HW submission

1.

A.

```
library(ISLR2)
df_Auto <- data.frame(Auto)
nrow(df_Auto)
ncol(df_Auto)
df_Auto$year <- as.factor(df_Auto$year)
df_Auto$origin_new <- relevel(factor(df_Auto$origin), ref = 3)
lm_fit <- lm(mpg ~ . -name-origin ,data=df_Auto)
summary(lm_fit)

mean(lm_fit$residuals ^2)
```

```
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)  39.5201863  2.1512384  18.371  < 2e-16 ***
cylinders    -0.1924289  0.3040482  -0.633 0.527195
displacement  0.0172507  0.0072044   2.394 0.017139 *
horsepower   -0.0240199  0.0136328  -1.762 0.078904 .
weight       -0.0061203  0.0006494  -9.424  < 2e-16 ***
acceleration  0.0543880  0.0919344   0.592 0.554480
year71        1.0461869  0.8730131   1.198 0.231538
year72        0.0330325  0.8531036   0.039 0.969134
year73       -0.5322929  0.7718143  -0.690 0.490835
year74        1.6545531  0.9129699   1.812 0.070750 .
year75        0.9415172  0.8953739   1.052 0.293695
year76        1.7486166  0.8573480   2.040 0.042100 *
year77        3.2399161  0.8759807   3.699 0.000249 ***
year78        3.0821303  0.8333179   3.699 0.000249 ***
year79        5.3812526  0.8791655   6.121 2.36e-09 ***
year80        9.5116004  0.9339482  10.184  < 2e-16 ***
year81        6.9070845  0.9223997   7.488 5.12e-13 ***
year82        8.6173419  0.9031369   9.542  < 2e-16 ***
origin_new1  -2.5002584  0.5225230  -4.785 2.47e-06 ***
origin_new2   0.0073266  0.5291805   0.014 0.988961
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.05 on 372 degrees of freedom
Multiple R-squared:  0.8547,    Adjusted R-squared:  0.8473
F-statistic: 115.2 on 19 and 372 DF,  p-value: < 2.2e-16
```

The qualitative variables here are year and origin. In order to interpret the results, we need to first see what are the reference group of those variables.

For example- the reference group for origin used here is 3 i.e. Japanese. The coefficient of origin_new1 can be interpreted as the mean difference in mpg is -2.500 if we choose origin_new1(American) over origin_new3(Japanese). Similarly, we can interpret the coefficients of the values of other qualitative variables.

B. The training mean squared error is 8.83.
C. The predicted gas mileage is 30.86 miles per gallon.
D. Keeping all the covariates the same, the difference between the mileage of Japanese and American cars is -2.5. So, choosing American-made over Japanese-made may lead to a decrease in mileage by a mean of 2.5 miles per gallon.
E. Keeping all the covariates the same, the change in mpg with 10 units change in horsepower is -0.24 mpg.

2.

A.

```
df_Auto2 <- data.frame(Auto)
df_Auto2$American <- ifelse(df_Auto2$origin == 1,1,0)
df_Auto2$European <- ifelse(df_Auto2$origin == 2,1,0)
lm_fit21 <- lm(mpg ~ American + European ,data=df_Auto2)
summary(lm_fit21)
```

	Coefficient estimates
Intercept	30.4506
American	- 10.4172
European	-2.8477

Equation : Y= 30.4506 - 10.4172*$I_{American}$ -2.8477 * $I_{European}$ + e, where 'I' is an indicator variable where,

$I_{American}$= 1 if origin = 'American' else 0
$I_{European}$= 1 if origin = European else 0

	Predicted values of mpg
American	30.4506-10.4172 = 20.0334
European	30.4506-2.8477 = 27.6029
Japanese	30.4506

B.

```
df_Auto2$Japanese <- ifelse(df_Auto2$origin == 3,1,0)
lm_fit22 <- lm(mpg ~ Japanese + European ,data=df_Auto2)
summary(lm_fit22)
```

	Coefficient estimates
Intercept	20.0335
Japanese	10.4172
European	7.5695

Equation : Y= 20.0335 + 10.4172*$I_{Japanese}$ + 7.5695 * $I_{European}$ + e , where I is an indicator variable.
$I_{Japanese}$= 1 if origin = Japanese else 0
$I_{European}$= 1 if origin = European else 0

	Predicted values of mpg
Japanese	20.0335+10.4172 = 30.4507
European	20.0335+7.5695 = 27.603
American	20.0335

C.

```
df_Auto2$American_new <- ifelse(df_Auto2$origin == 1,1,-1)
df_Auto2$European_new <- ifelse(df_Auto2$origin == 2,1,-1)
lm_fit23 <- lm(mpg ~ American_new + European_new ,data=df_Auto2)
summary(lm_fit23)
```

	Coefficient estimates
Intercept	23.8182
American	- 5.2086
European	-1.4238

Equation : Y= 23.8182 - 5.2086*$I_{American}$ -1.4238 * $I_{European}$ + e , where I is an indicator variable.
$I_{American}$= 1 if origin = American else -1

$I_{European}$= 1 if origin = European else -1

	Predicted values of mpg
Japanese	23.8182 + 5.2086 + 1.4238 = 30.4506
European	23.8182 + 5.2086 - 1.4238 = = 27.603
American	23.8182 - 5.2086 + 1.4238 =20.0334

D. **Using the new origin variable as factor:**

```
df_Auto2$origin_new <- as.factor(ifelse(df_Auto2$origin == 3,0,ifelse(df_Auto2$origin == 1,1,2)))
lm_fit24 <- lm(mpg ~ origin_new ,data=df_Auto2)
summary(lm_fit24)
```

	Coefficient estimates
Intercept	30.4506
origin_new1	-10.4172
origin_new2	- 2.8477

Y= 30.4506 -10.4172 * $origin\_new1_{(American)}$ - 2.8477*$origin\_new2_{(European)}$ + e, where
$origin\_new1_{(American)}$= 1 if origin = American else 0
$origin\_new2_{(European)}$= 1 if origin = European else 0

	Predicted values of mpg
Japanese	30.4506 - 0 - 0 = 30.4506
American	30.4506 - 10.4172 = 20.0334
European	30.4506 - 2.8477 = 27.6029

Without using the new origin variable as a factor:

	Coefficient estimates
Intercept	25.2395
origin_new	-1.8453

Y= 25.2395 -1.8453 * origin_new+ e, where
origin_new= 0 if origin = Japanese
origin_new= 1 if origin = American
origin_new= 2 if origin = European

	Predicted values of mpg
Japanese	25.2395 - 0= 25.2395
American	25.2395 - 1.8453= 23.3942
European	25.2395 - 3.6906 = 21.5489

E. In all the cases between A-C the predicted values of mpg for all the 3 origins **are same**. In D, if we use origin as a factor, we will get the same prediction for all the origins but if we use origin as a quantitative variable, we will get different results and the mean squared error is also higher.

3.

```
df_Auto3$origin <- as.factor(df_Auto3$origin)
lm_fit31 <- lm(mpg ~ origin + horsepower + origin * horsepower ,data=df_Auto3)
summary(lm_fit31)
```

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        34.476496   0.890665  38.709  < 2e-16 ***
origin2            10.997230   2.396209   4.589 6.02e-06 ***
origin3            14.339718   2.464293   5.819 1.24e-08 ***
horsepower         -0.121320   0.007095 -17.099  < 2e-16 ***
origin2:horsepower -0.100515   0.027723  -3.626 0.000327 ***
origin3:horsepower -0.108723   0.028980  -3.752 0.000203 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4.422 on 386 degrees of freedom
Multiple R-squared:  0.6831,    Adjusted R-squared:  0.679
F-statistic: 166.4 on 5 and 386 DF,  p-value: < 2.2e-16
```

Equation: Y= 34.48 + 11 * $origin_{European}$ + 14.34 * $origin_{Japanese}$ - 0.12 * horsepower - 0.10 * $origin_{European}$ * horsepower - 0.11 * $origin_{Japanese}$ * horsepower

Equation: Y= 34.48 + 11 * $origin_{European}$ + 14.34 * $origin_{Japanese}$ + (**-0.12 - 0.10 * $origin_{European}$ - 0.11 * $origin_{Japanese}$**) * horsepower

	Change in mpg with 1 unit change in horsepower
Japanese	-0.12 - 0.11 = -0.23

American	-0.12
European	-0.12-0.10 = -0.22

4.

A. With the given beta coefficients, the equation will look like below:
Y= -165.1 + 4.8*X1

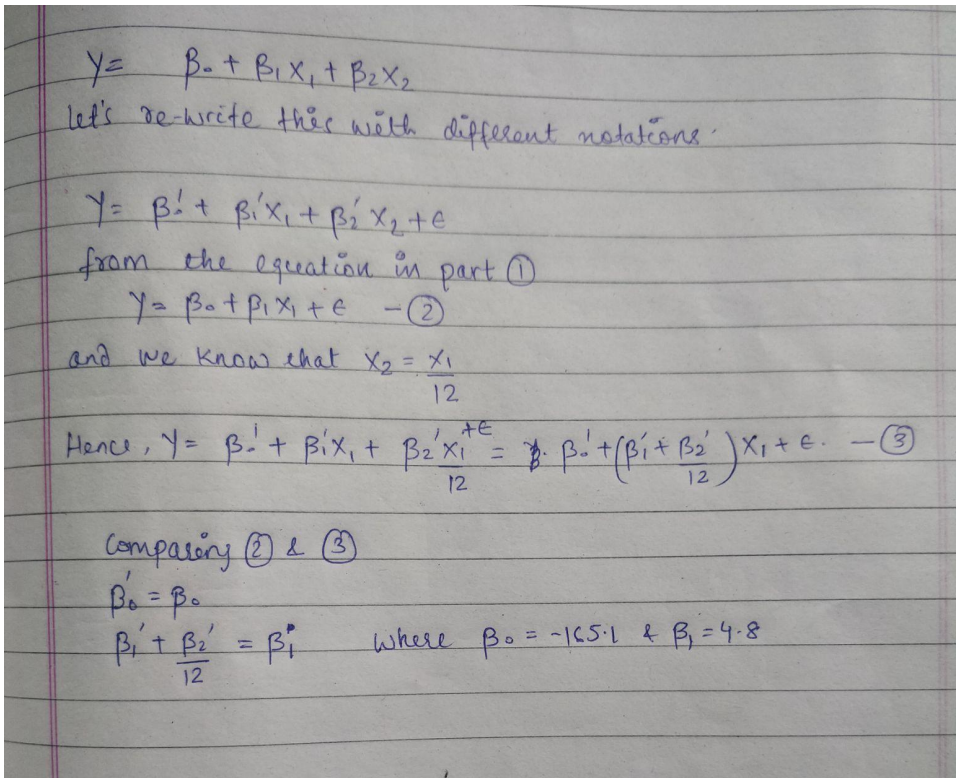
If X1 = 64, Predicted value of weight would be -165.1 + 4.8 * 64 which is 142.1 units.

B. $Y = \beta_{0*} + \beta_{1*}X_2 + \varepsilon.$

The estimate of coefficient β_{0*} which is the slope will remain the same as **-165.1** and the Estimate of β_{1*} will now be the average weight with 1 feet increase in height keeping the remaining covariates the same and will be equal to 4.8 * 12 = **57.6.**

Predicted weight for height of 5.333 feet = -165.1 + 57.6 * 5.33 = 142 units

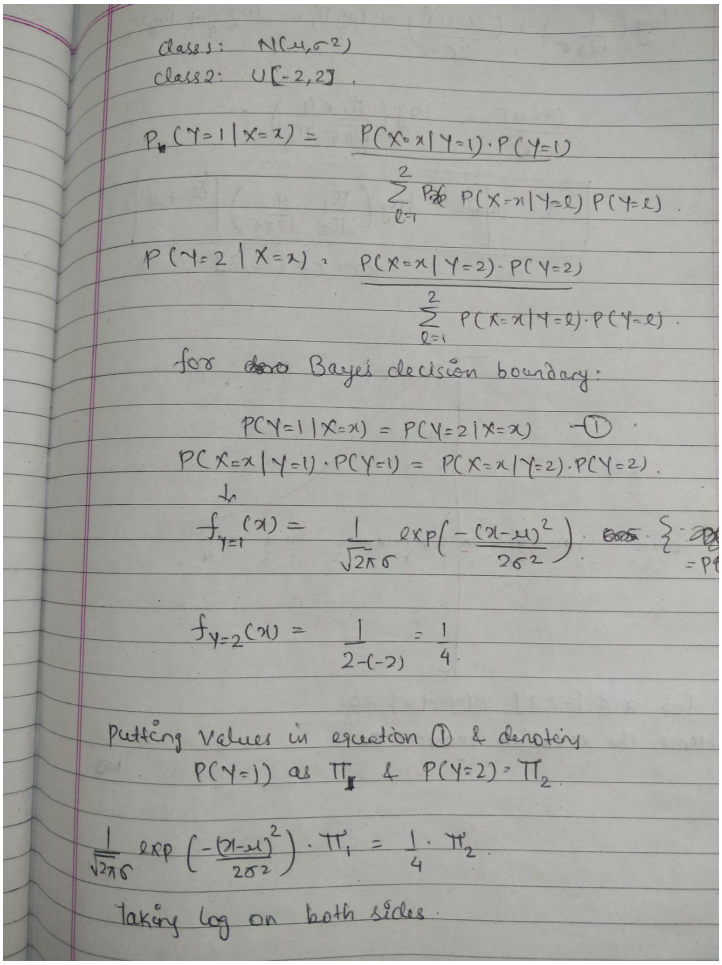
C.



D. Since, the predicted values are the same in all the three versions of the fitted model for the same values of the predictor variables, the mean squared errors are the same.

5.

A. Please find below the decision boundary of Baye's classifier for the given set of observations:



class 1: $N(\mu, \sigma^2)$
class 2: $U[-2, 2]$.

$$P_r(Y=1 \mid X=x) = \frac{P(X=x \mid Y=1) \cdot P(Y=1)}{\sum_{l=1}^{2} P(X=x \mid Y=l) P(Y=l)}.$$

$$P(Y=2 \mid X=x) = \frac{P(X=x \mid Y=2) \cdot P(Y=2)}{\sum_{l=1}^{2} P(X=x \mid Y=l) \cdot P(Y=l)}.$$

for Bayes decision boundary:

$$P(Y=1 \mid X=x) = P(Y=2 \mid X=x) \quad \text{--- (1)}$$

$$P(X=x \mid Y=1) \cdot P(Y=1) = P(X=x \mid Y=2) \cdot P(Y=2).$$

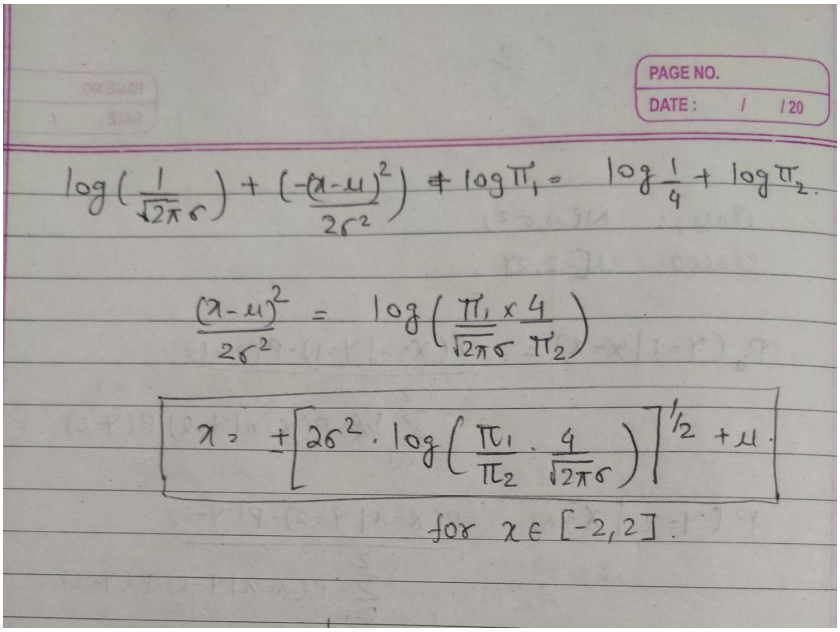
$$f_{Y=1}(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right).$$

$$f_{Y=2}(x) = \frac{1}{2-(-2)} = \frac{1}{4}$$

Putting values in equation (1) & denoting $P(Y=1)$ as π_1 & $P(Y=2) = \pi_2$.

$$\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \cdot \pi_1 = \frac{1}{4} \cdot \pi_2.$$

Taking log on both sides.

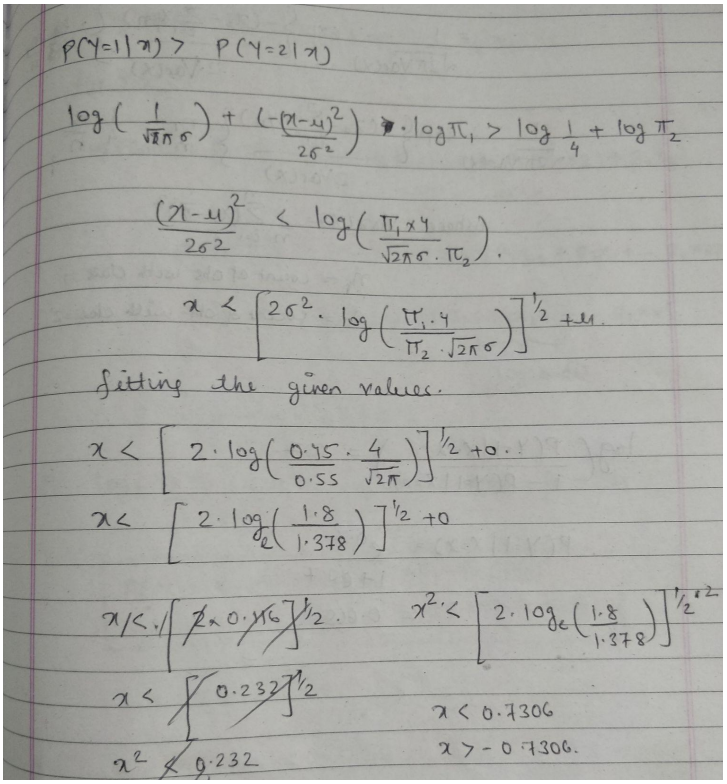


The values of X from this equation are only valid in the range of x between -2 and 2. Beyond that P(Y=2|X=x) is 0 for the uniform distribution and P(Y=1|X=x) is 1. Hence, X=-2 and 2 are two other decision boundaries.

B. By substituting the values of μ, σ *and* Π_1 in the equation derived above, we can find the ranges of x for which it can be classified in class1 or class2.

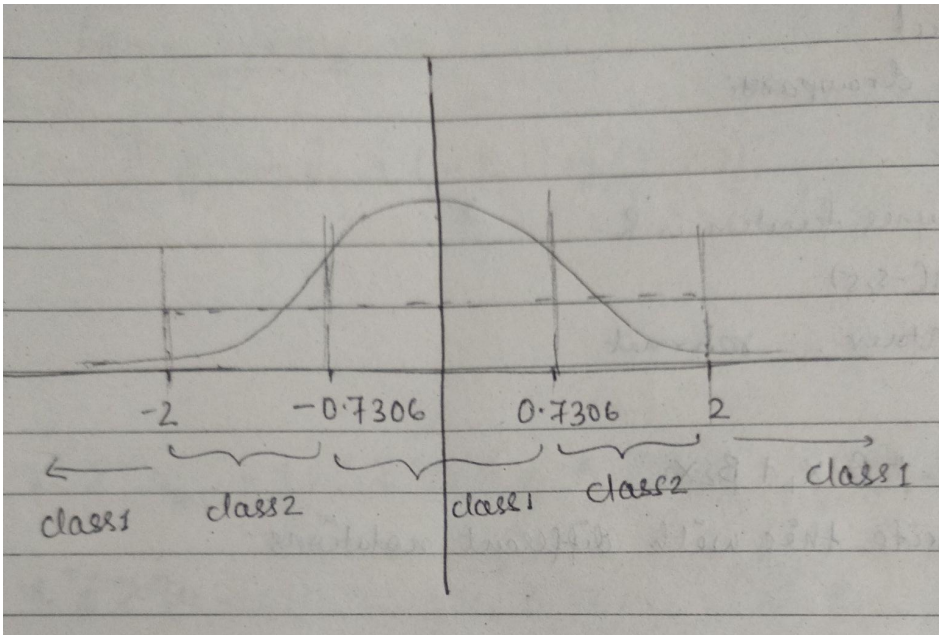
$\Pi_2 = 1 - \Pi_1 = 1 - 0.45 = 0.55$

We know that for X>2, and X<-2, the values will be assigned to class 1. For the other range of class1, we can derive that by solving P(Y=1|X=x) > P(Y=2|X=x).



From the solution, we found that for X>-0.7306 and X<0.7306, the values will be assigned to class1. Hence, class 2 will be assigned for values of X in the range : [-2,-0.7306] U [0.7306,2]

And class1 will be assigned for values of X in the range: [-Inf,-2) U (-0.7306,0.7306) U (2, Inf]



C. We have details of the training observations, we can estimate mu and sigma by first selecting observations for which Y=1. In this case, we know the X follows a normal distribution with mean μ and standard deviation σ. We can estimate these 2 parameters by calculating the mean and standard deviation of the samples for class 1. Similarly, we can estimate mean μ and standard deviation σ for other classes as well.

   For Π_1, the probability of Y=1 can be estimated by calculating the proportions of the sample in data belonging to class 1.

D. Given the training observations, we can estimate the values of μ, σ and Π_1 and $\Pi_2$from the data.

$$P(Y=1 \mid X=x_0) = \frac{f_1(x_0) \cdot P(Y=1)}{\sum_{l=1}^{2} f_l(x_0) \cdot P(Y=l)}$$

$$= \frac{\frac{1}{\sqrt{2\pi Var(X)}} \exp\left\{-\frac{(x_0-\bar{X})^2}{2Var(X)}\right\} \cdot \frac{n_1}{n}}{\frac{1}{\sqrt{2\pi Var(X)}} \cdot \exp\left\{-\frac{(x_0-\bar{X})^2}{2Var(X)}\right\} \cdot \frac{n_1}{n} + \frac{1}{4} \cdot \frac{n_2}{n}}$$

where, $Var(X) = \frac{1}{n-} \sum_{i=1}^{n} (x_i - \bar{X})^2$

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

$n_1 = \text{Count}\{x_1, x_2 \ldots x_n\}_{Y=1}$

$n_2 = \text{Count}\{x_1, x_2, \ldots x_n\}_{Y=2}$

$n = \text{Count}\{x_1, x_2, \ldots x_n\}$

6.

A.

n_2 = count

$$\log\left(\frac{P(Y=1|X=x)}{1-P(Y=1|X=x)}\right) = 0.7.$$

$$P(Y=1|X=x) = \frac{e^{0.7}}{1+e^{0.7}}$$

$$= 0.668$$

B.

Let the original logistic regression model be.

$$\log(\text{odds}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_p x_p.$$

for $X = x^*$.

$$\log(\text{odds}^*) = \beta_0 + \beta_1(x_1+1) + \beta_2(x_2-1) + \beta_3(x_3+2) + \beta_4 x_4 + \dots \beta_p x_p.$$

$$= \beta_0 + (\beta_1 - \beta_2 + 2\beta_3) + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots \beta_p x_p.$$

$$= (\beta_1 - \beta_2 + 2\beta_3) + \{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots \beta_p x_p\}.$$

$\log(\text{odds})$.

$$= (\beta_1 - \beta_2 + 2\beta_3) + 0.7$$

$$\Rightarrow \log\left(\frac{P(Y=1|X=x^*)}{1-P(Y=1|X=x^*)}\right) = 0.7 + \beta_1 - \beta_2 + 2\beta_3.$$

$$\boxed{P(Y=1|X=x^*) = \frac{e^{(0.7+\beta_1-\beta_2+2\beta_3)}}{1+e^{(0.7+\beta_1-\beta_2+2\beta_3)}}}$$

7.

A. Mvrnorm is used to generate bivariate normal distribution.

Below is the choice of sigma and mean vectors to generate data:

sigma <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)

mu1 <- matrix(c(0,1))

mu2 <- matrix(c(2,3))

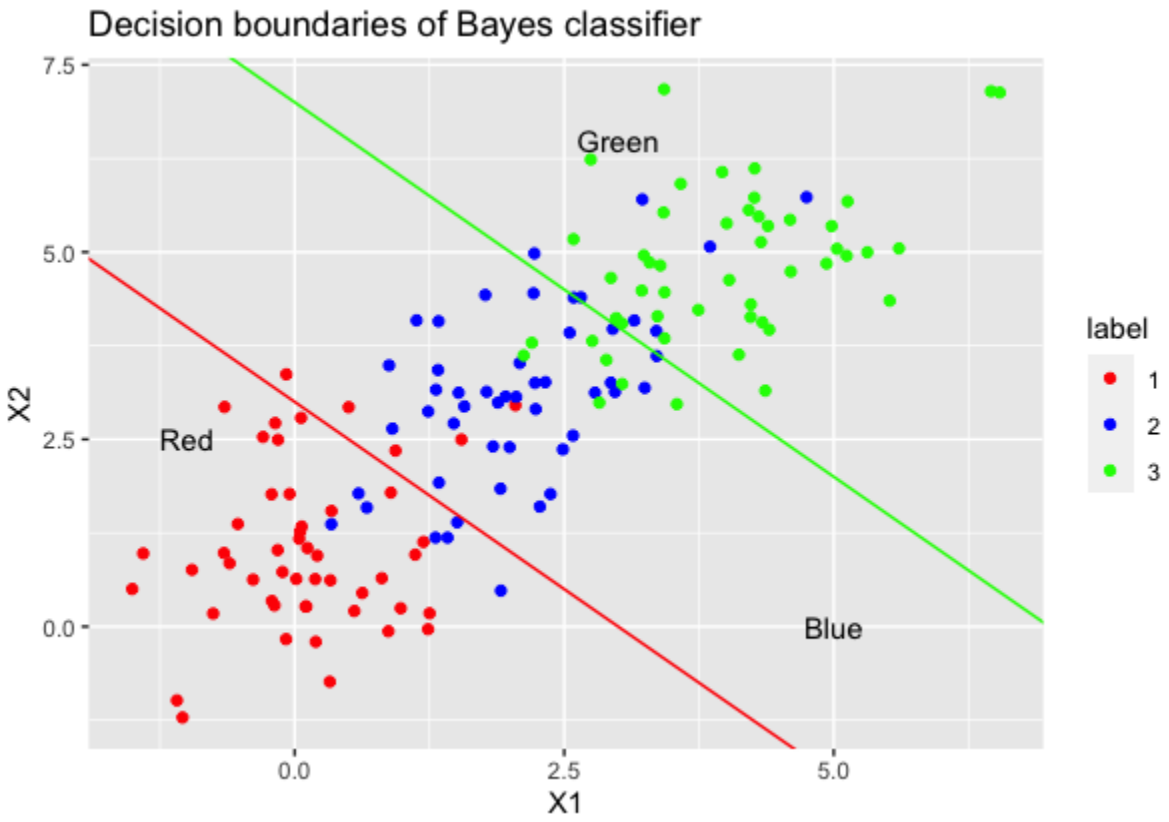
mu3 <- matrix(c(4,5))

```
library('MASS')
set.seed(100)
sigma <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)
mu1 <- matrix(c(0,1))
mu2 <- matrix(c(2,3))
mu3 <- matrix(c(4,5))

k1 <- mvrnorm(n=50,mu1,sigma)
set.seed(901)
k2 <- mvrnorm(n=50,mu2,sigma)
set.seed(302)
k3 <- mvrnorm(n=50,mu3,sigma)

df_k1 <- data.frame(k1)
df_k2 <- data.frame(k2)
df_k3 <- data.frame(k3)

df_k1$label <- as.factor('1')
df_k2$label <- as.factor('2')
df_k3$label <- as.factor('3')

data_training <- rbind(df_k1,df_k2,df_k3)
```

B. The decision boundary for Baye's classifier is derived analytically.

```
#Decision boundary for k=1 and k=2
A=(t(mu2) %*% solve(sigma) %*% mu2) -(t(mu1) %*% solve(sigma) %*% mu1)
#B=2 * t(solve(sigma) %*% (mu2-mu1))
B= 2*solve(sigma) %*% (mu2-mu1)
m1 <- -(B[1,1]/B[2,1])
c1 <- A[1,1]/B[2,1]

#Decision boundary for k=2 and k=3
A=(t(mu3) %*% solve(sigma) %*% mu3) -(t(mu2) %*% solve(sigma) %*% mu2)
#B=2 * t(solve(sigma) %*% (mu3-mu2))
B= 2*solve(sigma) %*% (mu3-mu2)
m2 <- -(B[1,1]/B[2,1])
c2 <- A[1,1]/B[2,1]

#Decision boundary for k=3 and k=1
A=(t(mu1) %*% solve(sigma) %*% mu1) -(t(mu3) %*% solve(sigma) %*% mu3)
#B=2 * t(solve(sigma) %*% (mu1-mu3))
B= 2* solve(sigma) %*% (mu1-mu3)
m3 <- -(B[1,1]/B[2,1])
c3 <- A[1,1]/B[2,1]
```

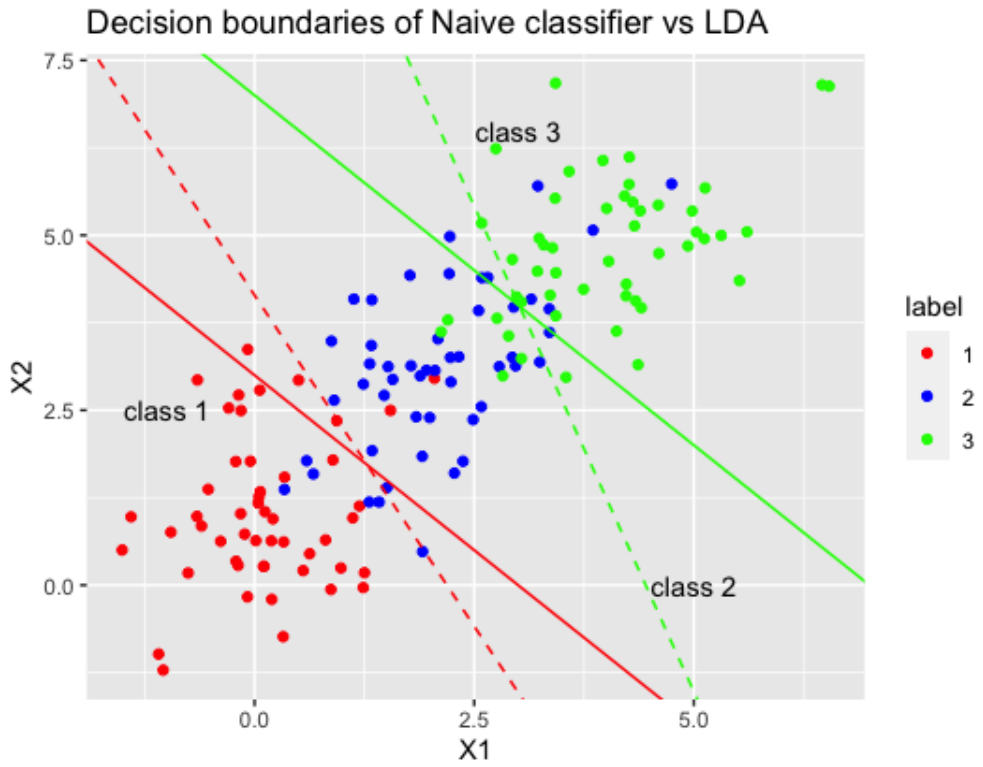
4



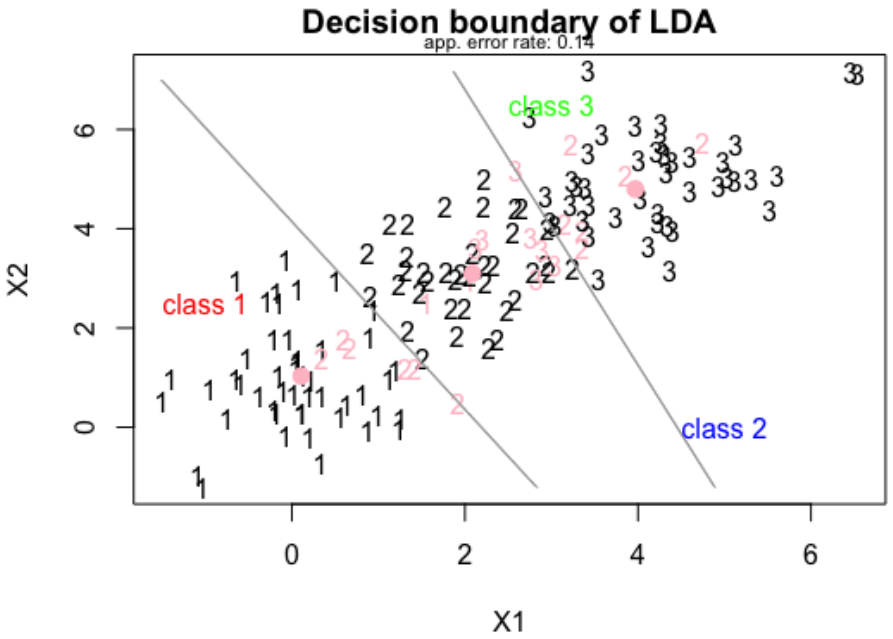
C. The decision boundary for LDA is calculated both analytically using matrix calculus and using 'drawparti' function.

```
model <- lda(factor(label)~., data = data_training)
predictions <- model %>% predict(data_training)

mu1_est=matrix(colMeans(df_k1[,c('X1','X2')]))
mu2_est=matrix(colMeans(df_k2[,c('X1','X2')]))
mu3_est=matrix(colMeans(df_k3[,c('X1','X2')]))
sigma_est<-(sigma_est_k1*50 + sigma_est_k2* 50 + sigma_est_k3*50)/(150-3)
#sigma_est <- matrix(cov(data_training[,c('X1','X2')]),nrow = 2)

#Decision boundary for k=1 and k=2
A=(t(mu2_est) %*% solve(sigma_est) %*% mu2_est) -(t(mu1_est) %*% solve(sigma_est) %*% mu1_est)
B=2 * t(solve(sigma_est) %*% (mu2_est-mu1_est))
#B= 2*solve(sigma_est) %*% (mu2_est-mu1_est)
m1_LDA <- -(B[1,1]/B[1,2])
c1_LDA <- A[1,1]/B[1,2]

#Decision boundary for k=2 and k=3
A=(t(mu3_est) %*% solve(sigma_est) %*% mu3_est) -(t(mu2_est) %*% solve(sigma_est) %*% mu2_est)
B=2 * t(solve(sigma_est) %*% (mu3_est-mu2_est))
#B= 2*solve(sigma_est) %*% (mu3_est-mu2_est)
m2_LDA <- -(B[1,1]/B[1,2])
c2_LDA <- A[1,1]/B[1,2]

#Decision boundary for k=3 and k=1
A=(t(mu1_est) %*% solve(sigma_est) %*% mu1_est) -(t(mu3_est) %*% solve(sigma_est) %*% mu3_est)
B=2 * t(solve(sigma_est) %*% (mu1_est-mu3_est))
#B= 2*solve(sigma_est) %*% (mu1_est-mu3_est)
m3_LDA <- -(B[1,1]/B[1,2])
c3_LDA <- A[1,1]/B[1,2]
```

```
plot1= ggplot(data=data_training,aes(X1,X2,color=label)) +
  geom_point()+scale_color_manual(values = c("1" = "red", "2" = "blue","3"="green"))
plot1 + ggtitle('Plot of training data')
plot1 + geom_abline(intercept = c2,slope = m2,colour='green') +
  #geom_abline(intercept = c3,slope = m3,colour='blue') +
  geom_abline(intercept = c1,slope = m1,color='red') +
  geom_abline(intercept = c2_LDA,slope = m2_LDA,colour='green',lty=2) +
  geom_abline(intercept = c1_LDA,slope = m1_LDA,color='red',lty=2)+
  #geom_abline(intercept = c3_LDA,slope = m3_LDA,colour='blue',lty=2) +
  annotate('text',x=-1,y=2.5,label='Red')+annotate('text',x=5,y=0,label='Blue')+
  annotate('text',x=3,y=6.5,label='Green') +
  ggtitle('Decision boundaries of Naive classifier vs LDA')
```



The Decision boundary of LDA using ‘drawparti’ function:



The dotted lines above are the decision boundaries of LDA. They are different from the decision boundaries of Baye's classifier but are able to separate the classes with separability close to Baye's classifier. Although the decision boundary of LDA is different from Baye's, the misclassifications are kind of the same as in Baye's classifier.

D.

```
p_train <- predictions$class
tab1 <- table(Predicted = p_train, Actual = data_training$label)
tab1
```

Predicted	Actual 1	2	3
1	48	6	0
2	2	38	7
3	0	6	43

The number of misclassified classes is = 6+7+2+6=21

Hence, the misclassification error rate in training is 21/150 = 14%

E. The confusion matrix

```
set.seed(800)
sigma <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)
mu1 <- matrix(c(0,1))
mu2 <- matrix(c(2,3))
mu3 <- matrix(c(4,5))

k1 <- mvrnorm(n=50,mu1,sigma)
set.seed(501)
k2 <- mvrnorm(n=50,mu2,sigma)
set.seed(802)
k3 <- mvrnorm(n=50,mu3,sigma)

df_k1 <- data.frame(k1)
df_k2 <- data.frame(k2)
df_k3 <- data.frame(k3)

df_k1$label <- as.factor('1')
df_k2$label <- as.factor('2')
df_k3$label <- as.factor('3')

data_test <- rbind(df_k1,df_k2,df_k3)

predictions_test <- model %>% predict(data_test)
p_test <- predictions_test$class
tab2 <- table(Predicted = p_test, Actual = data_test$label)
tab2
```

	Actual		
Predicted	1	2	3
1	46	4	0
2	4	41	5
3	0	5	45

The number of misclassified classes is = 5+4+4+5=18.

Hence, the misclassification error rate in the test data is 18/150 = 12%.

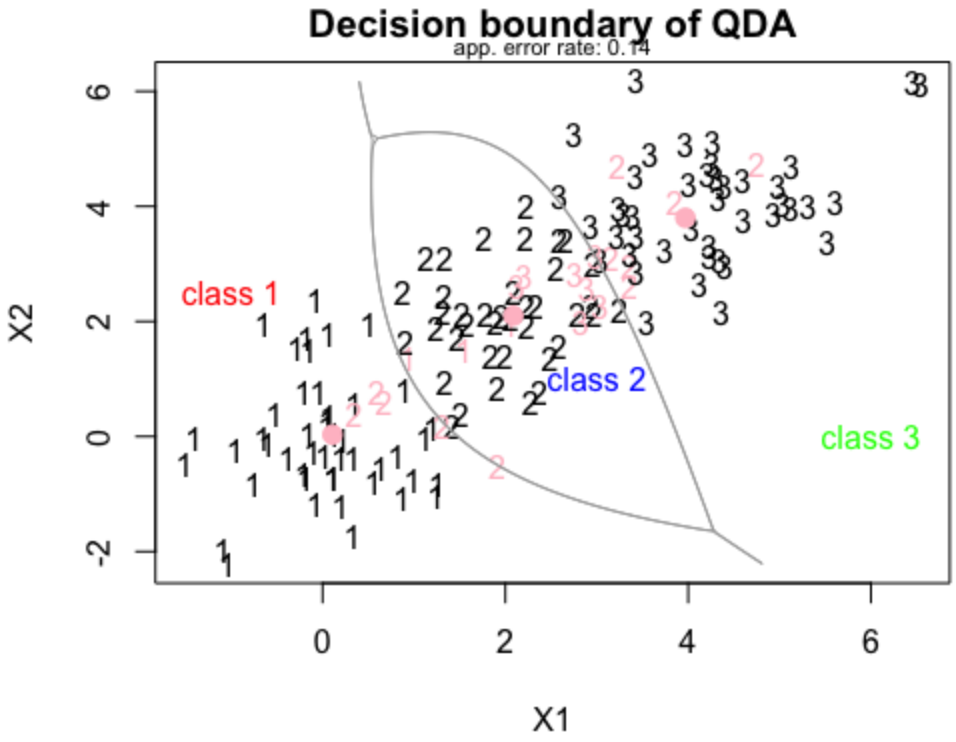
F. The training and test error rate are very similar which suggests that LDA is performing well on the data that we generated.

8.

A. model_QDA <- qda(factor(label)~., data = data_training)

The decision boundary is plotted using ‘drawparti’ function from klaR library in R.

```
library(klaR)
drawparti(data_training$label, data_training$X1, data_training$X2, method="qda", xlab="X1", ylab="X2"
          , col.wrong = "pink", imageplot = FALSE,
          legend.err= FALSE,
          col.mean = "pink")
text(-1, 2.5, label="class 1",col='red',cex=1)
text(3, 1, label="class 2",col='blue',cex=1)
text(6, 0, label="class 3",col='green',cex=1)
title(main = "Decision boundary of QDA")
```



The decision boundaries in QDA are non-linear unlike in Baye's decision boundaries. The boundaries are passing through the same region where Baye's decision boundaries were passing through but in a non-linear fashion.

B.

```
> predictions_QDA <- model_QDA %>% predict(data_training)
> p_train_QDA <- predictions_QDA$class
> tab1_QDA <- table(Predicted = p_train_QDA, Actual = data_training$label)
> tab1_QDA
         Actual
Predicted  1  2  3
        1 47  5  0
        2  3 39  7
        3  0  6 43
```

The number of misclassified classes is = 5+7+3+6=21.

Hence, the misclassification error rate in the test data is 21/150 = 14%.

C.

```
> p_test_QDA <- predictions_test_QDA$class
> tab2_QDA <- table(Predicted = p_test_QDA, Actual = data_test$label)
> tab2_QDA
         Actual
Predicted  1  2  3
        1 41  3  0
        2  9 42  5
        3  0  5 45
```

The number of misclassified classes is = 3+5+9+5=22.

Hence, the misclassification error rate in the test data is 22/150 = 14.7%.

D. The misclassification error rate in test data is higher than in the training data by 0.7%. The test data is unseen by the model and hence we can expect the error to be more. But since this is still close to training error, we can say that the model performs well in our data.
E. Both the models had the same training error i.e. 14% Since the training data has a small overlap between all classes and Baye's decision boundary is linear, both LDA and QDA are able to separate classes equally well.
F. LDA has a smaller test error of 12% than an error of 14.7% in QDA. The data generated in our case can easily be separated by linear boundaries with some misclassification error. Hence, the QDA performs equally well on the training data but it performs slightly poorer in test data because of its non-linearity which makes it flexible and that leads to low bias and high variance. Hence, it is performing poorer in test data.

9.

The ridge regression estimator is as follows:

$$\sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_{i1} + \cdots \beta_p x_{ip}))^2 + \lambda(\beta_1^2 + \cdots \beta_p^2).$$

where p is total number of features.

Here, $\beta_0 = 0$

Hence, the above equation can be reduced to

$$\|Y - X\hat{\beta}\|^2 + \lambda \hat{\beta}^T \hat{\beta} \quad \text{--- (1)}$$

where,

$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}_{n \times 1}, \quad X = \begin{pmatrix} x_{11} \cdots & x_{1p} \\ x_{21} & x_{2p} \\ \vdots & \\ x_{n1} & x_{np} \end{pmatrix}_{n \times p}.$$

$$\hat{\beta} = \begin{pmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \\ \vdots \\ \hat{\beta}_p \end{pmatrix}$$

differentiating eq (1) and equating to 0, we get:

$$\frac{\partial}{\partial \beta} \left(\|Y - X\hat{\beta}\|^2 + \lambda \hat{\beta}^T \hat{\beta} \right) = 0$$

$$\frac{\partial}{\partial \beta} \left(\|Y\|^2 + \hat{\beta}^T X^T X \hat{\beta} - 2\hat{\beta}^T X^T Y + \lambda \hat{\beta}^T \hat{\beta} \right) = 0$$

$$0 + 2X^T X \hat{\beta} - 2X^T Y + 2\lambda \hat{\beta} = 0.$$

$$X^T X \hat{\beta} - X^T Y + \lambda \hat{\beta} = 0.$$

$$\hat{\beta}(X^T X + \lambda I) = X^T Y.$$

$$\boxed{\hat{\beta} = [X^T X + \lambda I]^{-1} X^T Y}$$