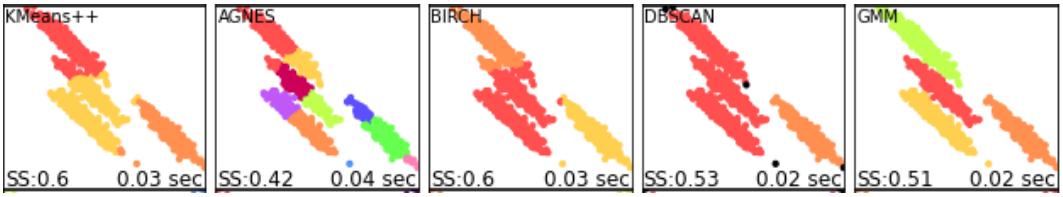
For the 'Rings' data-set, most methods produce similar results where the both the inner and outer rings are partitioned into clusters. The **DB-SCAN** algorithm produces a vastly different output for this data-set, where the inner ring is clustered as one entity, and the outer ring features multiple clusters where the threshold is met. Decreasing the threshold would result in a cluster output where two clusters are formed by the inner and outer rings. The **GMM** algorithm features similar output to the **K-Means**, **BIRCH**, and **AGNES**, but additionally exhibit overlapping cluster extremities.



Next, for the 'Eye' data-set, K-Means, **AGNES**, **BIRCH**, and the **GMM** algorithms all produce similar results where the outer ring is partitioned into several clusters reminiscent of the shape found in the 'Rings' data-set outputs. All algorithms consider the 'pupil' in the data to be one cluster, with the exception of the **GMM** algorithm, which designates two clusters to this region. Once again the **GMM** algorithm features overlapping cluster extremities. The **DB-SCAN** algorithm performs similarly to the previous data-set as for this one. Increasing the threshold will produce an output of two clusters of the 'pupil' and ring.

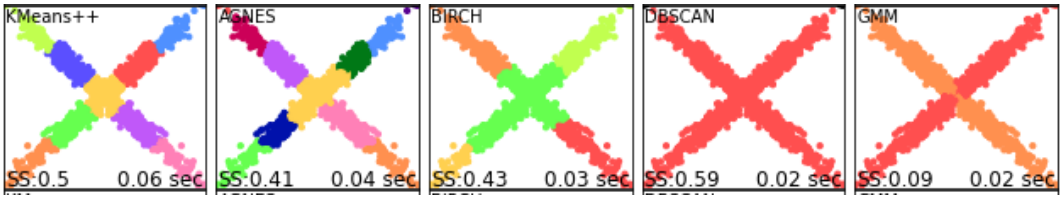


As for the 'Crescents' data-set, once again the **K-Means**, **AGNES**, **BIRCH**, and **GMM** methods produce similar results, where the crescents are separated into several clusters, only with varying cluster quantities. The highest cluster quantity produced by the above methods is with the **K-Means** algorithm, producing thirteen clusters for this data-set, while **AGNES** produces three clusters. The **GMM** algorithm was able to discover overlapping clusters in this data-set as well. When using a lenient threshold, the **DB-SCAN** method designated each crescent as its own cluster. Using the parameter selection implementation sets the cut-off point such that the crescent structures are formed of many clusters.

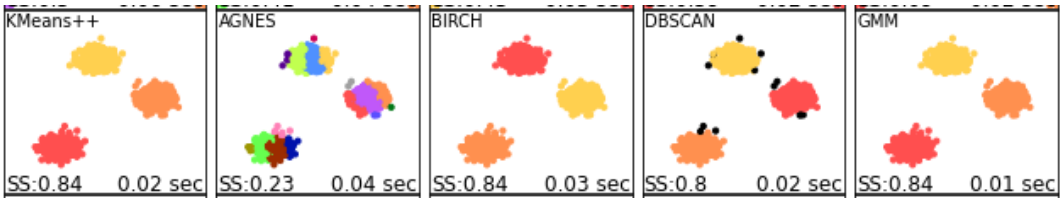


Concerning the 'Streaks' data-set, the **K-Means** and **BIRCH** methods produced very similar results, with minuscule differences in the extreme ranges of the clusters. The **AGNES** method

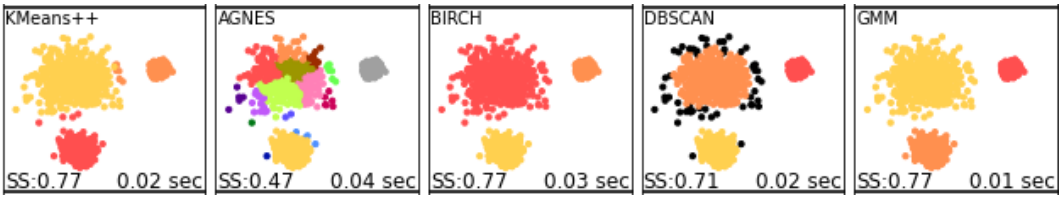
separated parts of the streaks into clusters. The proximity of the streaks did not enable the **DB-SCAN** algorithm to identify each streak as a separate entity, with the exception of the streak by the lower right. **GMM** however was able to produce four clusters, one for each streak.



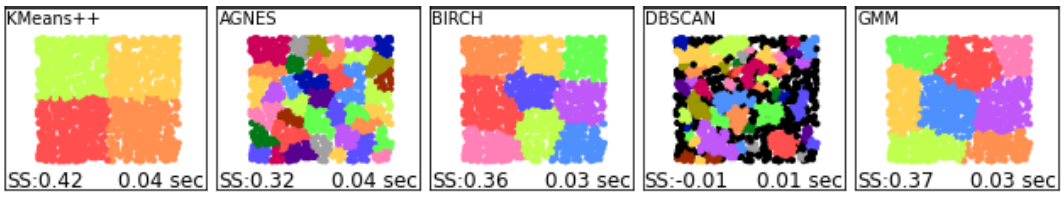
For the 'Cross' data-set, K-Means and **AGNES** both produced similar results in which the intersection of the two lines is a cluster region, and each limb of the cross contains two cluster regions, featuring nine clusters in total. The **BIRCH** algorithm produced a central cross cluster, with one cluster about each limb. **DB-SCAN** considers the entire cross to be one entity. **GMM** considers each of the two individual lines to be a cluster, overlapping by the intersection.



In regards to the 'Strict Blobs' data-set, all clustering methods produce identical results, excluding the **AGNES** method, which would produce a similar result with an lowered threshold, and **DB-SCAN** only inconsequentially differing slightly owing to the designation of some points as noise points.



Similar to the 'Strict Blobs' data-set, the 'Lax Blobs' data-set also elicits similar results from all clustering methods. **K-Means** however designated some points around the extremities of the yellow cluster to its sister clusters.



Finally, for the 'Uniform' data-set, the **K-Means** and **BIRCH** methods produced similarly shaped clusters, with the former designating each of the four quartiles as a cluster, and the latter producing nine clusters much similar to a 3x3 grid. The **AGNES** and **GMM** algorithms produced irregular cluster shapes for this data-set. The output of the **DB-SCAN** method clustered the observations with neighboring points, but the data is not equally spread throughout the data-set, and so it did not group all the data into one cluster.

6 Discussion

6.1 K-Means

K-Means consistently partitions non-circular shapes of groups of observations (e.g. the Rings, Eye, Crescent, Streak, Cross, and Uniform data-sets) into several sub-clusters. This method is

strictly concerned with the distance of observations from the centroids, and so any cluster formed by **K-Means** is naturally convex as a consequence.

Additionally, as is seen in the output for the 'Lax Blobs' distribution, this method labelled some observations on the extremities of the red cluster as belonging to the yellow or orange cluster. This may be an undesirable outcome as visually those observations seem to belong to the red cluster, and the other methods consistently behave as such.

While the Silhouette Score implies that the **K-Means** method is the best performing method, this is only due to the metric aligning with the mechanism of the method's clustering process.

This method could be applied on data-sets where partitioning the data irrespective of its overall shape is a desirable mechanism, such as effectively partitioning connected observations in the data into equally-sized sub-clusters, as could be applied to modelling the characteristics or behavior of specific entities. For example, if the data at hand is a two-dimensional distribution of the income and expenditure of customers at a commercial entity, **K-Means** can be used to equally partition the data-set into groups of customers (e.g. low earners, high spenders / high earners, high spenders, etc.) while providing the convenience of assisting the modelling process by providing the cut-off point for the groups.

6.2 AGNES

AGNES produces greatly varying cluster outcomes based on the linkage function used to calculate the dissimilarity between observations in the data. For example, using the 'single' linkage function, it produces an output similar to the outputs of **DB-SCAN**.

And so, this method may be of interest because of its customizability. Additionally, as it produces a dendrogram, it may be an attractive method if this form of representation is an outcome of interest.

This method's shortcoming is that the output may not be adjusted with the addition of new observations, meaning the output is exclusively applicable only to the data as is it is given during initiation, unlike the other methods which allow for dynamic insertion and updating the clustering outcome.

6.3 BIRCH

While **BIRCH** may have not produced any notably different clustering outcomes in the data, it is designed with efficiency in mind.

Its consistent time spent to complete the clustering process implies that indeed it is able to produce results in one scan of the data. **BIRCH** is a more desirable method if either system resources are limited (such as memory), or if the data is very large, because **BIRCH** only reads from the data-set once to facilitate its algorithm, and any following operations are applied on its constructed tree rather than the original data, so it is an efficient method.

6.4 DB-SCAN

DB-SCAN , with a conservative threshold, consistently discovers the shape of the distributions and clusters the data as such. However, due to the way the parameter selection is implemented, the cutoff point is relatively strict, resulting in a visually underwhelming output.

Overall, **DB-SCAN** excels at discovering the overarching shapes in the data and is not affected at all by outliers. This is a useful trait if clustering must be applied in object detection, with the support of machine learning algorithms, to classify the overall shape of an object as an entity.

6.5 GMM

GMM was the sole method able to identify overlapping clusters in some of the data-sets.

GMM's most interesting feature is that the designations of observations are 'soft' designations, where a given observation is only most likely to belong to one cluster, yet less likely for others. This is useful in, for example, image processing to classify images into groups based on the color profile of its content, where a user may choose to account for the two most likely designations.

7 Conclusion

The primary objectives of this project have been fulfilled with no issue. The mechanisms and outputs of each clustering method has been outlined and demonstrated on the various test distributions, with notable strengths and limitations included in the 'Discussion' section.

Throughout the course of the study, it has been noted that some scoring metrics are more suited to specific clustering methods than others, and as such the clustering outputs cannot be said to be definitively representative of the suitability of the clustering methods, but is rather dependant on the interpretation of the clustering output by the user.

Furthermore, real data is rarely two-dimensional. For visualization purposes, the data needs to be reduced to two or three dimensions, which could alter the performance of the clustering methods.

Going forward, it would be interesting to repeat this study for real, or high dimensional data, which could allow the methods to exhibit an affinity for specific types of data given the necessity of dimension reduction. It would also be desirable to perform an analysis on the parameter selection methodology to assess the information gain with labelled data.

8 References

[1]: David Arthur and Sergei Vassilvitskii, k-means++: The Advantages of Careful Seeding (2006). (PDF) The Advantages of Careful Seeding. Available here. [accessed Nov 2022]

[2]: Kaufman, L. and Rousseeuw, P.J, Agglomerative Nesting (Program AGNES) (1990) (PDF) Finding Groups in Data: An Introduction to Cluster Analysis Available here. [accessed Nov 2022]

[3]: Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, A Density-Based Algorithm for Discovering Clusters (1996). (PDF) A Density-Based Algorithm for Discovering Clusters – A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise Available here. [accessed Nov 2022]

[4]: Tian Zhang, Raghu Ramakrishnan, Maron Livny, BIRCH: An Efficient Data Clustering Method for Very Large Databases (1996). (PDF) BIRCH: An Efficient Data Clustering Method for Very Large Databases Available here. [accessed Nov 2022]

[5]: A. P. Dempster, N. M. Laird and D. B. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm(1977). (PDF)Maximum Likelihood from Incomplete Data via the EM Algorithm Available here. [accessed Nov 2022]

[6]: Pedregosa et al, 12(85):28252830, 2011, Scikit-learn: Machine Learning in Python (2011). (Python Module Documentation) Scikit-learn: Comparing different clustering algorithms on toy datasets. Available here. [accessed Nov 2022]

[7]: Peter J. Rousseeuw, Silhouettes: A graphical aid to the interpretation and validation of cluster analysis (1987). (PDF) A graphical aid to the interpretation and validation of cluster analysis. Available here. [accessed Nov 2022].

[8]: Schwarz, Gideon , Estimating the Dimension of a Model. The Annals of Statistics 6, no. 2(1978). (PDF) Estimating the Dimension of a Model. Available here. [accessed Nov 2022]

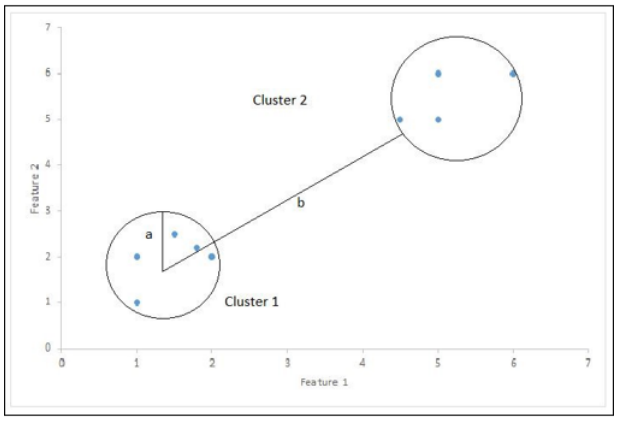
[9]: Ville Satopa, Jeannie Albrecht, David Irwin, and Barath Raghavan: Finding a “Kneedle” in a Haystack: Detecting Knee Points in System Behavior (2022). (PDF) Detecting Knee Points in System Behavior. Available here. [accessed Nov 2022]

A Appendix

A.1 Silhouette Score [7]

The Silhouette Coefficient is an internal cluster validation method proposed in 1987 that represents how well an algorithm has clustered the data points.

The Silhouette Coefficient for a given sample is equal to:-



$$\frac{b-a}{max(a,b)};$$

Where:-

- **a: Mean intra-cluster distance,**
- **b: Mean inter-cluster-distance.**

It is immediately recognizable that this metric rewards forming dense and tight clusters, and conversely penalizing sparse and spread clusters.

The value this metric can take is any number within the range of -1 to 1, where a greater value is better.

A.2 Bayesian Information Criterion [8]

Unlike the Silhouette Score, there is no set range of values that BIC must adhere to. However, the most fitting model is the model that possesses the lowest BIC value.

The BIC is given as:-

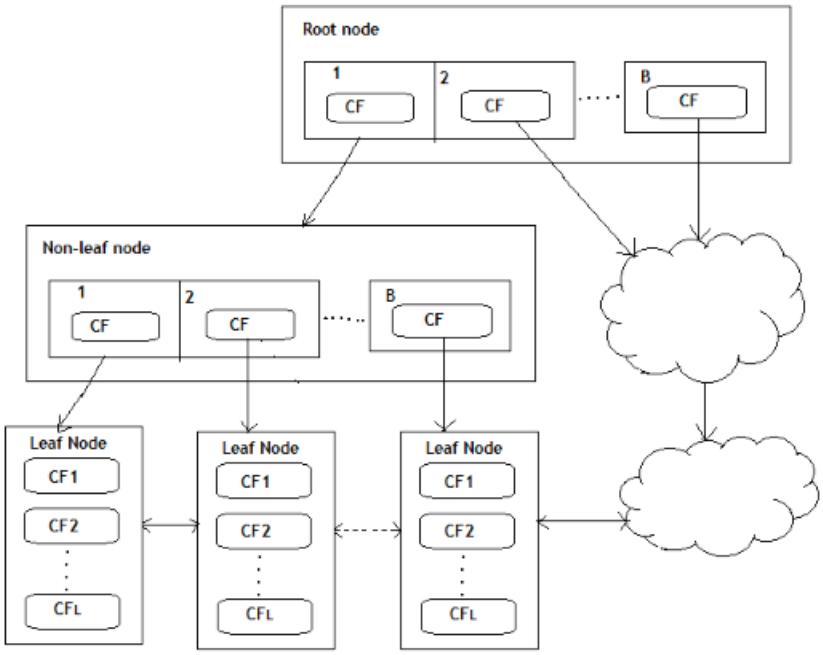
$$BIC = k * ln(n) - 2ln(\hat{L})$$

Where:-

- **k: Parameter quantity, which in the context of this project equates to the cluster quantity,**
- **n: Sample size, as in the quantity of observations in the data,**
- **: Maximized likelihood function value of the model.**

BIC awards superior goodness of fit of the model applied to the data, yet penalizes the use of more complex or otherwise denser data by increasing ‘n’ or over-fitting of the data by the increase of ‘k’.

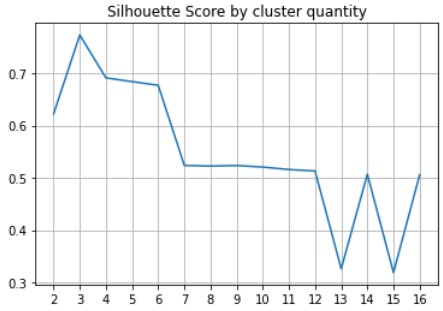
A.3 Clustering Feature Tree Schema



A.4 Parameter Estimation

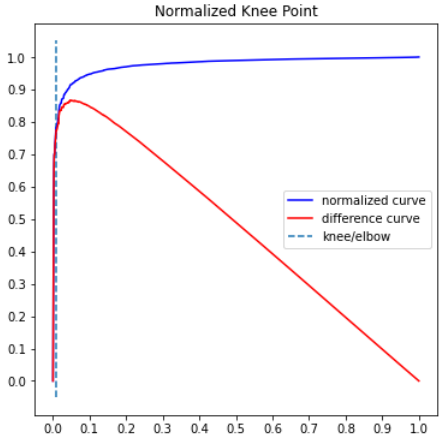
The following visualizations represent the output of the automatic parameter estimation functions as is applied on the 'Lax Blobs' data-set.

- **K-Means & BIRCH Cluster Quantity versus Silhouette Score**



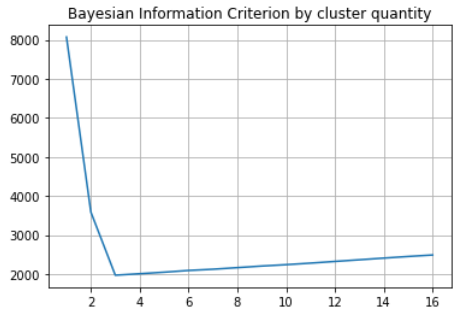
Select for highest Silhouette Score.

- **AGNES & DB-SCAN Dissimilarity/Epsilon versus Sample Size** [9]



Select for appropriate cut-off (knee) point.

- **GMM Cluster Quantity versus Bayesian Information Criterion**



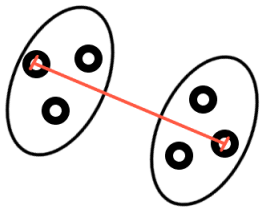

```
Optimal cluster quantity: 3
Corresponding to BIC value: 1968.4424973824628
```

Select for lowest BIC.

A.5 AGNES Linkage Functions

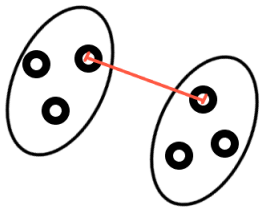
The output of this method differs based on the 'linkage' function selected. The linkage function is a function of dissimilarity between two sets of observations (inter-cluster) depending on the pairwise dissimilarity within the sets of observations (intra-cluster).

Commonly, there are four types of linkage functions:-



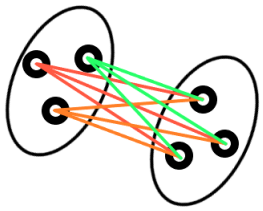
- **Complete:** The distance between two clusters is the maximum pairwise dissimilarity between observations in cluster A and cluster B.

$$max(d(a,b) : a \in A, b \in B)$$



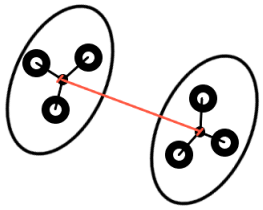
- **Single:** The distance between two clusters is the minimum pairwise dissimilarity between observations in cluster A and cluster B.

$$min(d(a,b) : a \in A, b \in B)$$



- **Average:** The distance between two clusters is the average of pairwise dissimilarities between observations in cluster A and cluster B.

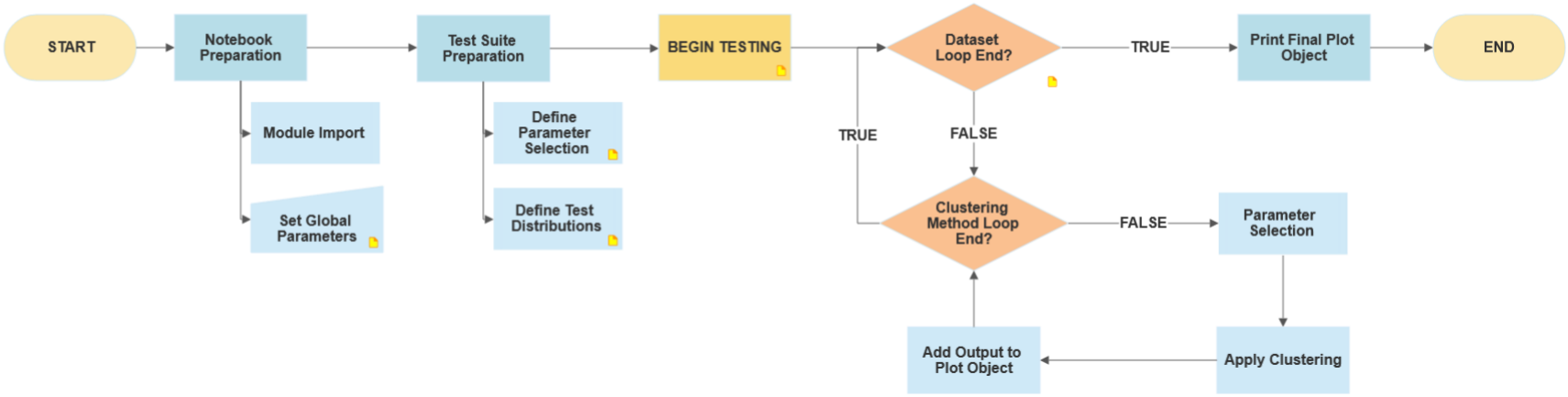
$$\frac{(\sum_{a \in A} \sum_{b \in B} d(a,b))}{\mid A \mid \times \mid B \mid}$$



- **Centroid:** The distance between two clusters is the distance between their respective centroids (The centroid is the intra-cluster average of distances (as given by the average linkage)).

$$d(C_a, C_b); C_x = \frac{\sum_{i=1}^{n} x_i}{n}$$

A.6 Program Code Flowchart



- **Set Global Parameters:**
  - Set random seed for consistency.
  - Set sample size of all distributions.
  - Set max cluster quantity to estimate parameters for (affects program run time).
  - Set **AGNES** and **DB-SCAN** sensitivity.
- **Define Parameter Selection:**
  - Cluster Quantity:-
    - **K-Means**
    - **GMM**
    - **BIRCH**
  - Dissimilarity Threshold:
    - **AGNES**
  - Epsilon:
    - **DB-SCAN**
- **BEGIN TESTING:** The testing is done in two nested for loops iterating on the data-sets and clustering methods with the following form:-
  - For Distribution:
    - For Method:
      - Invoke relevant parameter selection function
      - Apply clustering
      - Append output to plot object

A.7 Private Repository Link

Link to artefact repository

Instructions for use available in the repository, along with demo files for alternative inputs and outputs.

A.8 Required Software and Dependencies

Tested on a new system,

- Anaconda
- Jupyter Notebook (comes with Anaconda)
- KNEEDLE API Install by running 'pip install kneed' in the Anaconda prompt.