CS5691: Pattern Recognition and Machine Learning
Assignment 2
Course Instructor : Arun Rajkumar

Name: Amishi Panwar
Roll No. : CS22M009

(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} 10×5 matrix.)
i. Determine which probabilistic 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.

Probabilistic mixture could be Binomial Distribution, which is performing bernoulli trial 50 times for each data point. Result would be 0, 1.
As per the question, assume 4 mixtures.
Parameters will be :
Pi for each cluster, giving probability to select the cluster.
P probability of heading up 1 in each cluster.

PFA the derived algorithm and formulas used-

Derivation

considering binomial mixtures

$$L(p_i, \pi_i, x_i) = \prod_{i=1}^{n} f(x_i; p_1, p_2 \dots p_n; \pi_1, \pi_2 \dots \pi_n)$$

$$= \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k f(x_i; p_k)$$

For binomial

$$L(p, \pi, x) = \prod_{i=1}^{n} \sum_{k=1}^{k} \left(\pi_k p^{x_i} (1-p)^{50-x_i} \right)$$

taking log both sides.

$$\log(L(p, \pi, x)) = \log\left(\prod_{i=1}^{n} \sum_{k=1}^{k} \left(\pi_k p^{x_i} (1-p)^{50-x_i} \right) \right)$$

$$= \sum_{i=1}^{n} \log\left(\sum_{k=1}^{k} \left(\pi_k p^{x_i} (1-p)^{50-x_i} \right) \right)$$

By jensen's inequality

$$\log(L(\theta)) = \sum_{i=1}^{n} \left(\log\left(\sum_{k=1}^{k} \lambda_k^i \pi_k p^{x_i} (1-p)^{50-x_i} \right) \right)$$

here $\forall i \sum_{k=1}^{k} \lambda_k^i = 1 \quad 0 \le \lambda_k^i \le 1$

$$\log(L(\theta)) \ge \sum_{i=1}^{n} \sum_{k=1}^{k} \lambda_k^i \log\left(\frac{\pi_k p^{x_i} (1-p)^{50-x_i}}{\lambda_k^i} \right)$$

For maximizing p,

differentiate w.r.t p, taking other parameter const.

$$\text{eq}^n \log(L(\theta)) = \sum_{i=1}^{n} \sum_{k=1}^{k} \lambda_k^i \left[\log \pi_k + x_i \log p + (50-x_i) \log(1-p) - \log \lambda_k^i \right]$$

differentiate w.r.t p.

$$\log(L(\theta)) \ge \sum_{i=1}^{n} \sum_{k=1}^{k} \left(\frac{\lambda_k^i x_i}{p} - \frac{(50-x_i)\lambda_k^i}{1-p} \right) = 0$$

$$\sum_{i=1}^{n} \left(\frac{\lambda_k^i x_i}{p_k} - \frac{(50-x_i)\lambda_k^i}{1-p_k} \right) = 0 \quad \forall k \in K$$

$$\sum_{i=1}^{n}\left((1-P_k)\left(\partial_k^i x_i\right) - P_k\left(50-x_i\right)\partial_k^i\right)=0$$

$$\sum_{i=1}^{n}\left(\partial_k^i x_i - P_k\partial_k^i x_i - 50P_k\partial_k^i + P_k x_i\partial_k^i\right)=0$$

$$\sum_{i=1}^{n}\frac{\partial_k^i x_i}{50\,\partial_k^i} = P_k$$

$$\boxed{P_k = \sum_{i=1}^{n}\left(\frac{\partial_k^i x_i}{50\,\partial_k^i}\right)}$$

For maximizing π, differentiate w.r.t. π, taking other parameters constt.

$$\log(L(\theta)) \geq \sum_{i=1}^{n}\sum_{k=1}^{k}\partial_k^i \log\left\{\frac{\pi_k\, p^{x_i}\,(1-p)^{50-x_i}}{\partial_k^i}\right)$$

Differentiation w.r.t π

$$\sum_{i=1}^{n}\sum_{k=1}^{k}\frac{\partial_k^i}{\pi_k} = 0$$

For every k.

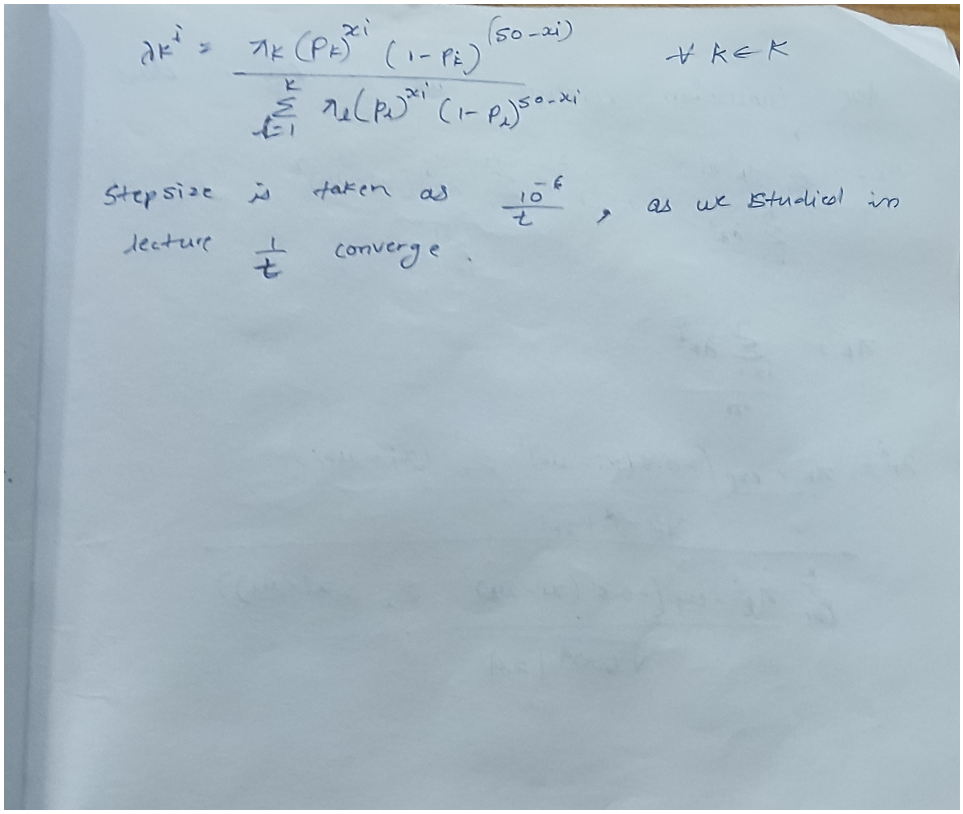
$$\sum_{i=1}^{n}\frac{\partial_k^i}{\pi_k} = 1$$

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

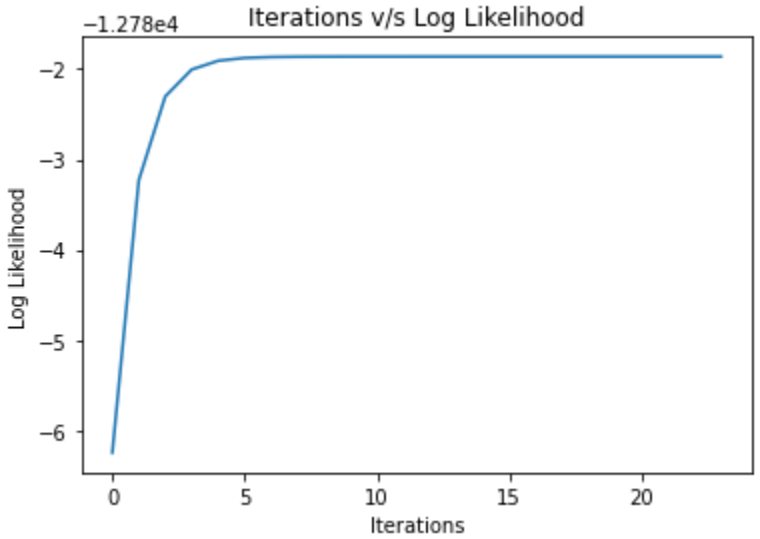
$$\boxed{\pi_k = \frac{\sum_{i=1}^{n}\lambda_k^i}{n}}$$

Find ∂_k^i using Bayes theorem

$$\partial_k^i = \frac{P(k^{th}\text{ cluster})\cdot P(x_i / k^{th}\text{ cluster})}{\text{Total probability of } x_i}$$



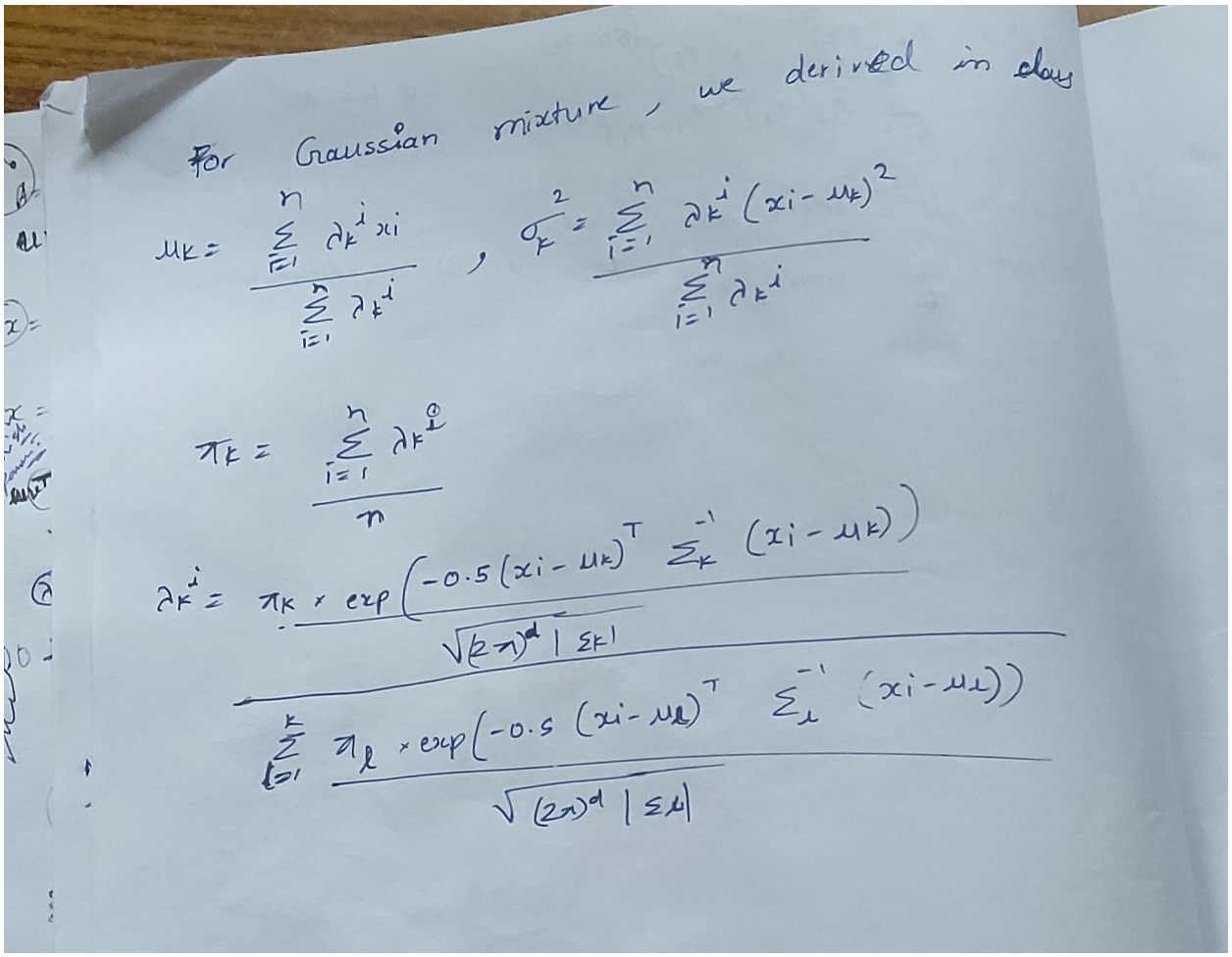
Graph obtained-



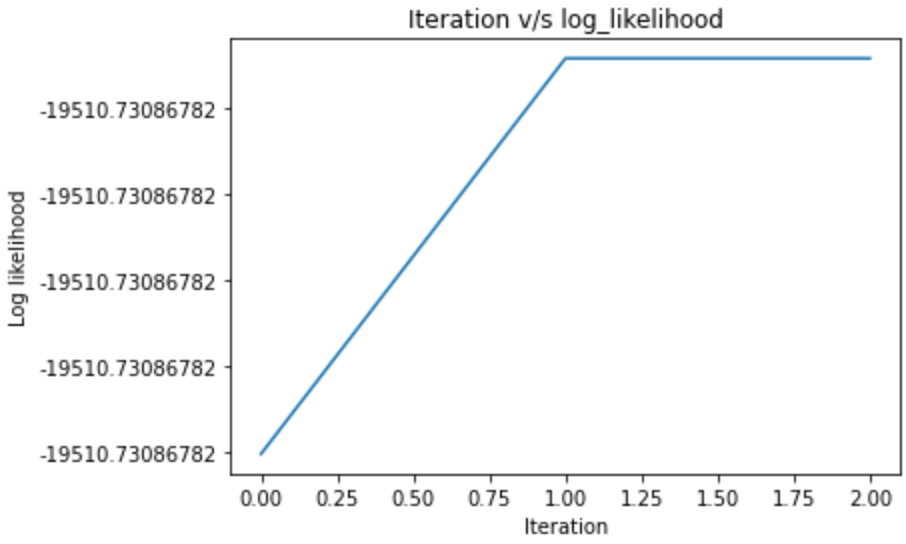
From the graph we can see that log likelihood averaged over 100 iterations is increasing till convergence.

ii. Assume that the same data was in fact 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.

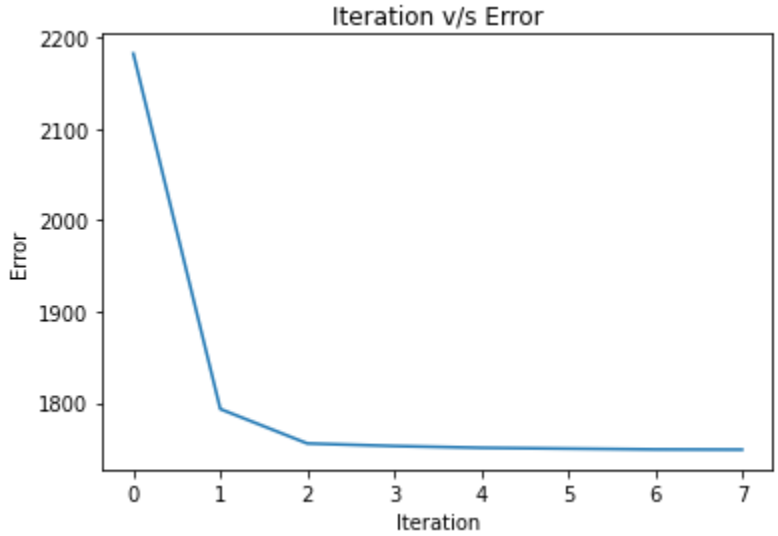
Assuming data generated from 4 gaussian mixtures.
Parameters will be:
Pi for each cluster, giving probability to select that mixture.
Mean and sigma of each of the 4 mixtures, assuming gaussian distribution.
Formulas for which are shown below, which are derived in the class-



From the graph obtained we can see that log likelihood is increasing with iterations till convergence.
The value of maximum log likelihood is more in case of binomial distribution, so chances of mixtures being binomial is more than that of gaussian



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



We can see from graph obtained that error is decreasing with iterations till convergence.

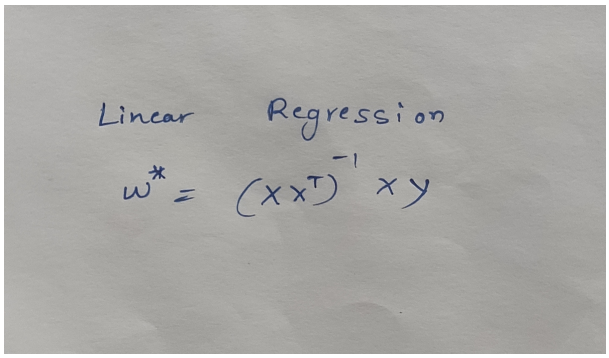
iv. Among the three different algorithms implemented above, which do you think you would choose for this dataset and why?

Three algorithms used are-
(i) assuming binomial distribution
(ii) Gaussian Distribution
(iii) Lloyds Algorithm for clustering

From the binomial and gaussian mixtures, binomial is giving maximum log likelihood, so I would choose binomial over gaussian. Lloyds gives us hard clusters but EM gives us soft clusters, that is the probability that data can be obtained from other clusters will be low but not exactly zero as in clustering.
Here I think hard clustering won't benefit much, therefore the algorithm we assumed in 1st part(for Binomial Distribution) is best for this dataset.

(2) You are given a data-set in the file A2Q2Data train.csv with 10000 points in (R100 , 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 wML to the regression problem using the analytical solution.

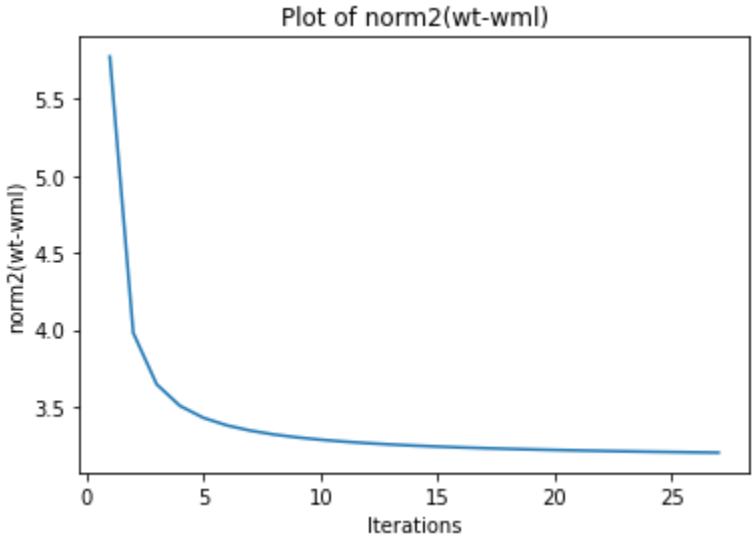


Wml =

[-7.84961009e-03 -1.36715320e-02 -3.61656438e-03 2.64909160e-03
  1.88551446e-01 2.65314657e-03 9.46531786e-03 1.79809481e-01
  3.73757317e-03 4.99608944e-01 8.35836265e-03 4.29108775e-03
  1.42141179e-02 3.94232414e-03 9.36795890e-03 -1.12038274e-03
  3.35727500e-03 1.16152212e-03 -9.40884707e-03 -2.45575476e-03
 -1.17409629e-02 -1.01960612e-02 7.95771321e-03 -1.00574854e-02
  6.04882939e-03 -4.67345192e-03 -3.09091547e-03 8.14909193e-03
  1.20264599e-02 -6.82458163e-03 -8.65405539e-03 9.86273479e-04
  4.92968011e-03 5.99772461e-03 -1.34667860e-02 1.07075729e-03
  1.32745992e-02 -1.14148742e-02 -2.01056697e-02 5.85096240e-01
  4.94483247e-04 -7.86666920e-04 -2.71926574e-03 -9.54021938e-03
 -5.44161058e-03 9.80679209e-03 -6.72540624e-03 -4.45414276e-04
  6.98516508e-03 3.16138907e-02 4.51763485e-01 -8.75221380e-03
  2.55167390e-03 4.24921150e-03 2.89847927e-01 7.03723255e-03
 -1.95796946e-03 1.41523883e-02 -1.06508170e-02 7.72743903e-01
 -5.67126044e-03 -6.30026188e-04 6.50943015e-03 -4.84019165e-03
  4.63832329e-03 4.54887177e-03 -2.99475114e-03 8.38781696e-03
 -2.47558716e-03 9.00947922e-04 1.14713514e-03 -1.87641345e-03
 -1.05175760e-02 -9.31304110e-03 -1.23550002e-03 5.97797559e-01
 -4.78625013e-03 -1.13727852e-02 2.88477060e-03 8.48999776e-01
 -1.08924235e-02 2.26346489e-03 -1.38099800e-03 -6.35934691e-03
  5.83784109e-03 5.69286755e-03 5.35566859e-03 -8.20616315e-03
  1.29884015e-02 -2.30575631e-03 -1.22263765e-04 8.66629171e-03
 -4.29446300e-03 5.69510898e-03 7.55483353e-03 -9.43540843e-03
  1.82905446e-02 -1.16998887e-03 -2.61599136e-03 -8.58616114e-03]

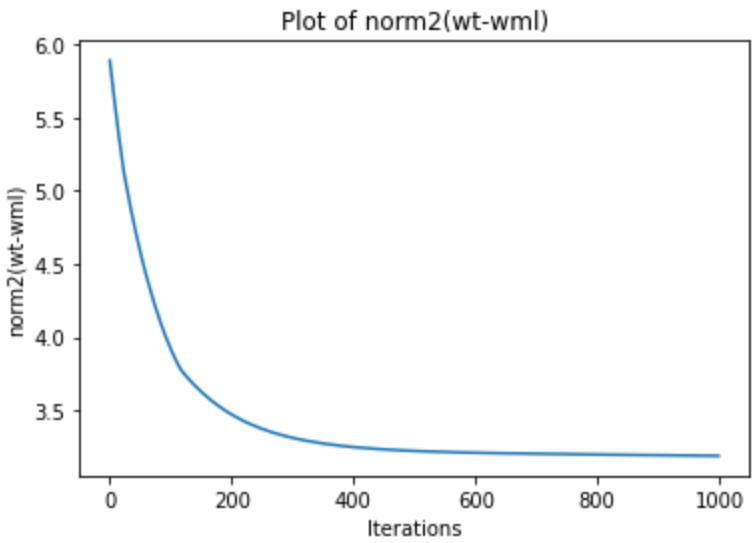
ii. Code the gradient descent algorithm with suitable step size to solve the least squares algorithms and plot ||wt − wML||2 as a function of t. What do you Observe?

According to the graph obtained, we can see that the value ||wt − wML||2 is decreasing with each iteration and is going to converge after a few iterations.
The value decreases with a rapid rate initially and then decreases gradually.



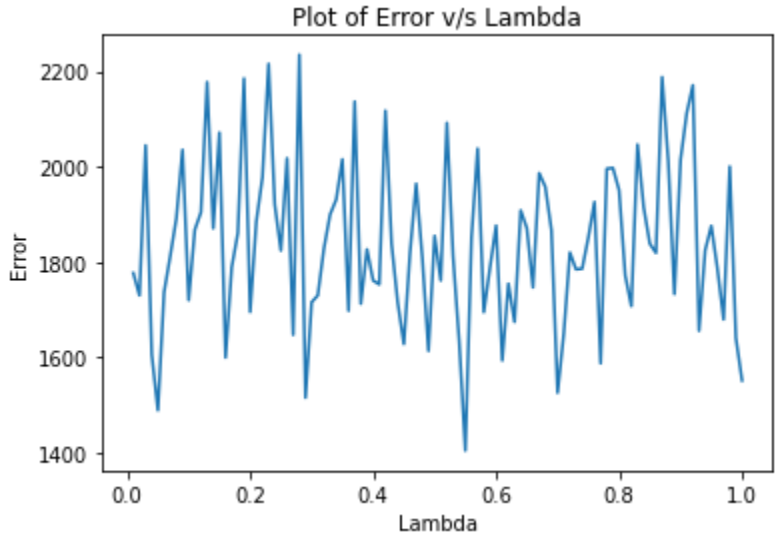
iii. Code the stochastic gradient descent algorithm using batch size of 100 and plot ||wt − wML||2 as a function of t. What are your observations?

W_stochastic =
[0.14456333 0.10579805 0.47880347 0.29382437 -0.41353698 0.08812265
-0.30300403 0.01936261 -0.07796245 0.47225258 -0.37812255 -0.15320068
-0.10865749 0.49609686 -0.1834782 0.56387057 0.00765658 -0.10424875
0.07798685 0.36763144 0.10320211 0.56006397 -0.22796925 -0.04480983
0.23333075 0.47106388 0.53264774 0.11594303 0.54481444 -0.12942557
-0.30355328 0.13445475 0.13479389 -0.13724751 -0.06546044 0.194664
0.35007982 0.31076162 -0.03854897 -0.16330761 0.5653606 0.37346641
0.0371909 0.3393196 -0.22911216 0.0336624 0.19908922 -0.17631016
0.21211812 -0.24420792 0.43008878 0.19172216 0.20604714 0.37522085
-0.3773873 -0.09275595 0.22418519 -0.28193672 0.06535784 0.5544575
0.3952764 0.30506615 0.14588438 0.0537052 0.38084198 0.37299631
-0.13794187 -0.28185981 0.39507395 0.56980194 0.00927276 0.27608272
0.29020341 0.44919183 0.32588349 0.29675802 0.29988851 -0.31162917
-0.37657592 -0.00199848 -0.3934353 0.24663957 -0.22725837 -0.26801811
0.44253498 0.37609916 -0.21369422 -0.10902035 -0.35895917 -0.16427946
-0.01408224 -0.20491747 0.43870212 -0.07566819 0.32073769 0.44315391
0.57113735 0.50571652 0.56847595 0.27400076]



From the graph we can see that the value ||wt − wML||2 is decreasing with each iteration till convergence.
But we can see that the initial drop is more in gradient descent as compared to the stochastic descent, this is because error decreases slowly in stochastic descent because of batches while in gradient descent we take whole data at a time and hence more drop in value.
For large datasets stochastic gradient descent is preferred.

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 wR . Compare the test error (for the test data in the file A2Q2Data test.csv) of wR with wM L. Which is better and why?



After testing for different values of lambda, we get min error for lambda = 0.55

And wRidge for this lambda =

```
[-7.40600431e-03 -7.01002593e-03 -1.76500381e-03  9.99605161e-04
  1.83663080e-01  8.58602716e-05  2.26759721e-03  1.74896711e-01
  8.49805595e-03  5.00803987e-01  1.13711513e-02  7.47997328e-03
  1.92201093e-02  9.68689688e-03  6.31170807e-03 -1.93691914e-03
  2.12282376e-04  1.36271769e-03 -1.28089402e-02 -3.09584969e-03
 -9.56380967e-03 -1.07419706e-02  5.77851414e-03 -1.21863481e-02
  1.04377738e-02 -3.19800016e-03  3.25680096e-03  5.55907760e-03
  1.32753118e-02 -6.73449691e-03 -8.36628306e-03 -5.99284438e-04
  8.11169351e-03  9.25333122e-03 -1.23424426e-02 -3.10561588e-03
  1.17032303e-02 -5.27536627e-03 -1.95893098e-02  5.89035305e-01
 -5.88075446e-03  5.39070508e-04 -1.34847111e-03 -8.95108706e-03
 -2.18271953e-03  8.29466055e-03 -1.08812182e-02 -2.97527808e-03
  6.68751359e-04  3.94190937e-02  4.50424208e-01 -1.33513950e-02
  4.95857992e-03  2.79323626e-03  2.95313903e-01  9.46842710e-03
 -2.02253105e-03  5.39745471e-03 -1.57917405e-02  7.73636777e-01
 -8.69152331e-03 -3.28774910e-03  6.58988358e-03 -1.03109586e-03
  1.64579654e-03  6.36103125e-03 -2.74124759e-03  3.58078068e-03
 -5.72326910e-03 -3.57321767e-03  5.04351456e-03  1.44027179e-03
 -1.48242158e-02 -1.43460597e-02 -3.85810110e-03  5.93043886e-01
 -4.95186423e-03 -1.33839551e-02 -9.19213095e-04  8.48315075e-01
 -8.03049793e-03  1.00654111e-03 -9.95813417e-04 -1.07446990e-03
  6.29042396e-03  9.66348926e-03  9.73631265e-03 -8.96964770e-03
  1.10173052e-02  2.31816670e-03  1.55826329e-03  1.26523405e-02
 -7.97628764e-04  3.02406437e-03  7.22491032e-03 -1.28698442e-02
  1.60783398e-02  6.60628303e-03  3.07104583e-04 -8.44145273e-03]
```

Test Error with wR = 185.9049574016168
Test Error with wml = 185.36365558489368

The error in both the cases is almost same that is w that we obtained by ridge is close to wml optimal.