CS5691: Pattern Recognition and Machine Learning
Assignment #2

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

Teammate 1: (Prakash A) (50% of contribution) **Roll number:** CS20B061
Teammate 2: (P Ayyappa Kethan) (50% of contribution) **Roll number:** CS20b058

- **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.

---

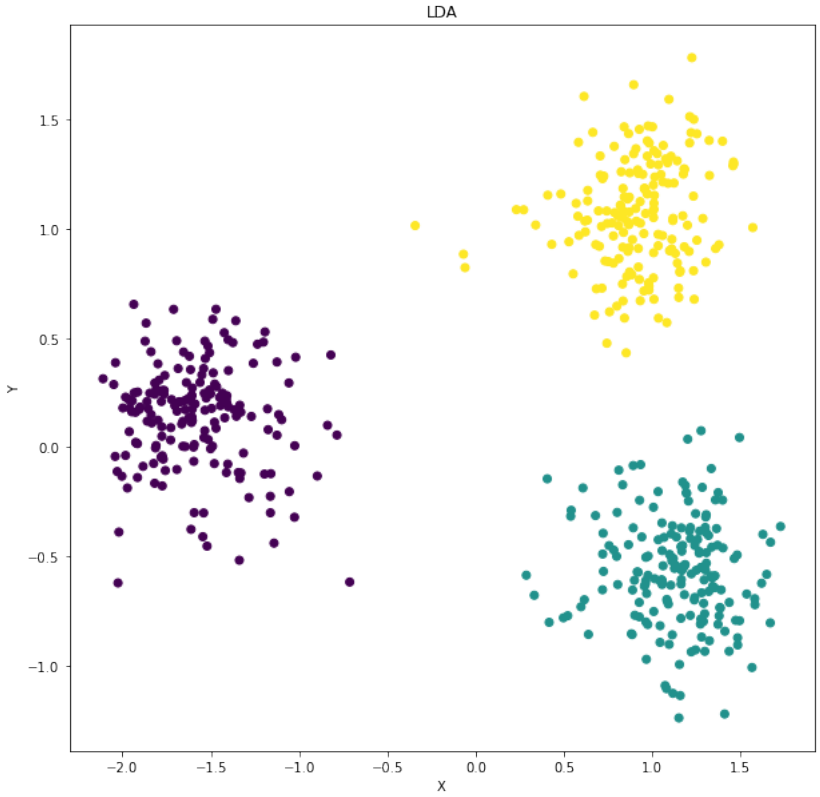
1. **[Linear Discriminant Analysis (LDA), Principal Component Analysis (PCA)**] You will implement dimensionality reduction techniques (LDA, PCA) as part of this question for the dataset1 provided here.

   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.

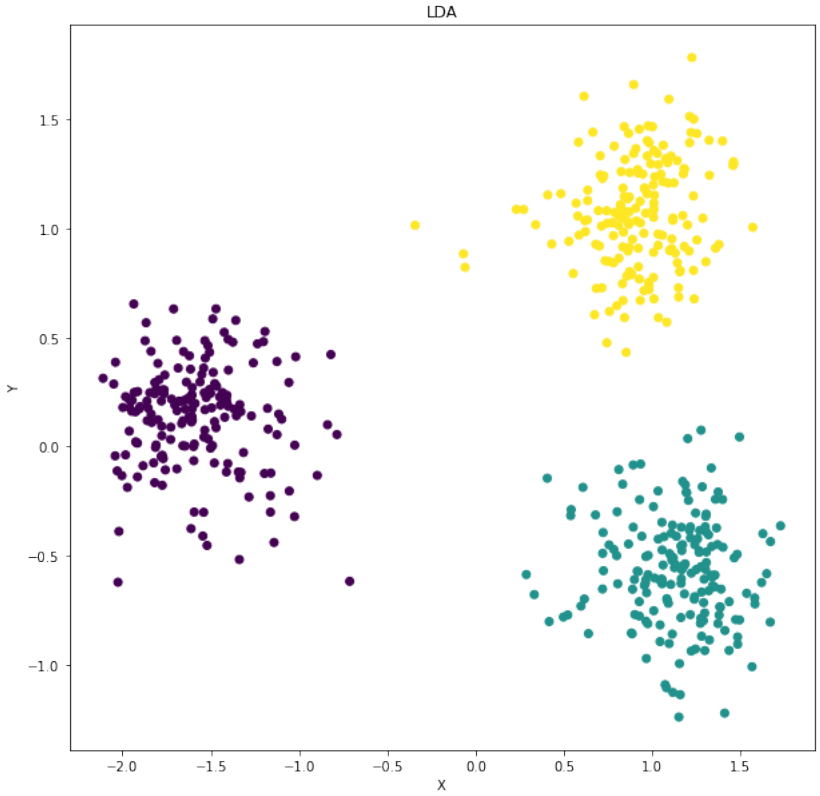
Solution:
LDA Transforms a D dimensional Data to smaller Dimension Data, In this assignment a 64 Dimension data need to converted in 2 Dimensional Data

The LDA Scatter Before Deleting Columns



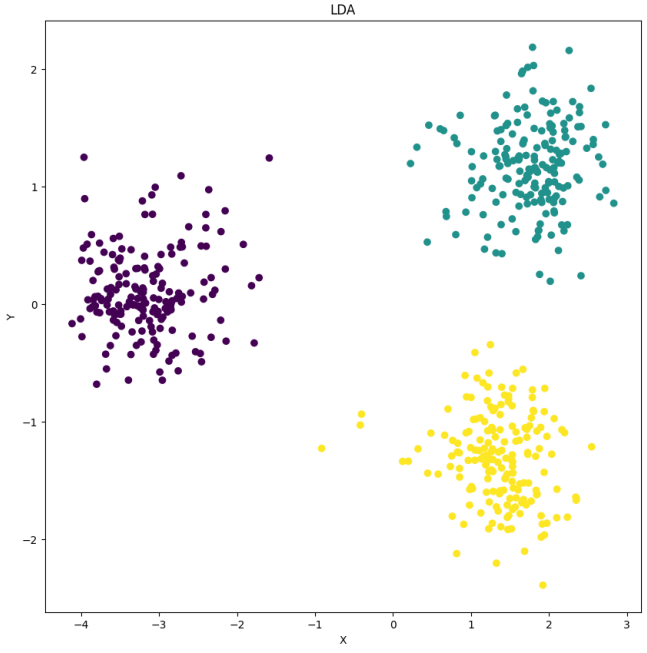
Since, The dataset1 has columns with variance = 0 , we cannot do zero mean, unit variance Normalization, so we have to delete those columns before Normalization.

After Deleting Columns LDA Scatter Plot



Now, LDA Scatter Plot after Normalization :

After Zero Mean, Unit Variance Normalization

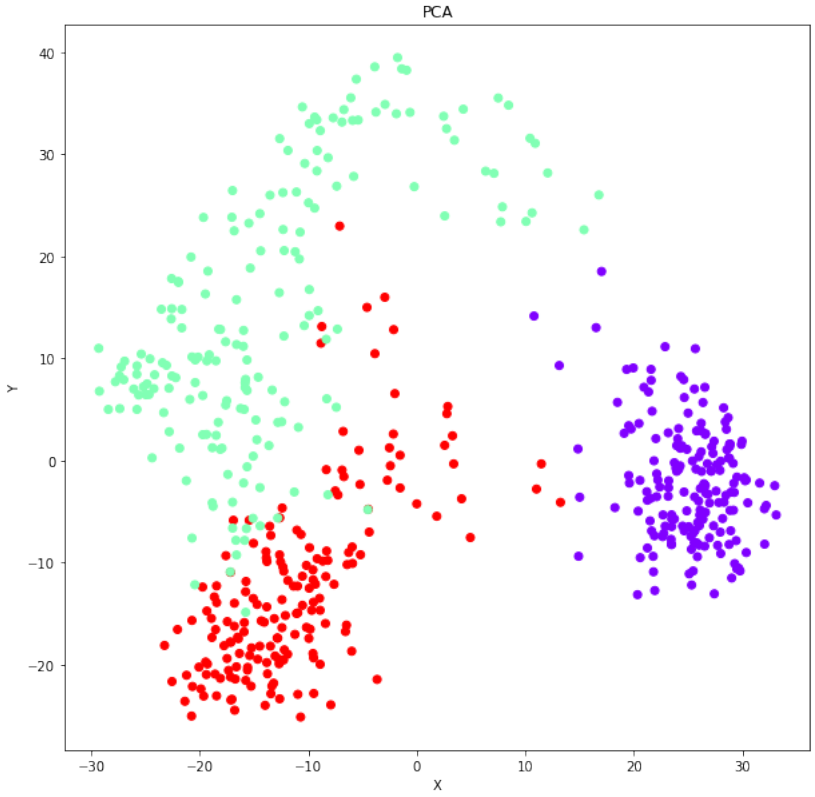


From The Above Plots one can see that even if the data is Normalized or the variance = 0 columns are deleted, the general trend or the class that point belongs to have not changed, That is the points even after normalization they still belong to same classes as before normalization only the points position/value changes due to normalization not the classification. This shows the robust quality of LDA Algorithm(classification is not changing). LDA gives same classification even after normalization.

NOTE : If the code is run in google colab with version 3.10.11 python then only we will arrive at the above plots , if it being run in different version of python such as 3.11 etc , the plot orientation may change but the points belonging to a class don't change

(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:
PCA Scatter Plot after Transformation into 2 Dimension



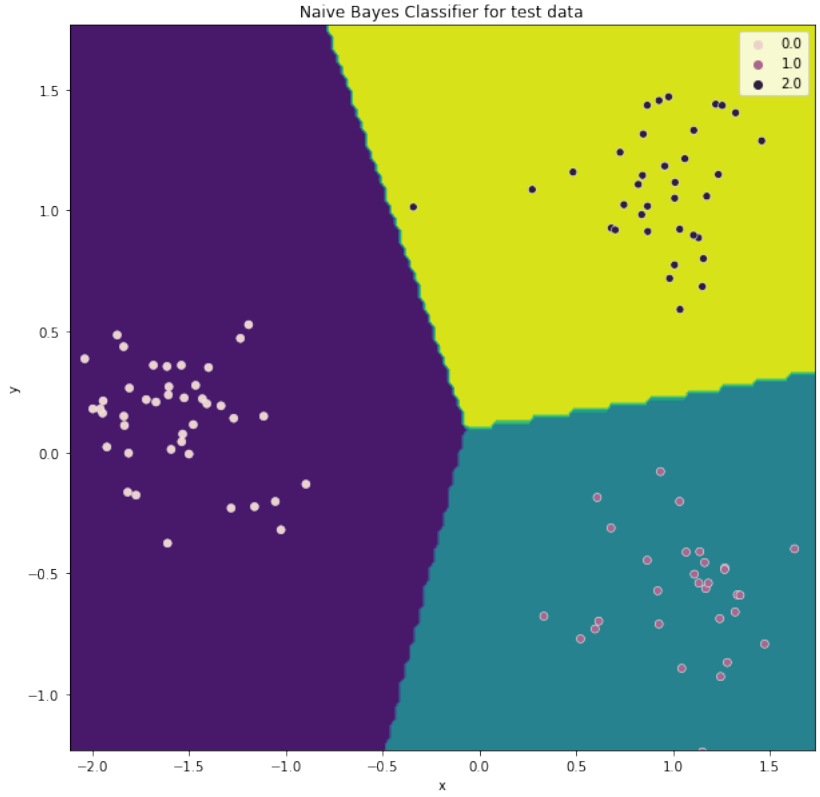
In PCA we are not using any criteria for classification into classes but in LDA we use scatter of the clusters.Due to this we are able to get good clusters in LDA.In LDA we use within class matrix and between class matrix, which uses the cohesion between

the points within the class and also between the classes , therefore when we transform the data it is more classified than PCA.

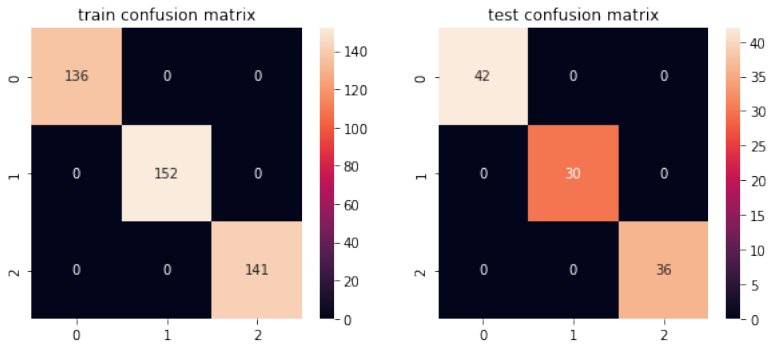
(c) (1.5 marks) Randomly shuffle and split the obtained dataset from part (a) into a training set (80%) and testing set (20%). Now build the Bayes classifier using the training set and report the following:

- 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 of Train Data : 1
Accuracy of Test Data : 1
Plot of the test data (Naive Bayes) with classification Boundary



Confusion Matrix of the test data

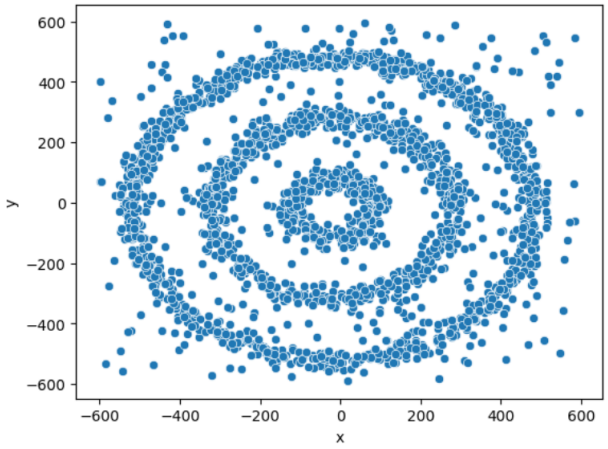


NOTE:

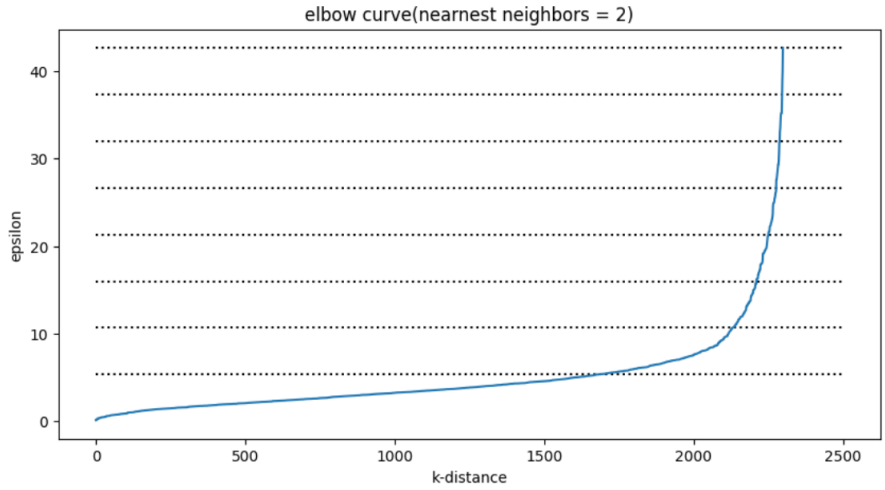
2. **[DBSCAN]** In this Question, you are supposed to implement **DBSCAN algorithm from scratch** on dataset2 provided here and dataset3 provided here. You also need to compare and contrast your observations from above with K-Means applied on both datasets. **However, you can use predefined libraries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

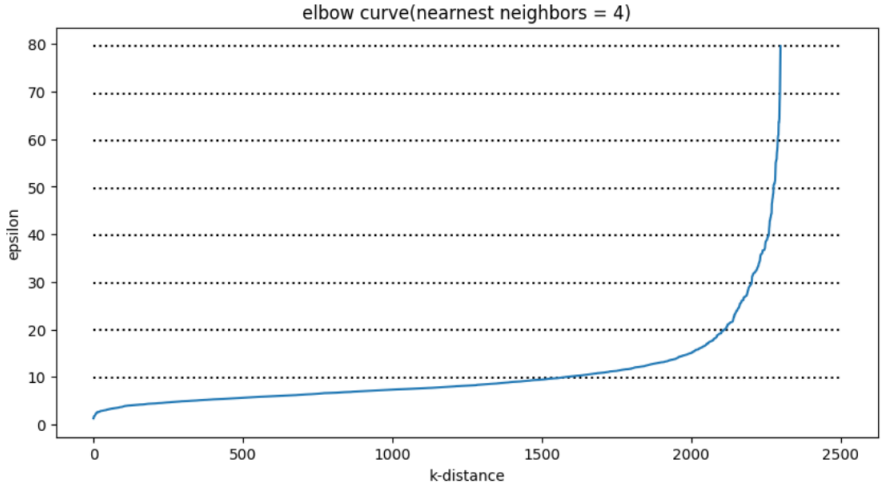
Solution:
In this whole question x is '0' in dataset and y is '1' in dataset.
Normalization is not done here as std is almost near.



Average distance of k-nearest points is checked here in estimation of epsilon range.



Considering nearest neighbors(k) = 2 elbow observed as epsilon = 10.For this the range of epsilon will be 5 to 20

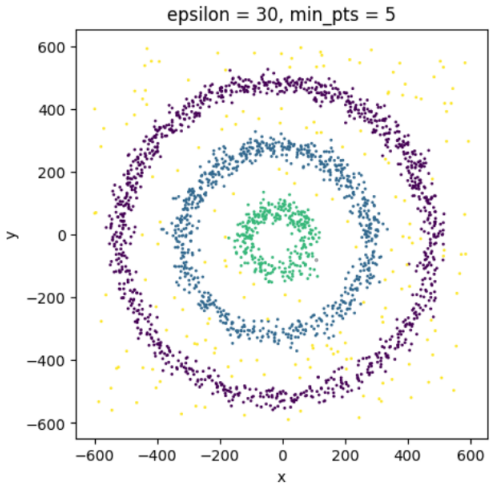


Considering nearest neighbors(k) = 4 elbow observed in between 20 to 30.For this the range of epsilon will be 10 to 40.
Final range of epsilon we are considering is 10 to 40.

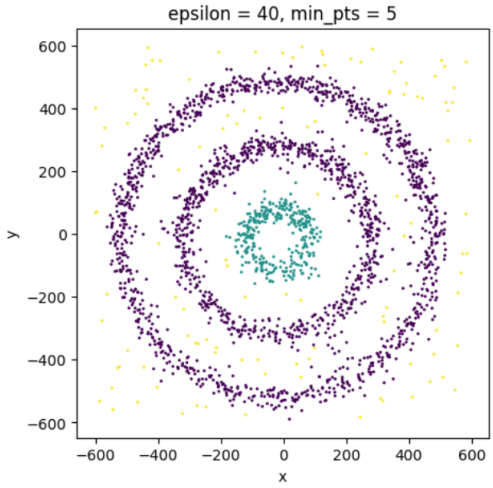
(b) (2 marks) Implement DBSCAN with the above suitable range of values of epsilon and detect the optimal value of epsilon, which gives the best clustering visually on the dataset. Show a visualization of the clusters formed for the best value of epsilon.

Solution:
Hyper parameter tuning(epsilon and minpts) is done by running loop through values of epsilon and minpts.

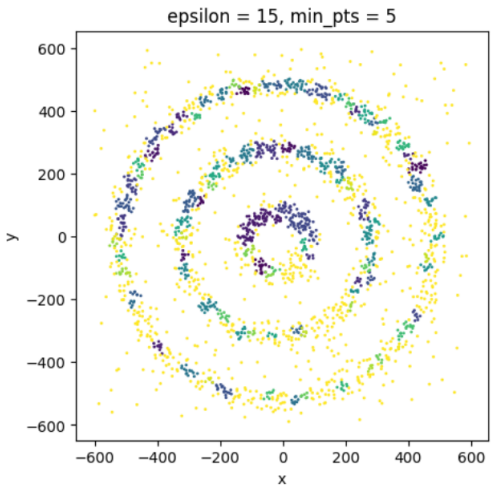
Epsilon = [10,15,20,25,30,35,40]
minpts = [4,5,6,7,8]
Optimal values : Epsilon = 30, minpts = 5 (3 clusters)



When we increase epsilon for same value of minpts the model is recording noise and decrease in number of clusters is observed in few cases.Below data has 2 clusters.

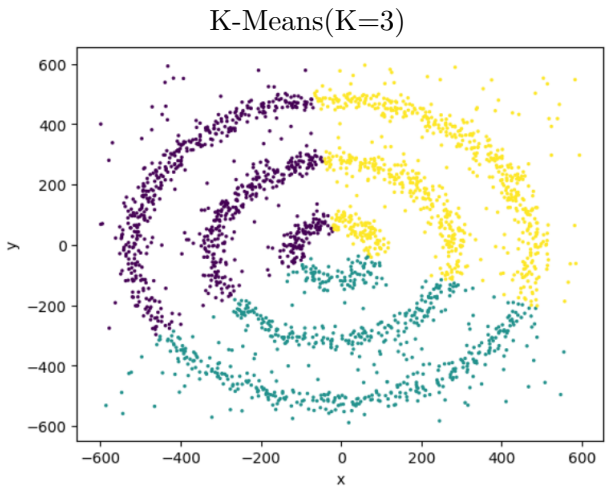


When epsilon is decreased many points which should be in class are becoming noise points.(yellow points)Below plot has 67 clusters.



(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:

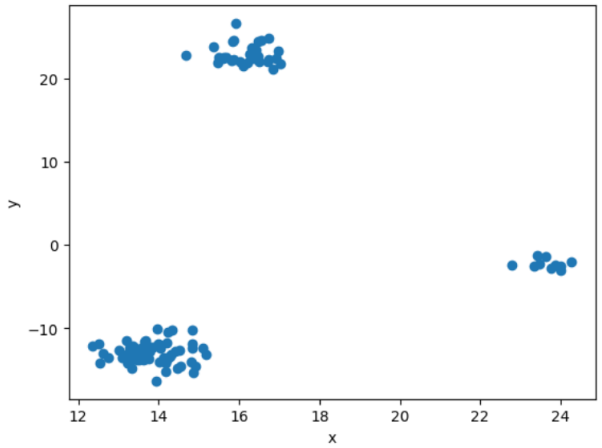


Techniques for improving K-Means clustering:
The problem here is K-Means is classifying the points of different clusters into the same cluster.To make sure that the points belonging to different clusters cluster into different clusters we increase the number of clusters.
The main idea is dividing the irregular shape into many regular shapes and join the regular shapes.
Here instead of k=3 we keep k=30 and obtain 30 clusters.Now we merge the similar clusters by using distance between centroids of 30 clusters.

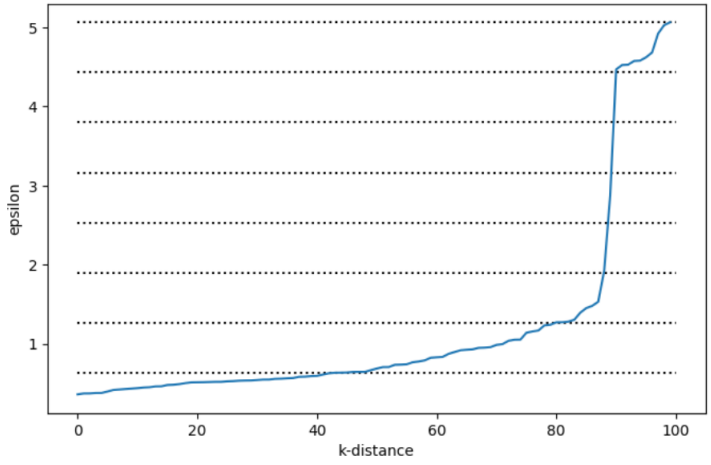
(d) (1.5 marks) Show a visualization of the data in dataset3. Use your implementation of DBSCAN with `minPts=15` on dataset3. Plot 'Elbow curve' to get an optimal range of values for `eps`. Detect the optimal value of epsilon which gives the best clustering visually on the dataset. Show a visualization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

Solution:

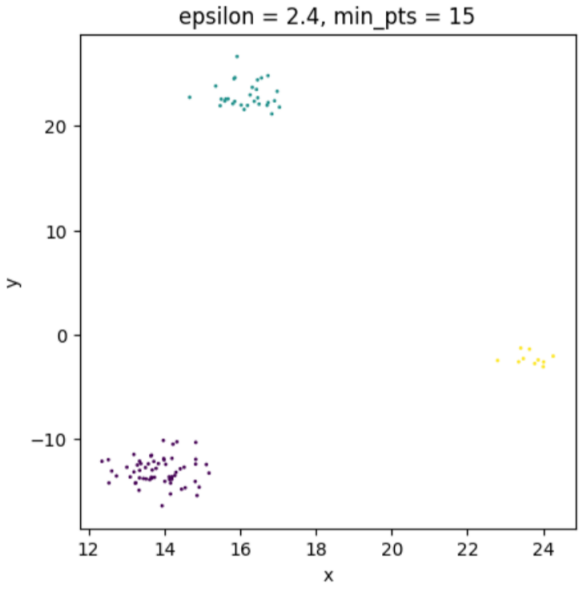
Visualizing data set 3



Elbow curve(minpts = 15)

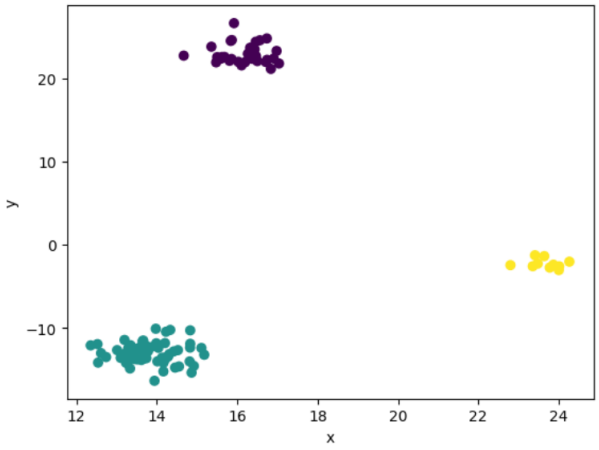


Above curve is elbow curve for minpts = 15 for dataset 3.
Epsilon range is taken form 1 to 2.5 with a difference of 0.1.
Optimal clustering is observed at epsilon = 2.4.(2.2,2.3,2.4,2.5 are also giving same cluster)



(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?

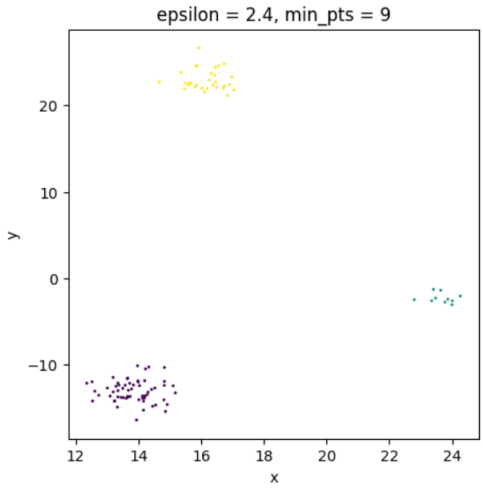
Solution:



KMeans(K=3)

K-Means have good initialization values i.e K=3. As there are three separate regions of data.
The given initialization values for dbscan are bad because dbscan is classifying the right bottom part of data as noise so reducing minpts will cluster this noise as one cluster.Having minpts less than 10 gives 3 clusters without noise.(with good value of

epsilon)



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

Solution:

- DBSCAN is robust to outliers while K-Means classifies every point i.e not robust to outliers
- KMeans output depends on initialization of centroids so it requires few iterations to get good cluster.
- KMeans require number of clusters to be specified whereas DBSCAN does not require initial number clusters.
- DBSCAN performs well with arbitrary shapes where as KMeans finds it difficult.
- DBSCAN does not perform well when there are varied density clusters.If two clusters have different densities one set of minpts,epsilon cannot classify that cluster.

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.

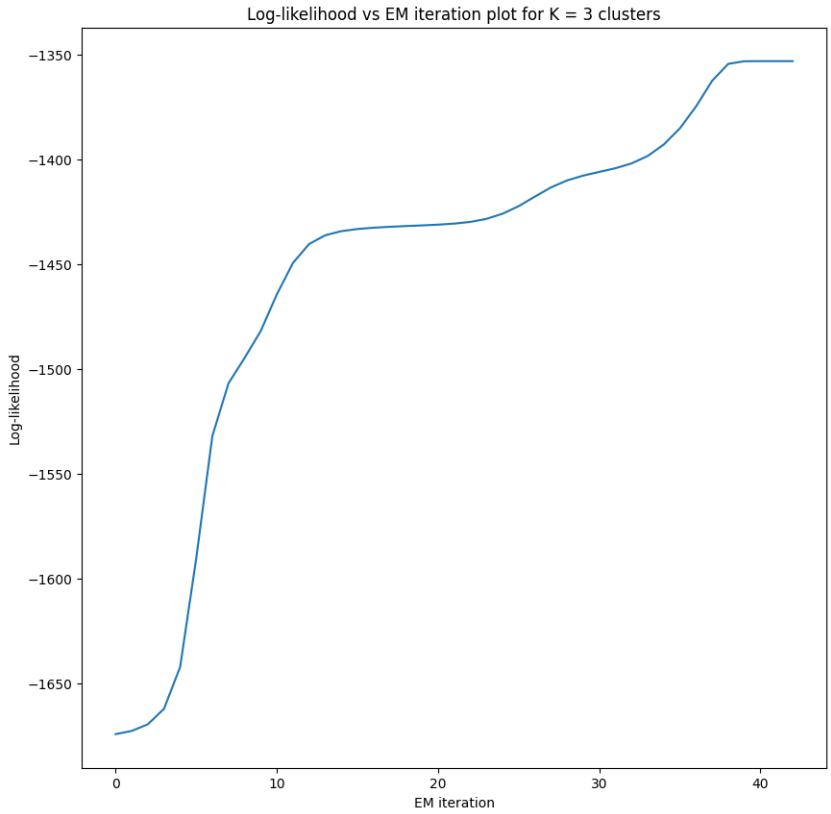
(a) (3 marks) Implement EM for GMM and plot the log-likelihood as a function of iterations.

Solution: In Implementation Of GMM, the intialization of pi, mean, covariance matrix are done as follows:
1. we have shuffled the data and divided into K clusters so pi = 1/K for all the K clusters.
2. After dividing the shuffled data into K clusters find the mean and covariance matrix for those and use them as initial values before starting EM.

Since, We have initialized the values , now start E and M step , for certain amount of iteration until the termination condition is reached that is :

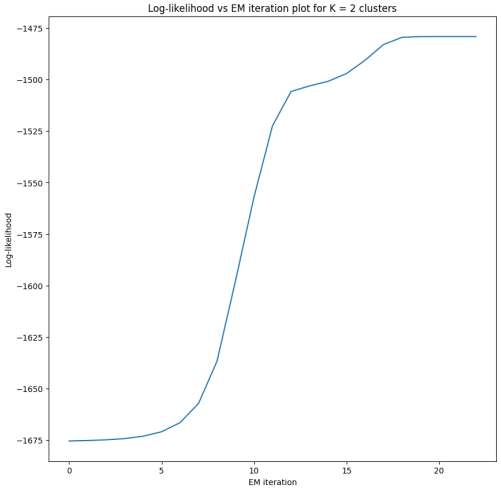
abs(log-likelihood[i-1] - log-likelihood[i]) ¡ convergence point, where log-likelihood[i] is the log-likelihood value at ith iteration.
In this Implementation Convergence Point is kept as 0.001

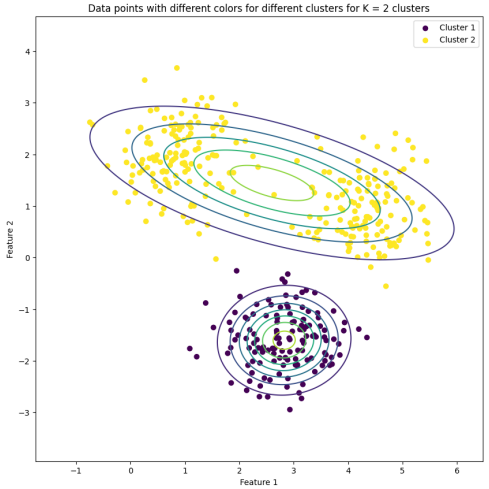
Here Example log-likelihood vs iteration for K = 3

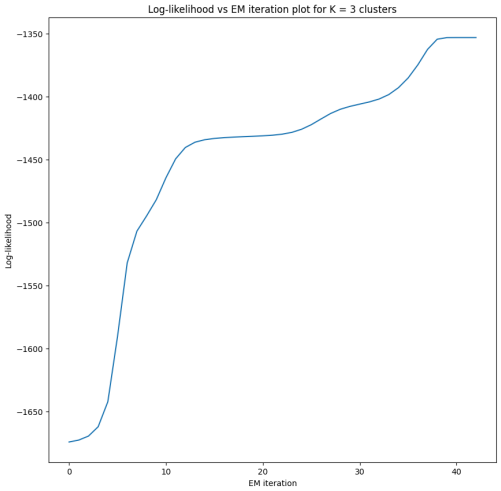


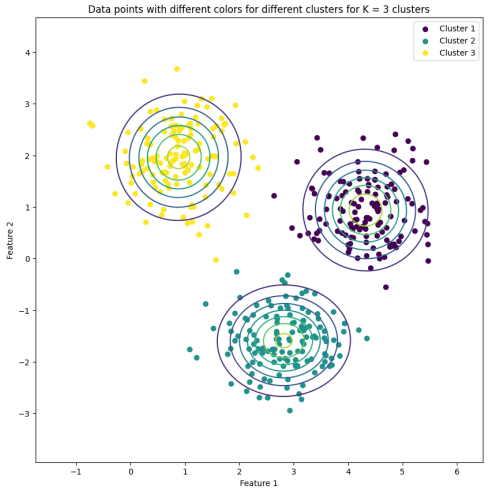
(b) (2 marks) Run EM for different numbers of Gaussians (k)(Try 2,3,4,5,6). Plot figures that can help in visualization and also log likelihood as a function of iteration for different values of k. Report the observations.

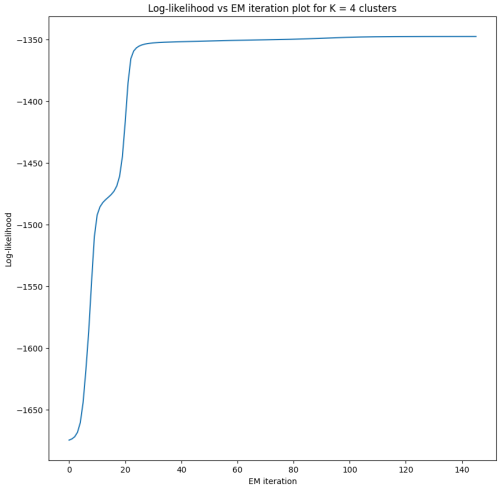
Solution:

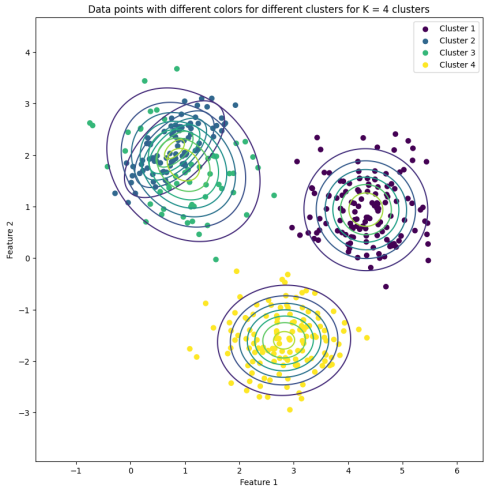

Log-likelihood vs EM iteration plot for K = 2 clusters
Log-likelihood
EM iteration

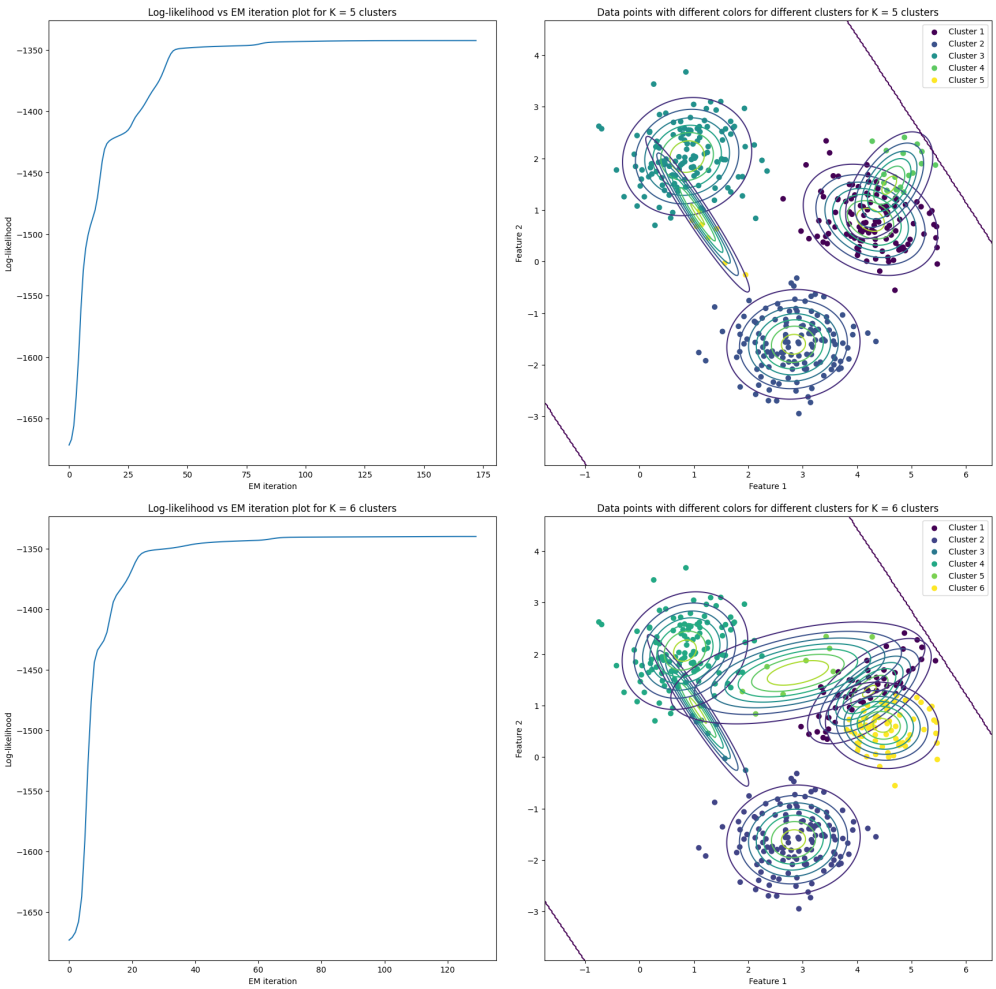

Data points with different colors for different clusters for K = 2 clusters
Cluster 1
Cluster 2
Feature 2
Feature 1


Log-likelihood vs EM iteration plot for K = 3 clusters
Log-likelihood
EM iteration


Data points with different colors for different clusters for K = 3 clusters
Cluster 1
Cluster 2
Cluster 3
Feature 2
Feature 1


Log-likelihood vs EM iteration plot for K = 4 clusters
Log-likelihood
EM iteration


Data points with different colors for different clusters for K = 4 clusters
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Feature 2
Feature 1



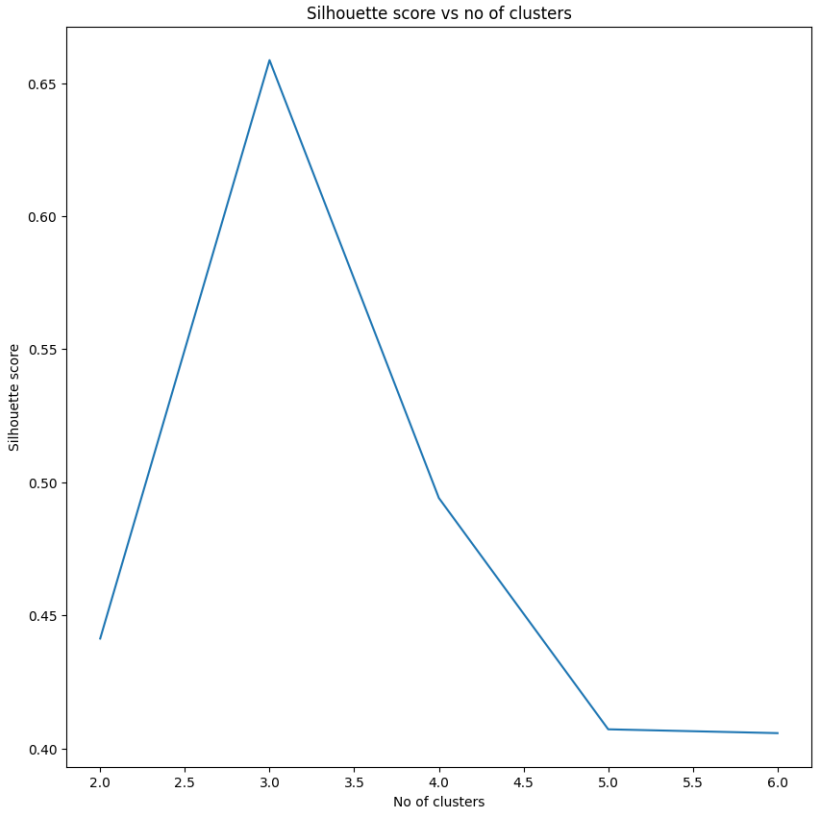
From the above observation it is clear that the K = 3 cluster is more clear and cohesive, the points fall in 3 different cluster and the Gaussians or not intersecting and the distance between points are cohesive compared to K = 2 therefore K = 3 looks more optimal value, but we have compare it with silhouette score to decide which is optimal and for K > 3 the Gaussians are intersecting so can't be optimal.

(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: Here To Find The Optimal Value of K we are using Silhouette Score.
The silhouette coefficient or silhouette score is a measure of how similar a data point is within-cluster (cohesion) compared to other clusters (separation), this can give a general idea how the clusters are arranged and cohesion between points, so larger the

silhouette score, better the clustering is , here we got the largest silhoutte score for K = 3, this means optimal K = 3.

Silhoutte Score Graph for various Value of K



From th Graph it is clear that the optimal value of K is 3, larger the silhouette score more clear and cohesive the clustering is, and we can see that for K = 3, the clustering is very neat and clear (from plot).

plot for K = 3