CS5691: Pattern Recognition and Machine Learning
Assignment #2

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

Teammate 1: (Debojyoti Mazumdar)(50% of contribution) **Roll number:** EE20B030
Teammate 2: (Sneha Reddy Palreddy)(50% of contribution) **Roll number:** EE20B129

- **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.

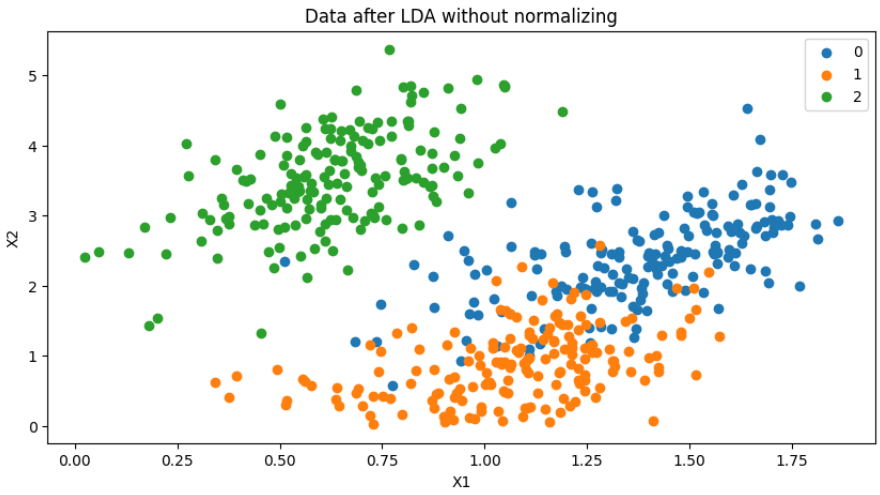
---

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.

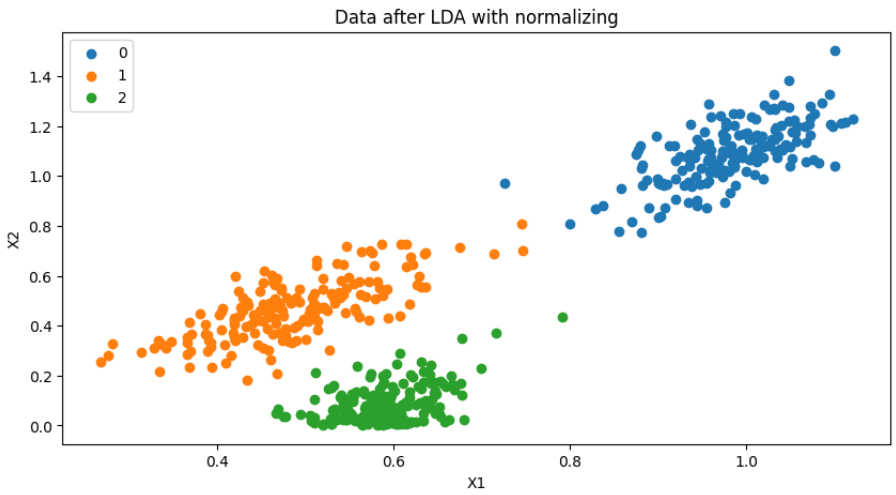
   Note that you have to implement **LDA from scratch** without using any predefined libraries (i.e. sklearn, scipy) . However, you can use **predefined libraries to implement PCA.**

   (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:
   The figure below is the scatter plot of the dataset-1 after converting it into a 2-dimensional dataset using LDA.

   After performing mean and unit variance normalization and then applying LDA we get the following scatter plot.

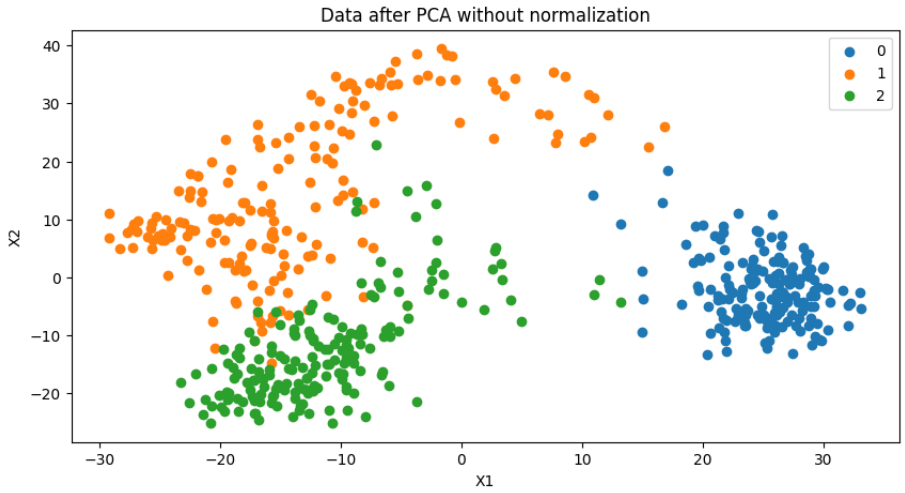


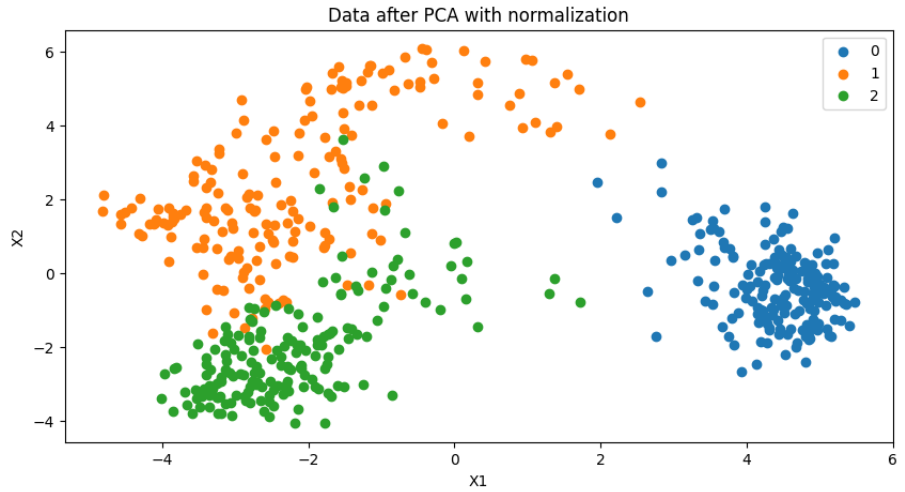
Mean normalization results in the mean of the new dataset being centered at the origin. Unit variance normalization results in the data being less scattered within each cluster. This leads to the clusters being more distinct. As seen the Silhouette score of the unnormalized data is 0.42 but of the normalized data it is 0.68. So the normalized data leads to better clustering than unnormalized data.

(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:

The figure below are the scatter plots of the dataset-1 with and without normalized after converting it into a 2-dimensional dataset using PCA.





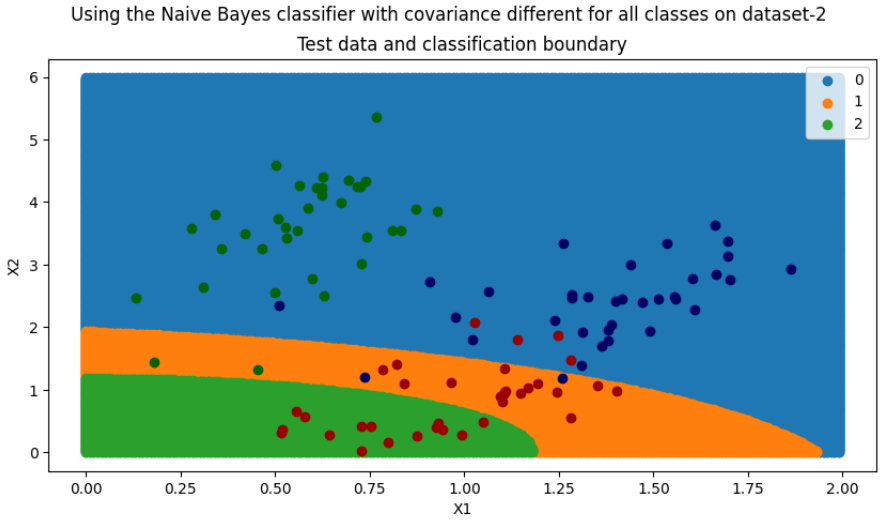
From the above 4 graphs we can see that for LDA the structure of the data (ignoring the value and just taking the positions of the data points) changes for normalized input dataset and unnormalized input dataset but for PCA the structure of the data (ignoring the value and just taking the positions of the data points) does not change for normalized input dataset and unnormalized input dataset.Also the dataset is more seperable into clusters after LDA than after doing 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:
Below is the naive bayes classified data considering I as covariance matrix of all classes. Dividing the transformed data into train and test data (80:20) and applying the above Naive Bayes classifier on it we have:

- Accuracy on train data = 0.9976635514018691
  Accuracy on test data = 0.9906542056074766
- Plot of the test data along with classification boundary:



- Confusion matrix of test data :

$$\begin{bmatrix} 33 & 4 & 33 \\ 2 & 18 & 2 \\ 0 & 15 & 0 \end{bmatrix}$$

Confusion matrix of train data :

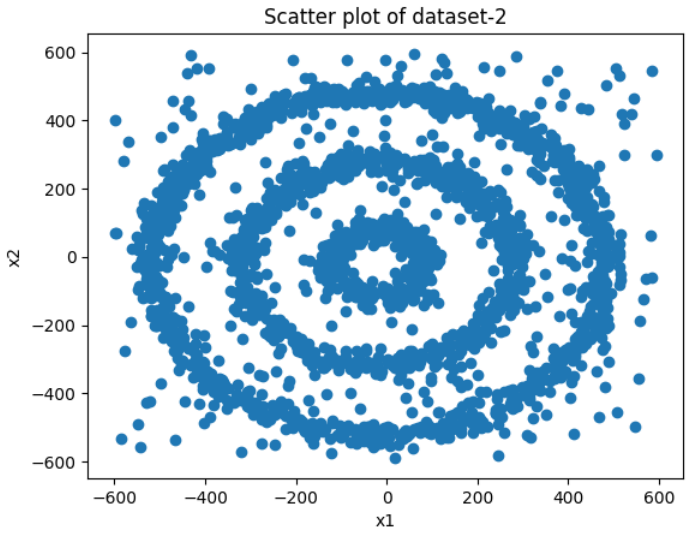
$$\begin{bmatrix} 108 & 35 & 0 \\ 29 & 93 & 24 \\ 4 & 17 & 118 \end{bmatrix}$$

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.**

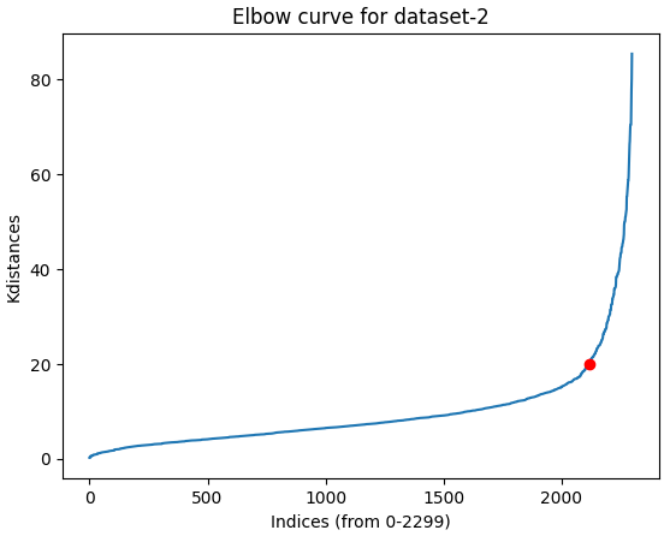
   (a) (1 mark) Visualize the data in dataset2. Then, find a suitable **range of values for epsilon** (a hyperparameter in DBSCAN algorithm) by using the 'Elbow Curve' of Datapoints plotted between K-Distance vs Epsilon. For simplicity, take only

integer values for epsilon. **You can use predefined libraries to implement K-distance.**

Solution:
We first plot the scatter plot of the dataset-2 to visualise how the points are distributed. The plot is as follows:



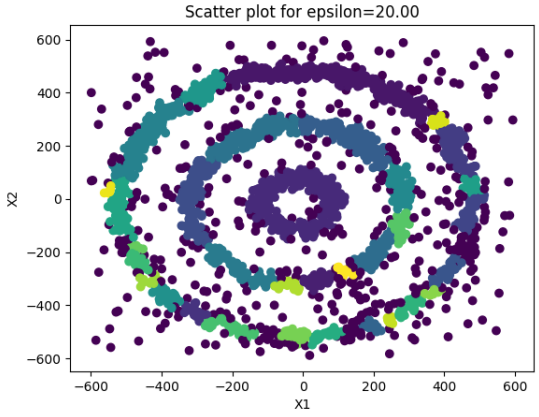
To perform the DBSCAN method of clustering on the given data we need two other inputs i.e. the epsilon value and number of minimum points. For this dataset we take minimum number of points as 5. We plot the elbow curve to find out the optimal range of epsilon values which give a good clustering result i.e. not too many clusters nor too small. The plot is as follows:

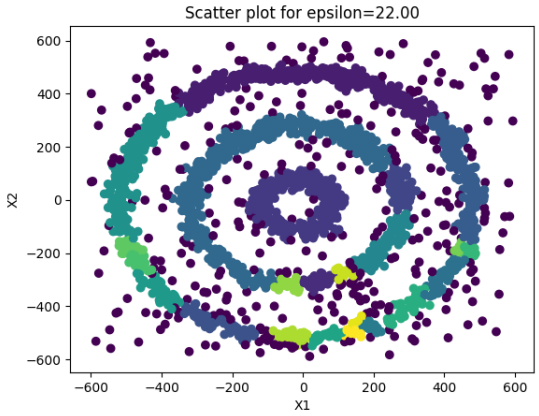


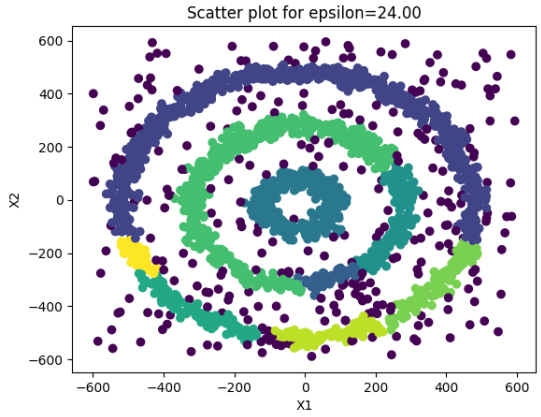
The optimum range of values of epsilon are (20-30)

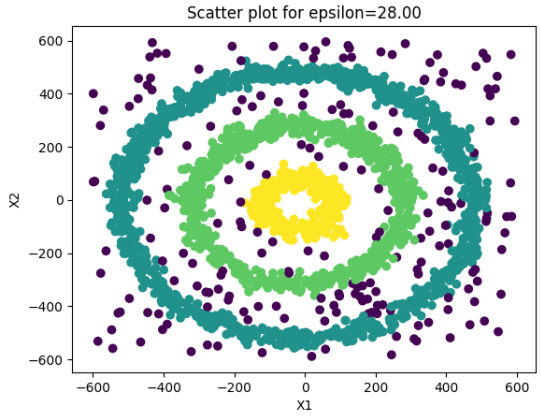
(b) (2 marks) Implement DBSCAN with the above suitable range of values of epsilon and detect the optimal value of epsilon, which gives the best clustering visually on the dataset. Show a visualization of the clusters formed for the best value of epsilon.

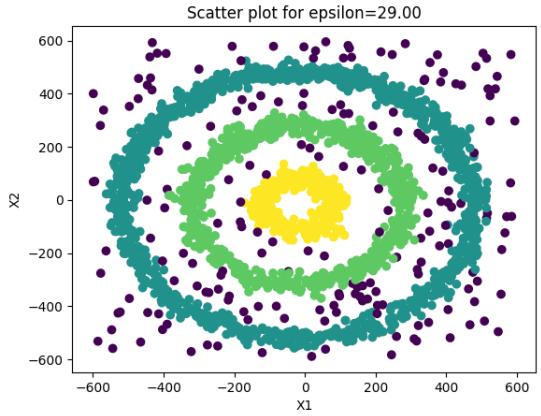
Solution:
We now implement the DBSCAN algorithm for the above range of epsilon values.For epsilon =20 the number of clusters are equal to that of 27 and they decrese as epsilon increases. A total of 4 clusters are formed for epsilon= 28, which is visually better than the lower epsilon values. The plots for various epsilon values are as follows:



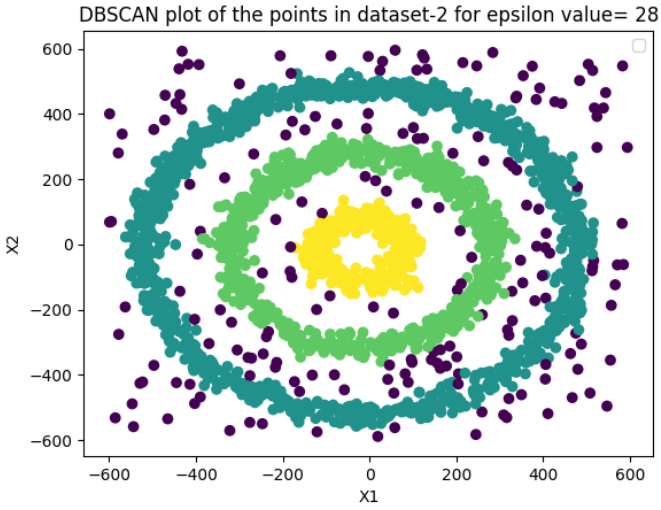






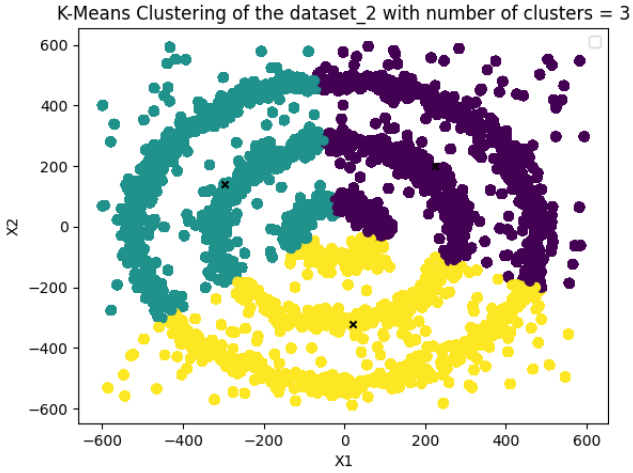


From the above plots we can observe that the graphs with increasing epsilon the total number of clusters decreases. For ep=28 the number of clusters=3. For epsilon values higher than this the number of clusters don't change till some extent but after which all the points form one cluster which is not effective.



(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:
For the K-Means we take the number of clusters=3 and try to plot it.
The plot is as follows: The black points mark the centroid of each cluster.



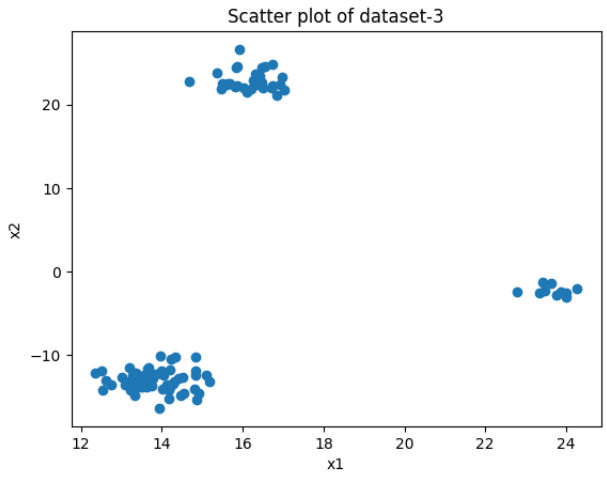
Few ways to improve the clustering by k-means are:

- Kernel Kmeans: Usually K-Means is implemented assuming that the boundaries are linear, for data that isn't linearly separable, we can use Kernel K-Means. Here the data is mapped to a higher dimension which can be linearly seperable and K-means can be applied properly then

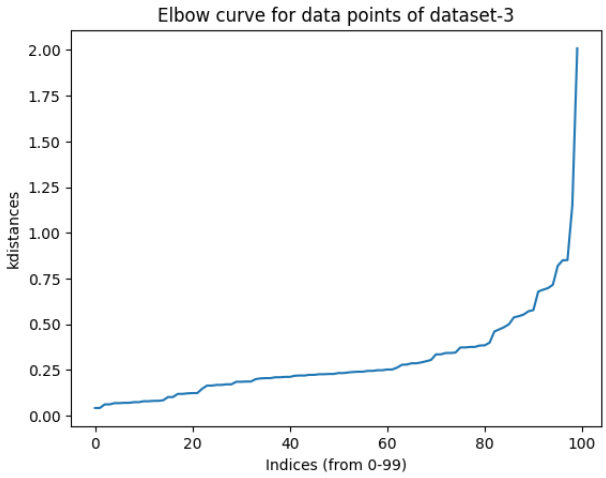
- With better initialisation techniques(maxmin) we can reduce the error significantly from an average of 15 percent to an average of 6 percent.
- Normalisation: By normalising we are increasing the convergence speed, i.e. it helps the optimisation algorithm cluster faster thus saving some time in calculations.

(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.

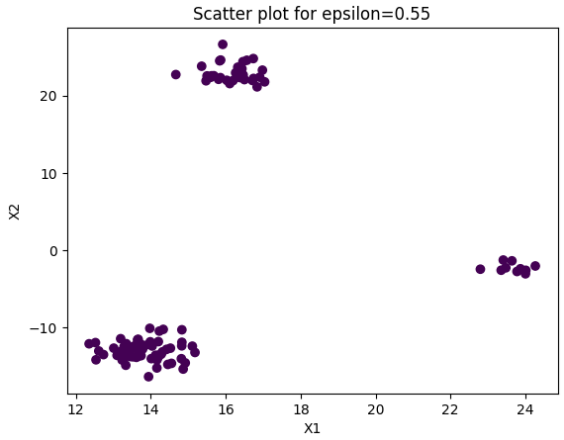
Solution:
We plot the all the datapoints of dataset-3 on a scatter plot which is as follows:

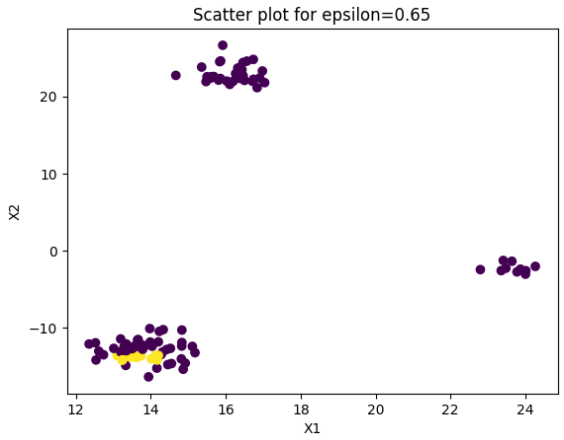


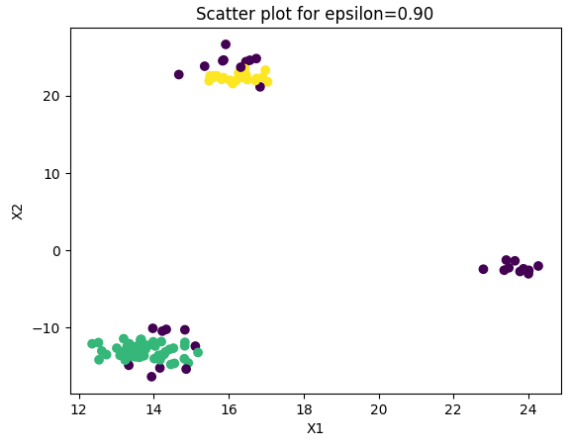
We then plot the elbow curve for the above datapoints-



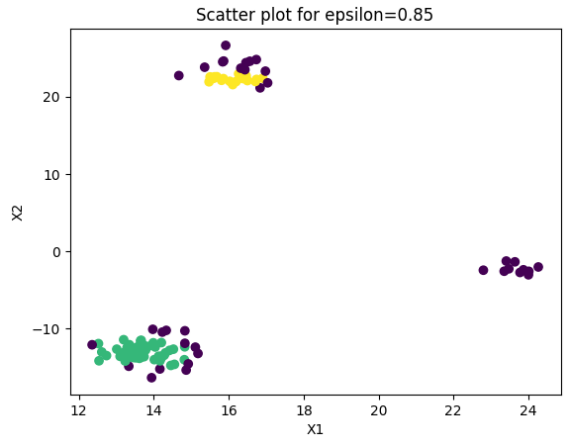
From the above graph we take epsilon values in the range (0.5,0.95) and perform DBSCAN on the data. Plots of few values of epsilon are as follow:





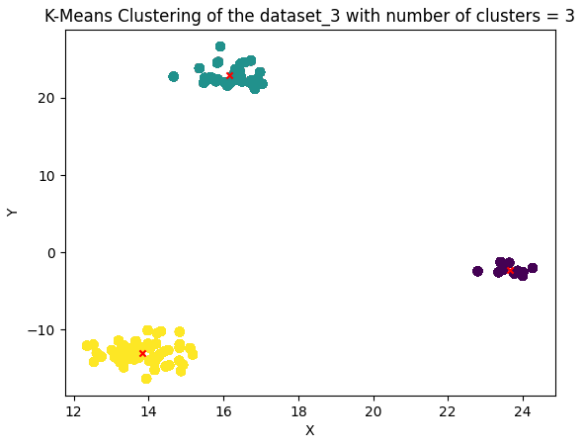


The best plot visually is achieved for epsilon = 0.85 with minimum number of points as 15. The plot looks as follow:



(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 for this dataset is as follows: The points marked red are centroid of each cluster.



- For the DBSCAN plot since the number of minimum points is high(15), we observe that for values of epsilon upto 0.6 we don't have any clusters formed, after which with incresing epsilon values we get two cluters with a lot of outlier points. Thus we can say that DBSCAN doesn't work very well for sparse datapoints.
- As for the kmeans the clusters are formed well as the data is seperated adequately. For this dataset K-means does a better job than DBSCAN method of clustering.

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

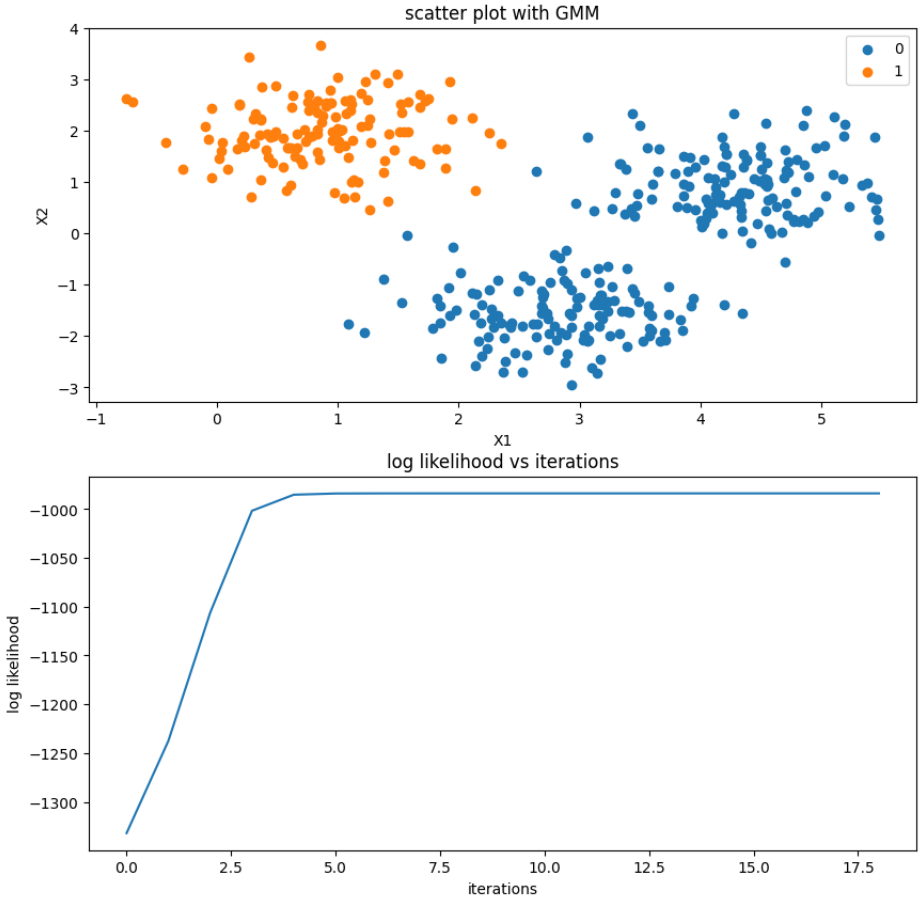
Solution:
Based on our observations wrt the two clustering methods on the two datasets. We can infer that: K-means doesn't go well with outliers as the centroid gets affected by them and differs quite a bit from the optimal value, whereas db-scan can identify the outliers easily, hence is suitable for noisy data sets like dataset-2.

K-Means is simple and easy to implement and works well for well seperated spherical clusters and usually is faster than other clustering methods.
DBSCAN cons: The output might not be very effective if the data has varying densities.

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.

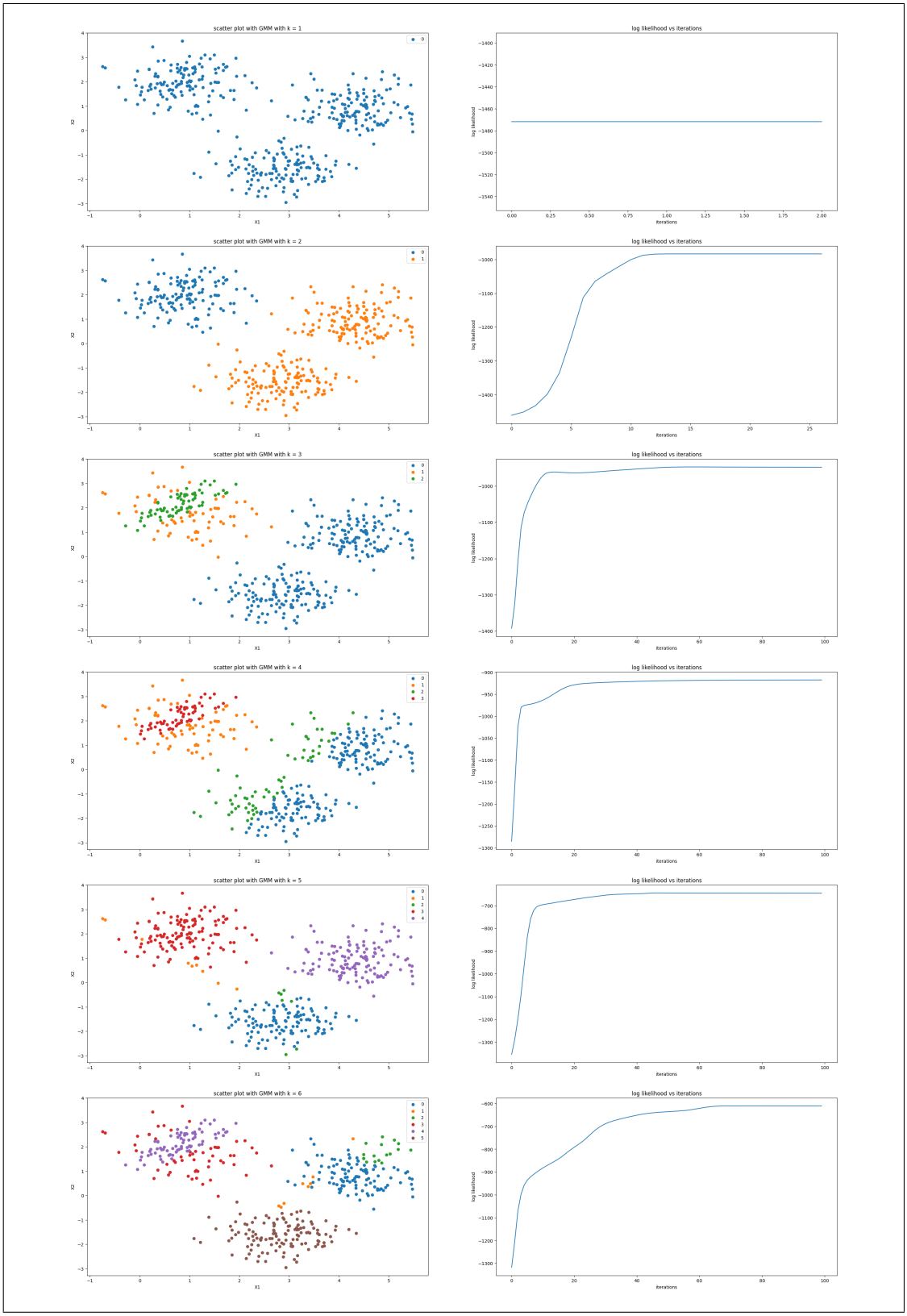
(a) (3 marks) Implement EM for GMM and plot the log-likelihood as a function of iterations.

Solution: Below is the scatter plot after clustering using the EM algorithm for GMM and the log-likelihood vs iteration plot of the dataset-4 for k = 2 (2 clusters)



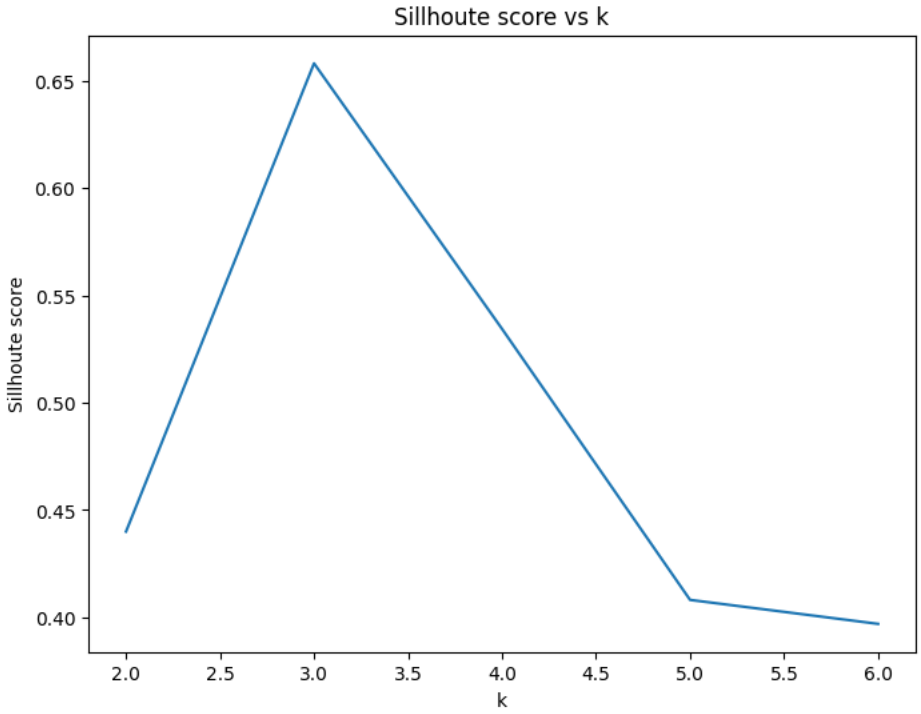
(b) (2 marks) Run EM for different numbers of Gaussians (k)(Try 2,3,4,5,6). Plot figures that can help in visualization and also log likelihood as a function of iteration for different values of k. Report the observations.

Solution:
Below are the scatter plot and log likelihood vs iteration plot for k = 1,2,3,4,5,6.


scatter plot with GMM with k = 1
log likelihood vs iterations
scatter plot with GMM with k = 2
log likelihood vs iterations
scatter plot with GMM with k = 3
log likelihood vs iterations
scatter plot with GMM with k = 4
log likelihood vs iterations
scatter plot with GMM with k = 5
log likelihood vs iterations
scatter plot with GMM with k = 6
log likelihood vs iterations
X1
X2
Iterations
log likelihood

(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 will be using the Silhouette score for finding the best value of k. Below is the plot for Silhouette score vs k



Silhouette score gives a value for how good the clustering is by finding how similar a data point is within-cluster (cohesion) compared to other clusters (separation). It is calculated as shown below.

$$s = \frac{b - a}{max\{a, b\}}$$

where b is the mean nearest-cluster distance for each sample and a is the mean intra-cluster distance. Closer the score value to 1, better is the clustering.
In the above graph we see that the score is close to 0.65 for k = 3 which is the maximum value of the score compared to the score values for all values of k.

So the optimal value of k is equal to 3.

4. **[Extras]** We have implemented FCM (Fuzzy C-Means) clustering algorithm on dataset-2 (used in question 2). The code of this is in `Assignment-2-Extras.ipynb`