1 question 1

1.1 (a)

as the given data points as 0's and 1's. we could say that the data might be generated from mixture of bernoulli's distribution(given not to assume gaussian).
EM for bernoulli mixture model
assuming the data generated is independent identical distribution.
let,$z_i \in \{1, 2, \ldots, k\}$, where k is the no of mixtures.
$p(z_i = l) = \pi_l, \sum_{l=1}^{l=k} \pi_l = 1$, where p is probability function
let,x_i be the ith data point generated from Ber(p_i), where Ber(p_i)is the bernoulli distribution with scucces probability as p_i and $i \in \{1, 2, \ldots, k\}$
so we have to find the parameters p_i and π_l, let call them as θ
so our likelihood function is
L$(x_1, x_2, \ldots, x_n, \theta) = \prod_{i=1}^{i=n} p(x_i; \theta)$ (where n is the no of data points)
$= \prod_{i=1}^{i=n} \sum_{k=1}^{k=K} p(x_i, z_i = k; \theta)$
taking log on both sides
$log(L) = \sum_{i=1}^{i=n} log(\sum_{k=1}^{k=K} p(x_i, z_i = k; \theta))$
$= \sum_{i=1}^{i=n} log(\sum_{k=1}^{k=K} \frac{\lambda_k^i p(x_i, z_i=k;\theta)}{\lambda_k^i})$
taking λ_k^i such that $\sum_{k=1}^{k=K} \lambda_k^i = 1$
using jensen's inequality we have
$log(L) \geq \sum_{i=1}^{i=n} \sum_{k=1}^{k=K} \lambda_k^i log(\frac{p(x_i, z_i=k;\theta)}{\lambda_k^i})$
so our modified Likelihood function is

$$\sum_{i=1}^{i=n} \sum_{k=1}^{k=K} [\lambda_k^i log(p(x_i, z_i = k; \theta)) - \lambda_k^i log\lambda_k^i] \tag{1}$$

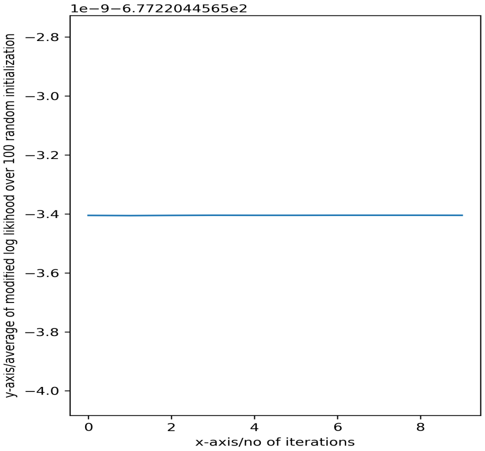
we have $p(x_i, z_i = k; \theta) = \pi_k p_k^{x_i}(1 - p_k)^{(1-x_i)}$
$\sum_{i=1}^{i=n} \sum_{k=1}^{k=K} [\lambda_k^i (log\pi_k + x_i logp_k + (1 - x_i)log(1 - p_k)) - \lambda_k^i log\lambda_k^i]$
for fixed value of $\lambda_k^i \forall i, k$ and taking derivative w.r.t. p_i for maximizing
we have
$\sum_{i=1}^{i=n} \lambda_k^i (\frac{x_i}{p_k} - \frac{1-x_i}{1-p_k}) = 0$
$\sum_{i=1}^{i=n} \lambda_k^i \frac{x_i}{p_k} = \sum_{i=1}^{i=n} \lambda_k^i \frac{1-x_i}{1-p_k}$
$\frac{\sum_{i=1}^{i=n} \lambda_k^i x_i}{\sum_{i=1}^{i=n} \lambda_k^i (1-x_i)} = \frac{p_k}{1-p_k}$
form the property of fractions we have
$\frac{2\sum_{i=1}^{i=n} \lambda_k^i x_i - \sum_{i=1}^{i=n} \lambda_k^i}{\sum_{i=1}^{i=n} \lambda_k^i} = 2p_k - 1$
$p_k = \frac{\sum_{i=1}^{i=n} \lambda_k^i x_i}{\sum_{i=1}^{i=n} \lambda_k^i}$
for fixed θ taking derivative w.r.t to λ_k^i, we have
$\lambda_k^i = \frac{p(x_i/z_i=k;\theta)p(z_i=k;\theta)}{\sum_{l=1}^{l=K} p(x_i/z_i=l;\theta)p(z_i=l;\theta)}$ (form the sir's lecture)
$\lambda_k^i = \frac{\pi_k p_k^{x_i}(1-p_k)^{1-x_i}}{\sum_{l=1}^{l=K} \pi_l p_l^{x_i}(1-p_l)^{1-x_i}}$ for π_k, taking derivative of for maximizing, for constraints as

$$\sum_{i=1}^{i=n} \lambda_k^i + \gamma\pi_k = 0 \tag{2}$$

$\sum_{i=1}^{i=n} \sum_{k=1}^{k=K} \lambda_k^i + \gamma \sum_{k=1}^{k=K} \pi_k = 0$
$\gamma = -n$

$\pi_{ml} = \sum_{i=1}^{i=n} \frac{\lambda_k^i}{n} (from2)$
so we have
$\pi_{ml} = \sum_{i=1}^{i=n} \frac{\lambda_k^i}{n},\ p_k = \frac{\sum_{i=1}^{i=n} \lambda_k^i x_i}{\sum_{i=1}^{i=n} \lambda_k^i}\ and\ \lambda_k^i = \frac{\pi_k p_k^{x_i} (1-p_k)^{1-x_i}}{\sum_{l=1}^{l=K} \pi_l p_l^{x_i} (1-p_l)^{1-x_i}}$

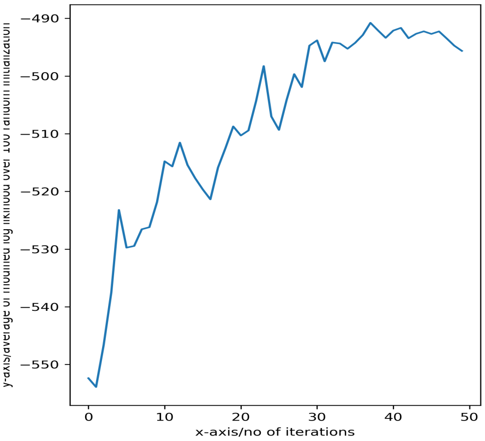
plot of modified loglikelihood as a function of iteration is shown in fig(1)



$figure : 1\ plot\ of\ modified\ log\ likilihood\ vs\ no\ of\ iterations\ for\ bernoulli\ mixture$

1.2 (b)

as for the given data set for some cluster the variance becomes zero and the computation of log liklihood is not possible for such data points (i.e $log(0) -> -\infty$). hence we have ignored such values and not included them for calculations of MLE.
plot for modified logliklihood vs iterations is shown in fig(2)



$figure : 2\ plot\ of\ modified\ log\ likilihood\ vs\ no\ of\ iterations\ for\ gaussian\ mixture$
plot of comparision of part a and part b is shown in fig(3)

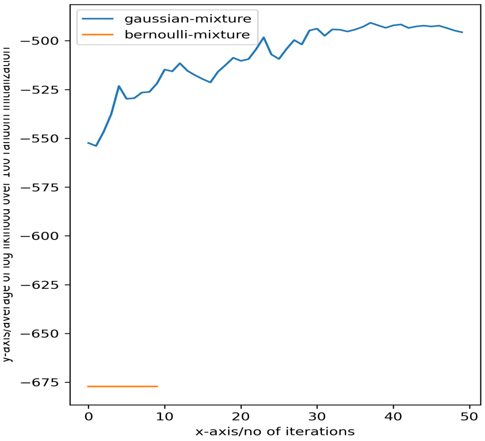


figure : 3 *plot of modified log likilihood vs no of iterations for bernoulli and gaussian mixture*

we can see form the above plot that the MLE for bernoulli distribution is converging faster than the gaussian distribution. the assignment of data points in clusters, for bernoulli mixture it is assigning them in one cluster and sometime, it is assigning them in two clusters (i.e the 0 and 1 data point are in different clusters), same goes for the gaussian mixture (i.e sometimes 0 and 1 are in same cluster and sometimes they are in different clusters)

1.3 (c)

plot of objective of k-means as function of iterations is shown in fig(4)

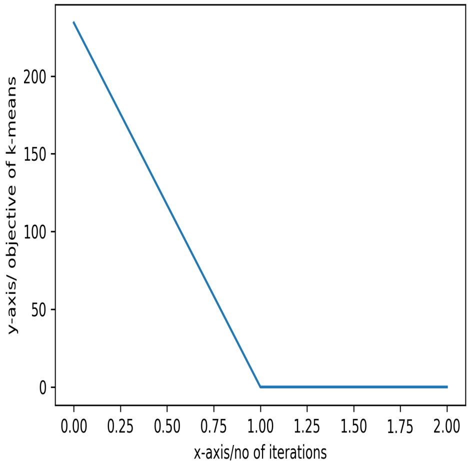


figure : 4 *plot of kmeans objective vs no of iteration*

1.4 (d)

Both benoulli mixture and gaussian mixture classifying 0 and 1 in same cluster or in two different cluster, while kmeans classify both of them in two different cluster with lesser computation than both of them, so even if the data points are from same cluster we could still say that obtained cluster for kmeans are from same class (i.e we could label the cluster as same), but if both of them are form different cluster than for bernoulli and gaussian they may assign them in same cluster and to assign them in different cluster we may have to run many trials, so the computaion increases.
considering above mentioned points i would choose kmeans for the given data.

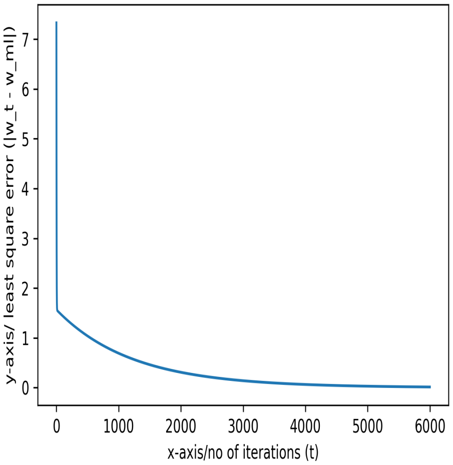
2 question 2

2.1 (a)

the code for W_{ML} is in the code library file named PRML assignment2 question 2

2.2 (b)

analytical solution require the large amount of computation if we have large no of components and have very large no of data points, as computation of $(\mathrm{X}^{\top}\mathrm{X})^{-1}$ becomes highly expensive, while for gradient descent we do not have to compute inverse, this reduces the computation to large amount for if we have large no of dimension.
plot of root squared error(i.e $L_2 normoferrorvector$) for gradient descent for no of iterations is shown in fig(5)

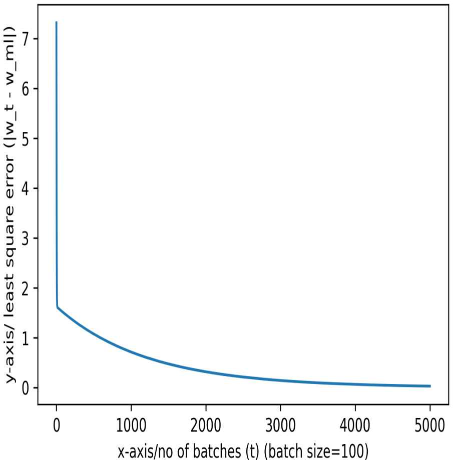


$figure : 5\ plot\ of\ |W_t - W_{ML}|_2\ vs\ no\ of\ iteration\ (t)$
as the given data has 100 dimension and 10000 data points, we have to compute the inverse of (100 x 100) matrix which is expensive, so using gradient descent, for some no iterations and a learning parameters, we can still compute W which is close to $W_m l$, without computing

inverse of (100 x 100) matrix.
so, using gradient descent we are able to find the W which is very close to W_{ML}
if we keep the value of learning parameter > 0.000001, we are having the problem of bursting gradient

2.3 (c)

if the given training set contains high no of data points the computation of $X^\top X$ becomes highly expensive, to reduce this we can use stochastic gradient descent for considering fewer no of samples at a time, and still we are able to manage to get the optimum solution
plot for stochastic gradient descent is shown in fig(6)



$figure : 6\ plot\ of\ |W_t - W_{ML}|_2\ vs\ no\ of\ iteration\ (t)$
the given test set contains 10000 points , it is expensive to compute $X^t opX$, if we use 100 points at a time. for stochastic gradient descent we have created 100 batches of size 100 data points for training, so while training we take one batch at a time and find W_t.from, the plot in fig(5) we can tell that we are able to reach the optimum value for W using stochastic gradient descent for lesser computation
we can see that for stochastic gradient descent we are having almost similar plot as gradient descent
for given data set, while using stochastic gradient descent we have to increase the value learning parameter(i.e. 0.0001), as by keeping the same learning parameter of gradient descent (i.e 0.000001) we are having the problem of vanishing gradient.

2.4 (d)

plot of λ vs error in validation set is shown in fig (6)
here we have used cross validation with $70\% - 30\%$ ratio for training and validating set from given training set

for , $\lambda = 4.2$ we are having minim error in validation set.
for the test data, MSE error for $W_{ML} = 0.3707$
for the test data, MSE error for $W_r = 0.3655$
hence we have less error for W_r as compared to the error w.r.t W_{ML}
so ridge regression is better as compared to analytic solution because, as analytic method is unbiased and it tries to minimize the error for training set as much as possible but this may lead to overfitting issues, while ridge regression is biased and it tries to minimize the sum of bias + variance hence it tries to find a balance for bias and variance (i.e when sum of both of them is minimum), so it avoids the overfitting of model

3 question 3

3.1 a

let, u = mean and $\sigma^2 = 1$ be the variance
we have access to only one data point and the distribution is gaussian, so, we have,
$u_{ML} = \sum_{i=1}^{i=n} \frac{x_i}{n}$, where n is the no of data points
$u_{ML} = u$, as we have access to only single sample
mean squared error = bias^2 + variance = variance = 1
our random variable is unbiased as $(E(\theta') - \theta)^2 = 0)$
given,
$u_{new} = u_{ML}/2 = u/2$
MSE = bias^2 + variance = $(E(u_new) - u)^2 + var(u_new)$
$= (u/2 - u)^2 + var(\sum_{i=1}^{i=n} \frac{x_i}{2n}) = u^2/4 + 1/4var(u) = (u^2 + 1)/4$

the new MSE is less for $(u^2 + 1) < 4$
$u^2 < 3$
$-\sqrt{3} < u < \sqrt{3}$
for, u $\in (-\sqrt{3}, \sqrt{3})$ we have lesser MSE than ML estimator
yes, this shrunk estimator have lesser MSE than the ML estimator for u in range $(-\sqrt{3}, \sqrt{3})$

4 question 4

4.1 a

MSE = bias^2 + variance
clearly, if we use u_{new} as $u + x$, we will not get what we require
let, $u_{new} = |u|/|x|$
consider x as positive,now,
$(|u|/x - u)^2 + var(|u|/x) < 1$
$(1 - x)^2/x^2 * u^2 + 1/x^2 < 1$ $\quad (var(|u|) = var(u) = 1)$

$u^2 < x^2 - 1/(1-x)^2$
$u^2 < (x+1)/(x-1)$
$-((x+1)/(x-1))^{1/2} < u < ((x+1)/(x-1))^{1/2}$
$|u| < ((x+1)/(x-1))^{1/2}$
as x is positive ,for $[a, b]$
$max(|a|, |b|) = c \; c = ((x+1)/(x-1))^{1/2}$
$c^2 = (x+1)/(x-1)$
$x = c^2 + 1/(c^2 - 1)$
so, $u_{new} = |u|/|x|$, where c=$max(|a|, |b|)$

all the code are in folder named code and, code file is different for each question and named as per question