PRML ASSIGNMENT - 2

Name : Saurabh Raj Ratan
Roll No. : CS22E009

Assignment Report

Q1.) 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.)

Q1(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 initialisations) as a function of iterations.

Sol.Q1(i) The given sample is derived from a Bernoulli mixture, which can be clearly seen by plotting the data points and looking at the sum along each axis.

Solution :

Let us assume a set of D binary variables x_i, i=1,2,3,……,D where each of which is governed by a Bernoulli distribution with parameter μ, where :

$$p(X=\mu) = \Pi^{D}_{i=1}\ \mu_i^{xi}(1-\mu_i)^{1-xi}$$

The mean and covariance of the distribution are μ and $diag(\mu_i(1-\mu_i))$.

Therefore, the finite mixture can be expressed as: $p(x|\mu,\pi)=\Sigma^{K}_{k=1}\pi_k p(x|\mu_k)$

Also ,

$p(x|\mu_k)=\Pi^{D}_{i=1}\ \mu_i^{xi}(1-\ \mu_i)^{1-xi}$

So, redefining the mean and covariance of the mixture distribution as below,

$E[x]=\Sigma^{K}_{k=1}\pi_k\mu_k$ and $\Sigma^{K}_{k=1}\pi_k\{\Sigma_k+\mu_k\mu_k^{T}\}-E[x]E[x]^{T}$

To derive the EM for this mixture, let **z** be the latent variable associated with each instance of **x**. Therefore, the conditional distribution of x, given the latent variable **z**, can be written as:

$p(X|z,\mu) = \Pi^{K}_{k=1}\ p(x|\mu_k)^{zk}$

With the prior distribution for the latent variable as:

$p(z|\pi)=\Pi^{K}_{k=1}\ \pi^{zk}_{k}$

Taking the expectation of the log-likelihood with respect to the posterior distribution of the latent variable, yields:

$Ez[\ln p(X, Z|\pi,\mu)]= \Sigma Nn=1\ \Sigma Kk=1\gamma(z_{nk})\{\ln \pi k +\Sigma Di=1[xni\ln \mu ki+(1-x_{ni})\ln (1-\mu_{ki}]\}$

In the **Expectation** step, these posterior probabilities are evaluated using the Bayes theorem.

Now, if we consider the sum over n in the likelihood expression, we can obtain the posterior probabilities entered only through two terms:

$N_k=\Sigma^{N}_{n=1}\gamma(z_{nk})$ and $X_k=(1/N-k)\Sigma^{N}_{n=1}\gamma(z_{nk}x_n)$

In the **Maximization** step, we maximize the expected complete data likelihood with respect to the parameters μ_k and π_k.

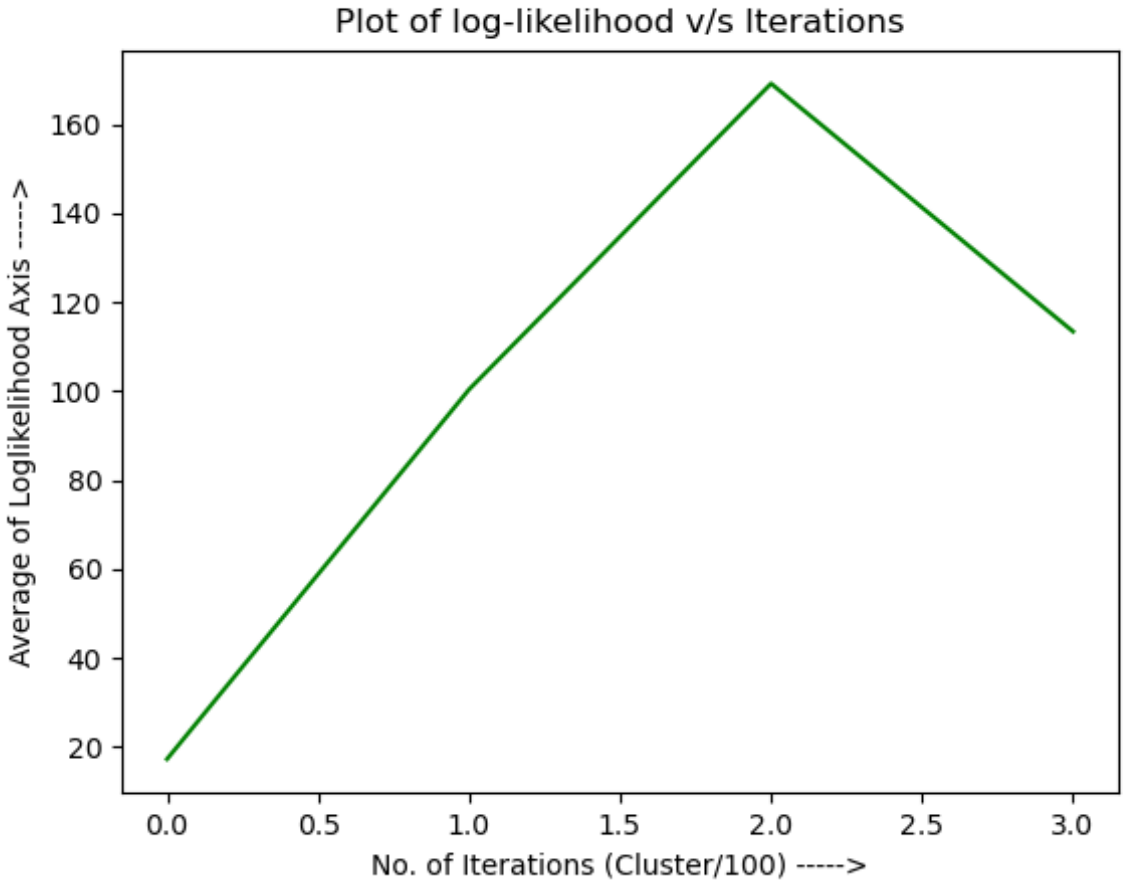
Equating derivative to 0. we get,

$\mu_k = x\_bar_k$

For the maximization with respect to π_k, we introduce a Lagrange multiplier, thus following steps analogous to Gaussian Mixtures. $\pi_k=N_k/N$

The above derivation is based on Latent Class Analysis.

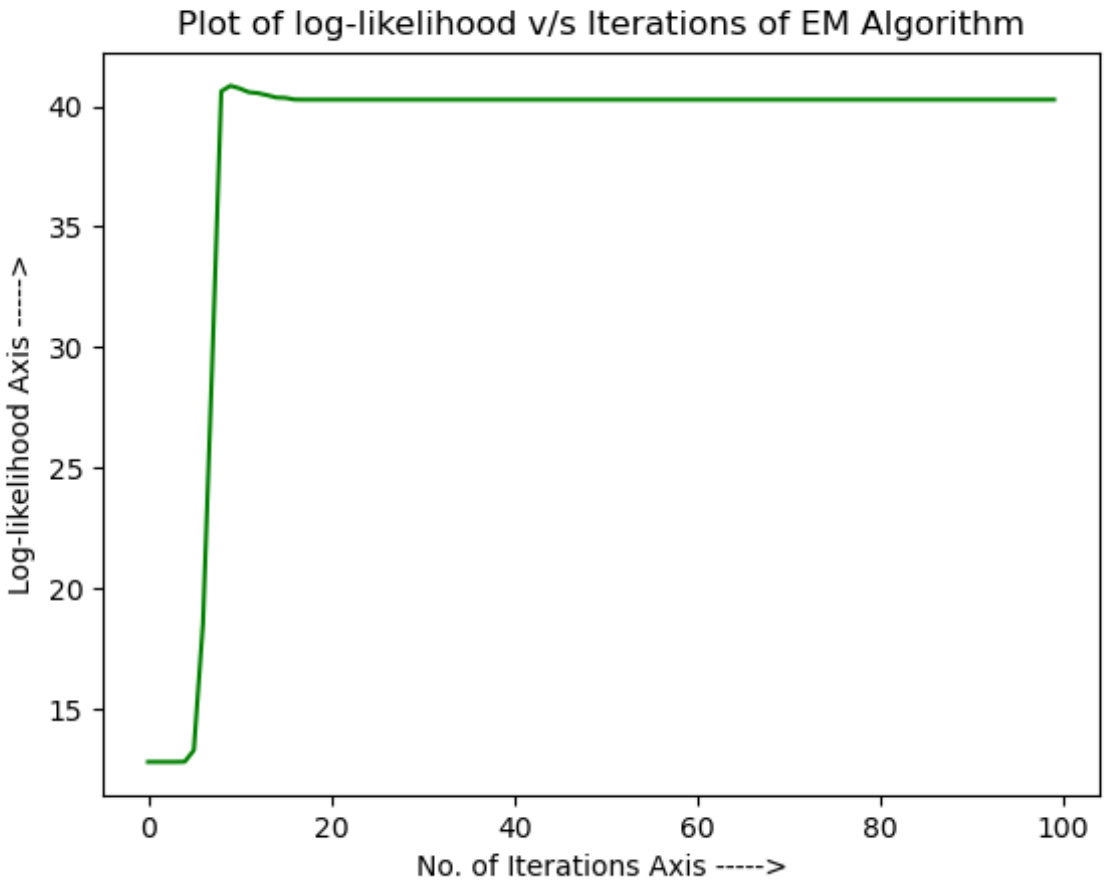
Curve plotted is shown below:



Q1(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 (aver- aged over 100 random initialisations 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.

Solution :

Assuming that data has generated from Gaussian(Normal) Distribution, following graph has plotted after EM Algorithm:

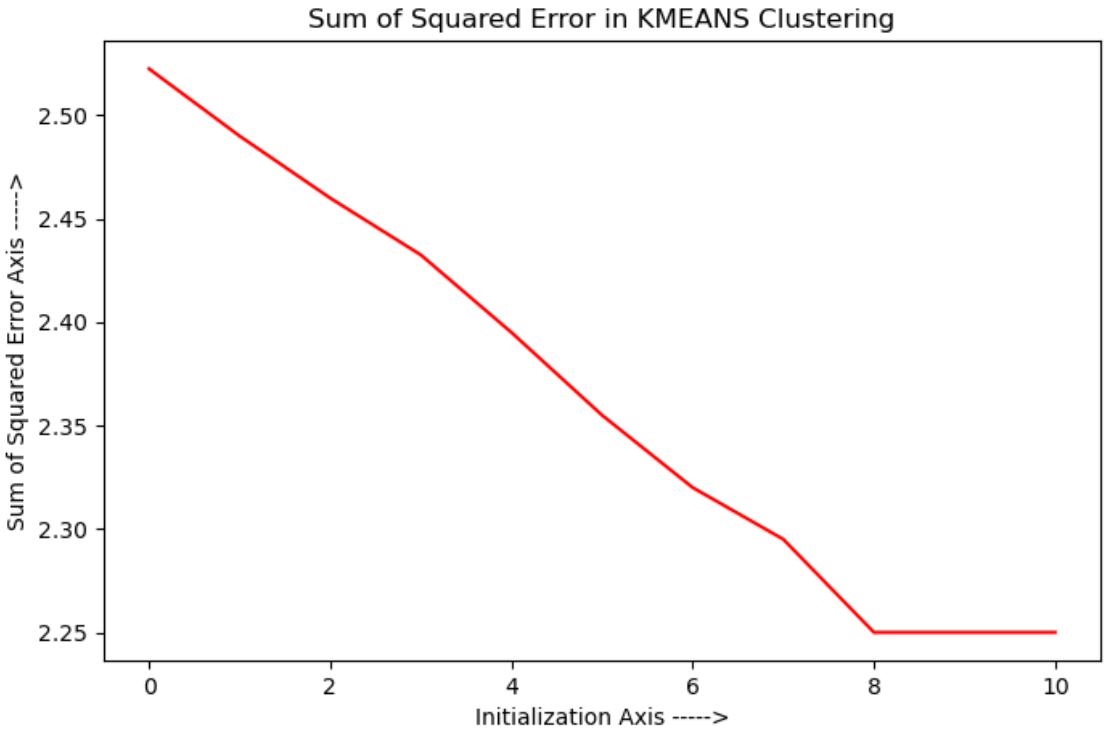


EM Algorithm assuming Gaussian Mixture Model

Q1(iii) Run the K-means algorithm with K = 4 on the same data. Plot the objective of K − means as a function of iterations.

Sum of Squared Error of each data points from its nearest centroid is plotted against each iterations and the corresponding plot is shown below:

It is observed that error becomes constant after certain no. of iterations.



Sum of Squared Error in K-Means Clustering Algorithm

Q1(iv) EM Algorithm for Bernoulli's Distribution can be chosen for this dataset because it can be claimed using above graphs and from analysis of plots of the dataset, that the dataset has been generated previously from a Mixture of Bernoulli Distributions could be true because of the reason than the Log-Likelihood for the implementation with a Mixture of Bernoulli shows a maximum value of over 100 random initializations in above plot.

Q(2) You are given a data-set in the file A2Q2Data train.csv with 10000 points in (R^{100}, R) (Each row corresponds to a datapoint where the first 100 components are features and the last component is the associated y value).

Q(2.i) Obtain the least squares solution w_{ML} to the regression problem using the analytical solution.

Solution:

First we will calculate error function or loss function from formula : loss = Sigma of square of (Y_predicted - Y_actual) over all datapoint.

After this we will differentiate this loss function and equate to zero in order to minimise the Loss Function.And Eventually after solving this we will get Least Square Solution as:

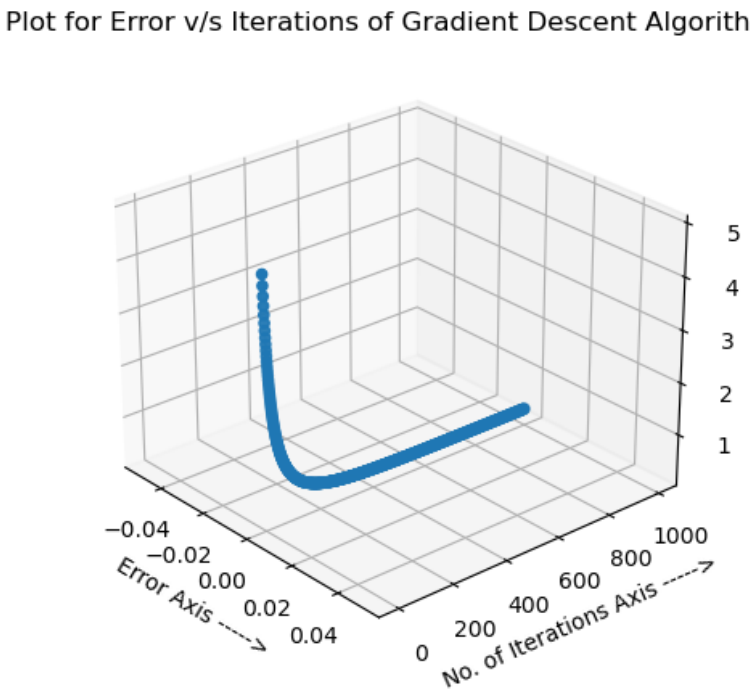
$$W_{ml}=(X^TX)^{-1}*X^T*B$$

Error obtained on given Test Data is : 13.61452712431025

Q(2.ii) Code the gradient descent algorithm with suitable step size to solve the least squares algorithms and plot $\|w^t - w_{ML}\|_2$ as a function of t. What do you observe?
Solution :

In Gradient Descent Algorithm after certain no. of Iterations the value of W_t-W_{ml} decrease gradually with iterations and eventually becomes saturated and tend to become constant and tend to become parallel to the x-axis.

W_t-W_{ml} vs iterations

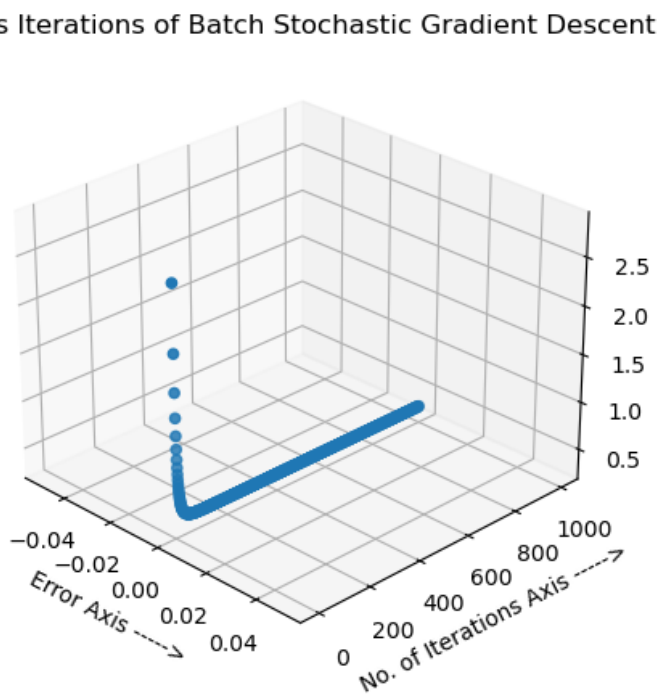


Error v/s iterations of Gradient Descent Algorithm

Q(2.iii) Code the stochastic gradient descent algorithm using batch size of 100 and plot $\|w^t - w_{ML}\|_2$ as a function of t. What are your observations?

Solution:

After running Batch Stochastic Gradient Descent Algorithm for the given Dataset it is noticed that although both Gradient Descent as well as Stochastic Gradient Descent Algorithm Converges eventually after certain no. of iterations but the rate of convergence in Stochastic Gradient Descent is way faster than Gradient Descent Algorithm.



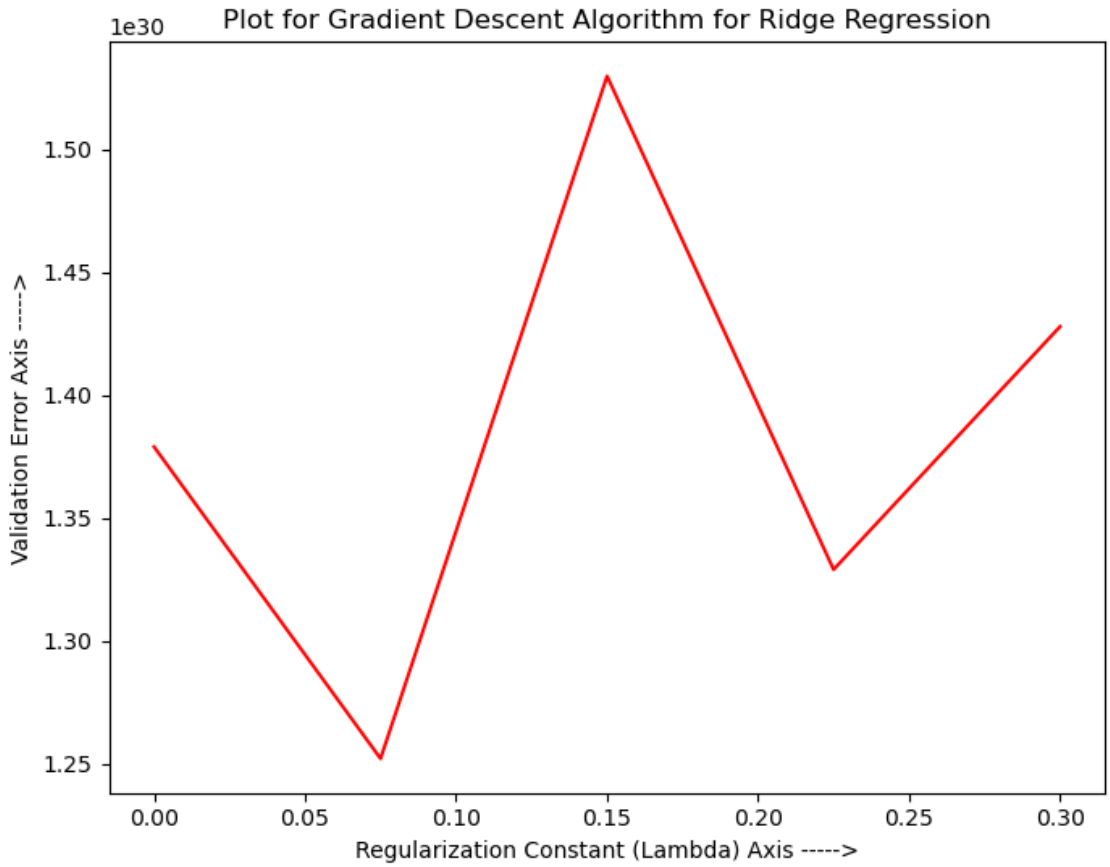
Batch Stochastic Gradient Descent Algorithm

Q(2.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 w_R. Compare the test error (for the test data in the file A2Q2Data_test.csv) of w_R with w_{ML}. Which is better and why?

Solution:

For the different values of lambda, the cross-validations has been performed and graph has been plotted.

Best Choice of Lambda = 0.15


Plot for Gradient Descent Algorithm for Ridge Regression
1e30
1.50
1.45
1.40
1.35
1.30
1.25
Validation Error Axis ----->
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Regularization Constant (Lambda) Axis ----->
Error with Least Square Solution of Linear Regression = 13.61452712431025
Error on Test Data with above Best choice of lambda value = 5.7469597 which is Lesser than above. Hence it can be claimed that due to Regularisation, performance of ridge regression is better than Least Square Solution of Linear Regression.