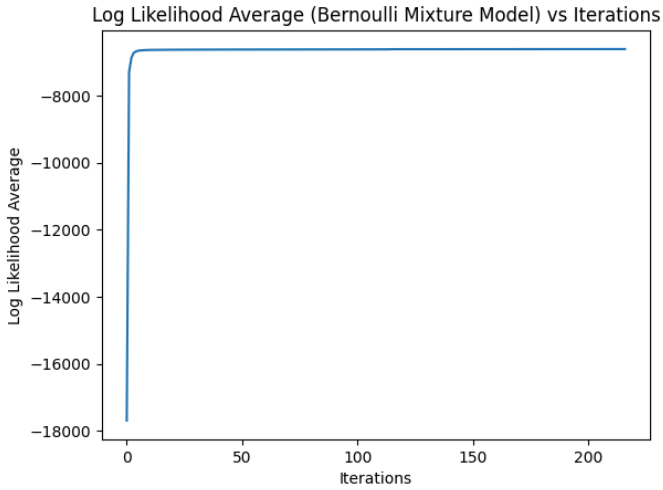
Question 1

i)

- As the data consists of only 0s and 1s, it is most likely generated from a Bernoulli Mixture Model.
- The following graph was obtained when averages of log-likelihood for each iteration was plotted against iterations.
- A tolerance of 10^{-3} was set to check between previous log-likelihood value and current log-likelihood values and if it was below the tolerance the algorithm was terminated.



- N is the number of data points, D is the number of dimensions, K is the number of mixtures.
- λ is a $N \times K$ matrix,
- μ is a $K \times D$ matrix and it is the Bernoulli probability matrix
- π is $1 \times K$ matrix and contains the mixture probabilities.

Expectation step:

$$\lambda_{n,k} = \frac{\pi_k \prod_{i=1}^{D} \mu_{k,i}^{x_{n,i}} (1-\mu_{k,i})^{1-x_{n,i}}}{\sum_{m=1}^{K} \pi_m \prod_{i=1}^{D} \mu_{m,i}^{x_{n,i}} (1-\mu_{m,i})^{1-x_{n,i}}}$$

Maximization step:

$$\mu_m = \frac{1}{\sum_{n=1}^{N} \lambda_{n,m}} \sum_{n=1}^{N} \lambda_{n,m} X_n$$

$$\pi_m = \frac{\sum_{n=1}^{N} \lambda_{n,m}}{N}$$

Calculation:

Bernoulli Mixture Model pdf

$$P(X_i;\mu) = \sum_{k=1}^{K} \pi_k \prod_{d=1}^{D} \mu_{k,d}^{x_{i,d}} (1-\mu_{k,d})^{(1-x_{i,d})}$$

Log-Likelihood:

$$l(X;\pi,\mu,\lambda) = \sum_{i=1}^{N} log\left(\sum_{k=1}^{K} \lambda_{i,k} \frac{\pi_k \prod_{d=1}^{D} \mu_{k,d}^{x_{i,d}} (1-\mu_{k,d})^{(1-x_{i,d})}}{\lambda_{i,k}}\right)$$

Modified log-likelihood:

$$L(X;\pi,\mu,\lambda) = \sum_{i=1}^{N}\sum_{k=1}^{K} \lambda_{i,k} log\left(\frac{\pi_k}{\lambda_{i,k}} \prod_{d=1}^{D} \mu_{k,d}^{x_{i,d}} (1-\mu_{k,d})^{(1-x_{i,d})}\right)$$

- On derivating the above equation wrt $\mu_{k,d}$ and equating it to zero, we get

$$\sum_{i=1}^{N} \lambda_{i,k} x_{i,d}(1-\mu_{k,d}) - \lambda_{i,k}(1-x_{i,d})\mu_{k,d} = 0$$

$$\mu_{k,d}\left(\sum_{i=1}^{N} \lambda_{i,k} x_{i,d} + \sum_{i=1}^{N} \lambda_{i,k} - \sum_{i=1}^{N} \lambda_{i,k} x_{i,d}\right) = \sum_{i=1}^{N} \lambda_{i,k} x_{i,d}$$

$$\mu_{k,d} = \frac{\sum_{i=1}^{N} \lambda_{i,k} x_{i,d}}{\sum_{i=1}^{N} \lambda_{i,k}}$$

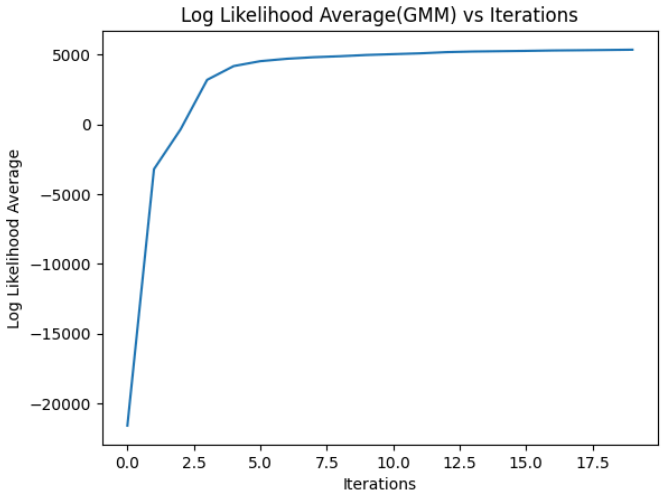
It can be written as

$$\mu_k = \frac{1}{\sum_{i=1}^{N} \lambda_{i,m}} \sum_{i=1}^{N} \lambda_{i,k} X_i$$

Optimizations for the other variables require constraint optimization techniques. The equations were similar to the GMM derivation for EM in class hence they were written directly by just replacing the probability functions.

ii) The below graph depicts the relation between the log-likelihood of the multivariate gaussian mixture model and the number of iterations performed in the EM algorithm averaged over 100 random initializations.

- Note that the algorithm was terminated at 20 iterations because the time needed to compute the required values at the E and M steps was large.



Observations:

- EM algorithm for this multivariate GMM takes a lot more time in comparison with Bernoulli when the same terminations criteria were set.
- Hence the graph was plotted by taking only up to 20 iterations.
- Also, GMM attains higher likelihood values than the Bernoulli mixture model which suggests that the data fits better with GMM.

Equations:

- As it is a Multivariate Gaussian Mixture Model, a few modifications were made to the original equations used in the class.

$$G\left(X|\mu,\Sigma\right)=\frac{1}{\sqrt[D]{2\pi}\ \sqrt{|\Sigma|}}exp\left(-\frac{1}{2}(X-\mu)\Sigma^{-1}(X-\mu)^{T}\right)$$

- D is the number of dimensions/features
- Here Σ is the covariance matrix of dimenions $D\ \times\ D$.
- μ is a $K\ \times\ D$ matrix which is the mean
- π is a $K\ \times\ 1$ matrix. Here K is the number of mixtures.
- The probability density is given by

$$p(X)=\sum_{k=1}^{K}\pi_{k}G(X|\mu_{k},\Sigma_{k})$$

- The log-likelihood function is as given below.

$$ln\left(p(X|\mu,\Sigma,\pi\right)=\sum_{i=1}^{N}ln\sum_{k=1}^{K}\pi_{k}G(X_{i}|\mu_{k},\Sigma_{k})$$

- **Expectation**-step:

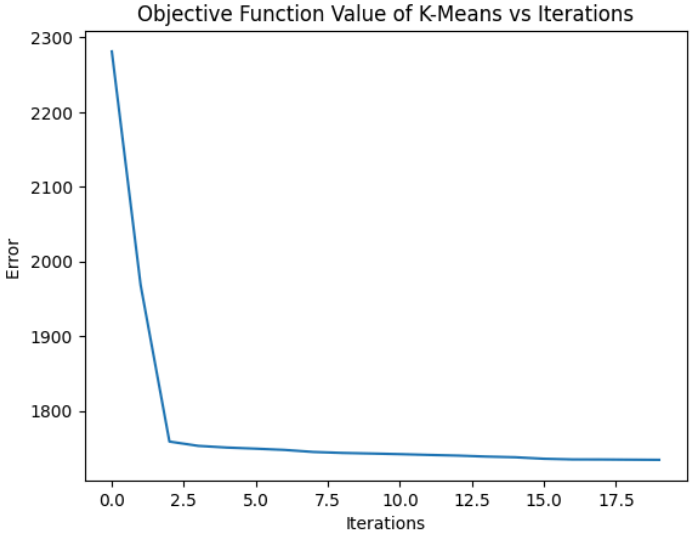
$$\gamma_k(X) = \frac{p(X|k)p(k)}{\sum_{k=1}^{K} p(k)p(X|k)}$$
$$= \frac{p(X|k)\pi_k}{\sum_{k=1}^{K} \pi_k p(X|k)}$$

- **Maximization**-step:

$$\mu_k = \frac{\sum_{n=1}^{N} \gamma_k(x_n)x_n}{\sum_{n=1}^{N} \gamma_n(x_n)}$$
$$\Sigma_k = \frac{\sum_{n=1}^{N} \gamma_k(x_n)(x_n - \mu_k)^T(x_n - \mu_k)}{\sum_{n=1}^{N} \gamma_k(x_n)}$$
$$\pi_k = \frac{1}{N}\sum_{n=1}^{N} \gamma_k(x_n)$$

Note: The above equations were obtained from various sources on the internet because GMM with more than 1 dimension wasn't discussed on how to apply the EM algorithm

iii) K-Means for the same data lead to the following graph of Error/Objective function vs Iterations



iv) Between mixture models and K-means, mixture models provide a better insight into the structure of data. In contrast, K-means just applies the algorithm disregards any structure in the data.

- Bernoulli harnesses the structure of the data because there are only 0s and 1s in the data
- But from the likelihood plots, we can see that Gaussian achieves higher

likelihood values than Bernoulli which suggests that Gaussian fits better with the data.

- Taking the Computational efficiency into consideration, GMM takes a large time (~3min) when compared to Bernoulli (~8sec).
- From the above points, I feel the **Bernoulli Mixture Model** is better because the data has only 0s and 1s, it is significantly faster.

Question 2

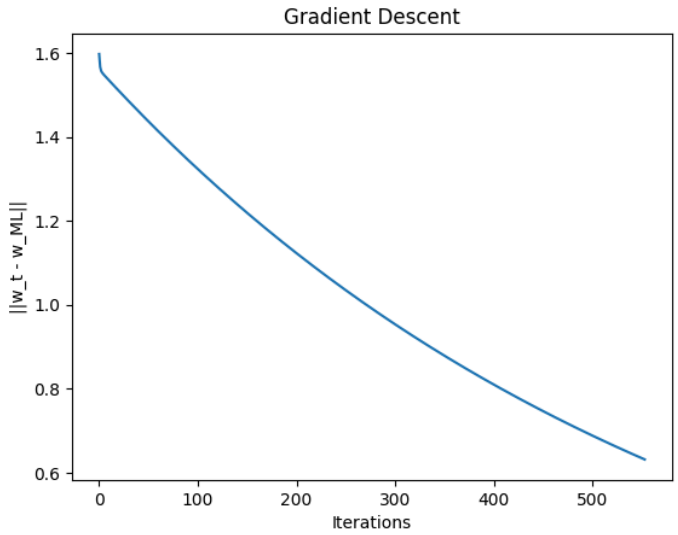
i) The least squares solution w_{ML} is obtained from the analytical solution as

$$w_{ML} = (XX^T)^{-1}XY$$

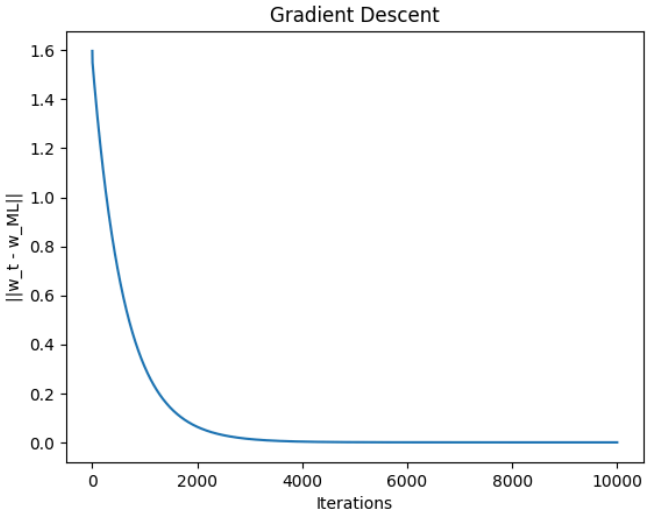
The train data and test data errors calculated from this w_{ML} are 396 and 185 respectively.

ii) The Gradient Descent algorithm was terminated when the percentage difference between w_grad_new and w_grad_old is less than 0.1%
Here w_grad_new is the new value of w obtained from gradient descent at this step whereas w_grad_old is the w value obtained in the previous step.

- When the percentage difference in w was set as 0.1% the algorithm terminated at 552 iterations. This is a very good approximation to w_{ML}.



- Here t is the denotes the t 'th' iteration.
- If the algorithm was simulated for 10000 iterations the following graph was obtained.



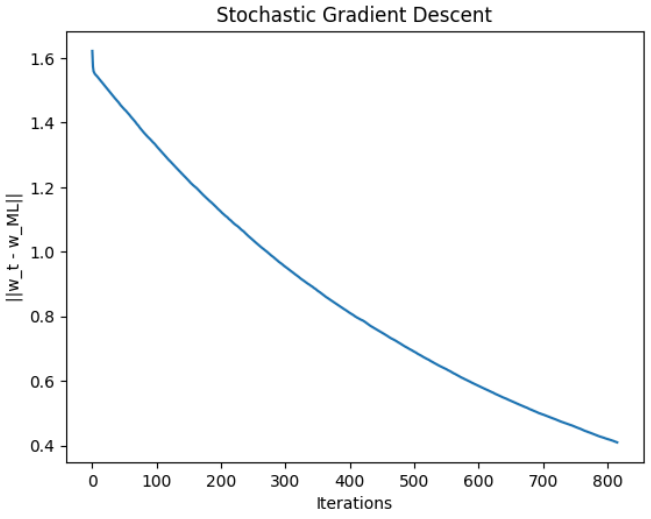
Observations:

- The gradient descent method eventually converges to the analytical solution of w. This is because the objective function in linear regression is a convex function.
- As we need not require a 100% accurate w_{ML} we can operate upto 99.9% accuracy at a very fast speed.
- The best result was obtained at a step size of 10^{-6}.
- The errors of the train and test data obtained are 721 and 131 respectively.
- The below data was obtained afterrunning the Gradient Descent algorithm for 1000 iterations. For step size of 10^{-5} and higher, the algorithm doesn't converge and diverges. Hence a step size of 10^{-6} best suits the data.

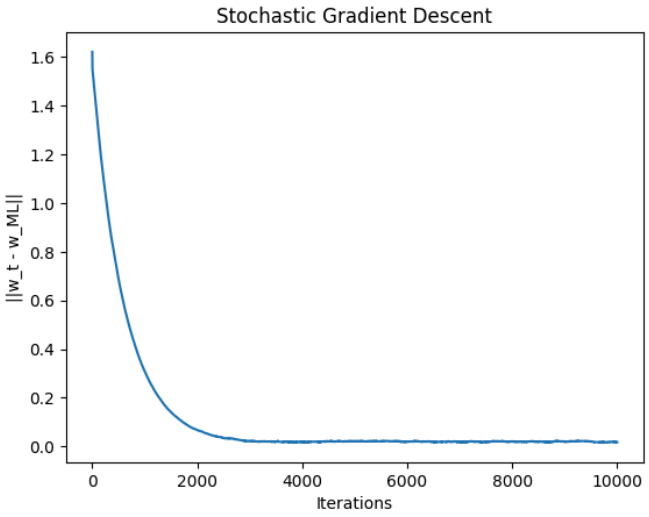
Step size	Error wrt train data
10^{-5}	Doesn't converge
10^{-6}	472
10^{-7}	1835
10^{-8}	2337

iii) Similar to the Gradient Descent algorithm, the SGD was terminated when the percentage difference between w_grad_new and w_grad_old is less than 0.1%.

- The algorithm terminated at ~900 iterations



- When the algorithm was continued for 10000 iterations the following graph was obtained.



Observations:

- Graphs obtained in SGD was similar to that obtained for normal Gradient Descent.
- SGD ultimately converges to w_{ML} because the objective function in linear regression is convex.
- The best result was obtained at a step size of 10^{-4}, the train data and test data errors are ~1000 and ~120 respectively.
- From the errors we can conclude that the best performance over test data was from the analytical method and the best performance over test data was from SGD.
- We can also observe that there is some noise in SGD due to random choice of data points which is evident from the 2nd graph in SGD when compared to the 2nd graph in GD. This noise will be eliminated when calculating w_{SGD} because it is the average of all w_ts.

$$w_{SGD} = \frac{1}{T}\sum_{t=1}^{T} w_t$$

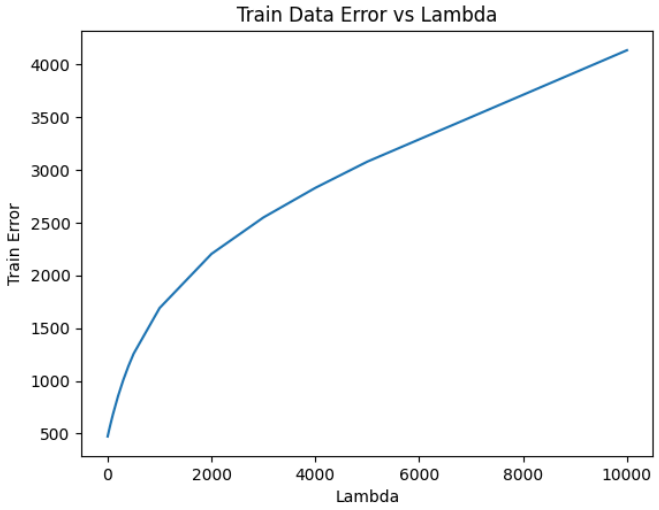
- The following data was obtained after 1000 iterations. As we can see a step size of 10^{-4} suits the data best. If 10^{-3} is chosen then the SGD algorithm diverges due to the large step size.

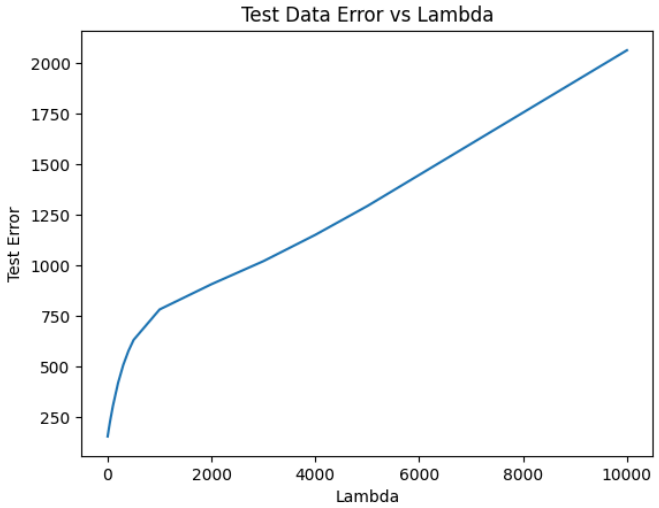
Step size	Error wrt train-data
10^{-3}	Doesn't converge

10^{-4}	910
10^{-5}	2118
10^{-6}	4273

iv)

The following graphs were obtained when λ were varied over multiple values and calculated an optimal w using gradient descent for ridge regression.
The obtained optimal w's were then used to calculate the errors wrt train and test data and the below graphs were plotted.





- From the graphs, it can be concluded that λ = 0 yields the best result.
- The following errors were obtained wrt test data

w_R	155.4
w_{ML}	185.3

- Hence we can conclude that ridge regression with $\lambda = 0$ is better than w_{ML} because the test data error for ridge regression is lower