

Ans: 

Model assummed: Two level estimation model with:

1. Level one choosing out of mixture of bernoulli and
2. the other level is generating the data from that bernoulli.

Now, each of the bernoulli distribution functions as toss of 50 coins to generate each data.

Choice of Distribution

Distribuion 4

Distribution 3

Distribution 2

Distribuion 1

ha

For our notation, we denote the probabilities of each coins to fall on heads to be: pkd

Where, k ranges from 0 to 3(corresponding to the 4 clusters). And d ranges from 0 to 49 (corresponding each dimension for every data)

For instance, the distribution 1 comprises of (p11 to p150) and varying the subsctript k for every other distributions.

Now, we have to obtain:

1. Probability of the generation of the data: P(data)
2. Probability of a particular distribution gets chosen: P(Zi=k) = πk
3. Probability of each of the bernoulli variables: pkd
4. Also the constants that we will be imposing in modified log likelihood, for each data in every cluster: λki

We begin with defining our likelihood:

L(parameters;data) = P(data/parameters)

= Πi=1400 Σk=14 πkΠd=150 (pkd)(data[i][d])(1-pkd)(1-data[i][d])

Now, to estimate parameters, we take the logarithm of the likelihood.

Log(L(parameters;data)) =

Σi=1400Log(Σk=14 πkΠd=150 (pkd)(data[i][d])(1-pkd)(1-data[i][d]))

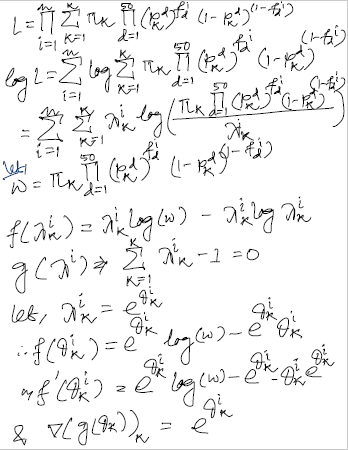
Hereafter, we introduce λki, to be able to get a lower bound which we can then differentiate partially and obtain estimations for parameters.

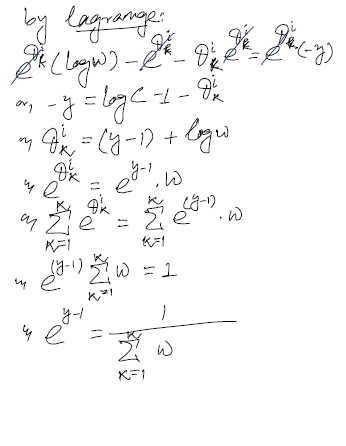
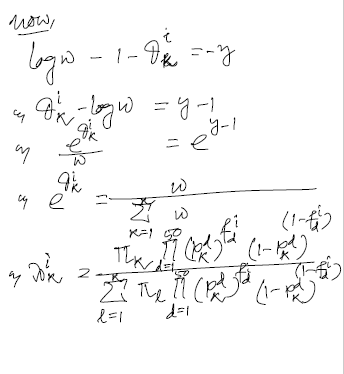
Therefore, Modified Log(L(parameters;data)) =

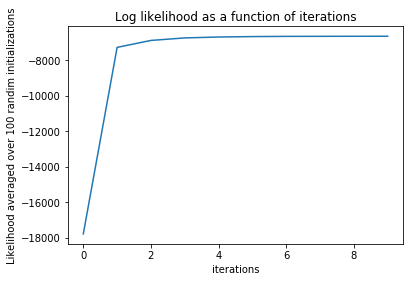
Σi=1400Log(Σk=14 λki (πkΠd=150 (pkd)(data[i][d])(1-pkd)(1-data[i][d]))/ λki)

= Σi=1400 Σk=14 λki Log(πk (Πd=150 (pkd)(data[i][d])(1-pkd)(1-data[i][d]))/ λki)

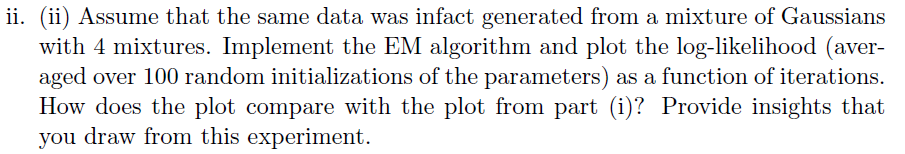
Differentiating this above equation partially with respect to λki fixing pkd & πk we obtain the equations. Here is the complete derivation as follows:





The plot of log likelihood as a function of iterations is presented above.



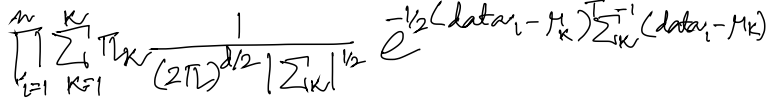
Ans: 

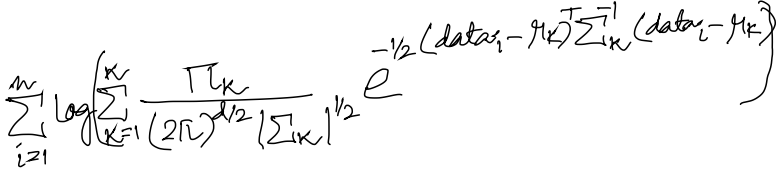
We use a multivariate gaussian distribution to generate the data in the following model.

As per our assumption, the individual mixtures has means as: (µ1 ,µ­­2­,µ3 & µ4)

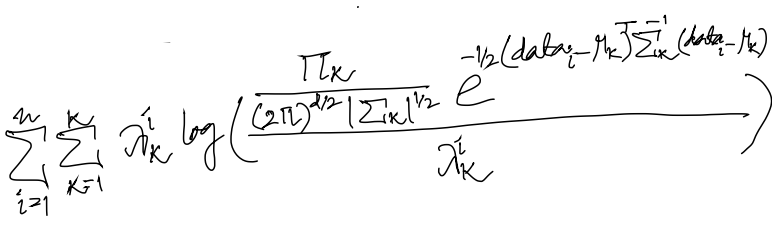
Also the covariances are assummed to be: (Σ1, Σ2, Σ3, Σ4)

The likelihood is as follows:

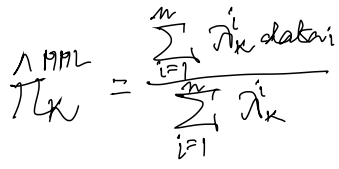
Therefore the Log Likelihood becomes:

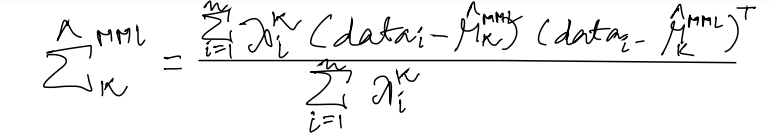


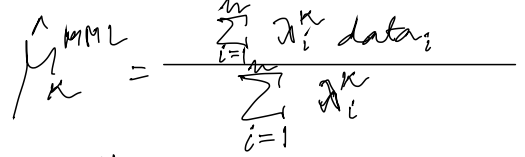
Converting it into modified log likelohood is:

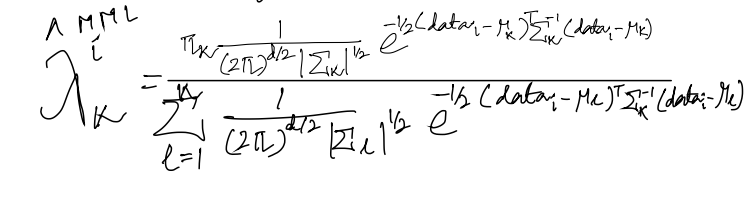


Partially differentiating the modified log likelihood to obtain different parameters gives:

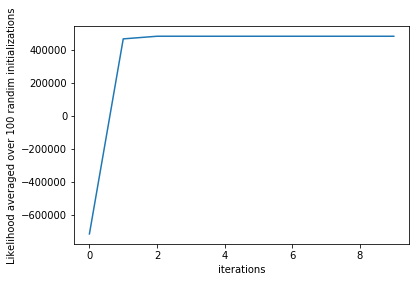








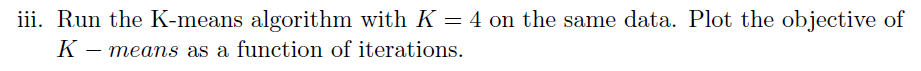
The plot of log-likelihood (averaged over 100 random initializations of parameters) as a function of iterations is as follows:



Comparision with the plot from part (i):

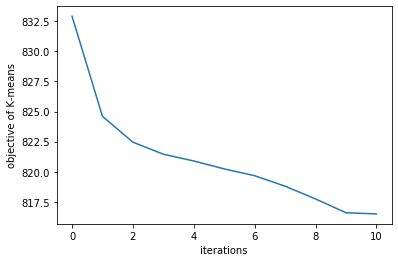
1. The log likelihood reaches to a higher value in the second model than the first.
2. The increase in the log likelihood is monotonus in this case, which was not quite there in the first.
3. The convergence in the log likelihood is comparatively faster in the second case.

Insights:

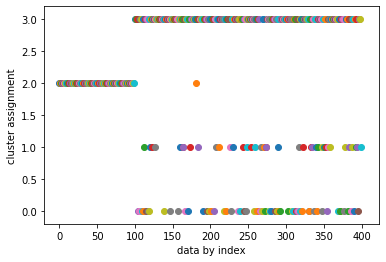


Ans: 

The plot of the objective of K-means as a function of iterations is as follows:

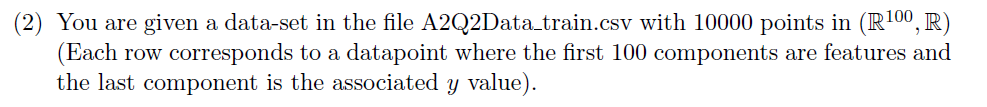


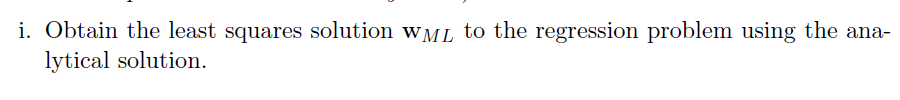
Assignment of each data points to their clusters is depicted as follows:





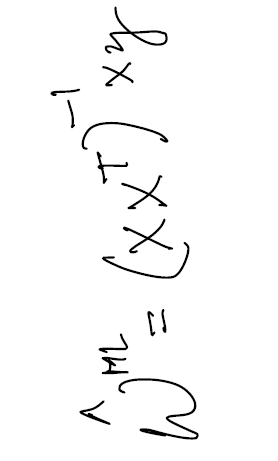
Ans: I would choose the mixture of bernoulli model as a generative model for this data. Since the data is a vector of either 1 or zero, it makes better sense to assume a bernoulli generative story. Also here if we try to assign hard-clustering of the data points with those clusters for which it has highest lambda, we can obtain our objective as the sum of the distances of the datas from their means. That value averaged over 100 random initializations comes as: 823.378. Which is pretty close to the naïve K-Means (817.84). We get a Lambda, that is soft clustering by our generative model for each point, which gives preference to this model over K-Means. In the other model(Gaussian Mixture) we get the sum of distances as: 930.368 which is significantly larger than the two.



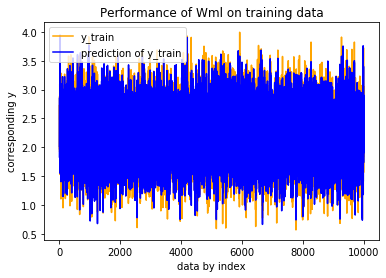




Ans: The least squares solution WML to the regression problem is calculated by the following equation(analytical solution):



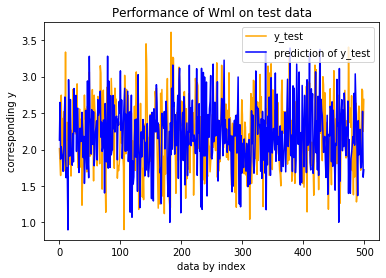
Here, X corresponds to the data in (d\*n) format where d is the dimension and n is the number of data points. Y is the corresponding dependent variable.



Here, our observation is the performance of Wml on train data is very close to the corresponding y\_train, i.e the approximation of y value is very close to the actual y value for train data with Wml.

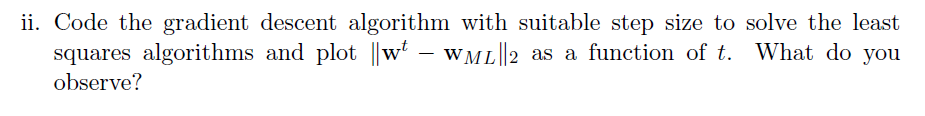
Also, averaging the train error over the number of data points we can find the norm to be:

0.001992145623761605

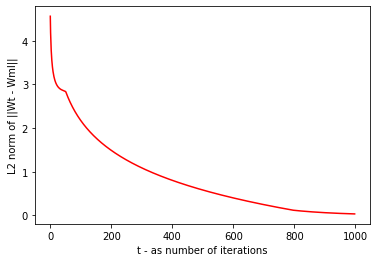


Here, we observe the performance of Wml on test data. This gives us an average error of:

0.02722966438169207



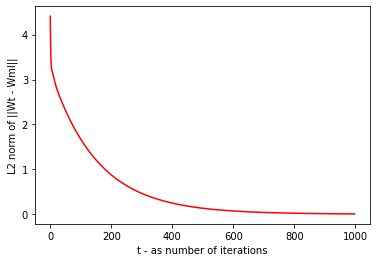
Ans: Step Size = (1/t) where t is taken in the range of 1 to 1000 as per the iterations. The plot of ||Wt – Wml||2 as a function of t is presented below:



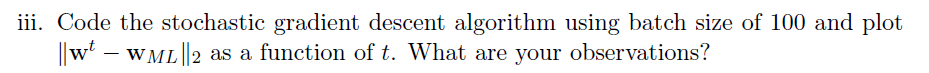
Here, the optimality of the Wt (i.e the solution of gradient descent at each step) is achieved gradually. As per the iteration progresses the difference with Wml gets minimised. This suits our assumption.

The least L2 norm achieved by the W(gradient descent) from Wml is 0.0347325 in 1000th iteration, taken (1/t) as step size. Also, tried with increased step size as (2/t) as t is the number of iteration. And, with 1000 iterations we observe the following plot of L2 norm of ||Wt – Wml||

as a function of t.



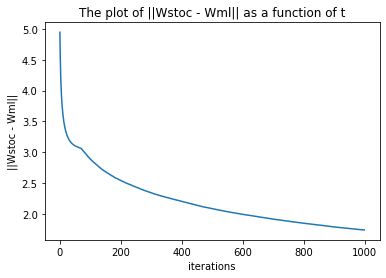
Here the least L2 norm we are able to get is: 0.0065722 at our 1000th iteration.



Ans:

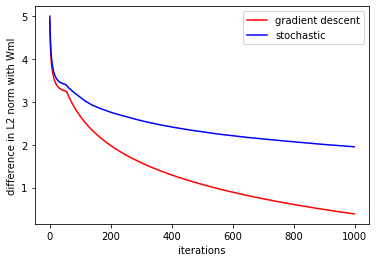
Here, the convergence of the L2 Norm with Wml of the Wstochastic is comparatively slower than the **gradient descent** that we performed earlier. L2 norm from Wml at 1000th iteration is around 2.23048, with step size (1/t) where t is the number of iterations.

So quite expectedly it does not perform as good as Gradient Descent performed using the same step size following the same number of iteartions.



Also the Stochastic Gradient descent approaches to its minimum difference with Wml monotonically because the objective function is quadratic. So with progress every updation is leading to lower the L2 norm, i.e getting the Wstochastic nearer to Wml.

Comparision with respecet to gradient descent is found to be as follows, plotted the L2 norm with Wml as a function of iterations:

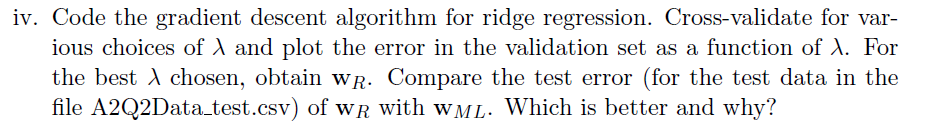


Here, we observe the difference in the final norm along with its propagation throughout the 100 iterations.

With step size (2/t) where t is the number of iterations, we produce the following plot of the norm as a function of the iteartions, here we observe that the L2 difference becomes lower as

|  |  |
| --- | --- |
|  |  |

compared to step size (1/t) such that at 1000th iteration, the minimum L2 difference we get is 0.12888. Though this is still not as good as Gradient Descent’s performance using same parameters but this is a significant improvement over the previous step size on Stochastic Gradient Descent. Also from the comparision plot it is also notable that the difference is very minimum between the two with this step size.

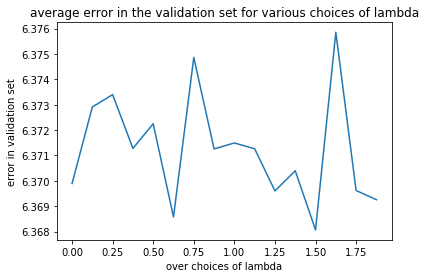


Ans: Choices of labmda: Lambda has been chosen between the range of 0 to 2 with a 0.125 difference between each.

Cross-validation: Here, K-fold cross validation is applied with k = 10.

Also, the step size is chosen to be (2/t), with t as the number of iterations in the gradient descent step.

Following is the plot of the error in the validation set as a function of λ.



Test error of Wridge with Wml: (averaged over number of data points in test data)

0.02720635668276448

Comparing with test error of Wml:

(Error obtained by Wridge – Error obtained by Wridge) = [-2.33076989e-05]

Therefore our observation is that the error we obtain by Wridge is lower in test data in comparision with Wml. Also our choice of lambda does not improve the Wridge by a wide margin, the difference is in the order of 10-5.