(1) You are given a data-set with 400 data points in $\{0,1\}^{50}$ generated from a mixture of some distribution in the file A2Q1.csv. (Hint: Each datapoint is a flattened version of a $\{0,1\}^{10\times5}$ matrix.)

i. (i) Determine which probabilisitic *mixture* could have generated this data (It is not a Gaussian mixture). Derive the EM algorithm for your choice of mixture and show your calculations. Write a piece of code to implement the algorithm you derived by setting the number of mixtures $K = 4$. Plot the log-likelihood (averaged over 100 random initializations) as a function of iterations.

ii. (ii) Assume that the same data was infact generated from a mixture of Gaussians with 4 mixtures. Implement the EM algorithm and plot the log-likelihood (averaged over 100 random initializations of the parameters) as a function of iterations. How does the plot compare with the plot from part (i)? Provide insights that you draw from this experiment.

iii. Run the K-means algorithm with $K = 4$ on the same data. Plot the objective of $K - means$ as a function of iterations.

iv. Among the three different algorithms implemented above, which do you think you would choose to for this dataset and why?

1.1)

I have used the Bernoulli model in the 4 models . each bernoulli model has different parameter p
Here the graph becomes parallel to the X-axis after running some iterations .so it means if we increase the step number the error more or less remains the same.
PI formula for bernoulli which gives probability that a model k is selected

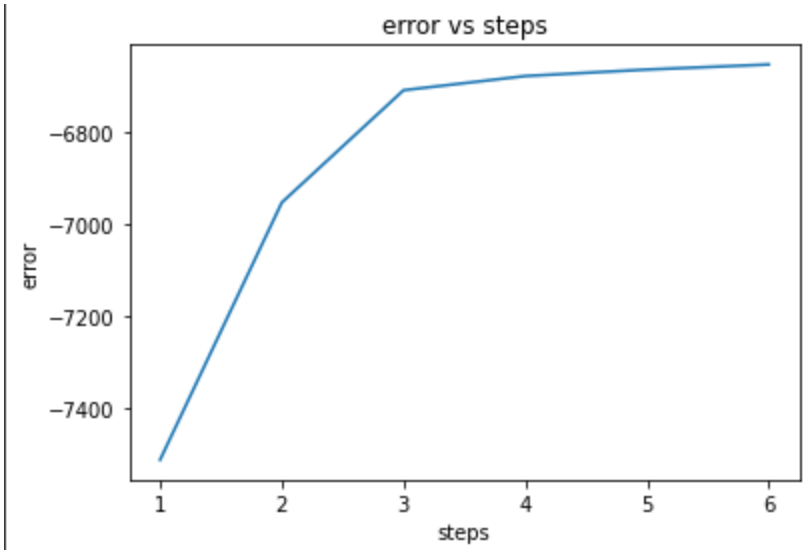
$$\pi_k = \frac{\sum_{i=1}^{n} \lambda_k^i}{n}$$

Each parameter of the model -

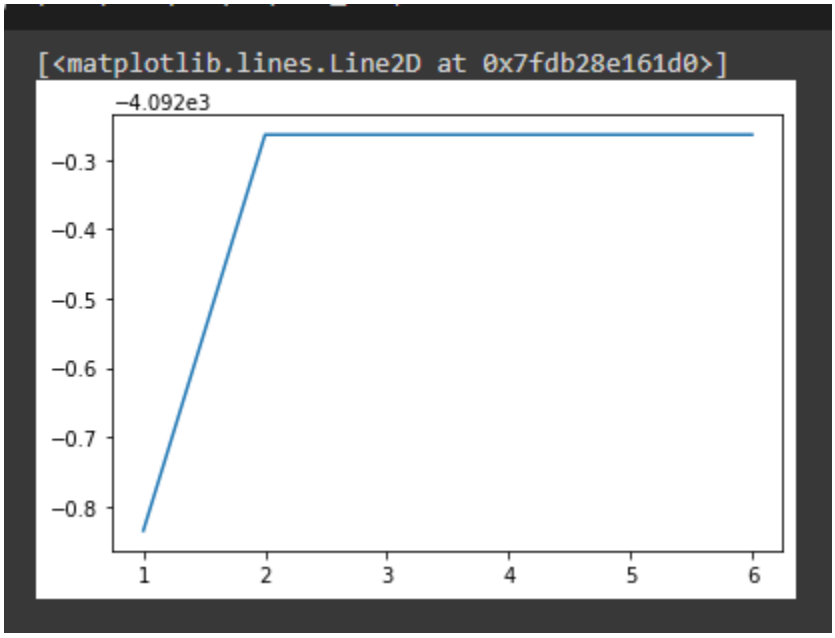
$$p_k^d = \frac{\sum_{i=1}^{n} \lambda_k^i f_d^i}{\sum_{i=1}^{n} \lambda_k^i}$$

I have made a matrix of this parameters
Each lambda value will be =

$$\lambda_k^i = \frac{\pi_k \prod_{d=1}^{50} (p_k^d)^{f_d^i} (1-p_k^d)^{(1-f_d^i)}}{\sum_{l=1}^{K} \pi_l \prod_{d=1}^{50} (p_l^d)^{f_d^i} (1-p_l^d)^{(1-f_d^i)}}$$

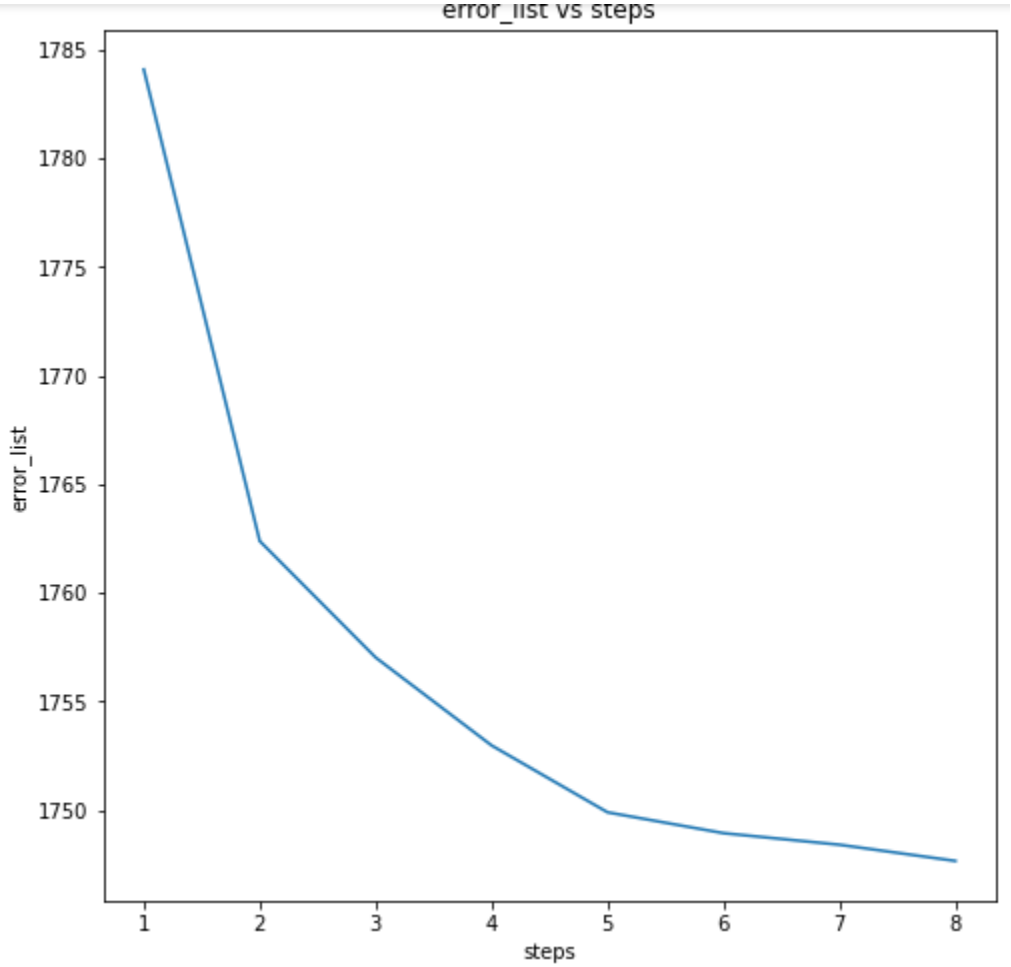


I tried to see how many points come from each of the clusters.
From cluster 1 = 122
From cluster 2= 110
From cluster 3 = 101
From cluster 4 = 67
So the points come from more or less all 4 models.
1.2)
Here we have 4 models and in each model there are 4 gaussian models with different mean and different covariance matrix . the objective function is =



Here almost all 400 points come from model 1
So it means in this case same model generates almost all data points

1.3)
Here the k means algorithm is used to determine from which cluster(model) a datapoint comes from.The error function of k means is written with the iteration number of each k means .
The graph is this =



But the exact graph changes each time as cluster assignment changes each time I run it.
I have run this for 8 steps.

Here model1 generates = 35 points
Here model2 generates = 235 points
Here model3 generates = 101 points
Here model4 generates = 29 points
1.4)
I have made an error function in which I am making an assignment matrix Z , which is basically the cluster number of each data point.
I am checking the distance of each point from the nearest mean and adding those -this serves as the error function.
With this function the bernoulli gives -1746 error
The gaussian gives = 2295 error

And the kmeans algo gives = 1746 error
It means the k means and bernoulli are more or less of the same performance but gaussian model gives worse performance that them.

(2) You are given a data-set in the file A2Q2Data_train.csv with 10000 points in $(\mathbb{R}^{100}, \mathbb{R})$ (Each row corresponds to a datapoint where the first 100 components are features and the last component is the associated y value).

i. Obtain the least squares solution $\mathbf{w}_{ML}$ to the regression problem using the analytical solution.

ii. Code the gradient descent algorithm with suitable step size to solve the least squares algorithms and plot $\|\mathbf{w}^t - \mathbf{w}_{ML}\|_2$ as a function of t. What do you observe?

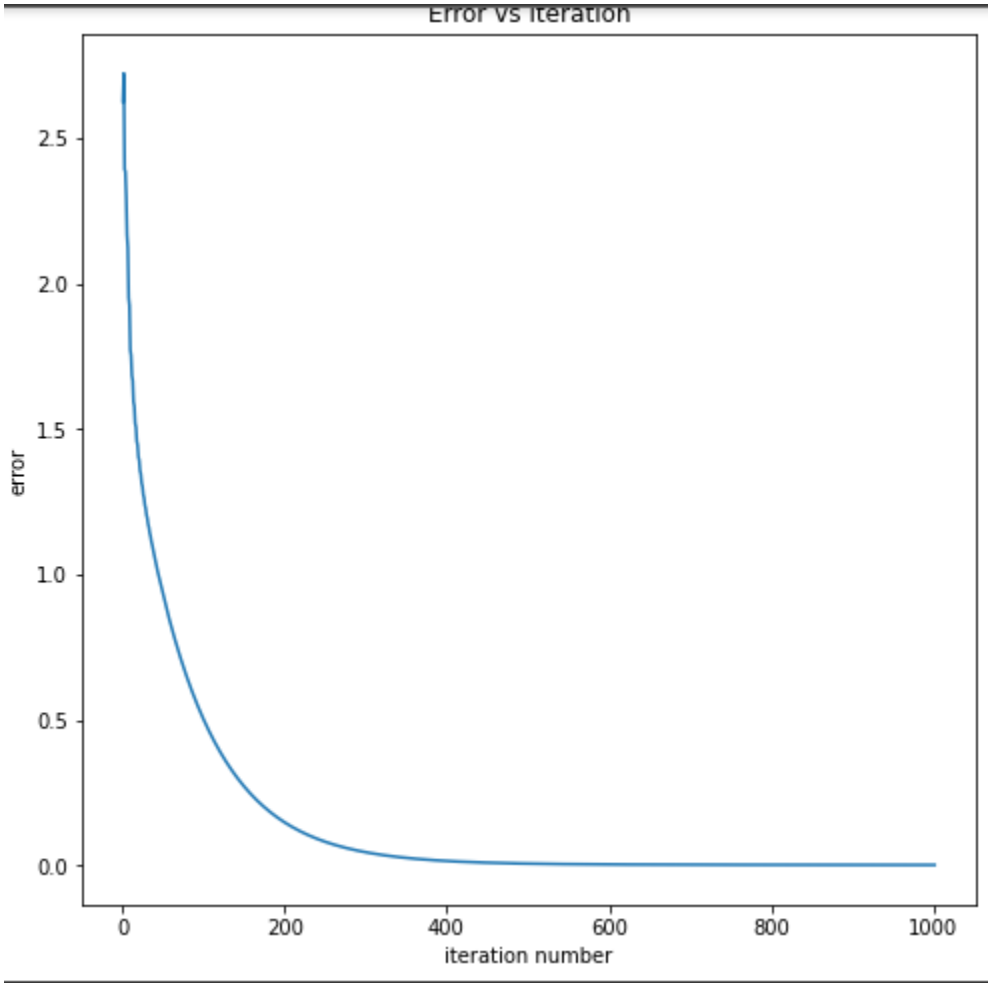
iii. Code the stochastic gradient descent algorithm using batch size of 100 and plot $\|\mathbf{w}^t - \mathbf{w}_{ML}\|_2$ as a function of t. What are your observations?

iv. Code the gradient descent algorithm for ridge regression. Cross-validate for various choices of λ and plot the error in the validation set as a function of λ. For the best λ chosen, obtain $\mathbf{w}_R$. Compare the test error (for the test data in the file A2Q2Data_test.csv) of $\mathbf{w}_R$ with $\mathbf{w}_{ML}$. Which is better and why?

2.1)
The dataset has 10000 points and each point has 100 features. A equation of line is y=mx+c form , so here I have introduced another parameter in parameter array and it becomes size 101, and in the dataframe , I have attached one column of 1's in dataframe .in every question of question 2, I have followed this.

Analytical error on test data = 185.375

2.2)
In the gradient descent algorithm I have used the learning rate as the inverse of current step size.
I have run the algorithm 1000 times.
This is the graph I found -



Where in Y axis I have plotted -

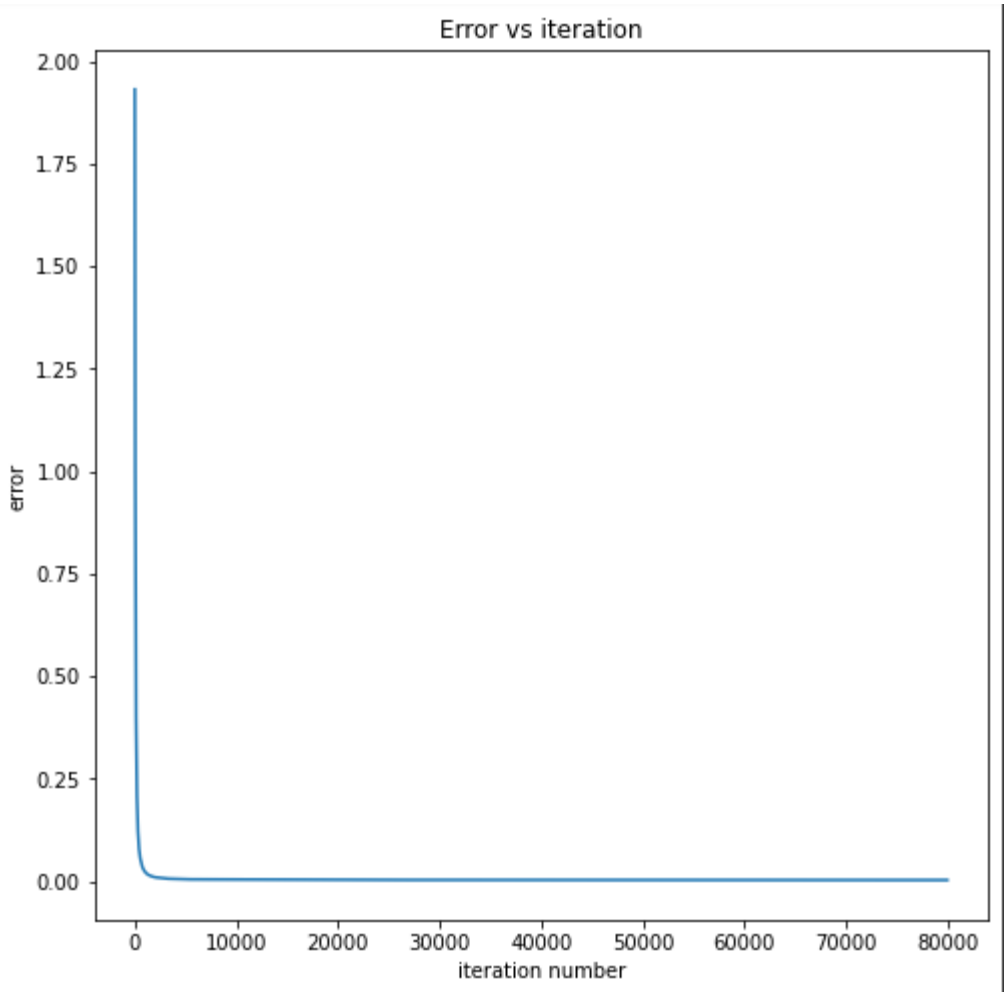
$\|\mathbf{w}^t - \mathbf{w}_{ML}\|_2$

In X axis I have plotted the iteration number .
I have plotted the graph for 1000 steps.
The parameter obtained from this gives a error of = 185.055 on the test data

2.3)

In stochastic gradient descent , I have run it for batch size of 100.
In test data = 183.925
In training data the error is =397.122

The graph is =



Where in Y axis I have plotted -

$\bar{\|\mathbf{w}^t - \mathbf{w}_{ML}\|_2}$

In X axis I have plotted the iteration number .

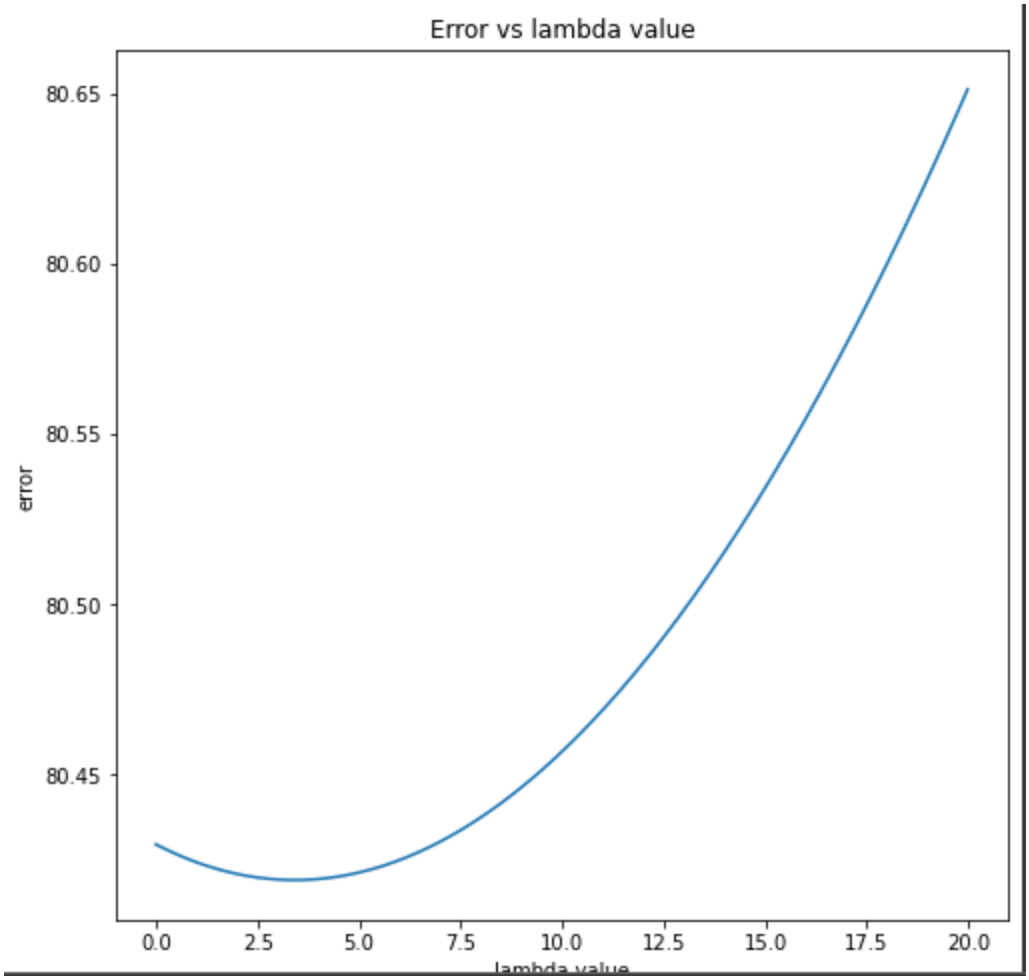
2.4)

In ridge regression I have divided the train data into 80% and 20% and trained the data and found the parameter from the 80% and I am printing the error value on 20% of the training data . I am doing these steps for multiple lambda values. The lambda value that is giving the minimum error is the best lambda .

In this case the lambda value is = 3.42

The test data error is =184.670

The cross validation graph which shows error on y axis as a function of lambda value is =


Error vs lambda value
80.65
80.60
80.55
80.50
80.45
error
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
lambda value