James Yang

DATA 558

Homework 2

1.

a. Fit a least squares linear model to the data, in order to predict mpg using all of the other predictors except for name. Present your results in the form of a table. Be sure to indicate clearly how any qualitative variables should be interpreted.

```
Residuals:
    Min      1Q  Median      3Q     Max
-9.5903 -2.1565 -0.1169  1.8690 13.0604

Coefficients:
                   Estimate Std. Error t value
(Intercept)       -17.218435   4.644294  -3.707
Auto$cylinders     -0.493376   0.323282  -1.526
Auto$displacement   0.019896   0.007515   2.647
Auto$horsepower    -0.016951   0.013787  -1.230
Auto$weight        -0.006474   0.000652  -9.929
Auto$acceleration   0.080576   0.098845   0.815
Auto$year           0.750773   0.050973  14.729
Auto$origin         1.426141   0.278136   5.127
                  Pr(>|t|)
(Intercept)        0.00024 ***
Auto$cylinders     0.12780
Auto$displacement  0.00844 **
Auto$horsepower    0.21963
Auto$weight        < 2e-16 ***
Auto$acceleration  0.41548
Auto$year          < 2e-16 ***
Auto$origin       4.67e-07 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.328 on 384 degrees of freedom
Multiple R-squared:  0.8215,    Adjusted R-squared:  0.8182
F-statistic: 252.4 on 7 and 384 DF,  p-value: < 2.2e-16
```

When doing summary of the lm, the standard error is the variability to expect in the coefficients. So for example, for cylinders we can expect a variation of 0.323 cylinders. The T value is the coefficient/std.error. When looking at the Pr(>= |t|) values, we can see that if the values are greater than the standard alpha of 0.05, then we know that the coefficients are not statistically significant in our model. The residual standard error is 3.328, which means it can be off by an average of that amount when predicting mpg. Multiple R-Squared is the ratio of (1-(sum of squared error/sum of squared total)).

b. What is the (training set) mean squared error of this model?

Taking the residuals of the summary squared, we get: 10.84748

c. What gas mileage do you predict for a Japanese car with three cylinders, displacement 100, horsepower of 85, weight of 3000, acceleration of 20, built in the year 1980?

```
Coefficients:
      (Intercept)     Auto$cylinders  Auto$displacement    Auto$horsepower
        -17.218435          -0.493376           0.019896          -0.016951
      Auto$weight  Auto$acceleration          Auto$year        Auto$origin
        -0.006474           0.080576           0.750773           1.426140
```

Add/Subtract the numbers:

$$-1.480128 + 1.9896 - 14.40835 - 19.422 + 1.61152 + 60.06184 = 28.352482$$

$$= 28.352482$$

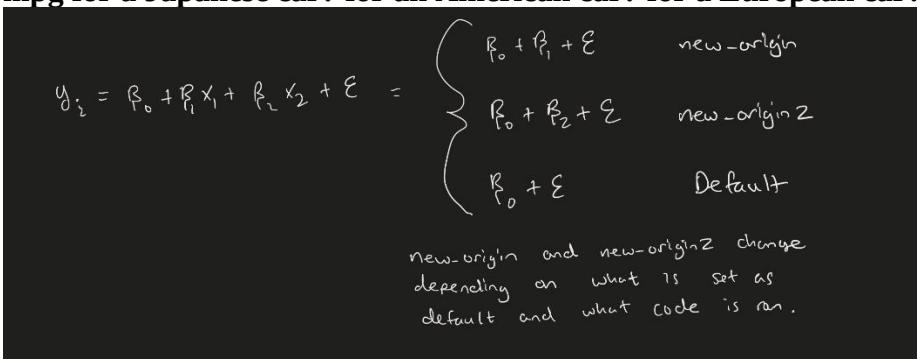
Multiplying these coefficients with the respective values above we get:
Mpg = 28.352

d. On average, holding all other covariates fixed, what is the difference between the mpg of a Japanese car and the mpg of an American car?
American cars get an average of 20.03 mpg and Japanese cars get an average of 30.45 mpg. The difference is 10.417 mpg.

e. On average, holding all other covariates fixed, what is the change in mpg associated with a 10-unit change in horsepower?
-0.016951*10 = -0.16951 MPG with every 10-unit increase. If we decrease, then it would be 0.16951 MPG increase.

2.

a. First, code the origin variable using two dummy (indicator) variables, with Japanese as the default value. Write out an equation like (3.30) in the textbook, and report the coefficient estimates. What is the predicted mpg for a Japanese car? for an American car? for a European car?

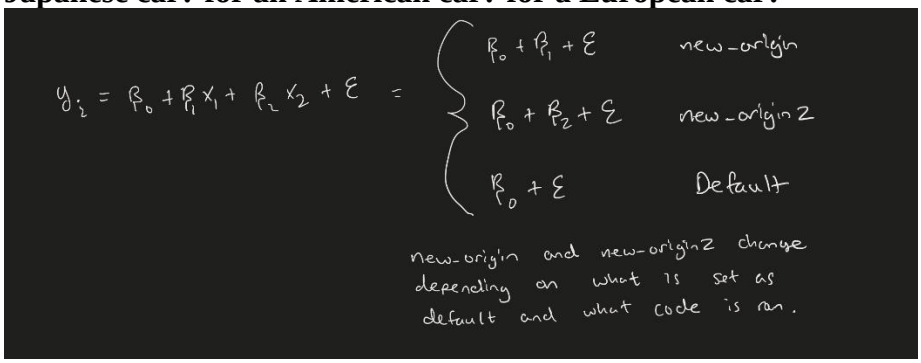


If new origin german 1, else 0.
If new origin2 american 1, else 0.

```
Call:
lm(formula = Autodummy$mpg ~ Autodummy$new_origin + Autodummy$new_origin2)

Residuals:
    Min      1Q  Median      3Q     Max
-12.451  -5.034  -1.034   3.649  18.966

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          30.4506     0.7196  42.314  < 2e-16 ***
Autodummy$new_origin -10.4172     0.8276 -12.588  < 2e-16 ***
Autodummy$new_origin2  -2.8477     1.0581  -2.691  0.00742 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.396 on 389 degrees of freedom
Multiple R-squared:  0.3318,    Adjusted R-squared:  0.3284
F-statistic:  96.6 on 2 and 389 DF,  p-value: < 2.2e-16
```

American MPG: 20.033
Japanese MPG: 30.451
Germany MPG: 27.603

b. Now, code the origin variable using two dummy (indicator) variables, with American as the default. Write out an equation like (3.30) in the textbook, and report the coefficient estimates. What is the predicted mpg for a Japanese car? for an American car? for a European car?

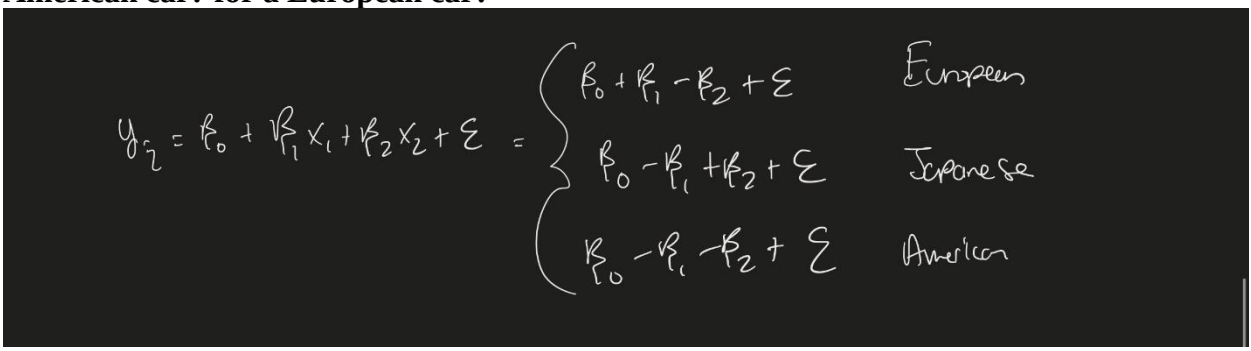


If new origin is Japanese 1, else 0.
If new origin 2 is german 1, else 0.

```
Call:
lm(formula = Auto$mpg ~ Auto$new_origin + Auto$new_origin2, data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-12.451  -5.034  -1.034   3.649  18.966

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         20.0335     0.4086  49.025   <2e-16 ***
Auto$new_origin     10.4172     0.8276  12.588   <2e-16 ***
Auto$new_origin2     7.5695     0.8767   8.634   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.396 on 389 degrees of freedom
Multiple R-squared:  0.3318,    Adjusted R-squared:  0.3284
F-statistic:  96.6 on 2 and 389 DF,  p-value: < 2.2e-16
```

American MPG: 20.033
Japanese MPG: 30.451
Germany MPG: 27.603

c. Now, code the origin variable using two variables that take on values of +1 or -1. Write out an equation like (3.30) in the textbook, and report the coefficient estimates. What is the predicted mpg for a Japanese car? for an American car? for a European car?

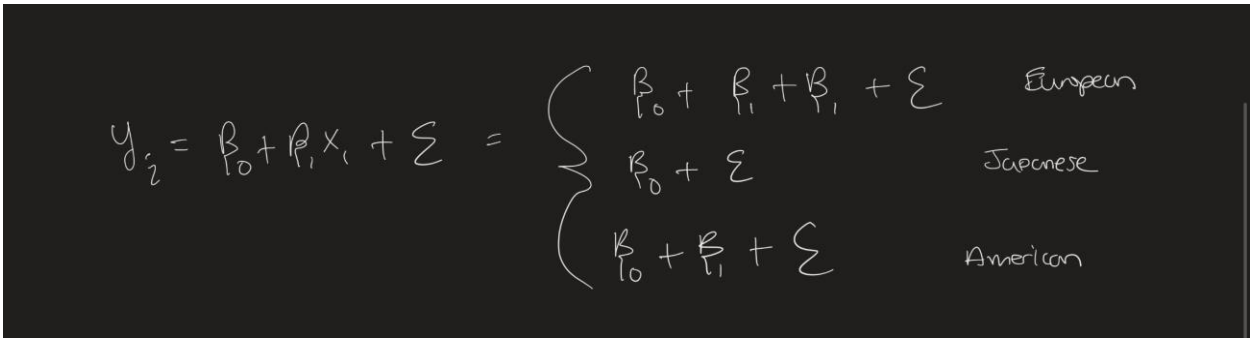


If new origin is German, it will give 1 else -1.
If new origin2 is Japanese, it will give 1 else -1.

```
Call:
lm(formula = Auto$mpg ~ Auto$new_origin + Auto$new_origin2, data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-12.451  -5.034  -1.034   3.649  18.966

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       29.0268     0.5290  54.867   <2e-16 ***
Auto$new_origin    3.7847     0.4384   8.634   <2e-16 ***
Auto$new_origin2   5.2086     0.4138  12.588   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.396 on 389 degrees of freedom
Multiple R-squared:  0.3318,    Adjusted R-squared:  0.3284
F-statistic:  96.6 on 2 and 389 DF,  p-value: < 2.2e-16
```

Japanese MPG: 30.532
American MPG: 19.952
Germany MPG: 27.522

d. Finally, code the origin variable using a single variable that takes on values of 0 for Japanese, 1 for American, and 2 for European. Write out an equation like (3.30) in the textbook, and report the coefficient estimates. What is the predicted mpg for a Japanese car? for an American car? For a European car?

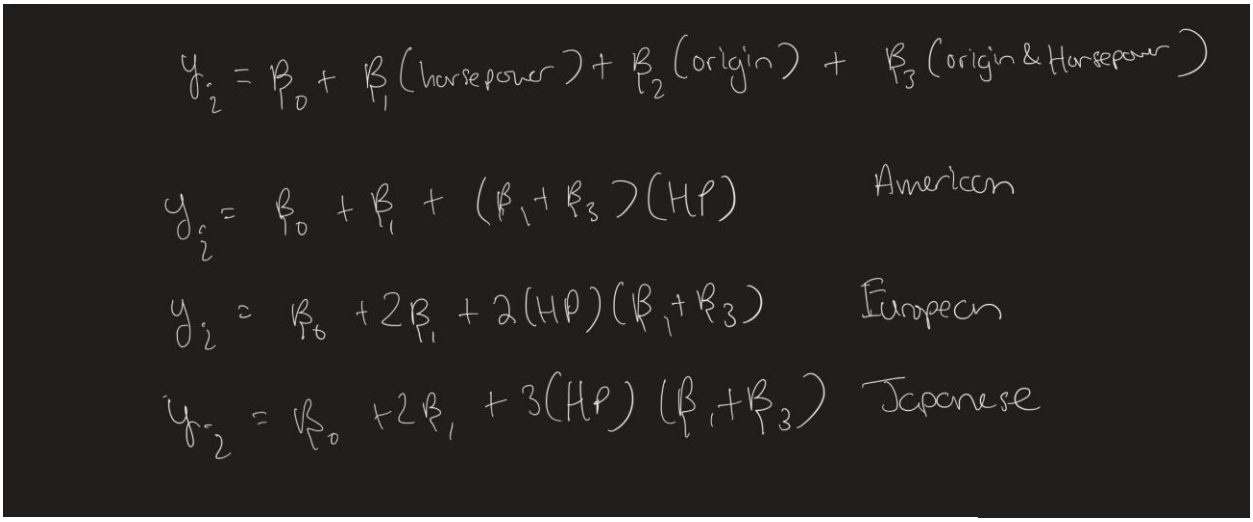



```
Call:
lm(formula = Auto$mpg ~ Auto$new_origin, data = Auto)

Residuals:
    Min      1Q  Median      3Q     Max
-14.394  -6.239  -1.167   5.761  22.751

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      25.2395     0.7332   34.42  < 2e-16
Auto$new_origin  -1.8453     0.6384   -2.89  0.00406

(Intercept)     ***
Auto$new_origin **
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.733 on 390 degrees of freedom
Multiple R-squared:  0.02097,   Adjusted R-squared:  0.01846
F-statistic: 8.354 on 1 and 390 DF,  p-value: 0.004063
```

American MPG: 23.3942
Japanese MPG: 25.2395
European MPG: 21.5489

e. Comment on your results in (a)-(d).
The values found in part c were a bit different than the ones found in the rest of the parts because there was a fluctuation in the values of points in which they were being put in. This caused a shift in values for the MPG. The cars that are American also were being valued less because they were no longer Japanese which meant that their MPG would get docked on the fact that they weren't Japanese.

3. **Fit a model to predict mpg on the Auto dataset using origin and horsepower, as well as an interaction between origin and horsepower. Present your results, and write out an equation like (3.35) in the textbook. On average, how much does the mpg of a Japanese car change with a one-unit increase in horsepower? How about the mpg of an American car? a European car?**




```
Call:
lm(formula = Auto$mpg ~ Auto$horsepower + Auto$origin + Auto$horsepower *
    Auto$origin)

Residuals:
     Min       1Q   Median       3Q      Max
-10.8206  -3.1504  -0.5536   2.3682  15.2386

Coefficients:
                             Estimate Std. Error t value
(Intercept)                  26.79098    1.69728  15.785
Auto$horsepower              -0.05942    0.01662  -3.574
Auto$origin                   7.87119    1.13907   6.910
Auto$horsepower:Auto$origin -0.06338    0.01312  -4.832
                             Pr(>|t|)
(Intercept)                   < 2e-16 ***
Auto$horsepower              0.000396 ***
Auto$origin                  2.00e-11 ***
Auto$horsepower:Auto$origin 1.95e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.424 on 388 degrees of freedom
Multiple R-squared:  0.6812,    Adjusted R-squared:  0.6788
F-statistic: 276.4 on 3 and 388 DF,  p-value: < 2.2e-16
```

The values change by a factor of the beta_2 and beta_3 multiplied by HP.
American Change in MPG: -0.1228
European Change in MPG: -0.2456
Japanese Change in MPG: -0.3684

4. For these questions, I have used the gcookbook dataset with a random set of height and weight in it.
    a. When solving the general equation, we get **Y = 142.1.**
    b. When plugging in a gcookbook heightweight dataset and performing an lm on it, I get that
       Y = 44.867x1 – 128.328. For someone to be 5.33 feet, we predict **Y = 110.96.**
    c. $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon$
       $= \boldsymbol{\beta_0 + \beta_1 X_1 + \beta_2 X_1 * 12 + \varepsilon}$
       $= \boldsymbol{\beta_0 + (\beta_1 + 12\beta_2) X_1 + \varepsilon}$
       We know the value of the beta coefficient is 4.8.
       $$(\boldsymbol{\beta_1 + 12\beta_2}) - \boldsymbol{4.8} + \boldsymbol{\varepsilon} = \boldsymbol{0}$$

d. According to the test set from gcookbook, the MSE values for all 3 models appear to be the same. The reason is because the error found is just going to be a factor of 12, meaning their errors should still differentiate the same and average out to be the same as well.

5.

Suppose we wish to perform classification of a binary response in a setting with $p = 1$: that is, $X \in \mathbb{R}$, and $Y \in \{1, 2\}$. We assume that the observations in Class 1 are drawn from a $N(\mu, \sigma^2)$ distribution, and that the observations in Class 2 are drawn from an Uniform$[-2, 2]$ distribution.

a.

$$P(y=1 \mid X=x) = P(y=2 \mid X=x)$$

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

However, $P(y=1 \mid X=x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-0.5\left(\frac{x-\mu_1}{\sigma}\right)^2}$

$$P(y=2 \mid X=x) = \frac{1}{2-(-2)} = \frac{1}{4}$$

$$\frac{1}{\sqrt{2\pi}\sigma} e^{-0.5\left(\frac{x-\mu}{\sigma}\right)^2} = \frac{1}{4}$$

b.

$$P(y=1 \mid x=x) = P(y=2 \mid x=x)$$

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

However, $P(y=1 \mid x=x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-0.5\left(\frac{x-\mu_1}{\sigma}\right)^2}$

$$P(y=2 \mid x=x) = \frac{1}{2-(-2)} = \frac{1}{4}$$

$$\pi_1 \frac{1}{\sqrt{2\pi}\sigma} e^{-0.5\left(\frac{x-\mu}{\sigma}\right)^2} = \frac{1}{4}\pi_2$$

$$P(y=1 \mid x=x)\,\pi_1 = P(y=2 \mid x=x)\,\pi_2$$

$$\frac{1}{\sqrt{2\pi}} e^{-0.5x^2} \cdot 0.45 = \frac{1}{4}(0.55)$$

$$\frac{1}{\sqrt{2\pi}} e^{-0.5x^2} = 0.3056$$

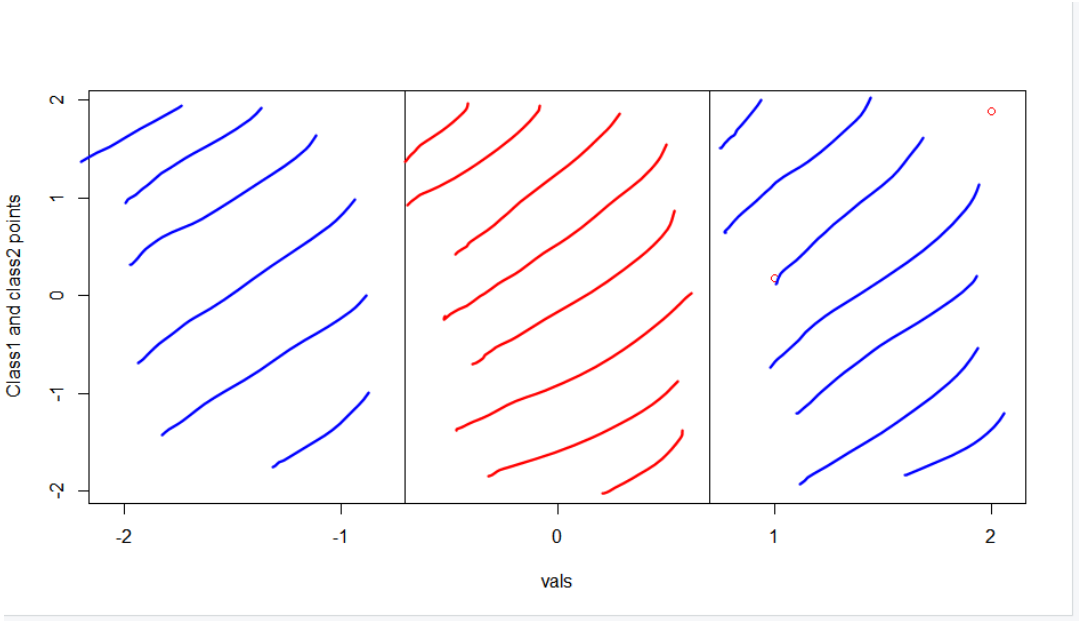
$$e^{-0.5x^2} = 0.7659$$

$$-0.5x^2 = \ln(0.7659)$$

$$-0.5x^2 = -0.2667$$

$$x \doteq 0.7303$$

+/- 0.703



Red is class1, Blue is class2. The black lines represent the +/- 0.703 value found in the previous section.

c. We can use the training observations to estimate n sample mean, sample SD, and pi values relating to class 1.

For π_i, we simply sum all of the values sampled where y = 1 multiplied by an identity matrix, and divide it by n number of samples.

For μ, we can take the sum of all the x observations where y = 1, and divide it by the sum of all the values sampled where y = 1 multiplied by an identity matrix.

For σ, we can get the sum of the x values subtracted by the average of x and divide it by the sum of all the values sampled where y = 1 multiplied by an identity matrix – 1.

d.

$$P(y=1 \mid X=x_0) = \frac{P(X=x_0 \mid y=1)\,P(y=1)}{P(X=x_0)} \quad \text{by Bayes Thm.}$$

$$= \frac{P(X=x_0 \mid y=1)\,P(y=1)}{P(X=x_0 \mid y=1)P(y=1) + P(X=x_0 \mid y=2)P(y=2)}$$

With these values:

$$P(X=x_0 \mid y=1) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$P(X=x_0 \mid y=2) = \frac{1}{4} \quad -2 \le x \le 2$$

$$\rightarrow \quad P(y=1) = \frac{\sum_{i=1}^{n} 1(y_i=1)}{n}$$

$$P(y=2) = 1 - P(y=1)$$

6.

a.

$$\log\left(\frac{Pr(y=1 \mid X=x)}{1-Pr(y=1 \mid X=x)}\right) = X\beta = \text{log odds}$$

$$= \log\left(\frac{Pr(y=1 \mid X=x)}{1-Pr(y=1 \mid X=x)}\right) = 0.7$$

$$\frac{Pr(y=1 \mid X=x)}{1-Pr(y=1 \mid X=x)} = e^{0.7}$$

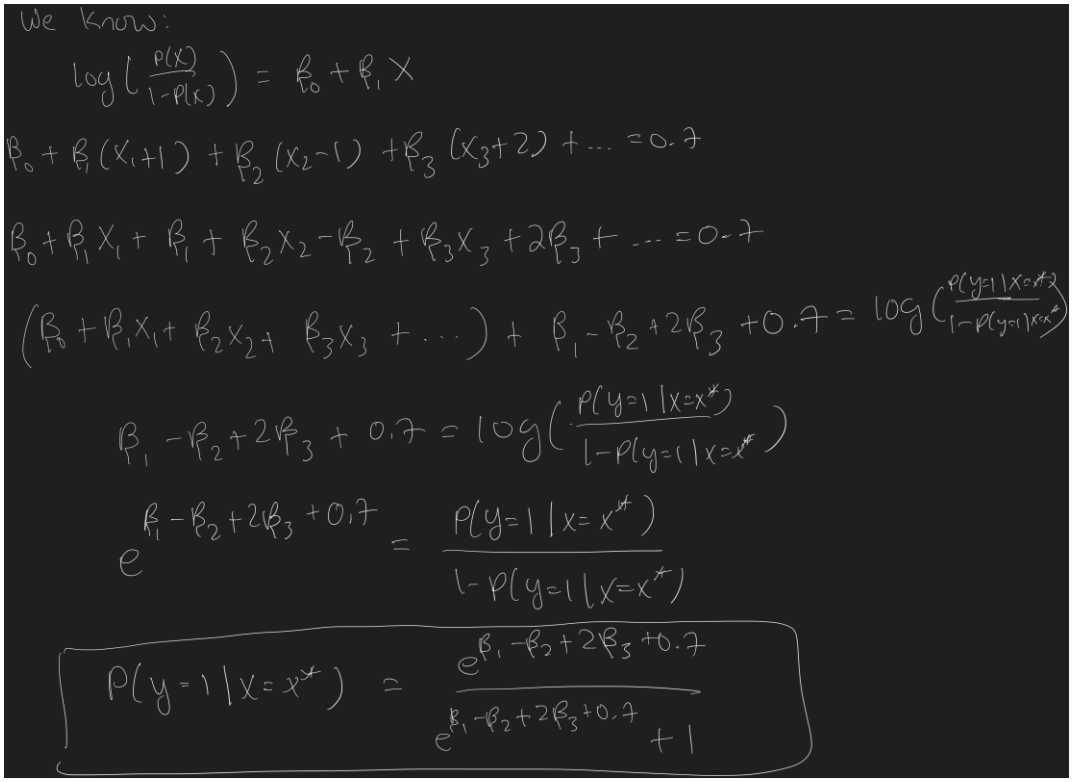
$$Pr(y=1 \mid X=x) = e^{0.7}(1 - Pr(y=1 \mid X=x))$$

$$Pr(y=1 \mid X=x) = e^{0.7} - e^{0.7} Pr(y=1 \mid X=x)$$

$$1 + e^{0.7}(Pr(y=1 \mid X=x)) = e^{0.7}$$

$$Pr(y=1 \mid X=x) = \frac{e^{0.7}}{1+e^{0.7}}$$

b.

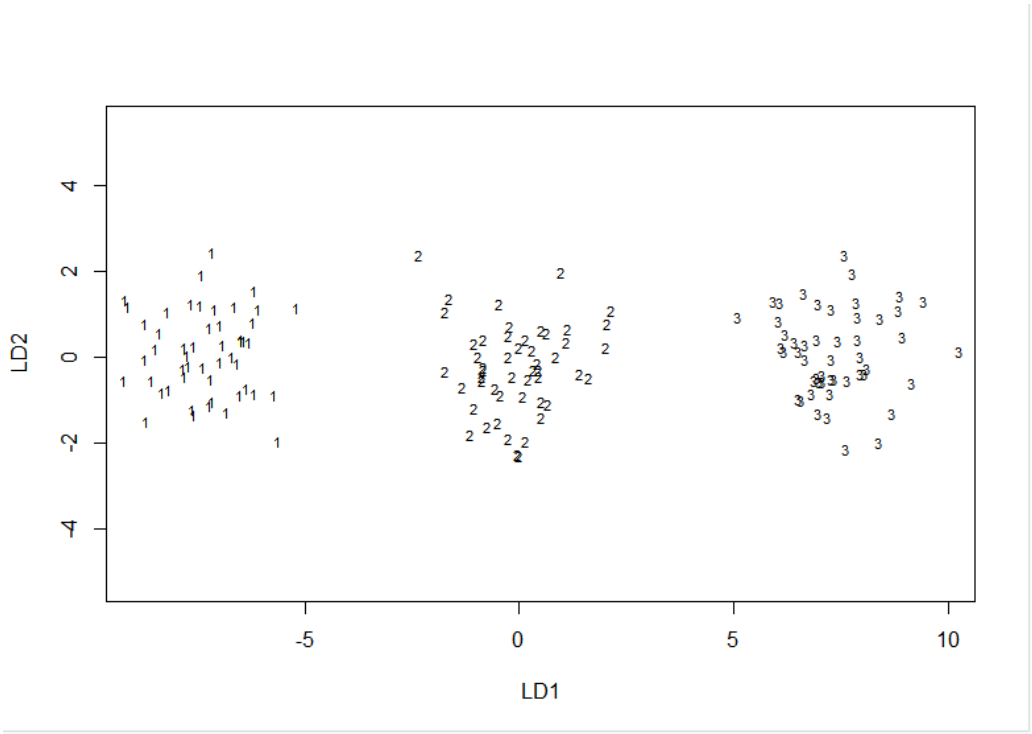


We can take advantage of beta + betax + … = 0.7 to substitute the values extracted from the x* to generate another 0.7 within the equation to make it look very similar to part a.

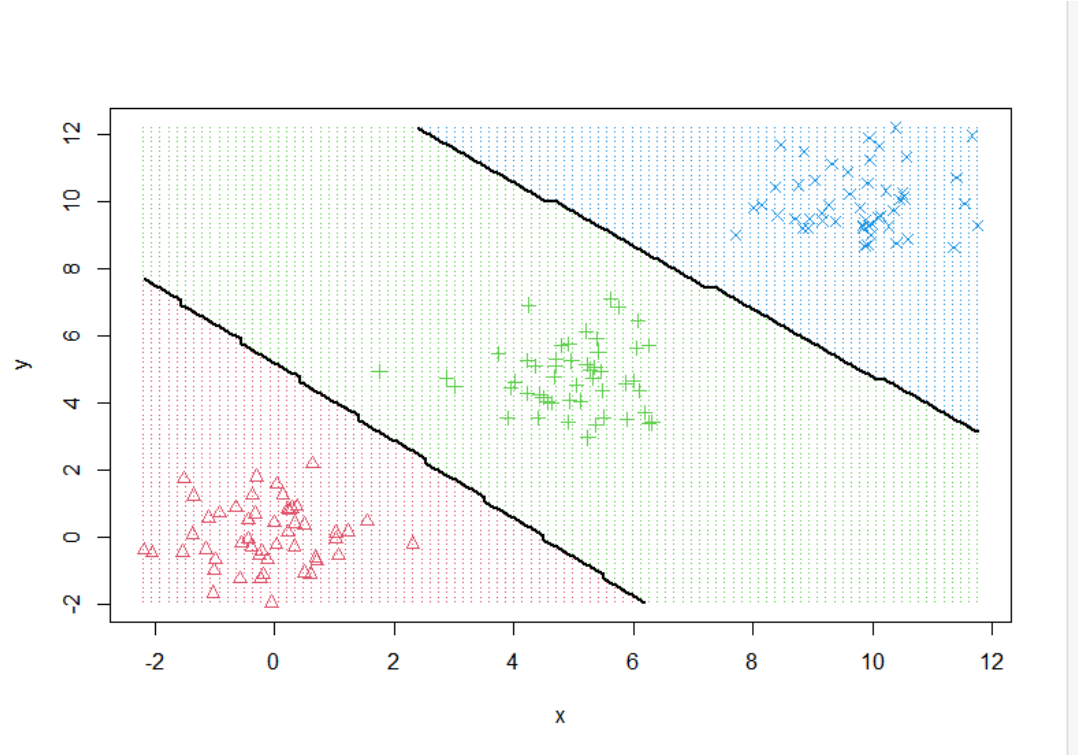
7.

a. $\boldsymbol{\Sigma = I \text{ or } 1}$
$\boldsymbol{\mu_1 = [0, 0]^T}$
$\boldsymbol{\mu_2 = [5, 5]^T}$
$\boldsymbol{\mu_3 = [10, 10]^T}$

b. .



c.



d.

Training	1	2	3
1	50	0	0
2	0	50	0

3	0	0	50

Training Error: 0%

e.

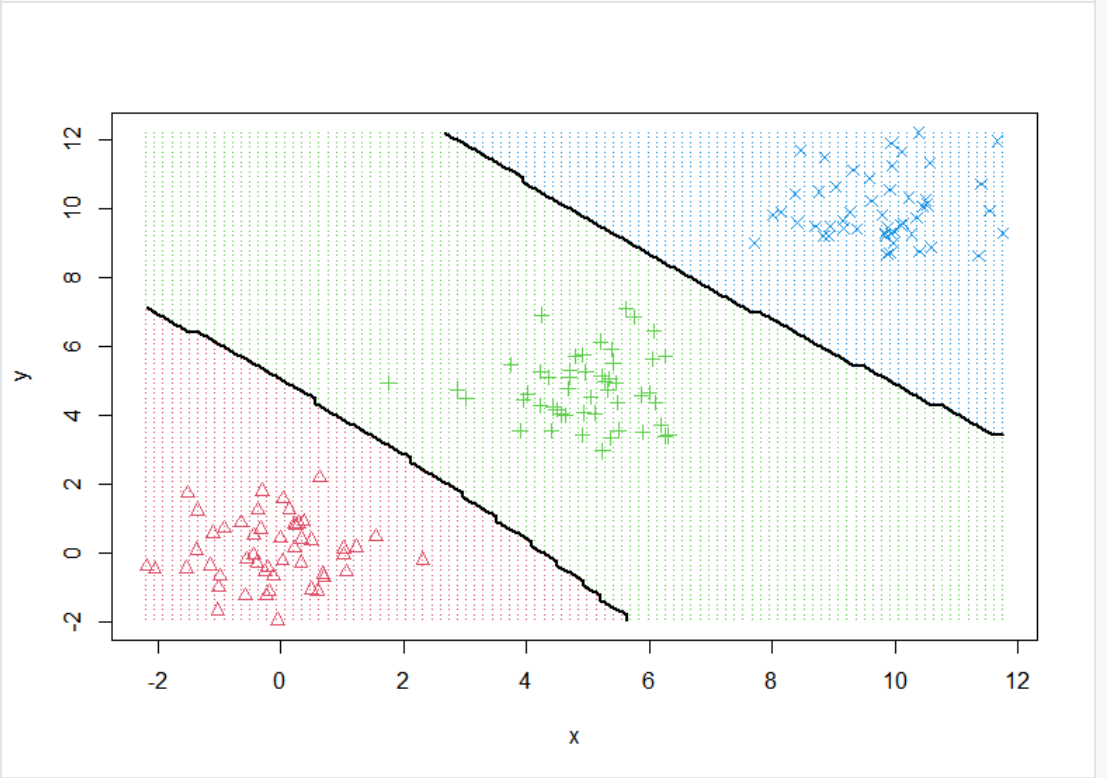
Testing	1	2	3
1	50	0	0
2	0	50	0
3	0	0	50

Testing error: 0%

f. It's interesting that my model received 0% training error but I can believe it because the distributions of my 3 variables are very different from each other and should separate as such. This makes it very difficult for the training dataset to receive any sort of error. It is interesting how the testing dataset found so much error, especially since the training found almost none but it also makes sense because LDA testing error has typically been much greater when placed on smaller sample sizes and I find this small sample to be no different. If we were to train an n amount of say 1000 or even 100000, it would most certainly have a lesser error rate.

8.

a. Using the same set of data from question 7 but switching to qda().



b. This time my training set got 1 incorrect, giving me a .67% error.

Training	1	2	3
1	49	0	0
2	0	50	0
3	1	0	50

c. This time my testing set got 0 incorrect, giving me an error of 0%.

Testing	1	2	3
1	50	0	0
2	0	50	0
3	0	0	50

d. I have been getting pretty consistent errors in my results. The main reason its giving me a about the same error in testing is most likely because my graphs are separated by a lot, but the middle values are being misclassified. LDA tends to be better than QDA if there are few training observations, but reducing variance is crucial. The variance in my third class is a lot greater than the variance found in my first class.

e. QDA had a much smaller error but had an increase in training error (although not very much – basically 0%). They both have relatively small training errors because their spacing between distributions is so far in relation to each other. If the space was more tightly pushed, there would be more discrepancies causing more error to occur.

f. Both models had some testing error but QDA had smaller testing error because of a few outlier points being pushed from class 1 and class 2. They are causing a shift in how the classifier is being generated, creating some variance in testing error. My values will shift from 24% to 30%, but I chose to keep the smaller error in the report. The reason behind this may be because of small sample size.

Code Referenced:

Q1:

```
dataset <- Auto

a <- lm(formula = Auto$mpg ~ Auto$cylinders + Auto$displacement + Auto$horsepower + Auto$weight + Auto$acceleration + Auto$year + Auto$origin)
temp <- summary(a)
mean(temp$residuals^2)



Americans <- Auto[Auto$origin == "1",]
ampg <- mean(Americans$mpg)

Japanese <- Auto[Auto$origin == "3",]
jmpg <- mean(Japanese$mpg)

vals <- c()



jmpg - ampg
```

Q2:

```
#################### Q2
Autodummy <- Auto
# for(i in 1:392) {
#   if(Autodummy$origin[i] != 3) { #if its japanese
#     Autodummy$new_origin[i] <- 1
#   }
#   else {
#     Autodummy$new_origin[i] <- 0
#   }
# }
# for(i in 1:392) {
#   if(Autodummy$origin[i] != 2) {
#     Autodummy$new_origin2[i] <- 1
#   }
#   else {
#     Autodummy$new_origin2[i] <- 0
#   }
# }
#gate missing
Auto$origineurope <- ifelse(Auto$origin == 2, 1, 0)
Auto$originamerica <- ifelse(Auto$origin == 1, 1, 0)
model_2a <- lm(mpg ~ origineurope + originamerica, data=Auto)

Auto$new_origin <- ifelse(Auto$origin == 3, 1, 0)
Auto$new_origin2 <- ifelse(Auto$origin == 2, 1, 0)
model_2b <- lm(Auto$mpg ~ Auto$new_origin + Auto$new_origin2, data = Auto)

Auto$new_origin <- ifelse(Auto$origin == 2, 1, -1)
Auto$new_origin2 <- ifelse(Auto$origin == 3, 1, -1)
model_2c <- lm(Auto$mpg ~ Auto$new_origin + Auto$new_origin2, data = Auto)

japanese <- c()
germany <- c()
usa <- c()

for(i in 1:392) {
  if(Auto$origin[i] == "3") {
    Auto$new_origin[i] = "0"
  }
  else if(Autodummy$origin[i] == "1") {
    Auto$new_origin[i] = "1"
  }
  else {
    Auto$new_origin[i] = "2"
  }
}
model_2d <- lm(Auto$mpg ~ Auto$new_origin, data=Auto)


################################### Q3
```

Q3:

```
################################### Q3
c <- lm(formula = Auto$mpg ~ Auto$horsepower + Auto$origin)
summary(c)
```

Q4: Used a sample dataset to get values.

```
################################Q4

library(gcookbook)

#Get inch to feet
val <- heightweight["heightIn"] / 12
heightweight["heightFt"] <- val


inches <- lm(weightLb ~ heightIn, data=heightweight)
feet <- lm(weightLb ~ heightFt, data=heightweight)
combined <- lm(weightLb ~ heightFt+heightIn, data=heightweight)

mean(inches$residuals^2)
mean(feet$residuals^2)
mean(combined$residuals^2)
```

Q6: Plotted horizontal +/- 0.703 lines

Q7/8:

```
matrixval <- c(1,0,0,1)
num = 50
################### 7A

### TRAINING ###
class_1 = mvrnorm(num, mu=c(0,0), Sigma=matrix(matrixval, 2))
colnames(class_1) <- c("x1","x2")

class_2 = mvrnorm(num, mu=c(5,5), Sigma=matrix(matrixval, 2))
colnames(class_2) <- c("x1","x2")

class_3 = mvrnorm(num, mu=c(10,10), Sigma=matrix(matrixval, 2))
colnames(class_3) <- c("x1","x2")
training = data.frame(x=append(append(class_1[,1], class_2[,1]), class_3[,1]), y=append(append(class_1[,2], class_2[,2]), class_3[,2]), actual=append(append(rep("1", 50), rep("2", 50)), rep("3", 50)))

all_vals = cbind(all_x, all_y)


### TESTING ####
class_1 = mvrnorm(num, mu=c(0,0), Sigma=matrix(matrixval, 2))
colnames(class_1) <- c("x1","x2")

class_2 = mvrnorm(num, mu=c(5,5), Sigma=matrix(matrixval, 2))
colnames(class_2) <- c("x1","x2")

class_3 = mvrnorm(num, mu=c(10,10), Sigma=matrix(matrixval, 2))
colnames(class_3) <- c("x1","x2")
testing = data.frame(x=append(append(class_1[,1], class_2[,1]), class_3[,1]), y=append(append(class_1[,2], class_2[,2]), class_3[,2]), actual=append(append(rep("1", 50), rep("2", 50)), rep("3", 50)))

all_vals = cbind(all_x, all_y)
#################### 7a
```

```
first <- training[training$actual == 1,]
second <- training[training$actual == 2,]
third <- training[training$actual == 3,]

plot(first$x, first$y, col = "red", xlim = c(-5, 15), ylim = c(-15, 15), xlab = "x values", ylab = "y values")
points(second$x, second$y, col = "blue")
points(third$x, third$y, col = "green")

library(klaR)
partimat(all_vals ~ x + y, data = df_7a, method ="lda")
mdl <- lda(training$actual ~ x + y, data = training)
pred_mdl <- predict(mdl, testing[,-3])
cm = as.matrix(table(Actual = training$actual, Predicted = pred_mdl$class))
cm
```

```
################################### 8A
first <- training[training$actual == 1,]
second <- training[training$actual == 2,]
third <- training[training$actual == 3,]

plot(first$x, first$y, col = "red", xlim = c(-5, 15), ylim = c(-15, 15), xlab = "x values", ylab = "y values")
points(second$x, second$y, col = "blue")
points(third$x, third$y, col = "green")

library(klaR)
partimat(all_vals ~ x + y, data = df_7a, method ="lda")
mdl <- qda(training$actual ~ x + y, data = testing)
pred_mdl <- predict(mdl, testing[,-3])
cm = as.matrix(table(Actual = training$actual, Predicted = pred_mdl$class))
cm
```

```r
###################################### 7/8
decisionplot <- function(model, data, class = NULL, ...) {
  cl <- data[,class]
  data <- data[,1:2]
  k <- length(unique(cl))
  plot(data, col = as.integer(cl)+1L, pch = as.integer(cl)+1L, ...)
  r <- sapply(data, range, na.rm = TRUE)
  xs <- seq(r[1,1], r[2,1], length.out = 100)
  ys <- seq(r[1,2], r[2,2], length.out = 100)
  g <- cbind(rep(xs, each=100), rep(ys, time = 100))
  colnames(g) <- colnames(r)
  g <- as.data.frame(g)
  p <- predict(model, g, type = "class")
  p <- p$class
  p <- as.factor(p)
  points(g, col = as.integer(p)+1L, pch = ".")
  z <- matrix(as.integer(p), nrow = 100, byrow = TRUE)
  contour(xs, ys, z, add = TRUE, drawlabels = FALSE, lwd = 2, levels = (1:(k-1))+.5)
}
ldamodel <- lda(actual ~ x + y, data=training)
decisionplot(ldamodel, training, class="actual")

qdamodel <- qda(actual ~ x + y, data = training)
decisionplot(qdamodel, training, class="actual")
```