CS5691: Pattern Recognition and Machine Learning
Assignment #2

Topics: LDA, GMM, DBSCAN **Deadline:** 28 April 2023, 11:55 PM

Teammate 1: Kodari Saipranav Reddy (50% of contribution) **Roll number:** CS20B040
Teammate 2: M.D Chakradar (50% of contribution) **Roll number:** CS20B050

- **For any doubts regarding questions 1 and 2**, you can mail cs22s013@smail.iitm.ac.in and cs21s043@smail.iitm.ac.in
- **For any doubts regarding question 3**, you can mail cs21d015@smail.iitm.ac.in and cs22s015@smail.iitm.ac.in
- Please refer to the **Additional Resources** tab on the Course webpage for basic programming instructions.
- This assignment has to be completed in teams of 2. Collaborations outside the team are strictly prohibited.
- Any kind of plagiarism will be dealt with severely. These include copying text or code from any online sources. These will lead to disciplinary actions according to institute guidelines. Acknowledge any and every resource used.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- You should submit a zip file titled **'rollnumber1_rollnumber2.zip'** on Moodle where rollnumber1 and rollnumber2 are your institute roll numbers. Your assignment will **NOT** be graded if it does not contain all of the following:
    1. Type your solutions in the provided LaTeX template file and title this file as **'Report.pdf'**. **State your respective contributions in terms of percentage at the beginning of the report clearly.** Also, embed the result figures in your LaTeX solutions.
    2. Clearly name your source code for all the programs in **individual Google Colab files**. Please submit your code only as Google Colab file (.ipynb format). Also, embed the result figures in your Colab code files.
- We highly recommend using `Python 3.6+` and standard libraries like `NumPy, Matplotlib, Pandas, Seaborn`. Please use `Python 3.6+` as the only standard programming language to code your assignments. Please note: the TAs will only be able to assist you with doubts related to Python.
- You are expected to code all algorithms from scratch. **You cannot use standard inbuilt libraries for algorithms until and unless asked explicitly**.
- **Any graph that you plot is unacceptable for grading unless it labels the x-axis and y-axis clearly.**

- Please note that the TAs will **only** clarify doubts regarding problem statements. The TAs won't discuss any prospective solution or verify your solution or give hints.
- Please refer to the CS5691 PRML course handout for the late penalty instruction guidelines.

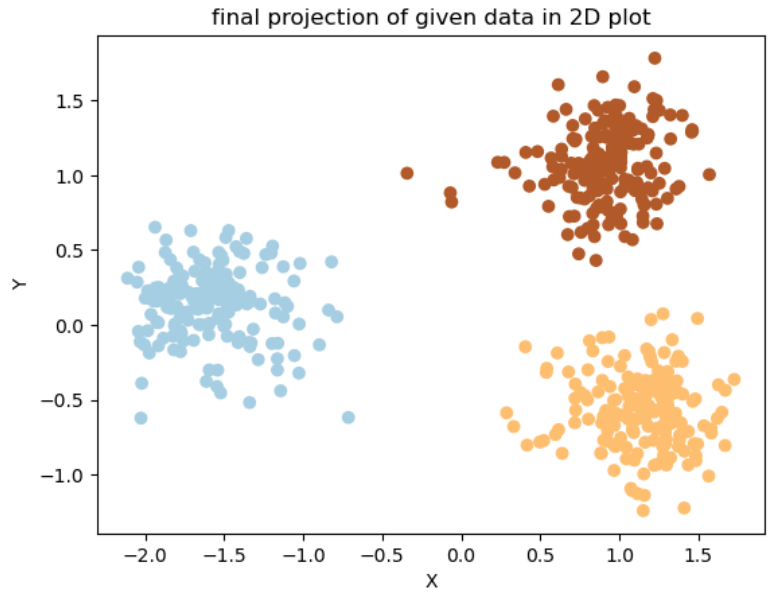
---

1. **[Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA)**] You will implement dimensionality reduction techniques (LDA, PCA) as part of this question for the dataset1 provided here.

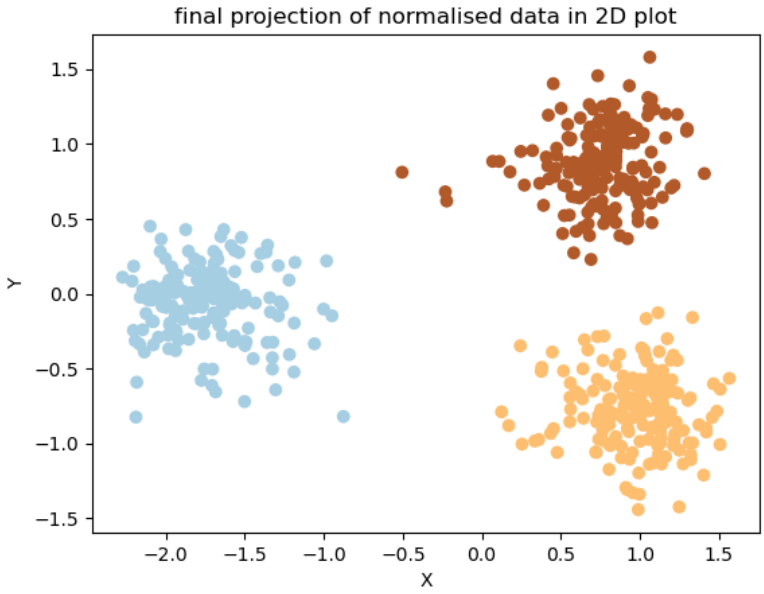
   Note that you have to implement **LDA from scratch** without using any predefined libraries (i.e. sklearn, scipy) . However, you can use **predefined libraries to implement PCA.**

   (a) (2 marks) Use Linear Discriminant analysis (LDA) to convert dataset1 into the two-dimensional dataset and then visualize the obtained dataset. Also, perform an analysis on how results will change if we perform normalization (i.e., zero mean, unit variance normalization) on the initial dataset before applying LDA.

Solution:
We have converted the dataset1 with 64 features into 2 features using LDA.
The plot of the points after performing LDA is given below.



Observations:
We can see the three clusters in the plot since there are 3 clusters.X, Y are the reduced data features using LDA computed with first 2 greatest eigen values.
And we have done the same after normalising the given dataset , i.e., subtracting mean from every x in dataset by dividing with standard deviation. We got the following plot.

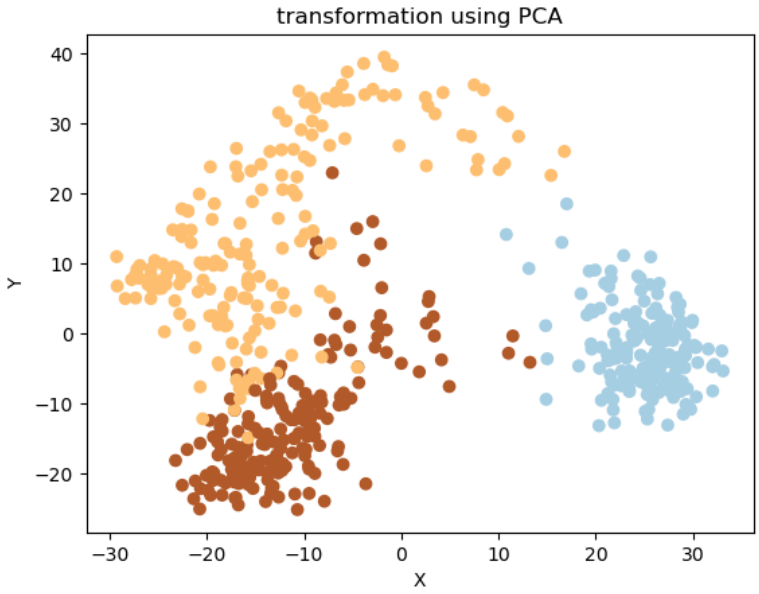


Normalisation is a common pre-processing step, this is just to improve the performance of any model.If we do normalisation before doing LDA the results might change like better seperation of classes,and performance of algorithm.

(b) (1.5 marks) Use PCA to convert dataset1 into two-dimensional data and then visualize the obtained dataset. Now, compare and contrast the visualizations of the final datasets obtained using LDA and PCA.

Solution:

Using PCA the visualisation of obtained dataset is given below.

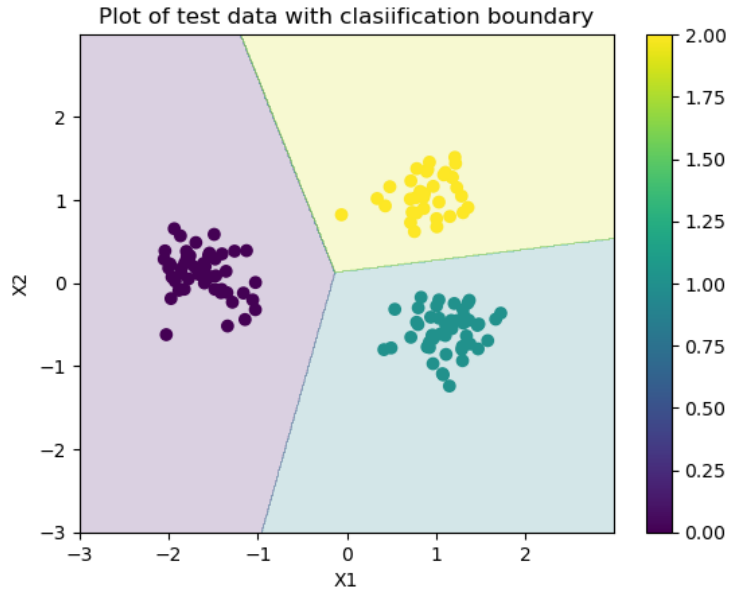


The plots look completlely different even though for same dataset.Because PCA tries to capture overall variance but LDA tries to maximize separabilty between the classes.PCA is likely to see overall structure of data whereas LDA sees class specific structure.It is the reason why we have clear clusturing in LDA than in PCA.

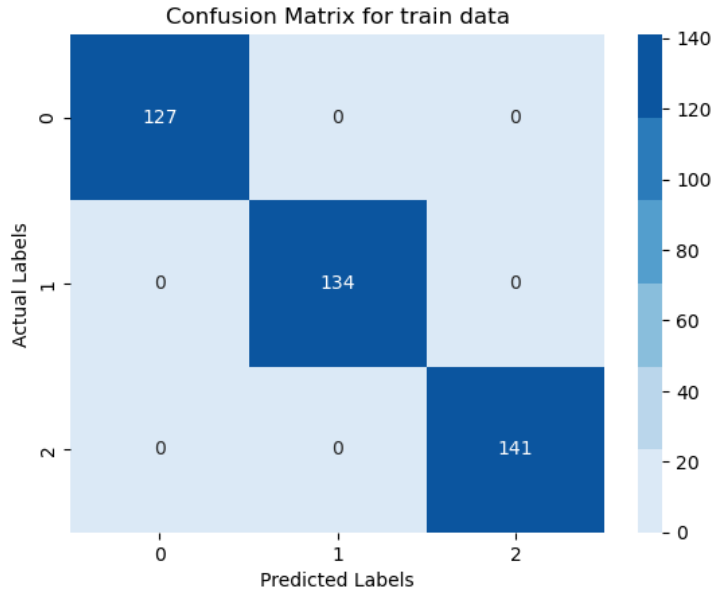
(c) (1.5 marks) Randomly shuffle and split the obtained dataset from part (a) into a training set (80%) and testing set (20%). Now build the Bayes classifier using the training set and report the following:

- Accuracy on both train and test data.
- Plot of the test data along with your classification boundary.
- confusion matrices on both train and test data.

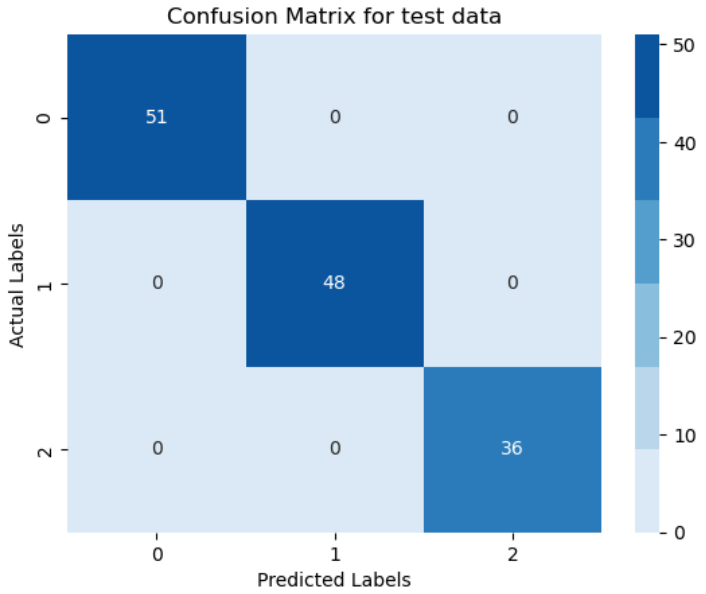
Solution:
Accuracy on traindata is 1.0
Accuracy on testdata is 1.0
Plot of test data along with classification boundary



Confusion matrices on train data



Confusion matrix on test data

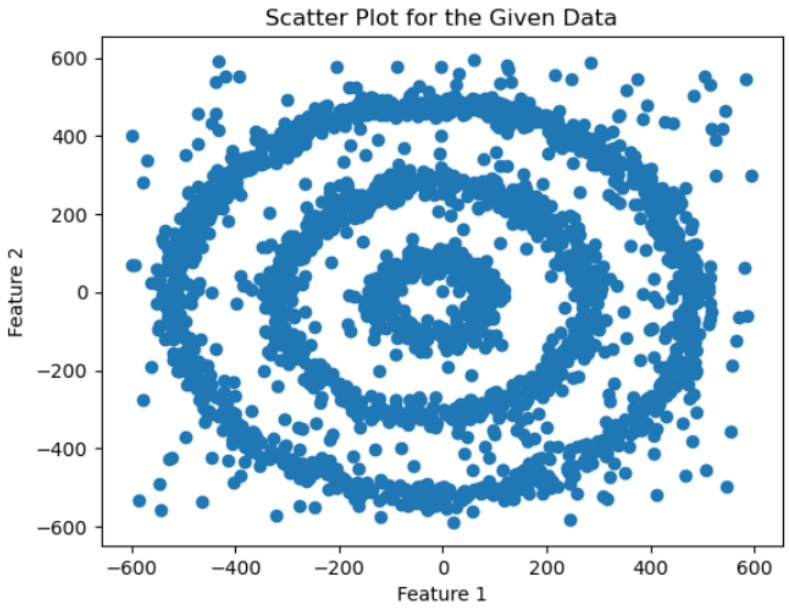


2. **[DBSCAN]** In this Question, you are supposed to implement **DBSCAN algorithm from scratch** on dataset2 provided here and dataset3 provided here. You also need to compare and contrast your observations from above with K-Means applied on both datasets. **However, you can use predefined libraries to implement K-means.**

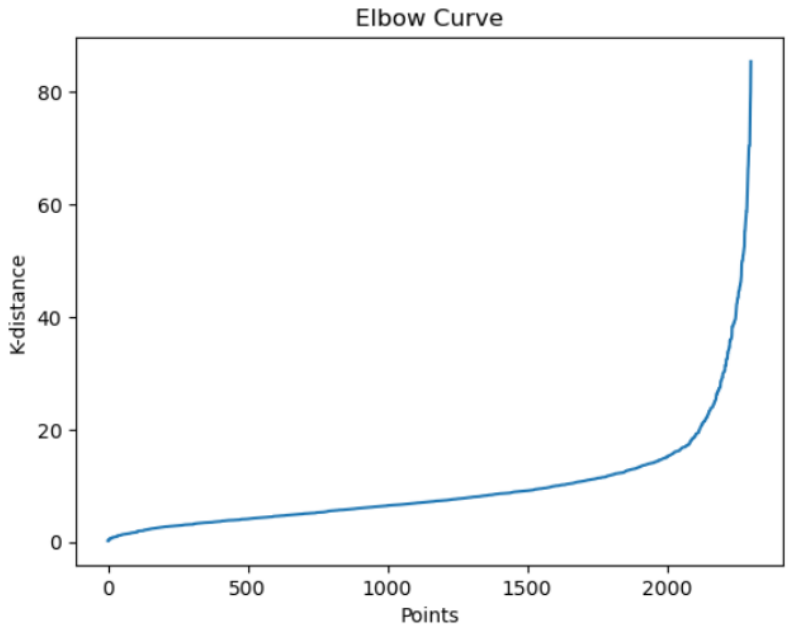
   (a) (1 mark) Visualize the data in dataset2. Then, find a suitable **range of values for epsilon** (a hyperparameter in DBSCAN algorithm) by using the 'Elbow Curve' of Datapoints plotted between K-Distance vs Epsilon. For simplicity, take only integer values for epsilon. **You can use predefined libraries to implement K-distance.**

   Solution:

   Visualisation of Dataset 2 is as follows:



For finding the suitable range for the values of epsilon we plot the 'Elbow Curve'. It is as follows:



From the above graph, we can infer that the suitable range of values for Epsilon is 20 to 40.

(b) (2 marks) Implement DBSCAN with the above suitable range of values of epsilon and detect the optimal value of epsilon, which gives the best clustering visually on the dataset.

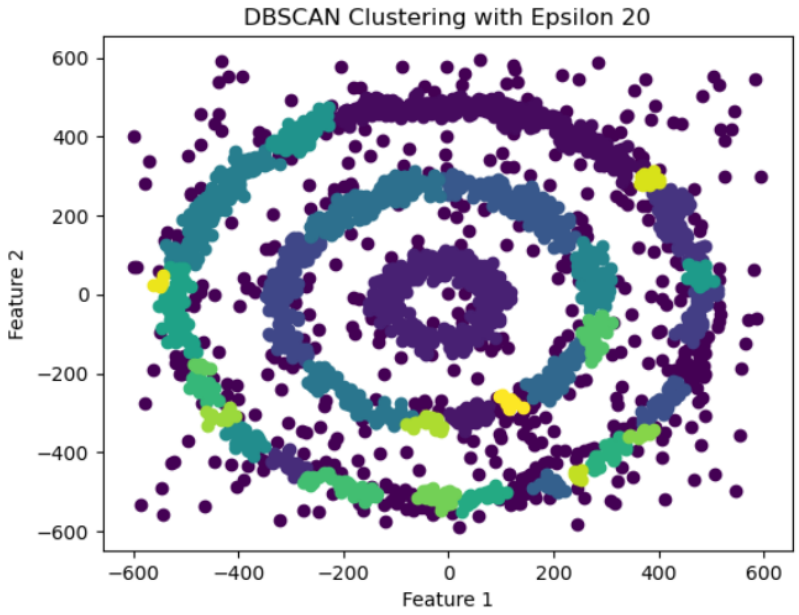
Show a visualization of the clusters formed for the best value of epsilon.

Solution:

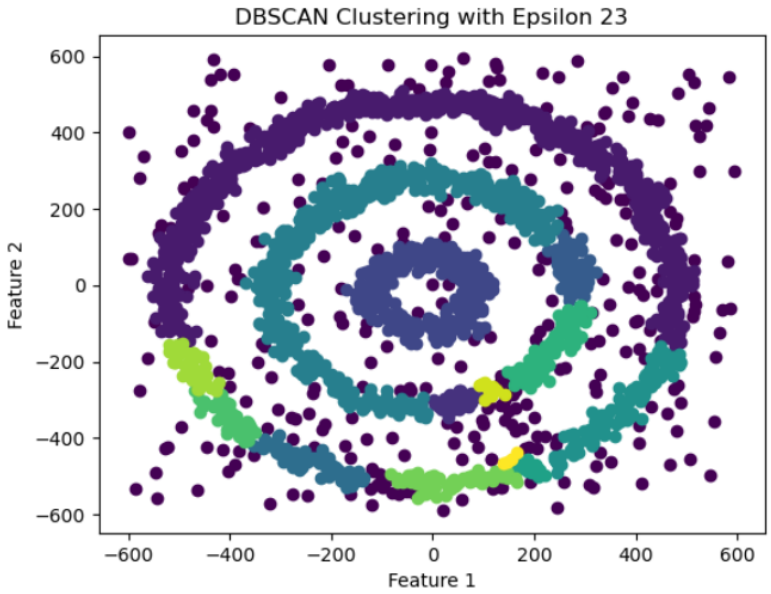
We implement the DBSCAN in the above obtained optimal range of Epsilon and with Minpts ranging from 4 to 10

So, we choose some values from the epsilon range, with Minpts=5 and plot the visualizations.

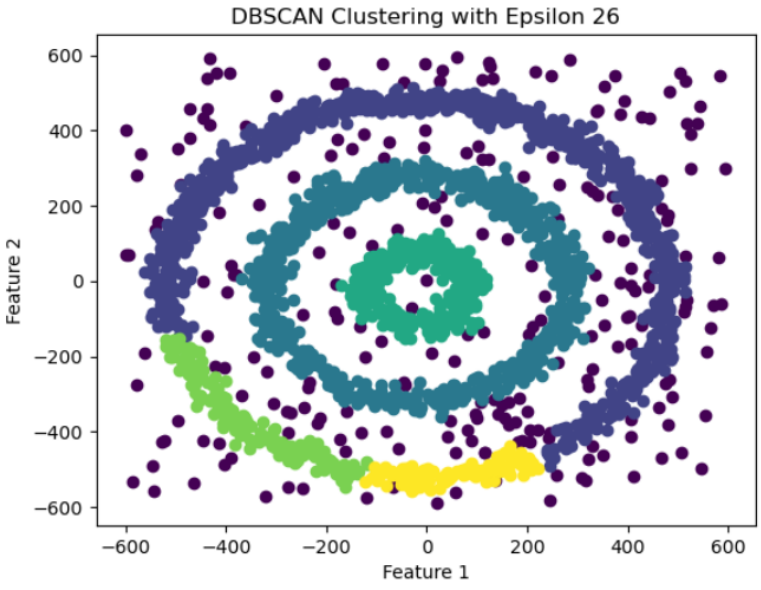
Eps = 20 , Minpts = 5



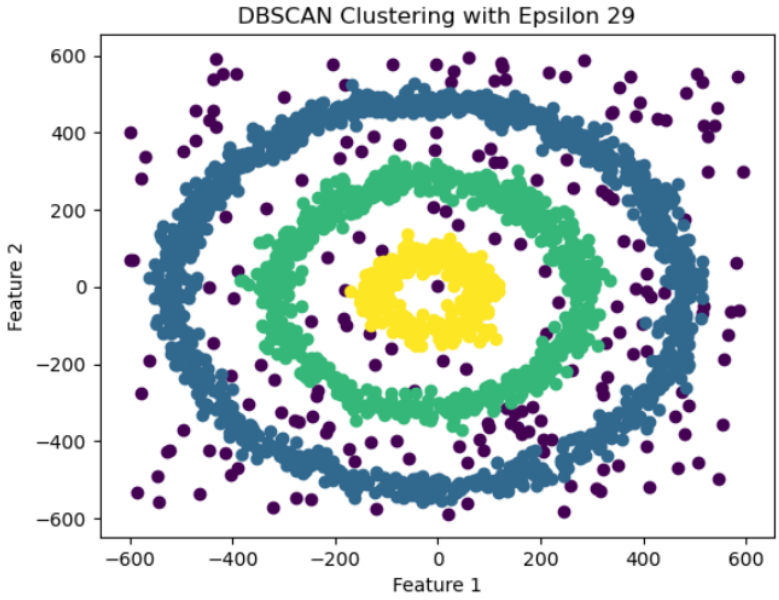
Eps = 23 , Minpts = 5



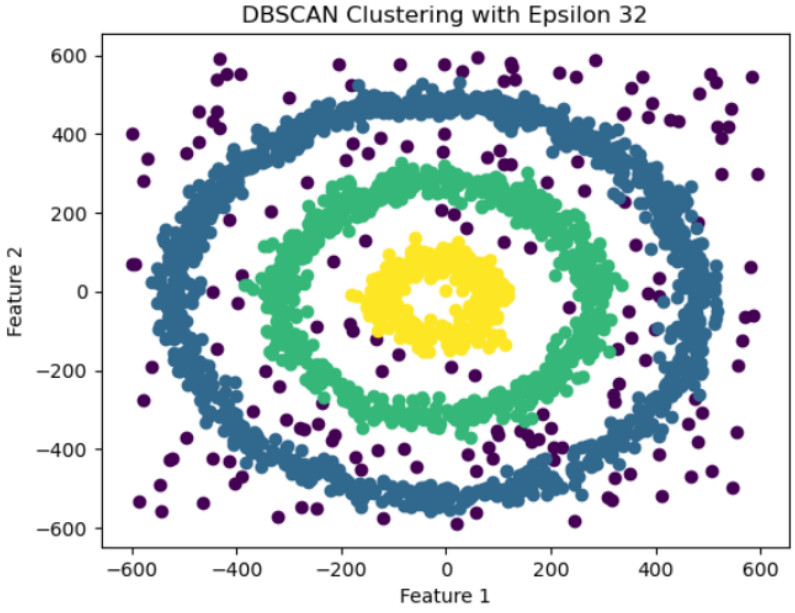
Eps = 26 , Minpts = 5



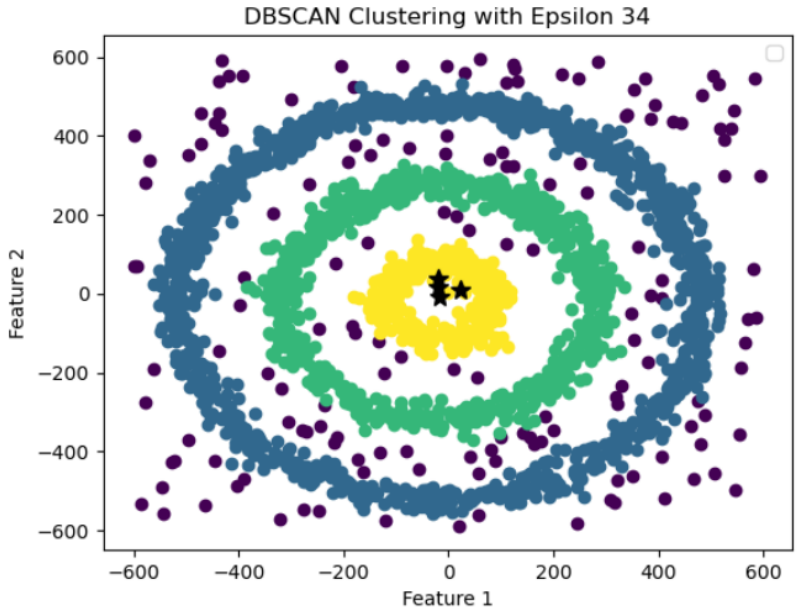
Eps = 29 , Minpts = 5



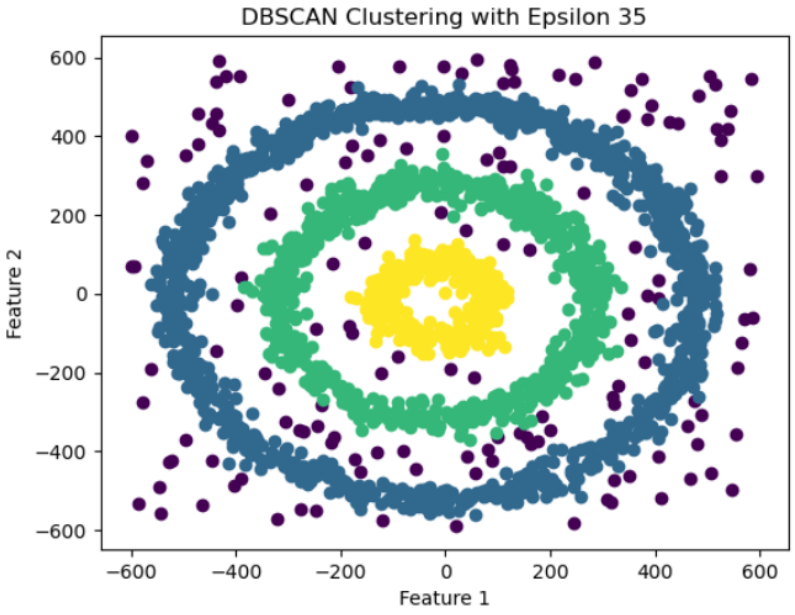
Eps = 32 , Minpts = 5



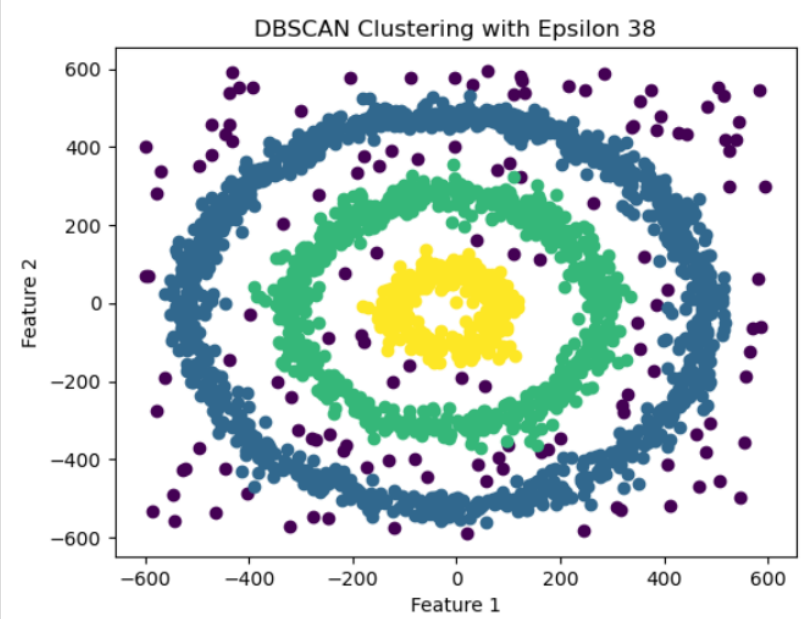
Eps = 34 , Minpts = 5



Eps = 35 , Minpts = 5



Eps = 38 , Minpts = 5



So from the above visualizations we can infer that Epsilon = 35 is the optimal Epsilon Value.

At this epsilon all the labeled points in the visualizations are clustered and the remaining remain as the noise points.

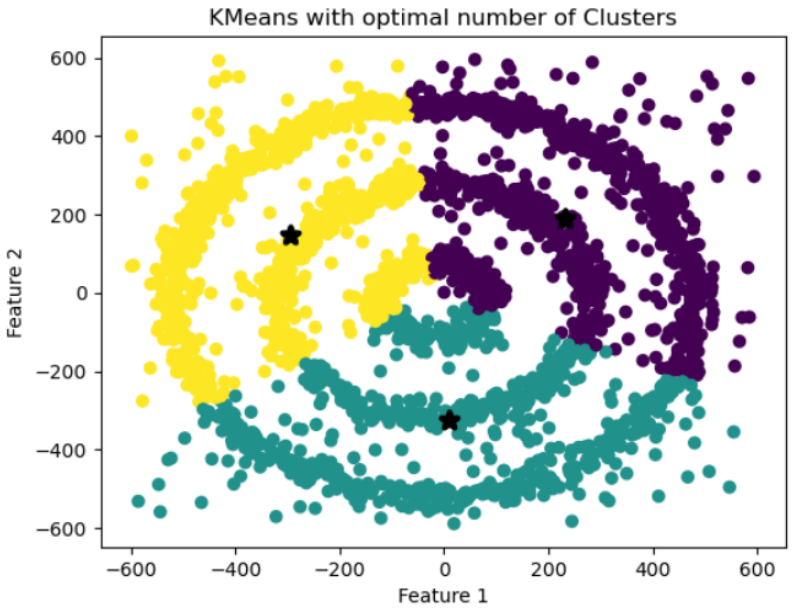
So the optimal value of epsilon which gives the best clustering is Epsilon = 35

(c) (1.5 marks) Implement K-Means and use it on dataset2 with value of K (number of clusters) set to the optimum number of clusters that you get from (b) above. Suggest various techniques to improve the clustering by KMeans in this case.

Solution:

The optimum number of clusters obtained from the above part are 3.

Now,the graph obtained by applying K-Means algorithm(K=3) on dataset 2 is as follows:



Techniques to improve KMeans :

1) Usage of a different distance Metric : The Euclidean distance Metric used in KMeans may be the best choice for the concentric circle dataset. Instead exploring different distance metrics such as cosine distance or Manhattan distance might be useful.

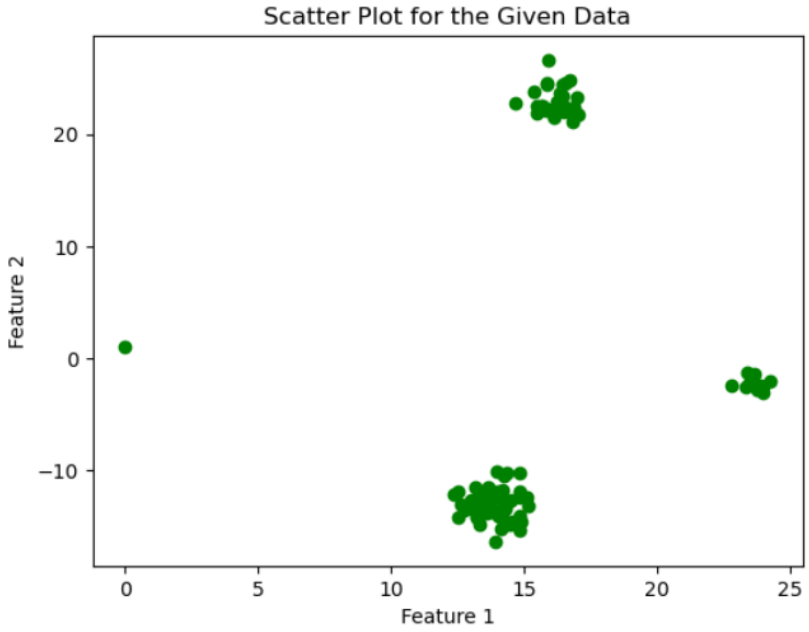
2) Coordinates : In this case as the clusters are spherical and isotropic, usage of polar coordinates could be better rather than using raw x,y coordinates. It could be beneficial to transform the data into polar coordinates, with radius and angle. It could reduce the effect of circular shape of the dataset.

3) We could create new features which would capture the circular nature of the data better and could reduce the complexity and could be useful while computation. For example, let us say we can use a feature that represents the distances of each points from the centres of the clusters.

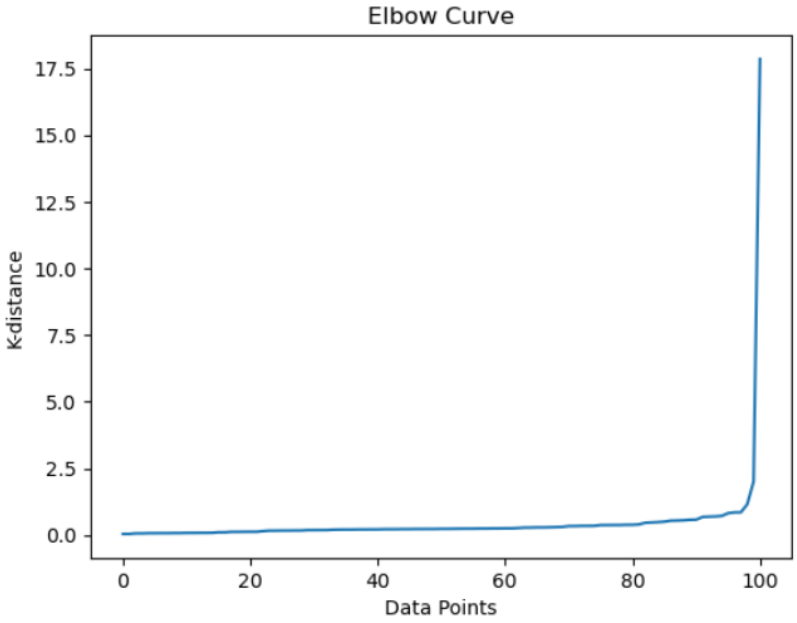
(d) (1.5 marks) Show a visualization of the data in dataset3. Use your implementation of DBSCAN with `minPts=15` on dataset3. Plot 'Elbow curve' to get an optimal range of values for `eps`. Detect the optimal value of epsilon which gives the best clustering visually on the dataset. Show a visualization of the clusters formed for the best value of epsilon.

Solution:

Visualisation of Dataset3 is as follows:



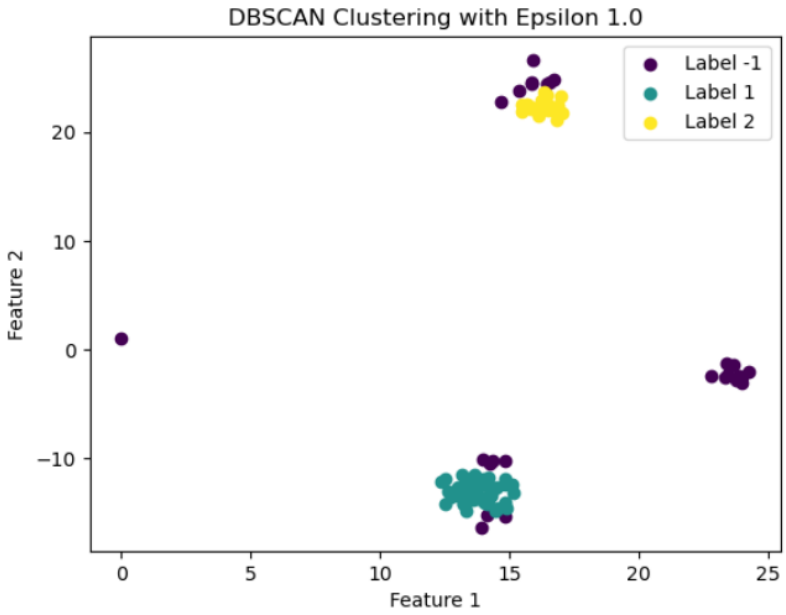
For finding the suitable range for the values of epsilon we plot the 'Elbow Curve'. It is as follows:



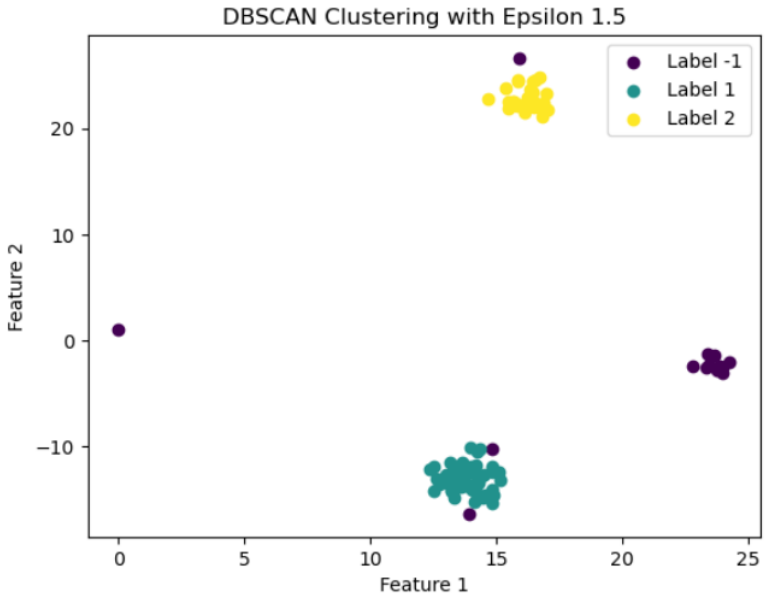
As we can see from the curve, the optimal range of Epsilon is 1 to 3.

So to detect the optimal values we can plot the results obtained from DBSCAN using these epsilon values :

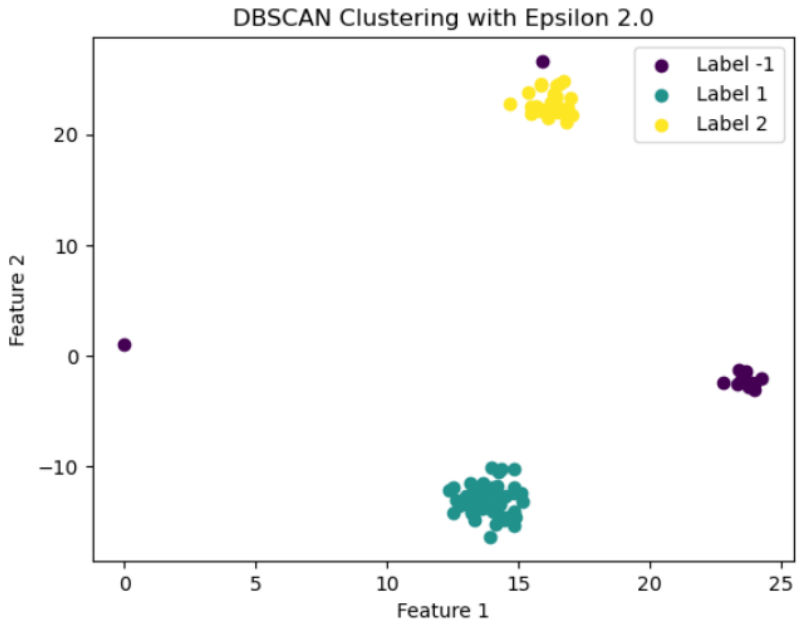
Eps : 1 , Minpts : 15:



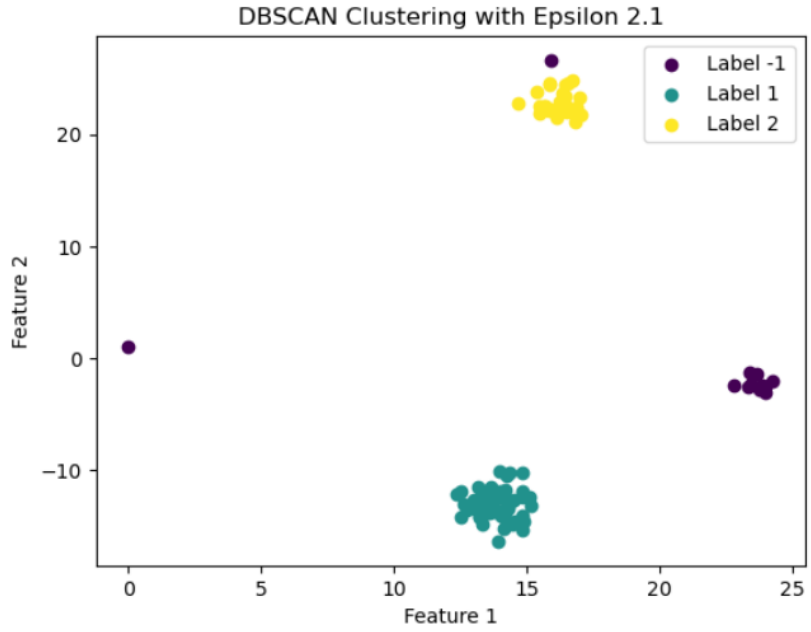
Eps : 1.5 , Minpts : 15:



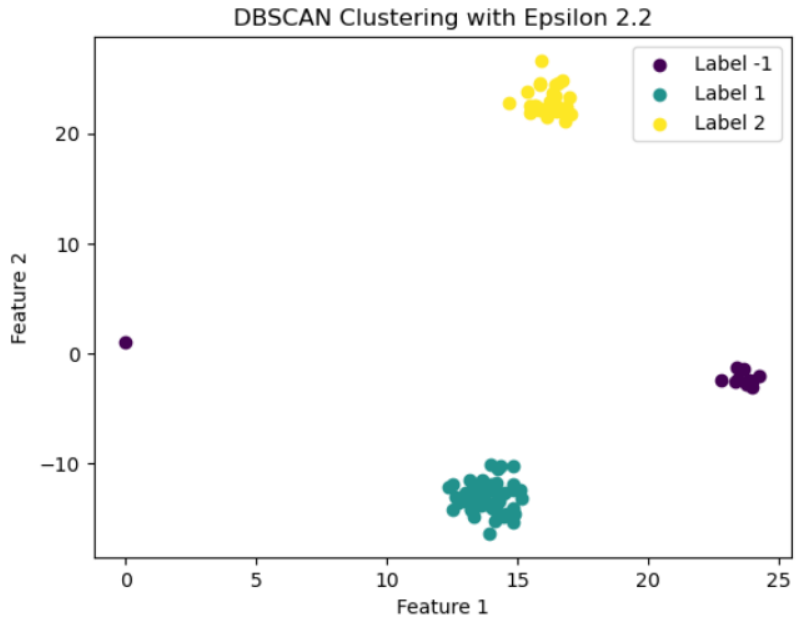
Eps : 2 , Minpts : 15:



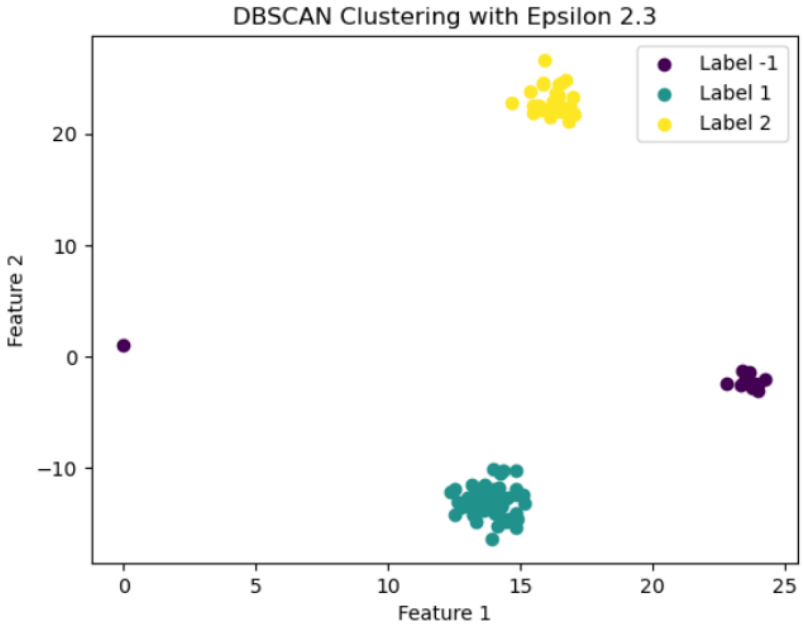
Eps : 2.1 , Minpts : 15:



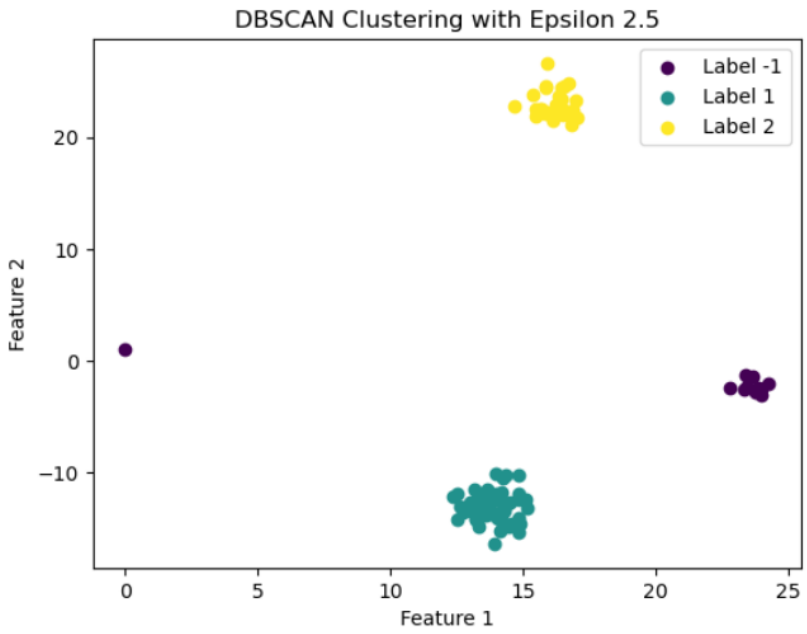
Eps : 2.2 , Minpts : 15:



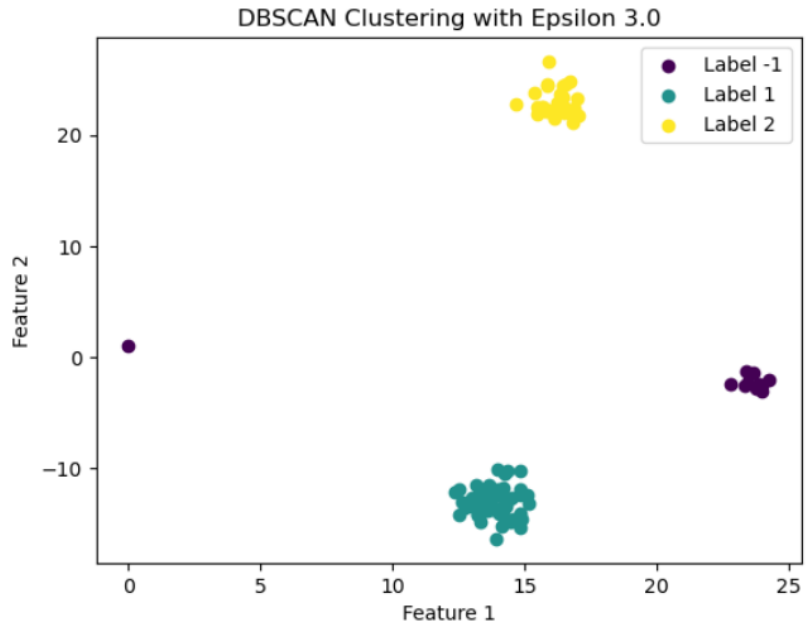
Eps : 2.3 , Minpts : 15:



Eps : 2.5 , Minpts : 15:



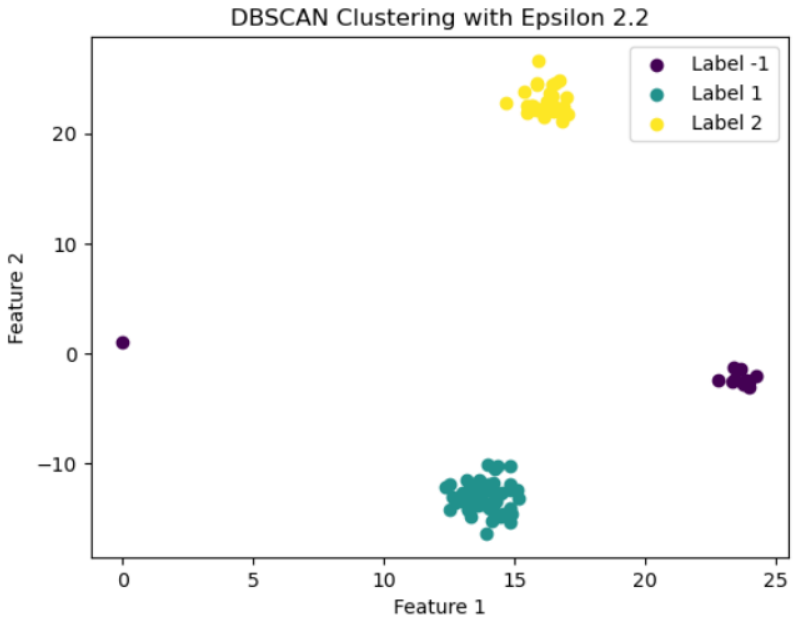
Eps : 3 , Minpts : 15:



As we can infer from the above graphs that the optimal value of epsilon is 2.2 when Minpts = 15

So, the visualisation of the clusters for the best value of the Epsilon is :

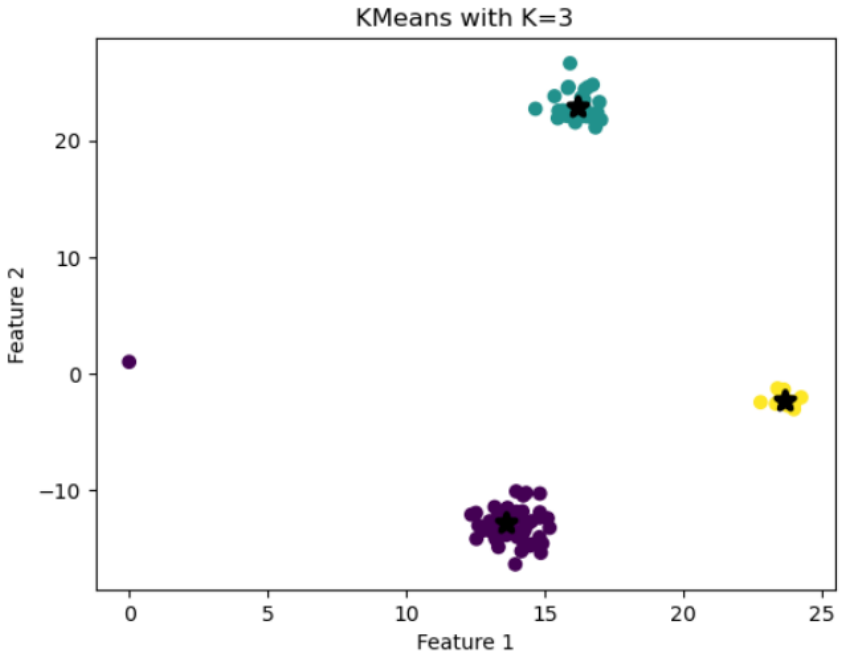
Eps : 2.2 , Minpts : 15:



(e) (1 mark) Now perform KMeans with K=3. Write your observations for obtained results in (d) and (e). Did we give you bad initialization values?

Solution:

KMeans Clustering Visualisation with K=3 for dataset3 :



Observations :

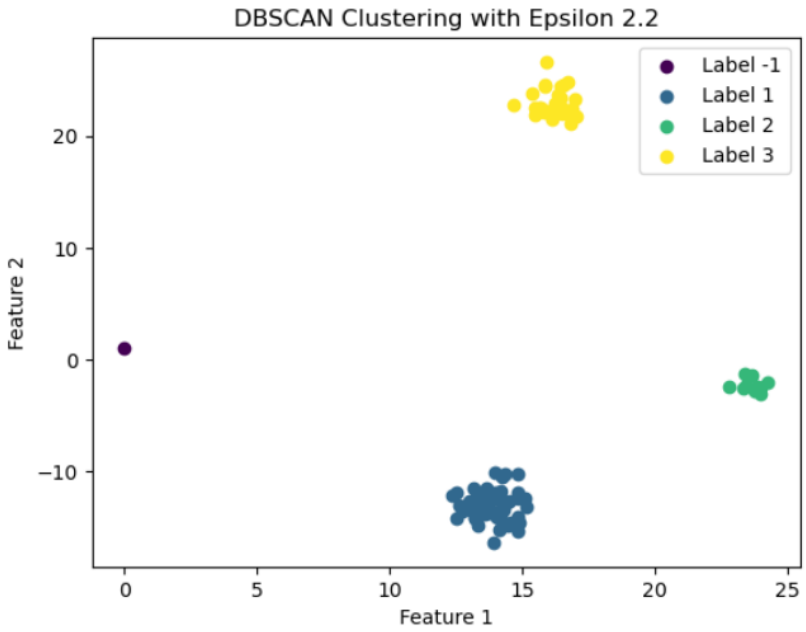
1) KMeans gives us better clustering in this case than the DBSCAN with the above initialisation, because in dataset 3 the right most cluster formed in the scatter plot is considered as noise by DBSCAN.

2) Unlike DBSCAN, KMeans couldn't fine the noise points and joined them in the nearest cluster.

3) When we had the concentric circular dataset2, KMeans failed in that case and didn't give the desired result but in this case for dataset3, KMeans has given better results than DBSCAN under the given conditions.

The initialization given is bad for dbscan, i.e Minpts = 15 is bad, because even the dense cluster to the right of the plot is considered as noise, as it doesn't have 15 number of points in it. Suppose if we reduce the minpts to 9 and see what happens:

Eps : 2.2 , Minpts : 9:



So as we can see the right most group of points is detected as a cluster and not as noise.
The given initialization values are bad.

(f) (1 mark) Based on all your learnings from this question, state the relative pros and cons of KMeans vs DBSCAN.

Solution:

KMeans

Pros :

1) KMeans is easy to implement and interpret. It is a based on the principle of minimizing the sum of squared distances between each point and its assigned centroid, making it an 'easy to go with algorithm for clustering tasks.

2) KMeans is computationally efficient and is suitable for large datasets as well.

3) KMeans works well with higher dimensional data.

4) KMeans performs best when the clusters are clearly seperated and distinct from each other.

Cons :

1) One of the major drawback for KMeans is that the number of clusters must be specified in advance, this could be difficult especially when dealing with complex datasets.

2) KMeans is sensitive to the initializations(for example: initial placement of centroids). The results could be different when the algorithm multiple times with different initializations.

3) KMeans is sensitive to noise, which can result in inaccurate clustering.

4) KMeans cannot handle clusters that have irregular shapes and sizes.

DBSCAN

Pros :

1) DBSCAN can handle arbitrary shapes and sizes of clusters making it more flexible than KMeans

2) DBSCAN does not require the number of clusters to be specified beforehand.

3) DBSCAN can deal with outliers and noise, so it can be more useful than kmeans in the case of noise datasets.

4) DBSCAN can perform well on large datasets.

Cons :

1) DBSCAN may not work well for datasets with varying densities.

2) DBSCAN can be computationally expensive and may require the use of approximation methods for large datasets.

3) The performance of the algorithm is sensitive to the choice of distance metric and parameters, which can make it difficult in choosing the optimal values for these parameters.

4) DBSCAN can struggle with higher dimensional data, as distance metric becomes less meaningful in higher dimensional spaces and the data becomes more sparse.

3. **[GMM]** In this question, you are supposed to implement the Expectation-Maximization algorithm for Gaussian mixture models on the given dataset4. The data can be found here.

   (a) (3 marks) Implement EM for GMM and plot the log-likelihood as a function of iterations.

Solution: In the case of GMM, EM algorithm is used to estimate the parameters of the model, such as the means, variances, and mixture coefficients of the Gaussian components. The algorithm has two main steps: the E-step and the M-step.

In the E-step, the algorithm computes the posterior probabilities (responsibiltites) of the latent variables, i.e., the probabilities of each data point belonging to each Gaussian component, given the current estimates of the model parameters.

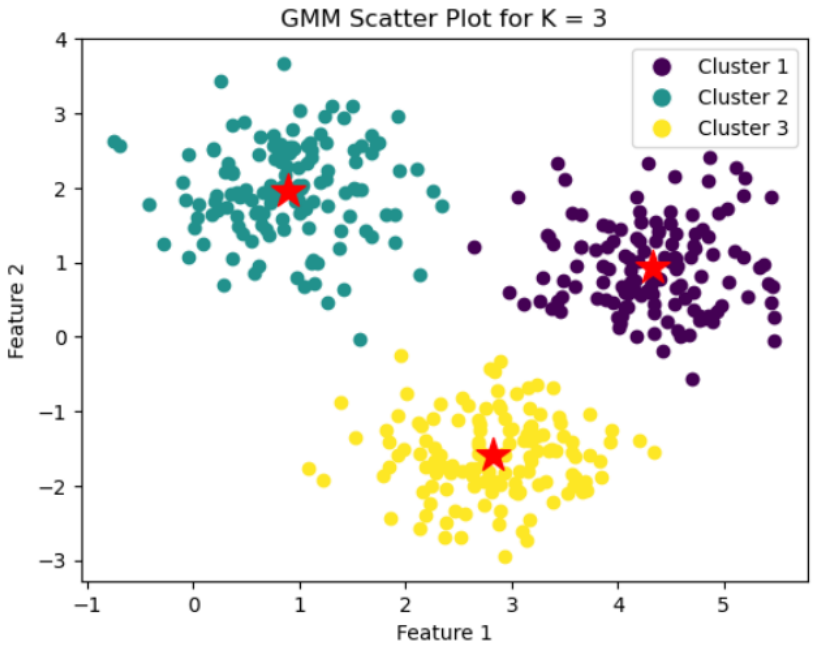
In the M-step, the algorithm updates the estimates of the model parameters based on the responsibilities computed in the E-step.

The algorithm iterates between the E-step and M-step until convergence, i.e., until the change in the log-likelihood of the data between successive iterations falls below a certain threshold.

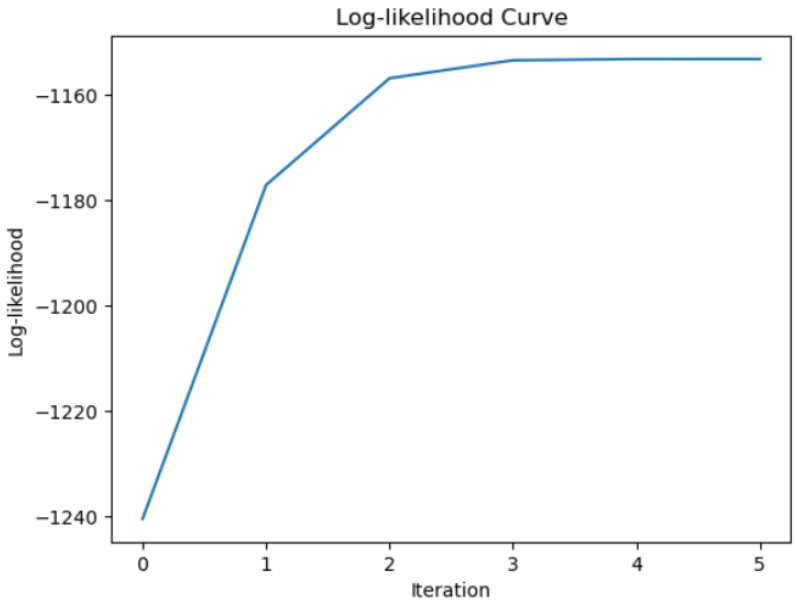
EM function : First we initialise our model parameters and then perform the Expectation step on those model parameters following by the Maximization Step, where we will update the parameters. Then, we compute the log-likelihood function.

Now, for K-value = 3, Let us plot the GMM scatter and the log-likelihood as the function of iterations :

The GMM Scatter plot for K = 3:



The log-likelihood vs iterations plot for K = 3:



(b) (2 marks) Run EM for different numbers of Gaussians (k)(Try 2,3,4,5,6). Plot figures that can help in visualization and also log likelihood as a function of iteration for different values of k. Report the observations.

Solution:

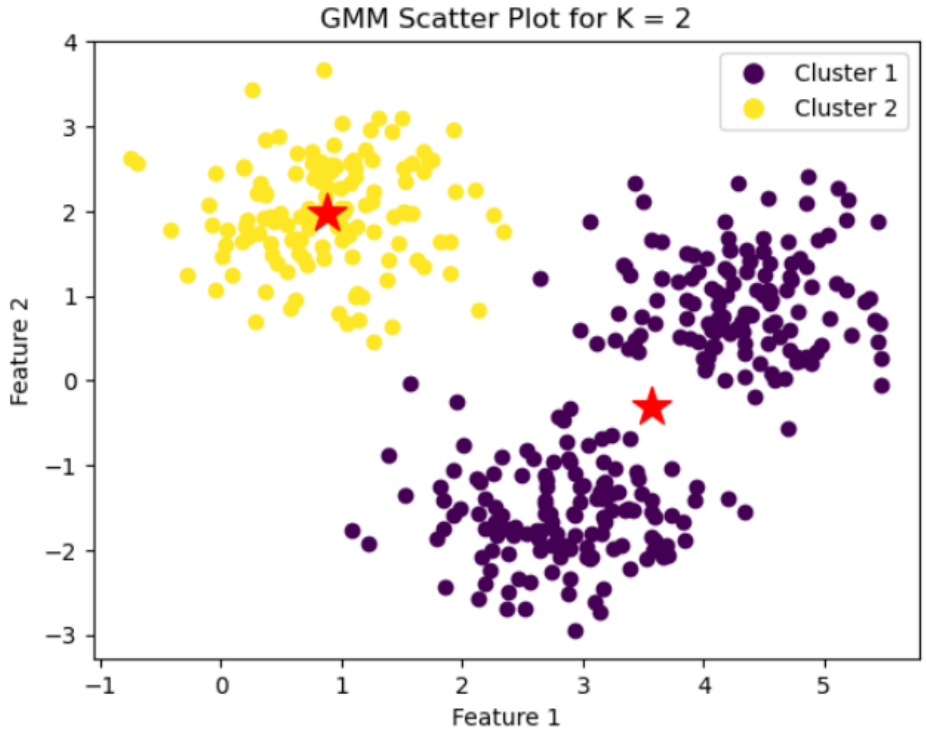
We will run the EM algorithm for different K-values given below:

K = [2,3,4,5,6]

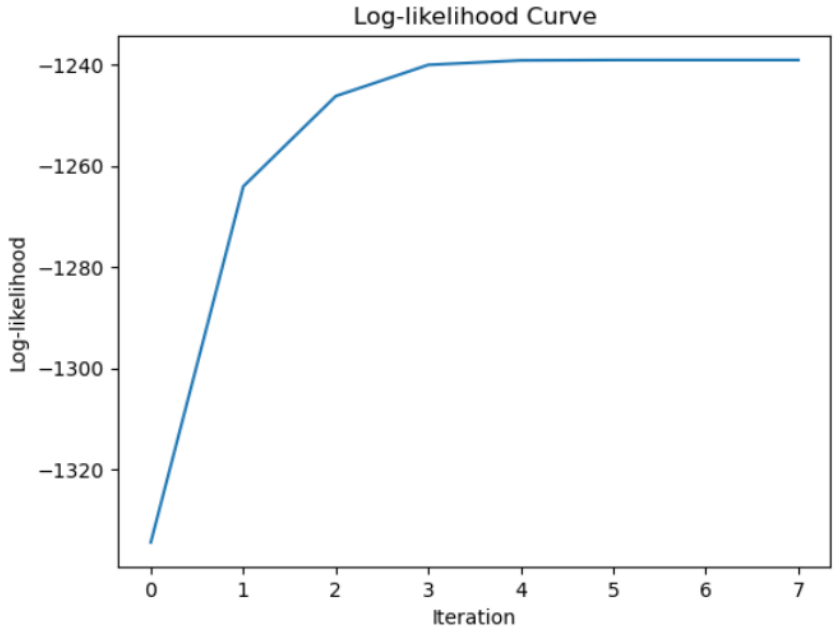
Now, we will plot the Scatter plots and the log-likelihood vs iteration plots from the data which we have obtained upon running the algorithm.

K = 2 :

Scatter Plot :

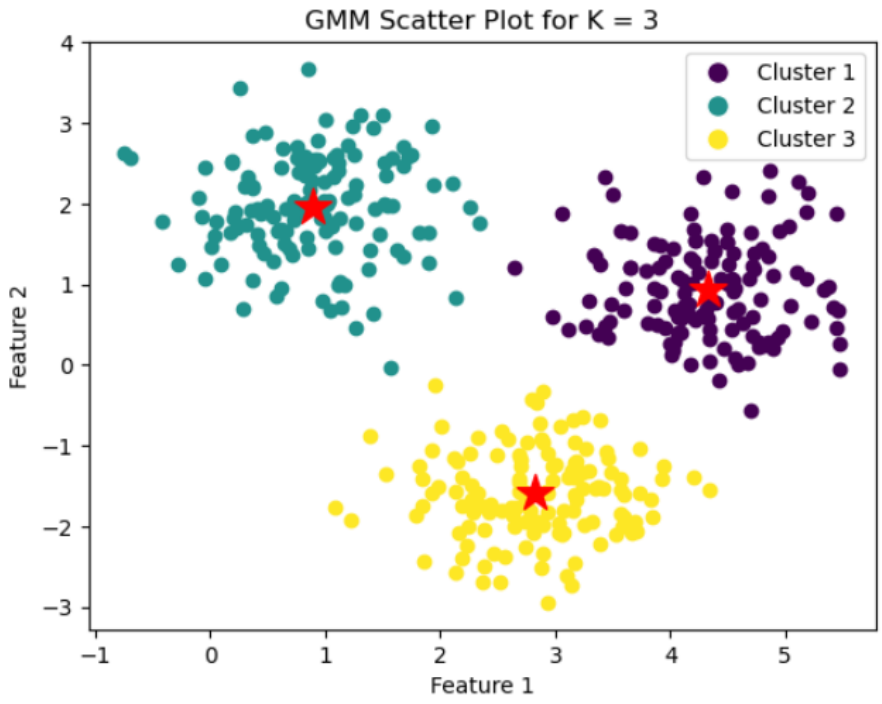


Log-likelihood vs iterations Plot :

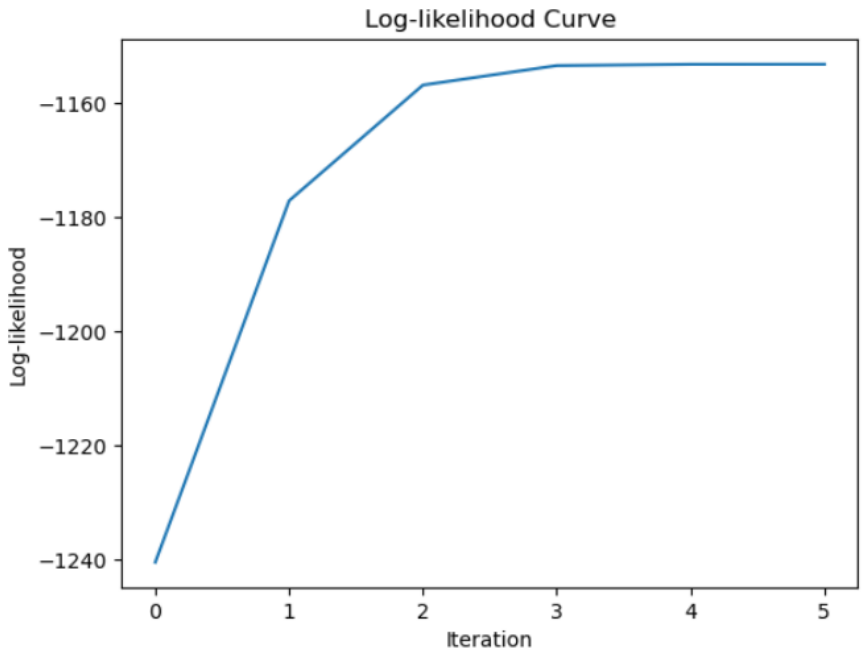


K = 3 :

Scatter Plot :

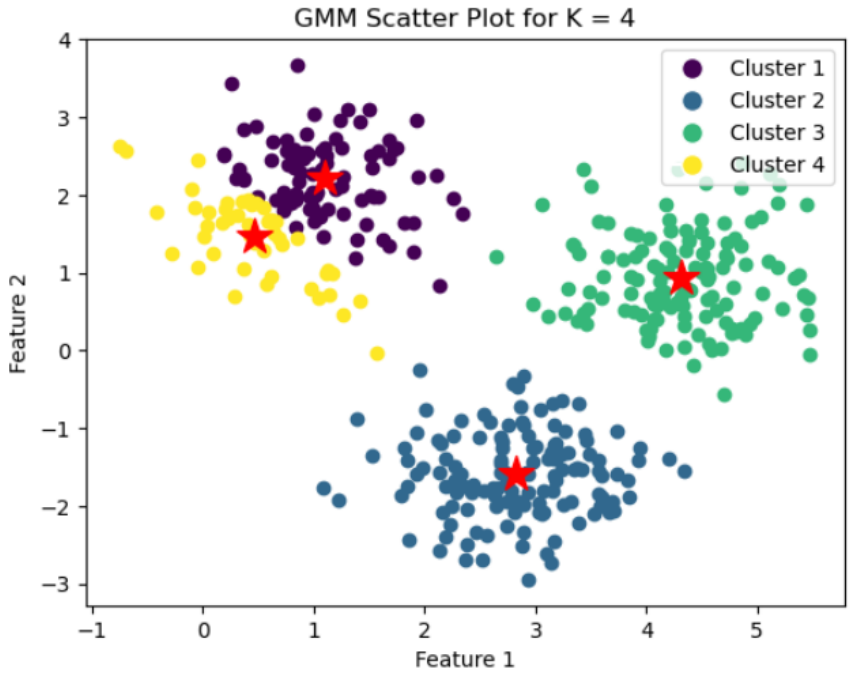


Log-likelihood vs iterations Plot :

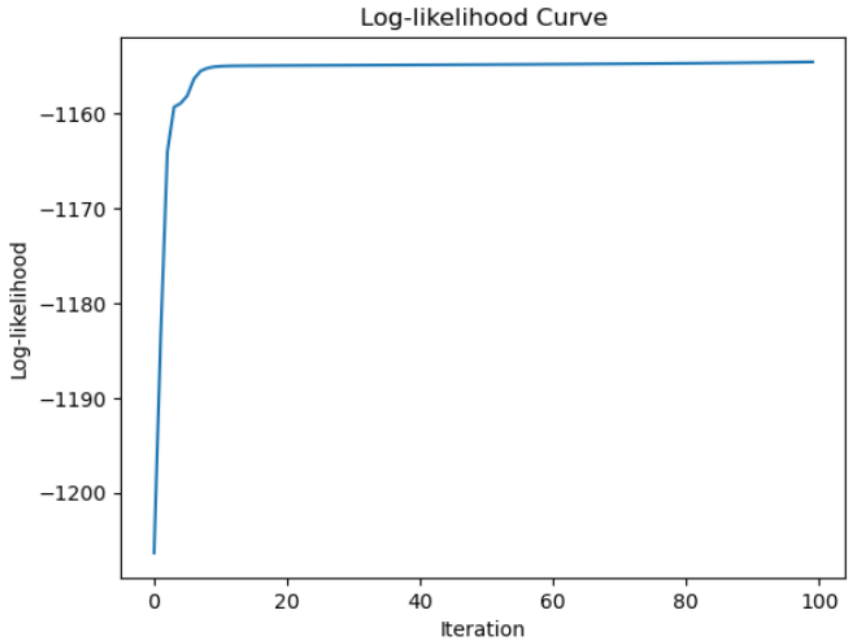


K = 4 :

Scatter Plot :

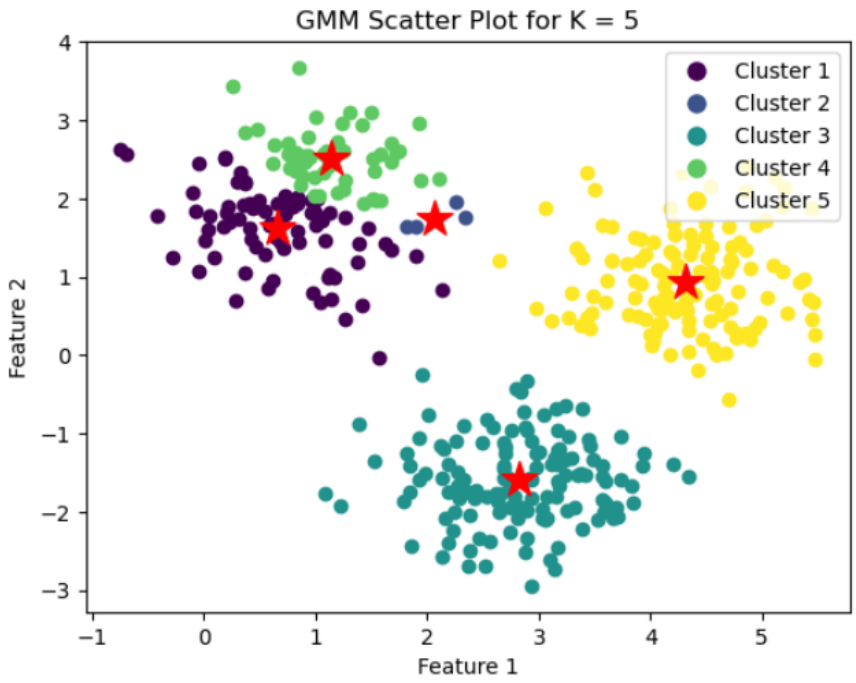


Log-likelihood vs iterations Plot :

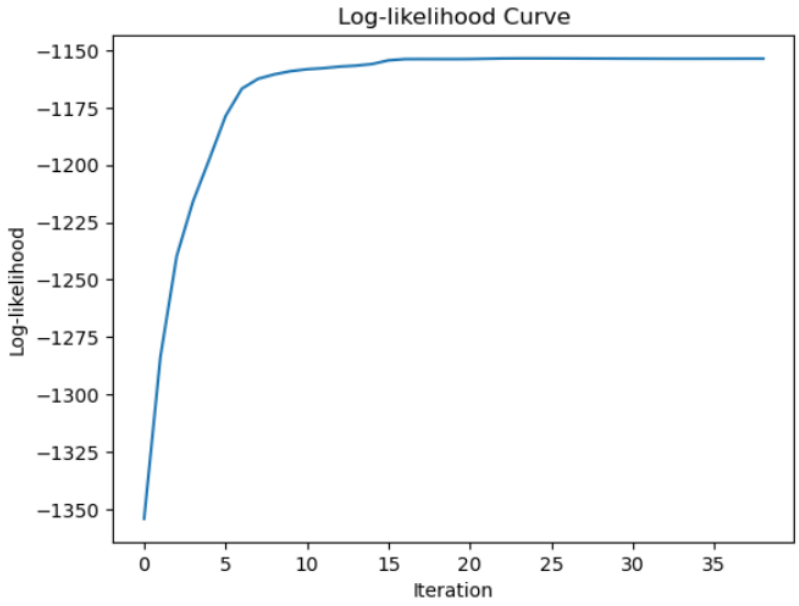


K = 5 :

Scatter Plot :

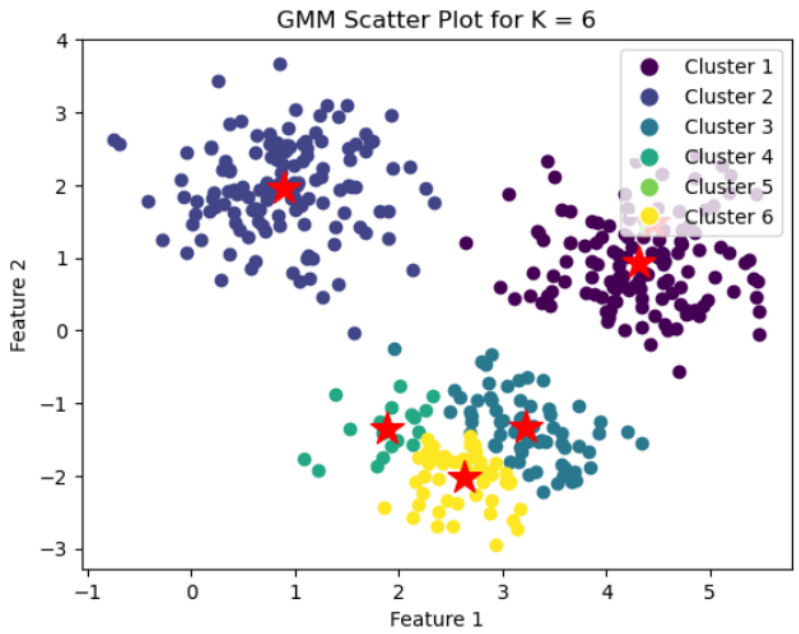


Log-likelihood vs iterations Plot :

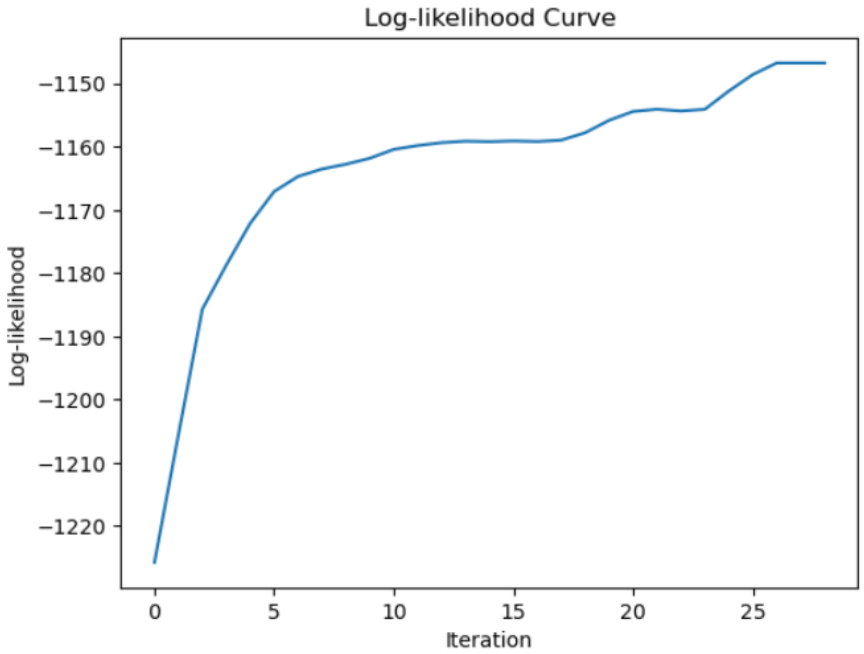


K = 6 :

Scatter Plot :



Log-likelihood vs iterations Plot :



So these are the Scatter plots for different K values and in each graph we can see that the curves get converged after certain number of iterations. The convergence of the log-likelihood graph indicates that the algorithm has found a good set of model parameters that maximize the log-likelihood of the data.

(c) (2 marks) Find the optimal k. There are several metrics like Silhouette score, Distance between GMMs, and Bayesian information criterion (BIC), or even you can use log-likelihood from the last question to infer. Give a clear explanation for your decision.
Note: **You can use third-party libraries - sklearn or any other only in this subsection.**

Solution:

We have to find the optimal K from the above given K values.

We will measure on the basis of the Silhouette Score, Bayesian information criterion(BIC) and the log-likelihood Curve from the previous part,to find the optimal K-value.

The **silhouette score** is a metric that measures how similar an object is to its own cluster compared to other clusters. A higher Silhouette Score indicates that the data point is well-matched to its own cluster and poorly matched to neighboring clusters, which suggests a good clustering result.

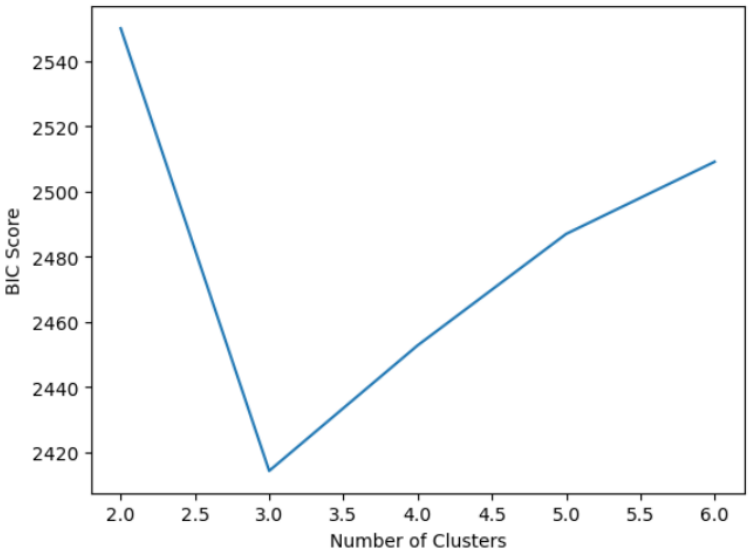
BIC is based on the principle of Bayesian model selection, which seeks to find the model that best fits the data while also being the simplest.The model with the highest posterior probability i.e with lower bic value is chosen as the best fit with fewer parameters.

So, I have implemented Silhouette Score and bic functions and I had plotted the values of these metrics with the K-values, The plots are :

Silhouette Score vs K-values :

As we can infer from the graph that the maximum value for the curve is at K = 3.

BIC vs K-values :



As we can see from the graph that the minimum value for the curve is at K = 3.

So from both the above scores, we can say that optimal K-value is 3.