Pavan Rathnakar Shetty
EECE 5644
HW 4
Due date: December 14, 11:59 pm
GitHub: Files in the hw4 folder
https://github.com/Pavan-r-shetty/5644.git

Question 1

Problem 1

Multilayer Perceptron was used to approximate class label posteriors.

Loss used for training < Minimum average cross-entropy loss.

Trained models were then used to approximate a MAP classification rule to achieve minimum probability of error on validation dataset.

Data: 4 classes with uniform priors

class 0:

$p(L=0) = 0.25$, $m0 = \begin{bmatrix} 4 \\ 2 \\ 1 \end{bmatrix}$ $cov\,0 = \begin{bmatrix} 1 & 0 & 0.8 \\ 0 & 0.7 & 0.3 \\ 0.5 & 0.6 & 0.8 \end{bmatrix}$

class 1:

$p(L=1) = 0.25$, $m1 = \begin{bmatrix} 1 \\ 3 \\ 4 \end{bmatrix}$ $cov\,1 = \begin{bmatrix} 1 & 0 & 0.6 \\ 0.6 & 1 & 0 \\ 0 & 0.6 & 0.8 \end{bmatrix}$

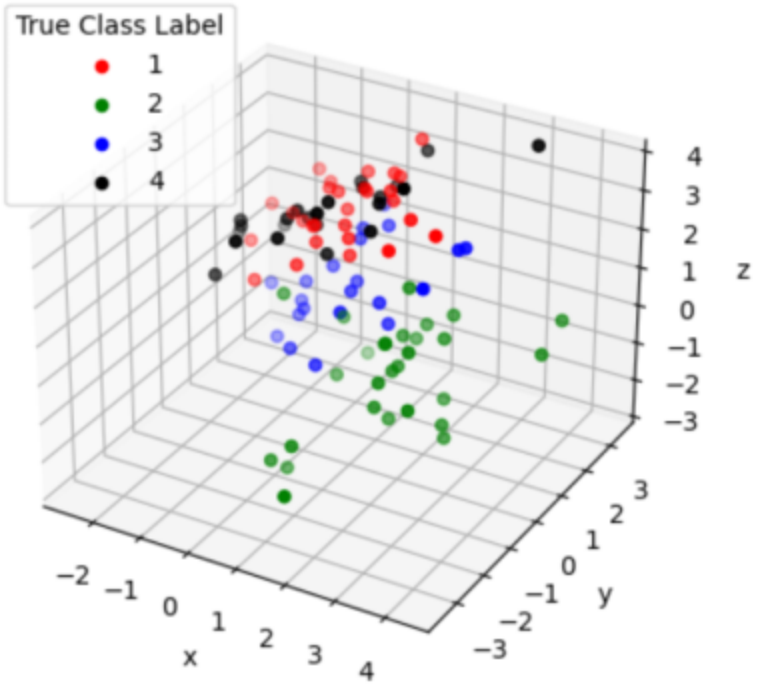
class 2:

$p(L=2) = 0.25$, $m2 = \begin{bmatrix} 2 \\ 4 \\ 2 \end{bmatrix}$ $cov\,1 = \begin{bmatrix} 0.8 & 0.1 & 0.7 \\ 0.4 & 0.9 & 0.6 \\ 0.7 & 0.8 & 0.9 \end{bmatrix}$

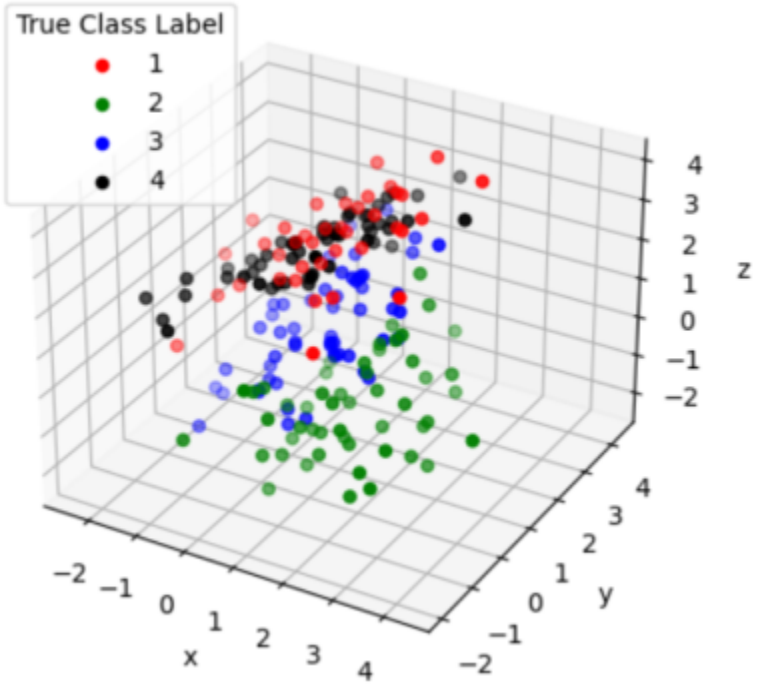
class 3:

$p(L=3) = 0.25$ $m3 = \begin{bmatrix} 3 \\ 1 \\ 3 \end{bmatrix}$ $cov\,3 = \begin{bmatrix} 1 & 0.6 & 0.9 \\ 0.1 & 0.8 & 0.3 \\ 0.7 & 0.3 & 1 \end{bmatrix}$

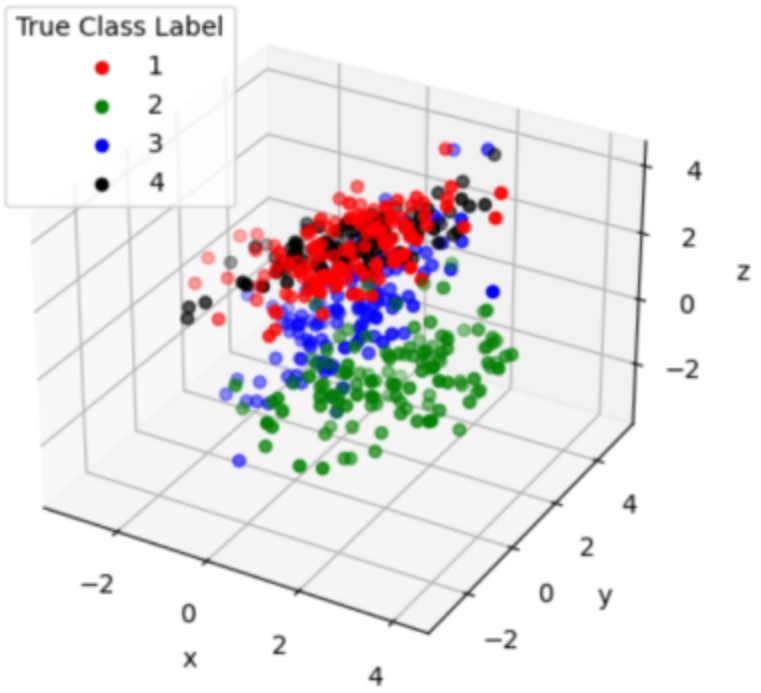
The scattered plots for all the dataset are:



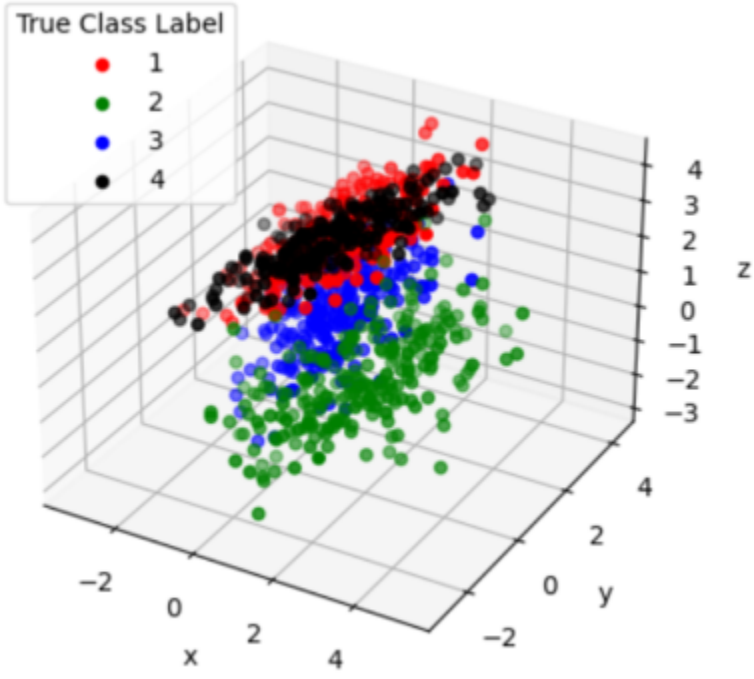
train_data_100



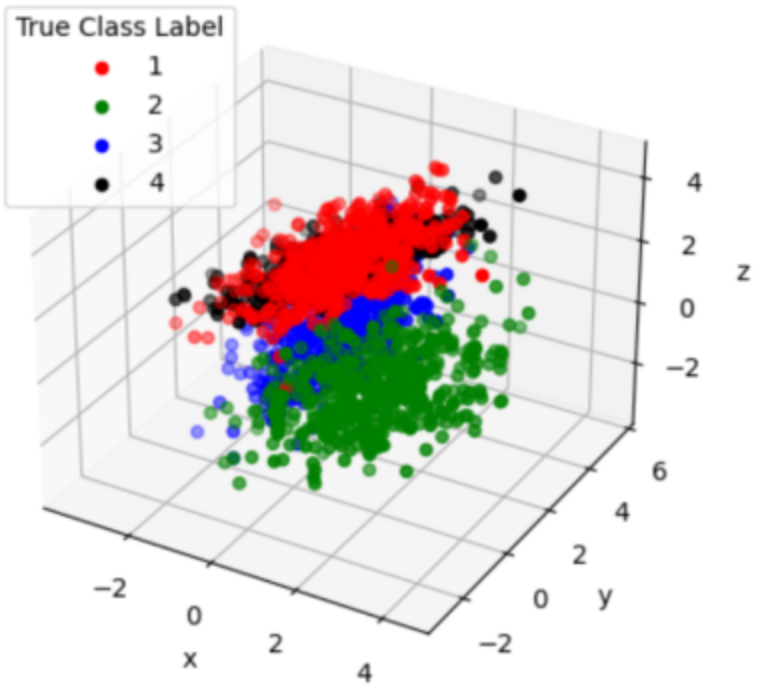
train_data_200



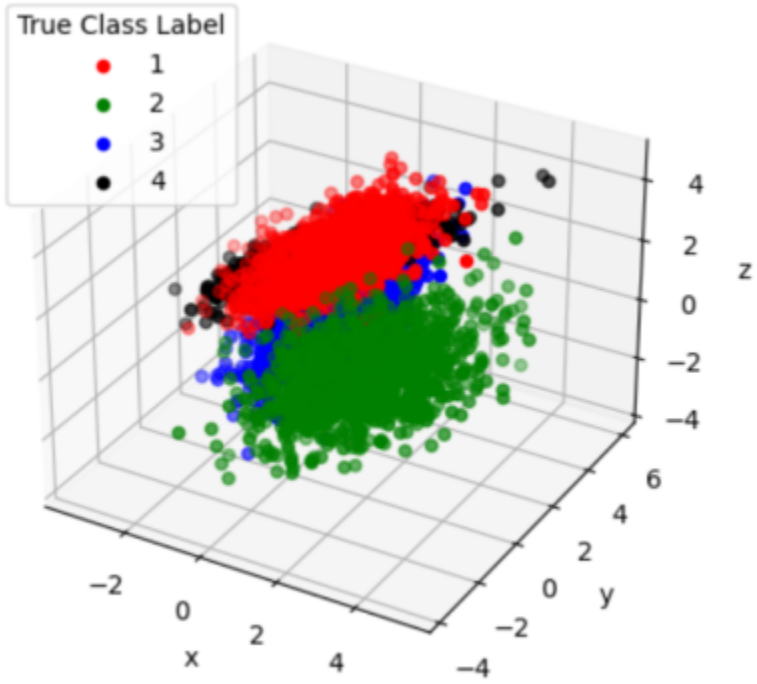
train_data_500



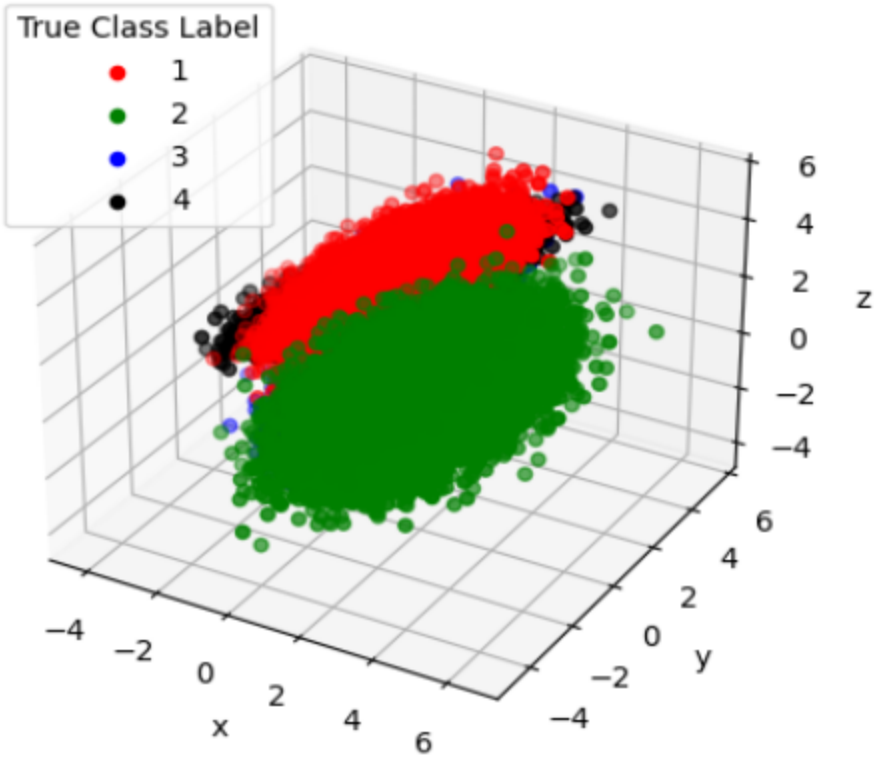
train_data_1000



train_data_2000



train_data_5000



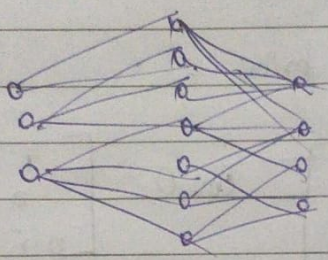
train_data_10000

Multi layer Perceptron:

- A model is created with an input layer, one hidden layer and output layer.

- The number of nodes in the hidden layer is varied from 1 to 10 to select the best.

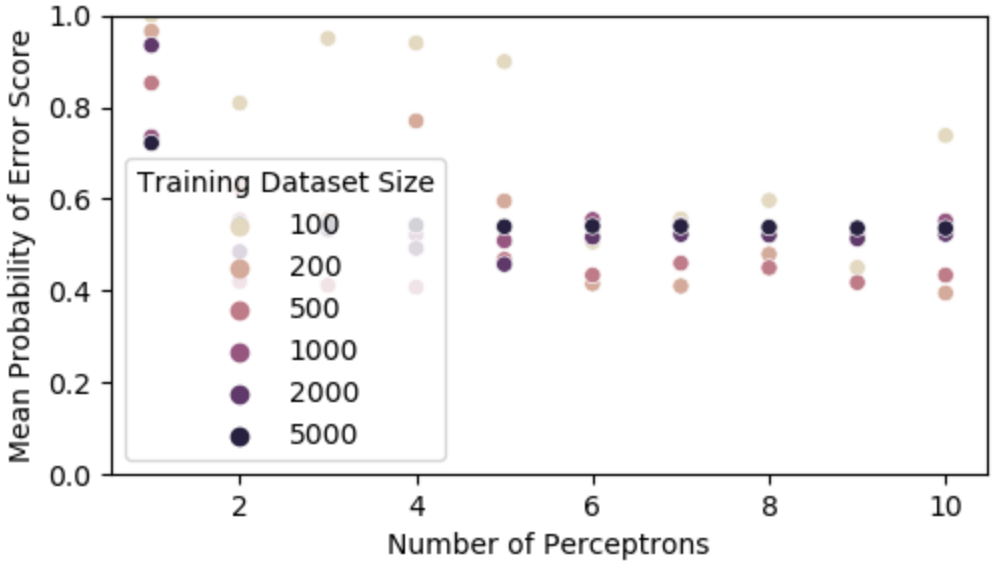
Eg:



Hyper parameter tuning using 10-fold cross validation

Figure below shows the performance of MLP with varying number of perceptrol.

- 10 fold cross validation was performed to determine the optimal number of MLP.



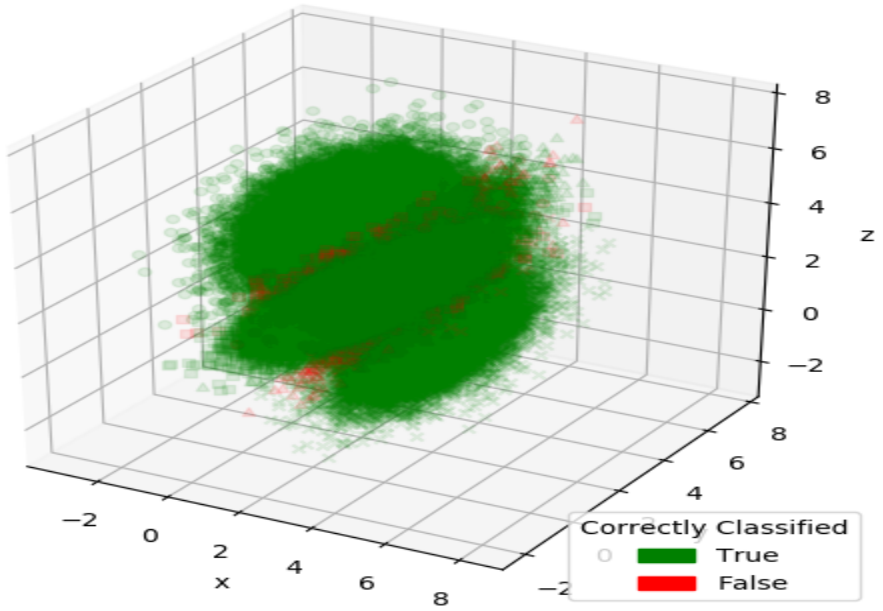
Observation and Inference:

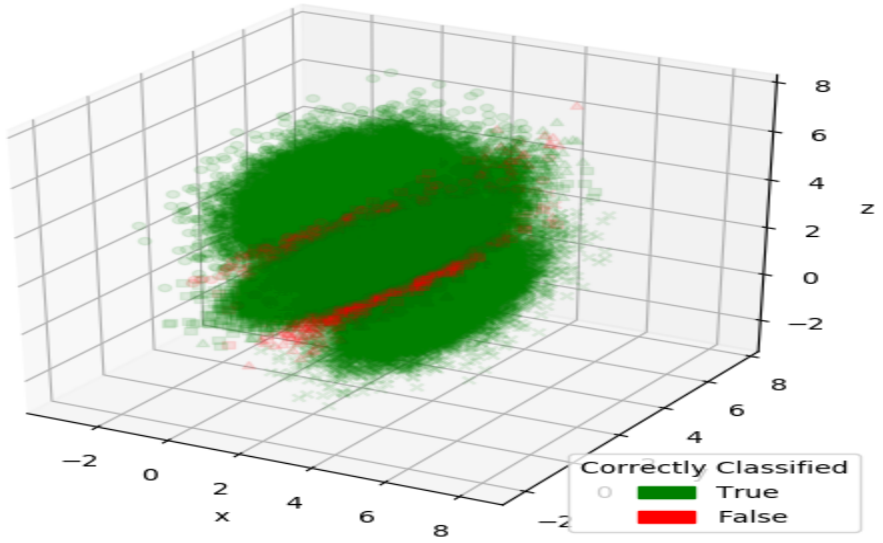
- As perceptron increases, Error reduced, however for 4–6 perceptron there is no much significant reduction in the error score.
- The score saturated & sometimes overfit.
- larger the training set, better is the performance.

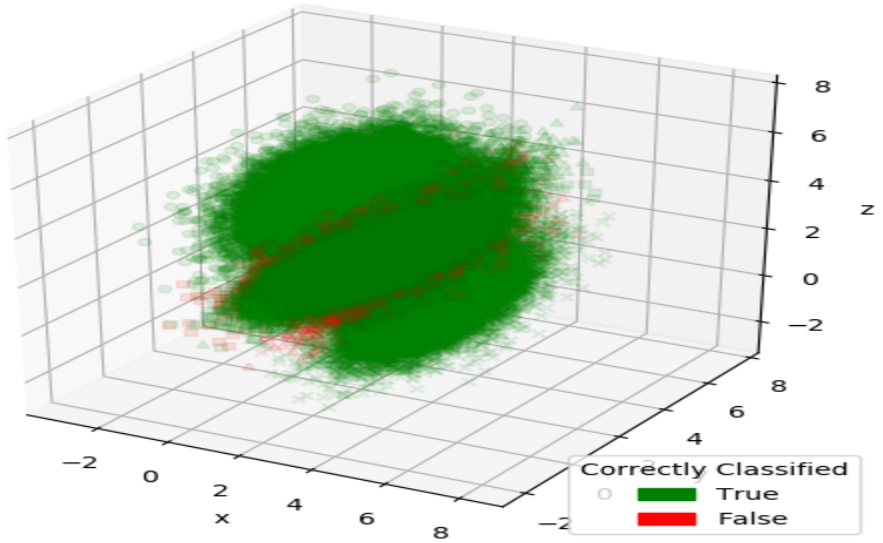
- Optimal Perceptron (minimum error) was trained on all the dataset (100 – 500 train-data)
- Theoretical optimal classifier was used using minimum probability of error classification rule.
- Theoretical optimal error = 16.8%

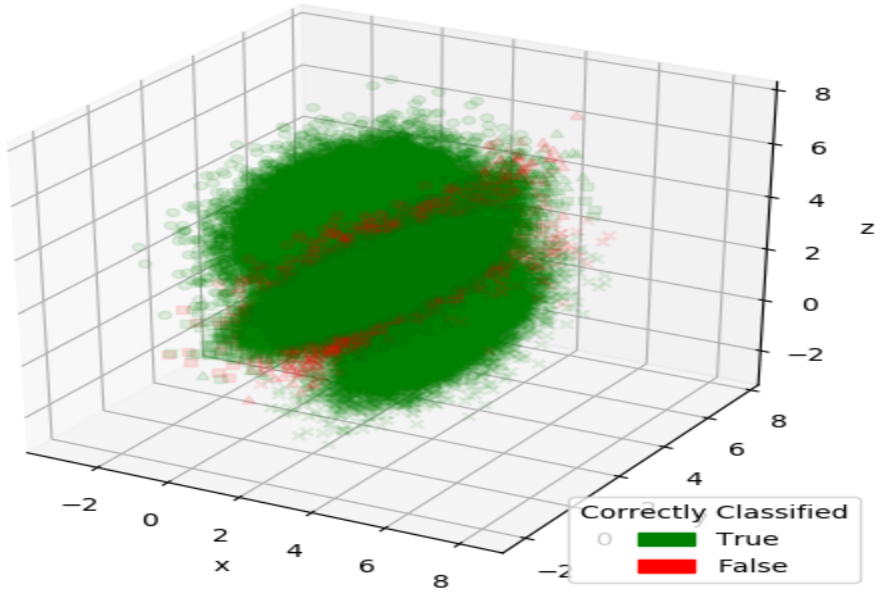
Figure below shows the classification done by the model on dataset. data - 10,000 was used to validate.

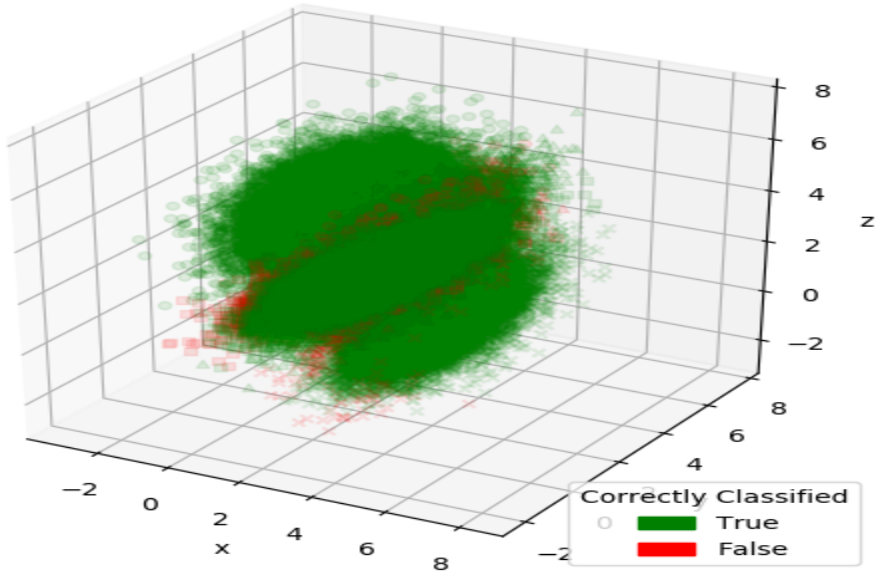
Classification with 5000-100 Train data is shown below

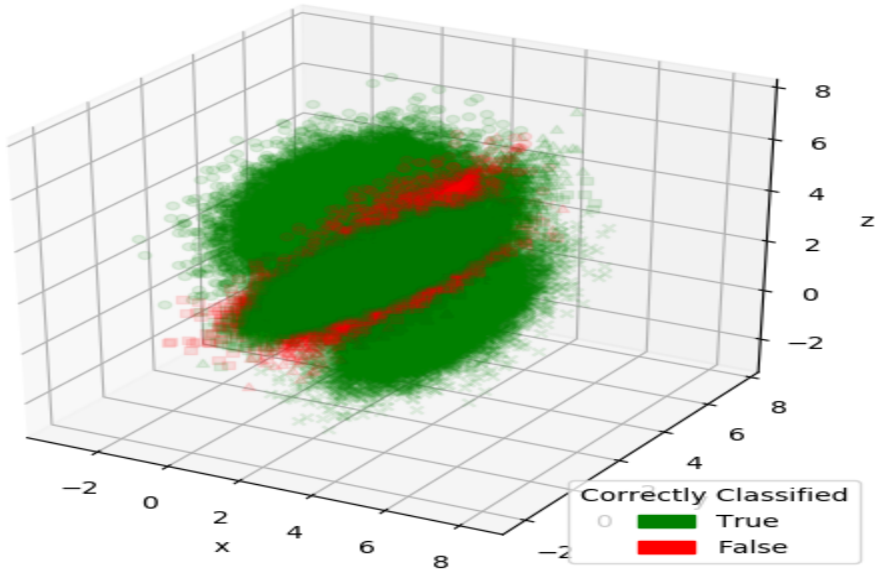

8
6
4
z
2
0
−2
8
6
4
−2
0
2
4
6
8
x
Correctly Classified
True
False


8
6
4
z
2
0
−2
8
6
4
−2
0
2
4
6
8
x
Correctly Classified
True
False

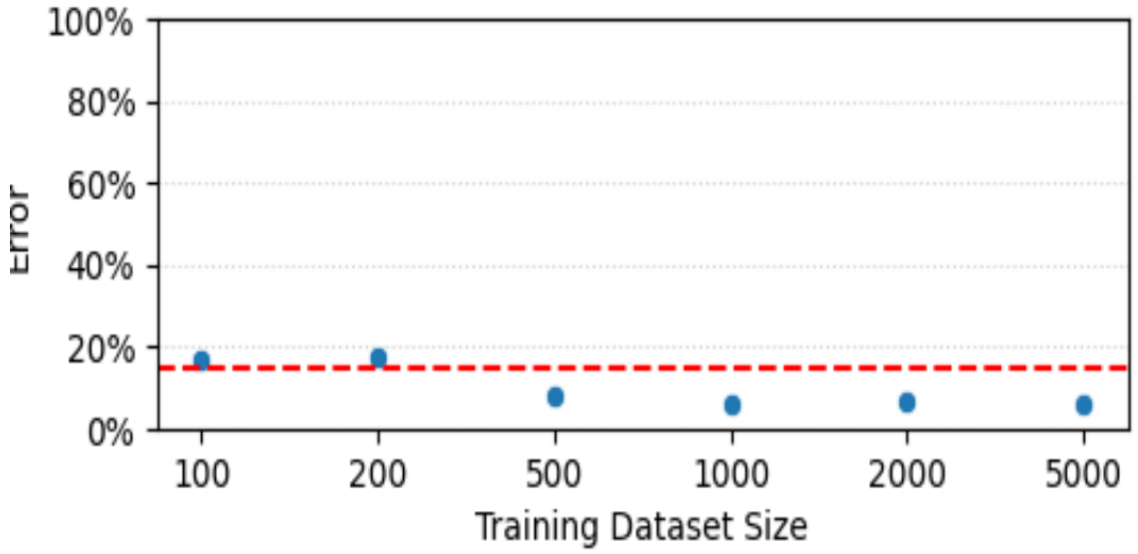

8
6
4
z
2
0
−2
8
6
4
−2
0
2
4
6
8
x
Correctly Classified
True
False


8
6
4
z
2
0
−2
8
6
4
−2
0
2
4
6
8
x
Correctly Classified
True
False





The figure below shows error v/s training data size for each MLP along with MAP classifier. (Red line = 16.8% optimal classifier)

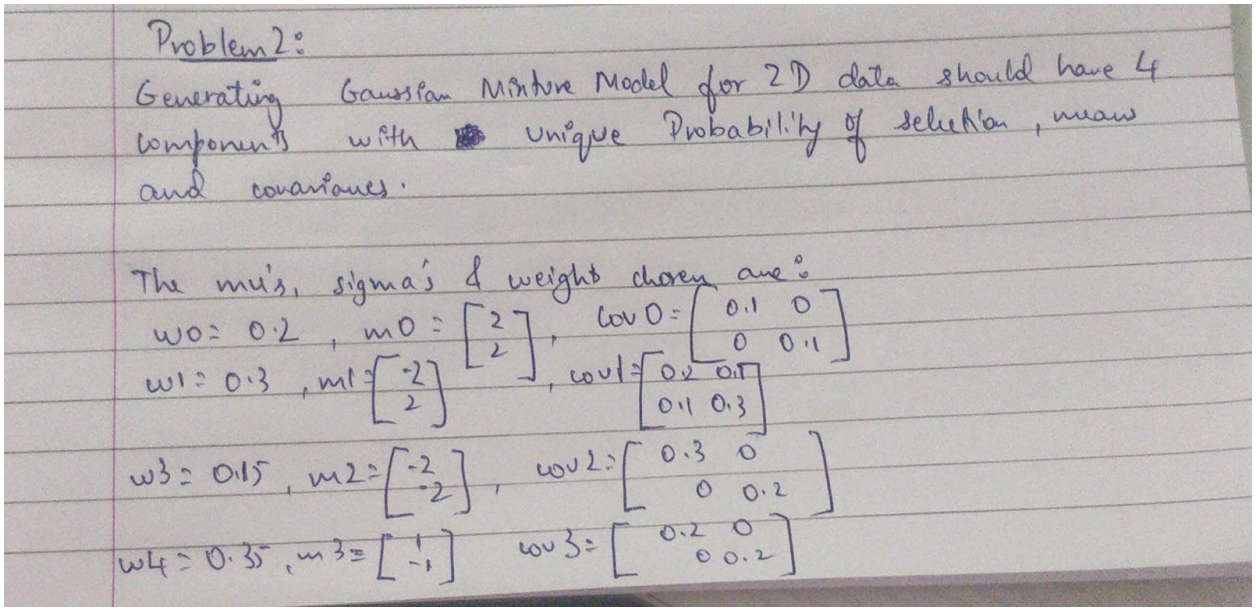


Summary of Performance

Model	Size	Probability of Error (%)
train_data_100	100	17.12
train_data_200	200	0.175
train_data_500	500	7.5
train_data_1000	1000	6.4
train_data_2000	2000	6.7
train_data_10000	10000	6.1

Packages Used: Keras, Scikit Learn, Pandas, Matplotlib

Question 2:



Problem 2:

Generating Gaussian Mixture Model for 2D data should have 4 components with unique Probability of selection, means and covariances.

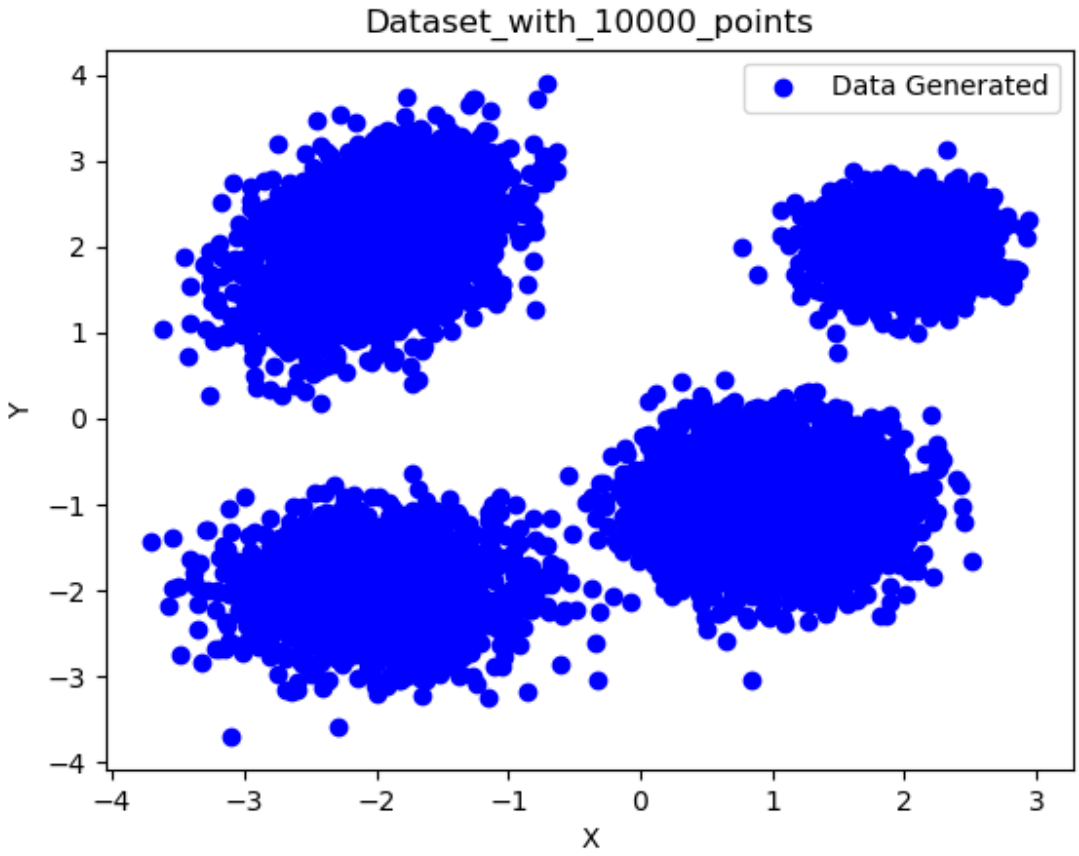
The mu's, sigma's & weights chosen are:

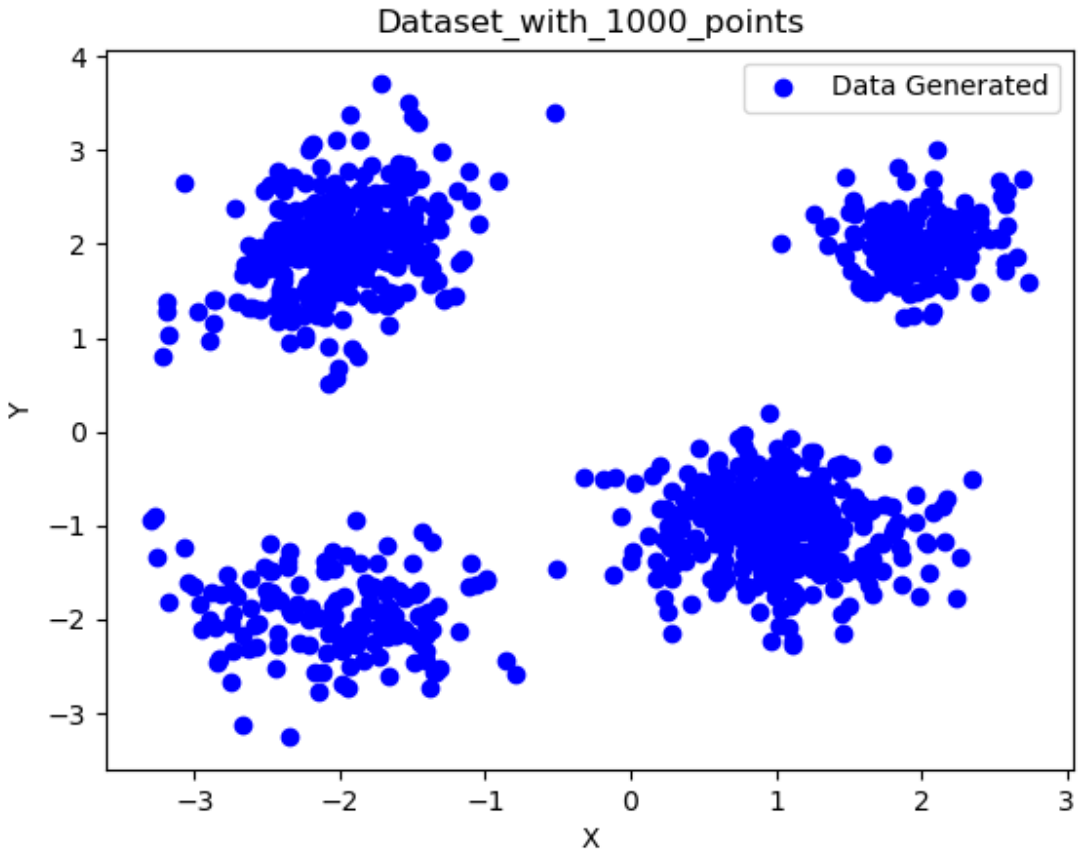
$w0 = 0.2$, $m0 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $cov0 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$

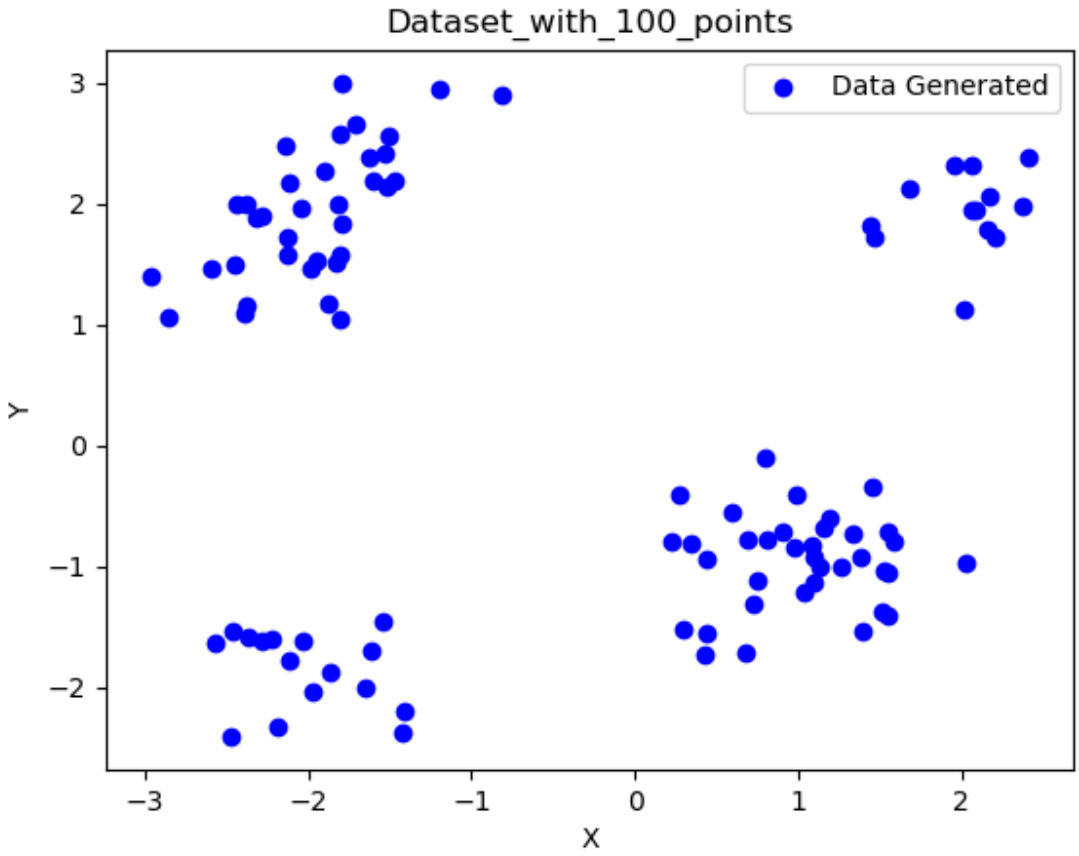
$w1 = 0.3$, $m1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$, $cov1 = \begin{bmatrix} 0.2 & 0.1 \\ 0.1 & 0.3 \end{bmatrix}$

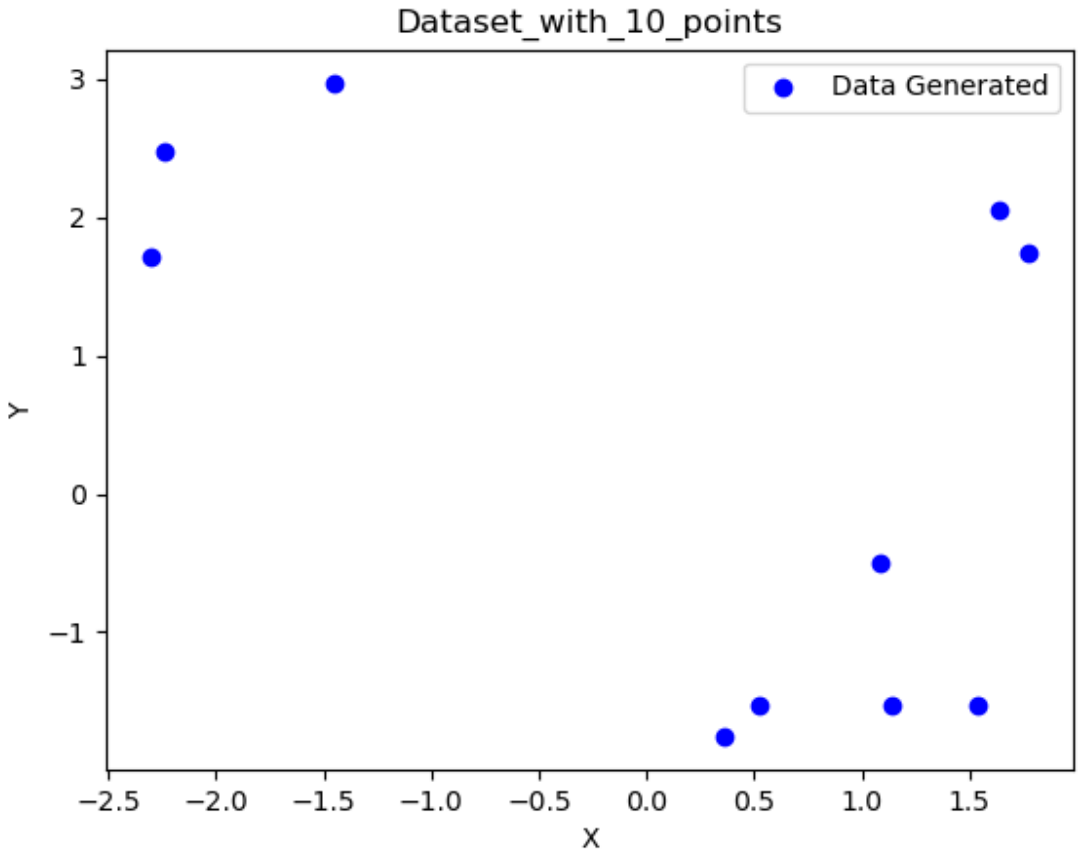
$w3 = 0.15$, $m2 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}$, $cov2 = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.2 \end{bmatrix}$

$w4 = 0.35$, $m3 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ $cov3 = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}$


Dataset_with_10000_points
Data Generated
Y
X
4
3
2
1
0
−1
−2
−3
−4
−4
−3
−2
−1
0
1
2
3


Dataset_with_1000_points
Data Generated
Y
X
4
3
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
3


Dataset_with_100_points
Data Generated
3
2
1
0
−1
−2
Y
−3
−2
−1
0
1
2
X



For each dataset using MLE and 10-fold cross validation, the gaussian components were evaluated with different model ranging 1-6.

The Bayesian Information Criteria and Akaike's Information Criteria were also calculated with -ve log likelihood.

The experiment was carried out 30 different times for each dataset. The frequency was noted down.

10 points dataset:

GMM order selection based on (-) log-likelihood	Frequency
1	30

100 points dataset:

GMM order selection based on (-) log-likelihood	Frequency
4	25
5	5

1000 points dataset:

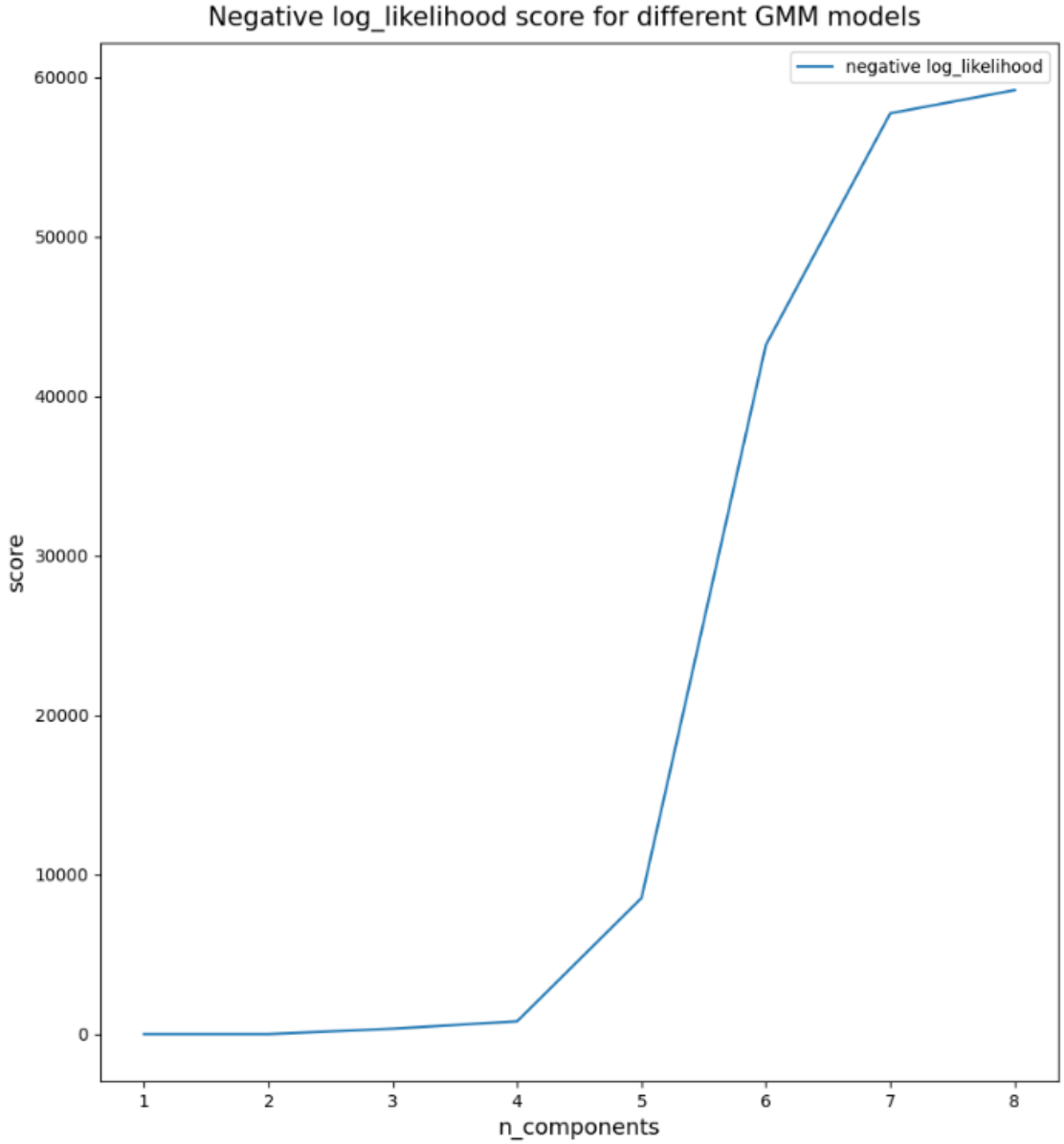
GMM order selection based on (-) log-likelihood	Frequency
4	23
5	5
6	2

10000 points dataset:

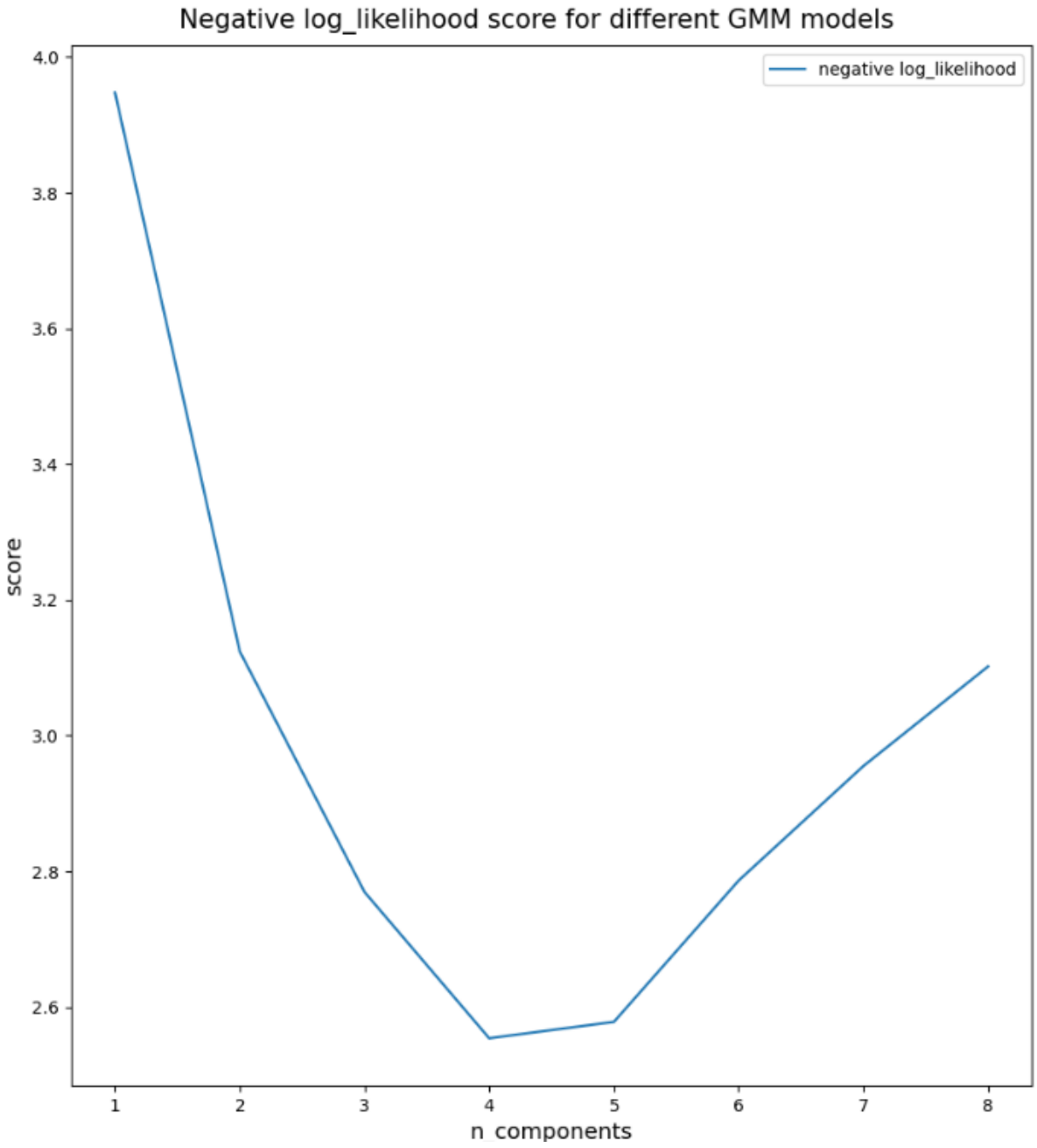
GMM order selection based on (-) log-likelihood	Frequency
4	26
5	4

The (-ve) log-likelihood vs n_component graphs for one iteration (of the 30 experimental cycles) of every dataset is plotted and shown below:

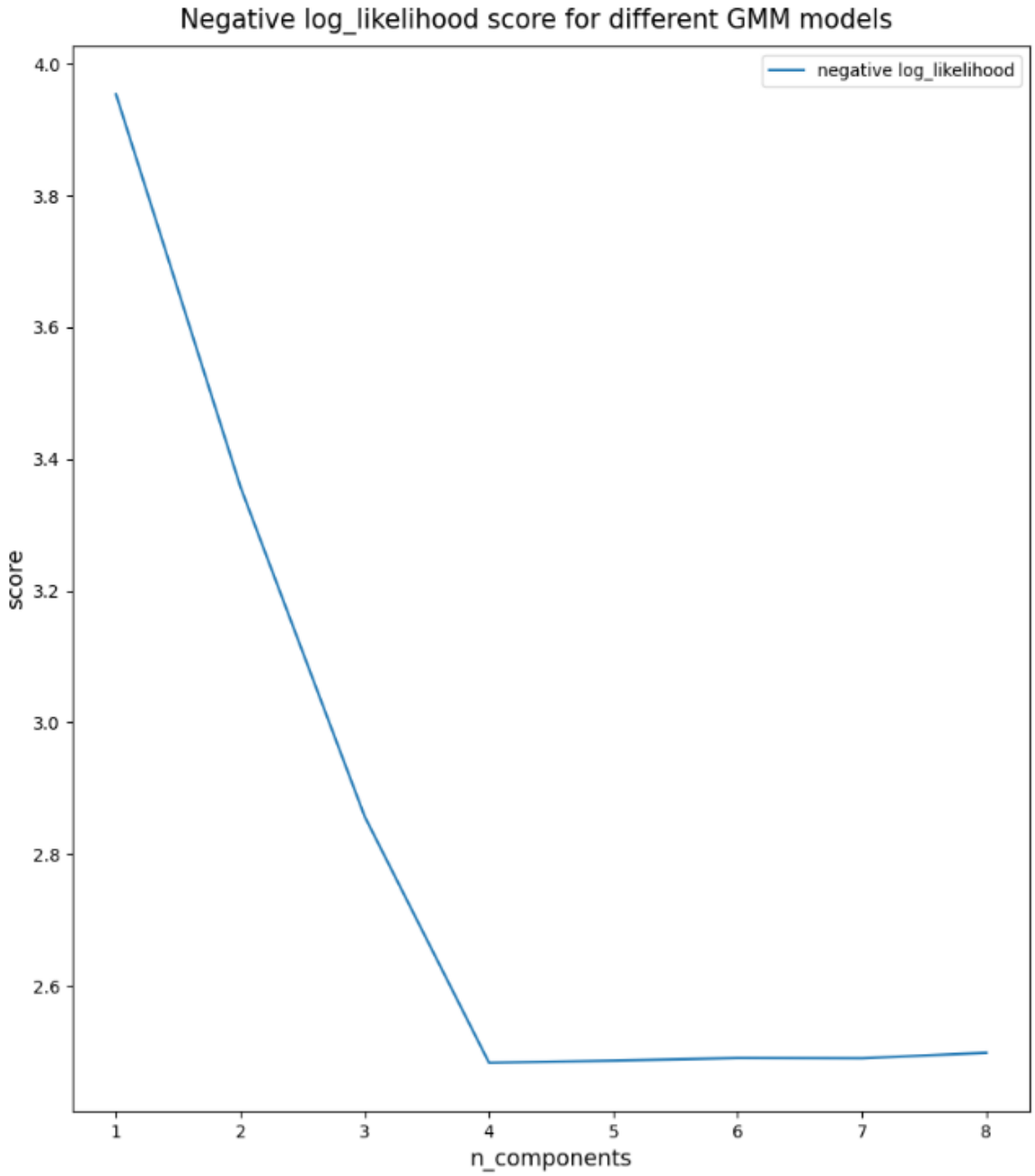
Dataset with 10 points:



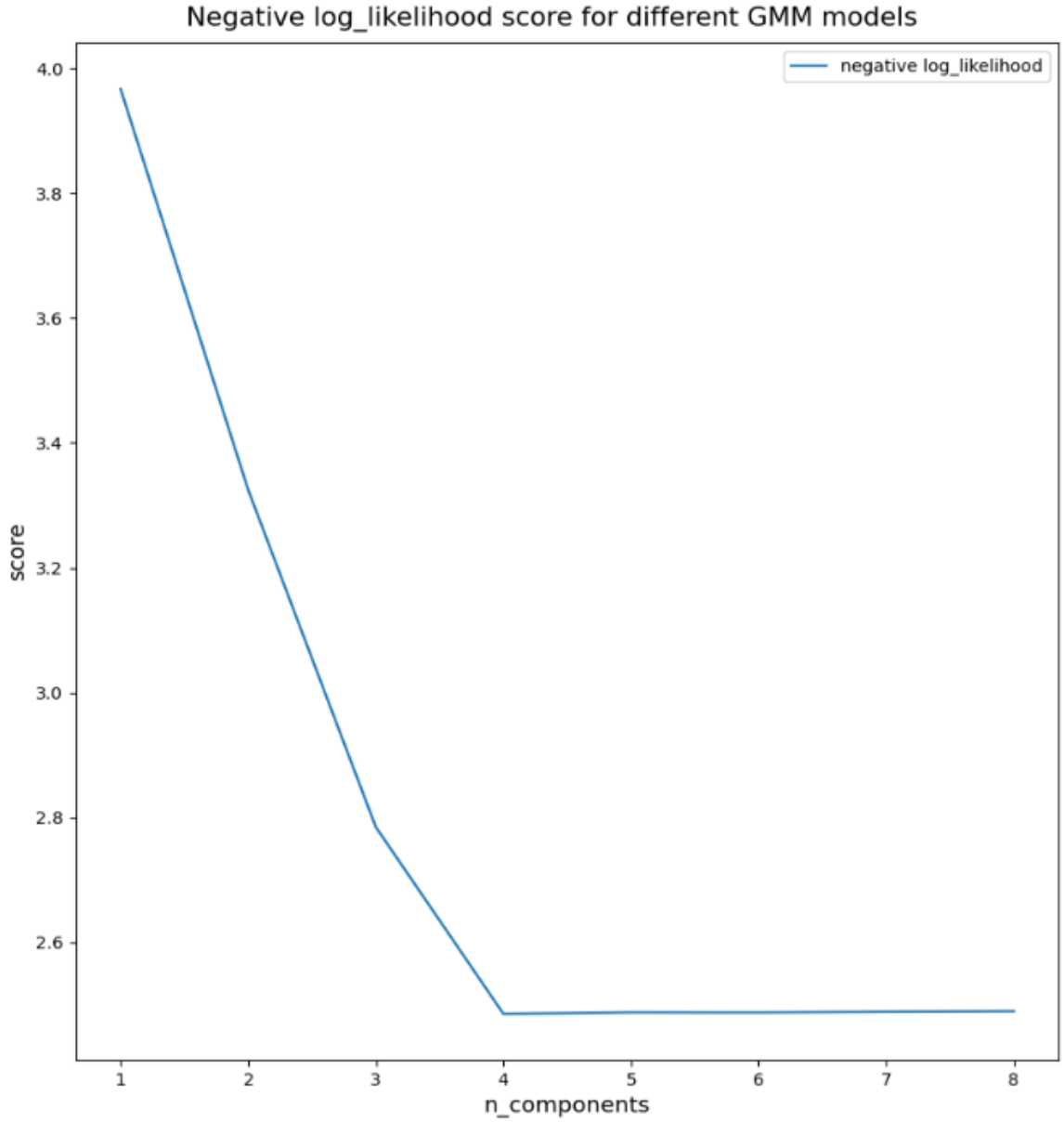
Dataset with 100 points:



Dataset with 1000 points:



Dataset with 10000 points:



Observations and Inference:

1) For data sets with 100 points, 1000 points, and 10000 points the order that is mostly chosen based on minima of -ve likelihood score is 4. There are few occurrences of 5 and 6 but the most dominant order is 4.
2) The -ve likelihood vs n_component graphs show which order produces the least -ve likelihood score and will be chosen accordingly. (best)