Mini Project 2

John Kenney

9/21/2021

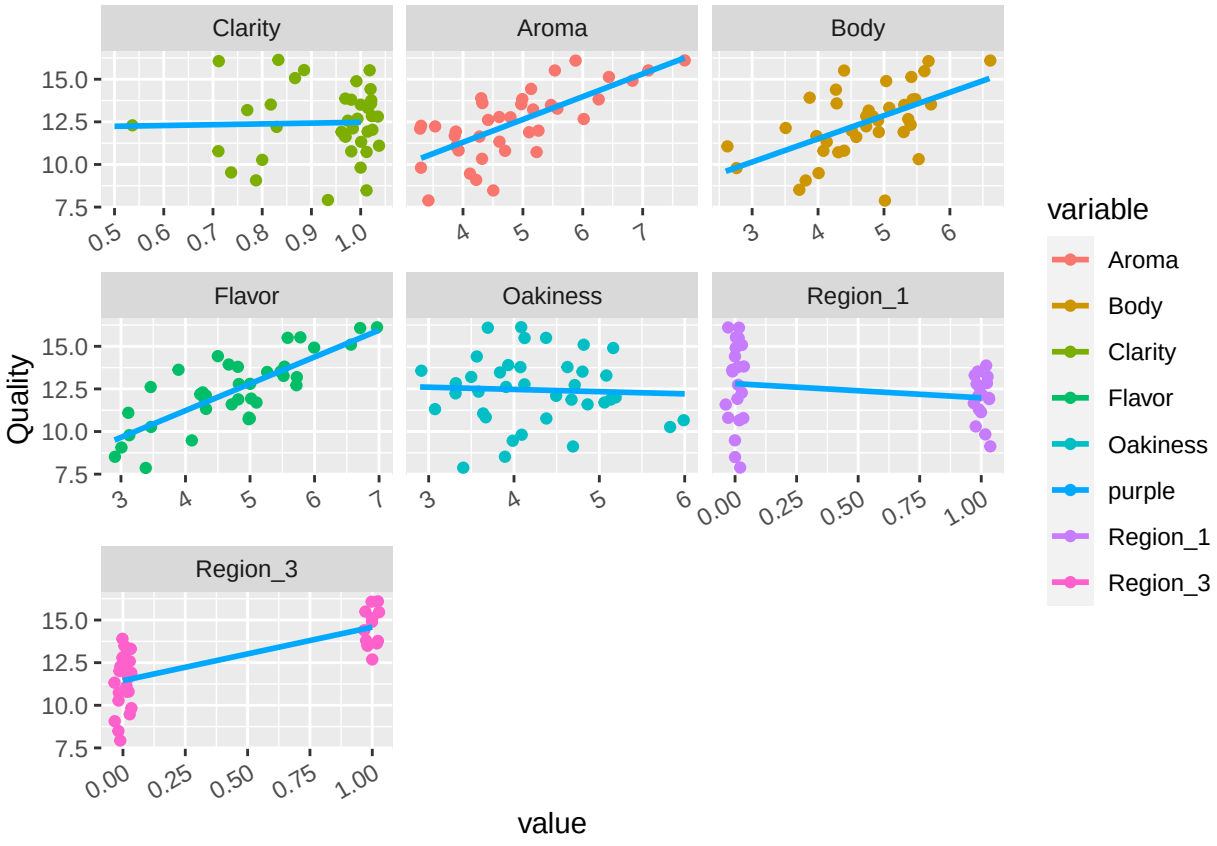
Section1

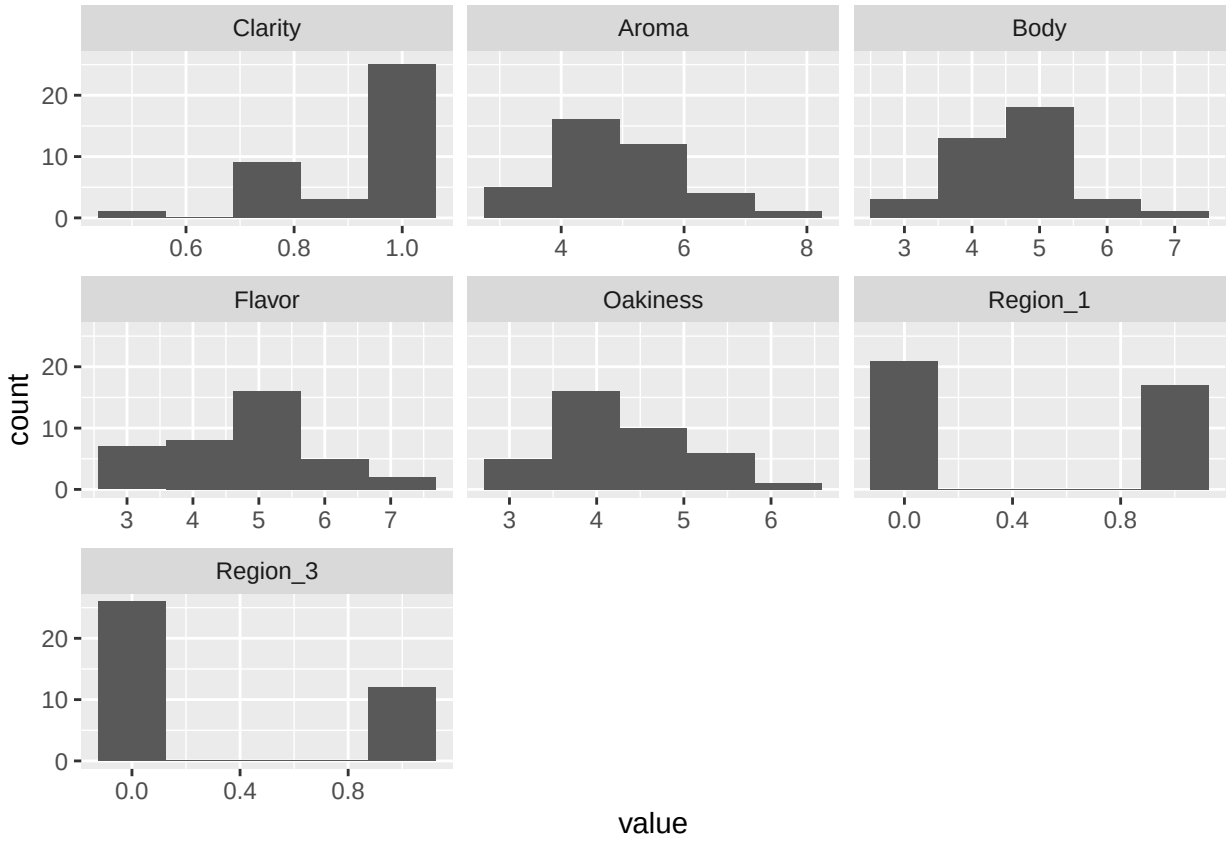
Question 1

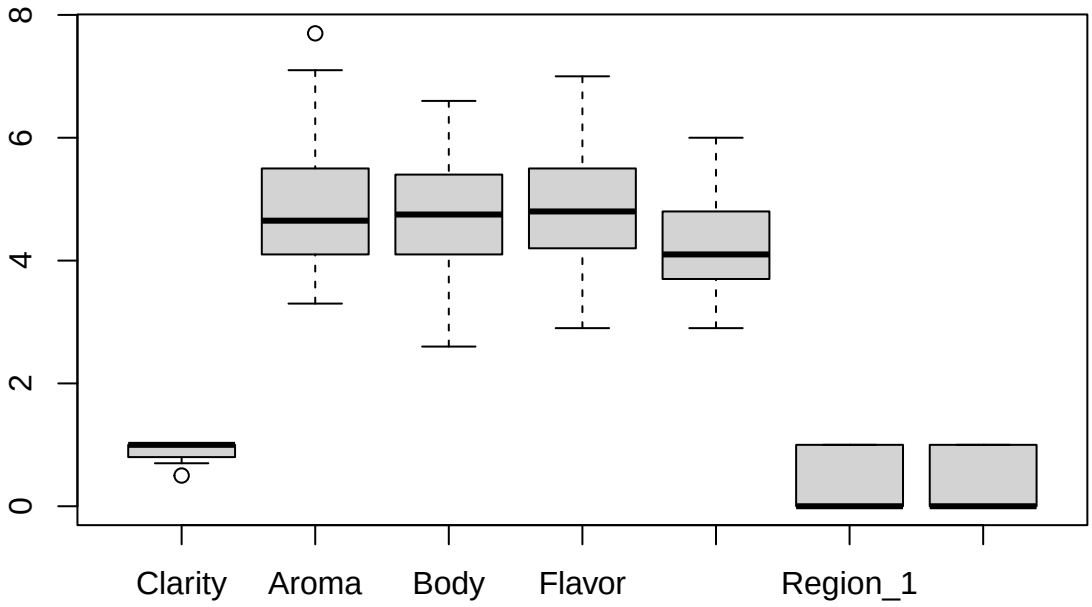
(a)

From the scatter plot we can see that aroma, body, and flavor look like good predictors to predict Quality when plotted by themselves. Also looking at the histograms Aroma, body, flavor, and oakiness seem approximately normal.

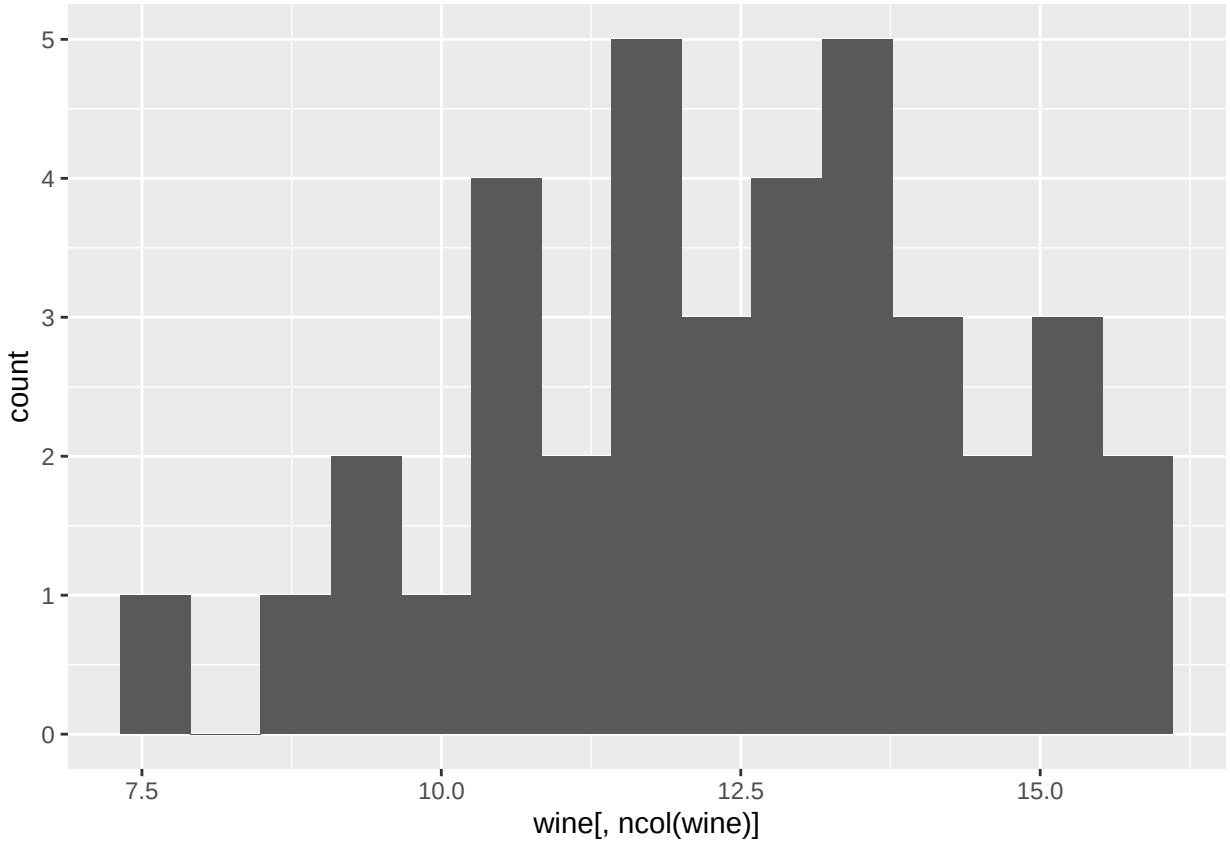
```
## `geom_smooth()` using formula 'y ~ x'
```




Clarity
Aroma
Body
Flavor
Oakiness
Region_1
Region_3
count
value
0
10
20
0.6
0.8
1.0
4
6
8
3
4
5
6
7
0.0
0.4
0.8

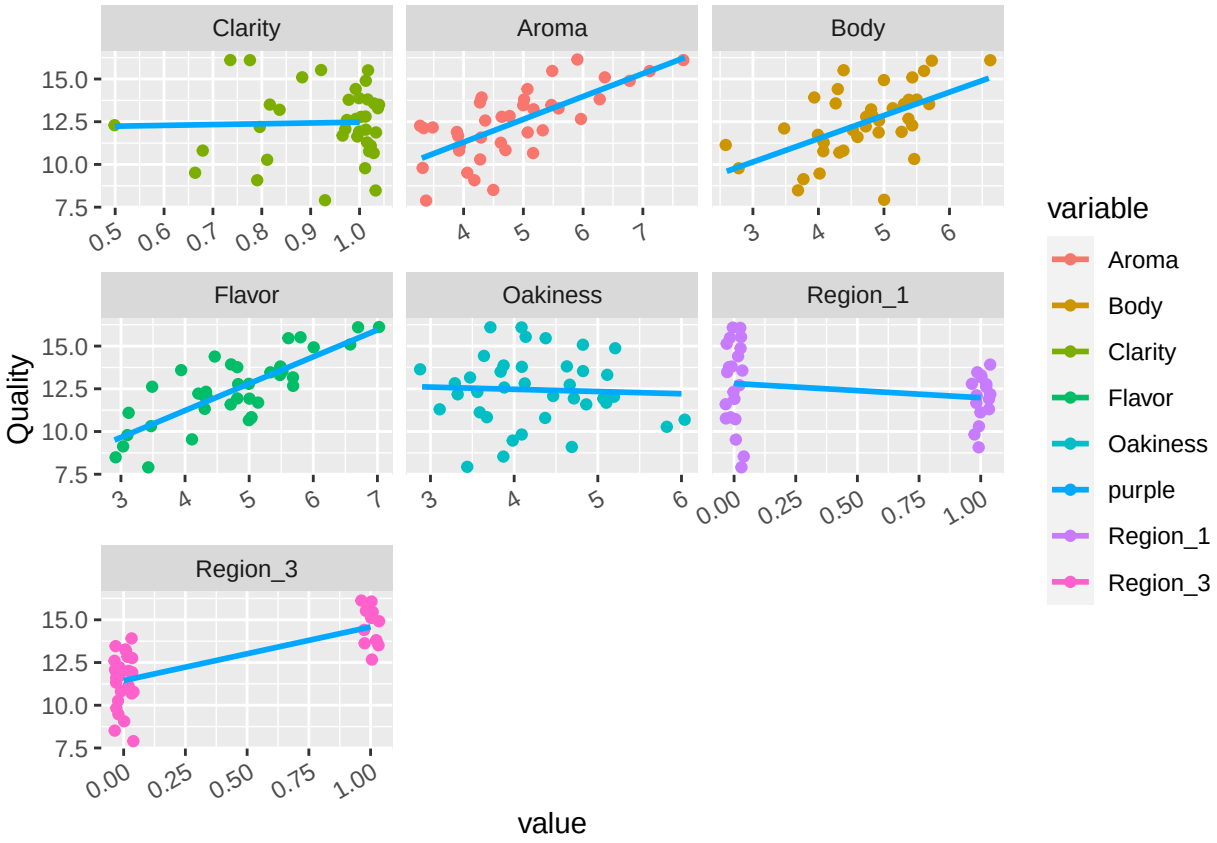


(b) From the histogram the response looks approximately normal. Therefore no transformation is necessary.



(c) From plotting a simple linear fit for each predictor I found from their summary pages that the only predictors significant to the Quality of Win are Aroma, Body, Flavor, and Region_3. This can be further shone in the plots for each fit.

```
## `geom_smooth()` using formula 'y ~ x'
```



(d)

From the summary below we can see that after fitting the full model that only the intercept, Flavor, Region1, and Region_3 are significant and we can see that our model has an adjusted R-squared of .7997 which is pretty good.

```
##
## Call:
## lm(formula = Quality ~ ., data = wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.80824 -0.58413 -0.02081  0.48627  1.70909
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.30151    1.90975   3.300 0.002502 **
## Clarity      0.01705    1.45627   0.012 0.990736
## Aroma        0.08901    0.25250   0.353 0.726908
## Body         0.07967    0.26772   0.298 0.768062
## Flavor       1.11723    0.24026   4.650 6.25e-05 ***
## Oakiness    -0.34644    0.23301  -1.487 0.147503
## Region_1     1.51285    0.39227   3.857 0.000565 ***
## Region_3     2.48544    0.58868   4.222 0.000207 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.9154 on 30 degrees of freedom
## Multiple R-squared:  0.8376, Adjusted R-squared:  0.7997
## F-statistic:  22.1 on 7 and 30 DF,  p-value: 3.295e-10
```

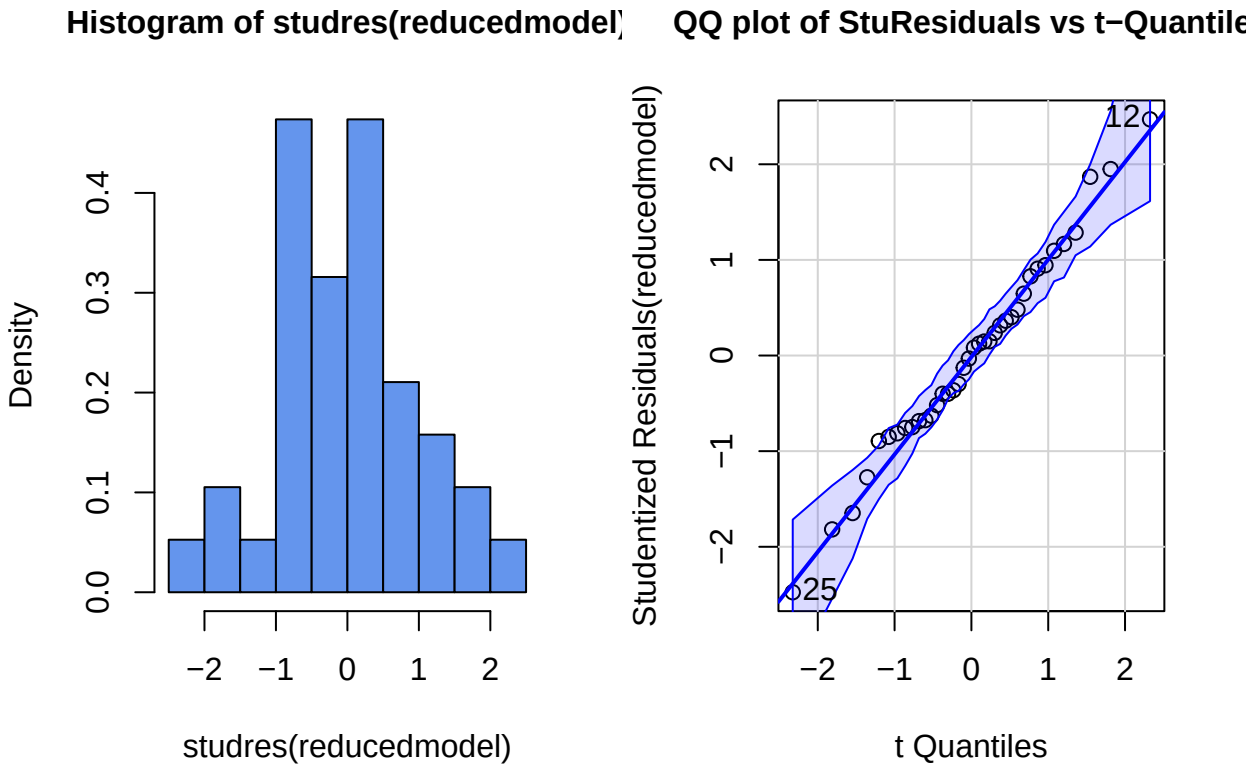
(e)

First we created a reduced model where the output can be seen below. After performing an anova test to confirm that our reduced model is better that the full model we can see from the output below that none of the predictors we took away were significant to our model. We can also see that region is an import factor when predicting the Quality of wine. I also dropped Region 2 as a variable because we turned Region into a dummy variable.

```
##
## Call:
## lm(formula = Quality ~ Flavor + Region_1 + Region_3, data = wine)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.97630 -0.58844  0.02184  0.51572  1.94232
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.5608     0.8174   6.803 7.96e-08 ***
## Flavor        1.1155     0.1738   6.417 2.49e-07 ***
## Region_1      1.5335     0.3688   4.158 0.000205 ***
## Region_3      2.7569     0.4495   6.134 5.78e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8946 on 34 degrees of freedom
## Multiple R-squared:  0.8242, Adjusted R-squared:  0.8087
## F-statistic: 53.13 on 3 and 34 DF,  p-value: 6.358e-13
```

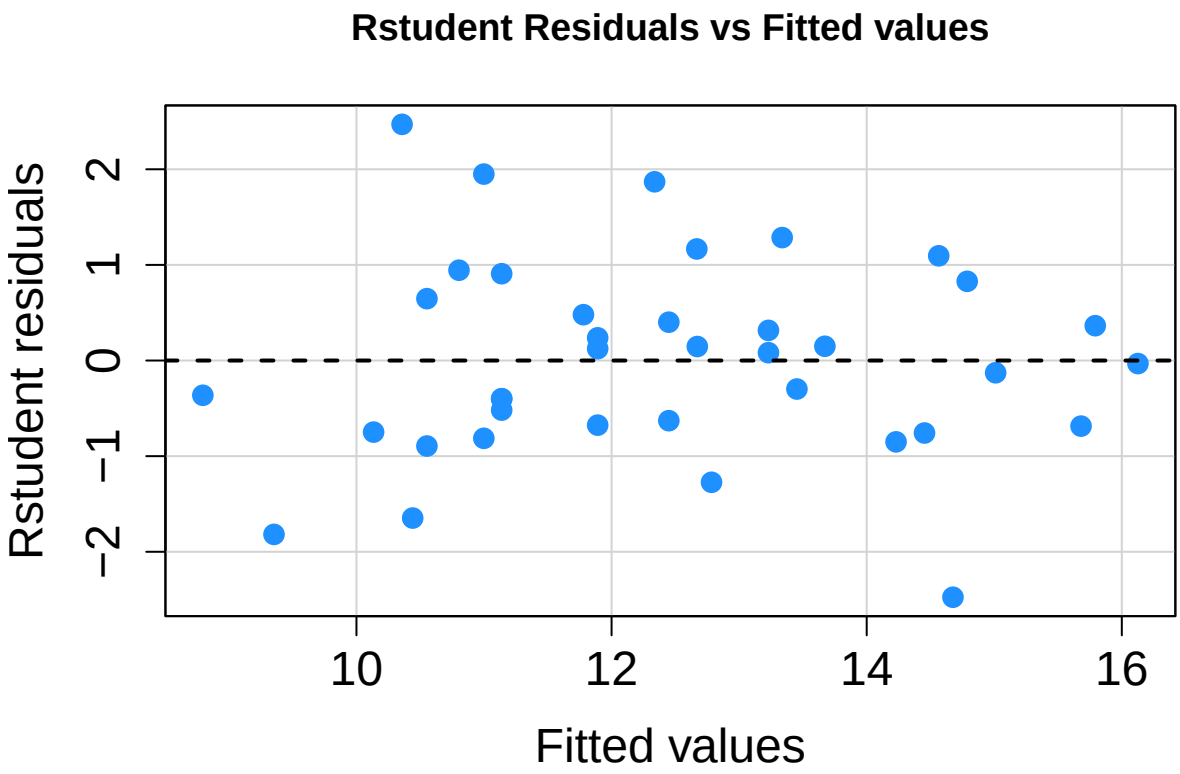
```
## Analysis of Variance Table
##
## Model 1: Quality ~ Flavor + Region_1 + Region_3
## Model 2: Quality ~ Clarity + Aroma + Body + Flavor + Oakiness + Region_1 +
##     Region_3
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1     34 27.213
## 2     30 25.140  4     2.073 0.6184 0.6528
```

In addition, we can see from the QQplot and histogram that everything is normal.




```
## integer(0)
```

Lastly we can see that this is a good linear model because after plotting our residual vs fitted values we found that each point was randomly distributed and formed a horizontal band accross the x-axis.




```
## numeric(0)
```

(f) Predicted Quality of Wine = (5.5608) + (1.1155)Flavor + (1.5335)Region_1 + (2.7569)Region_3

(g) From the predictions we can see that we the PI is wider and this makes sense because we are predicting a future observation. For CI we see a much smaller interval than PI that also agrees with what we would expect, because here we are are calculating the mean response so we are saying that we are 95% confident that the true population mean falls within this interval.

```
## [1] "95% Confidence interval for the mean response"
```

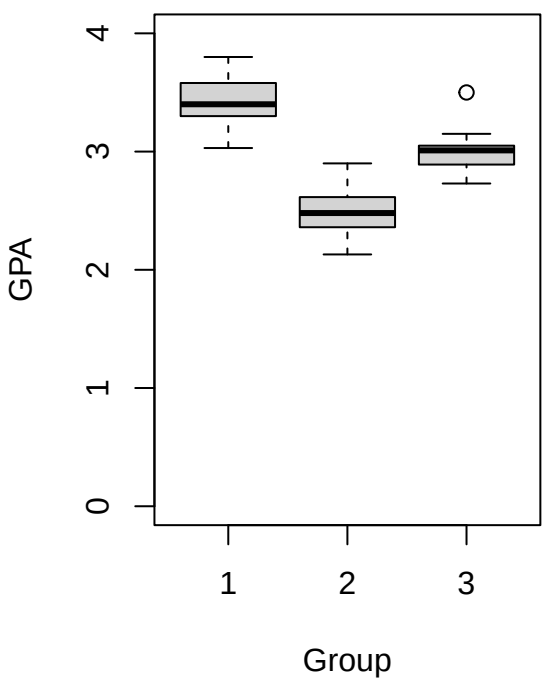
```
##        fit      lwr     upr
## 1 12.41371 11.95152 12.8759
```

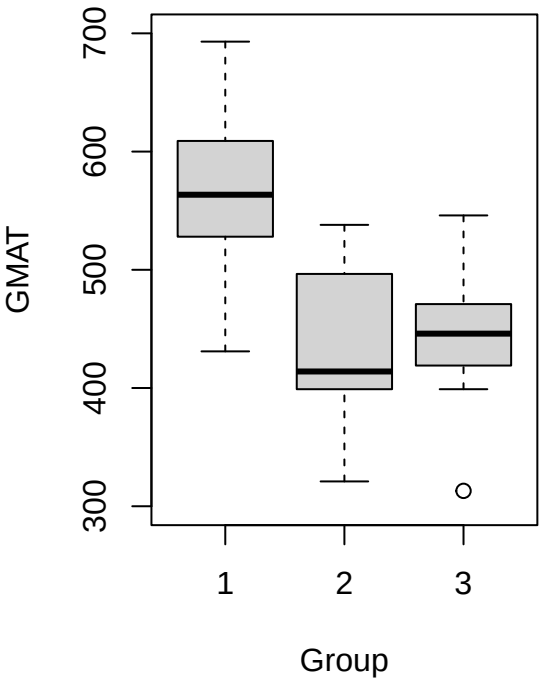
```
## [1] "95% Prediction interval for the response"
```

```
##        fit      lwr      upr
## 1 12.41371 10.53775 14.28967
```

Question 2

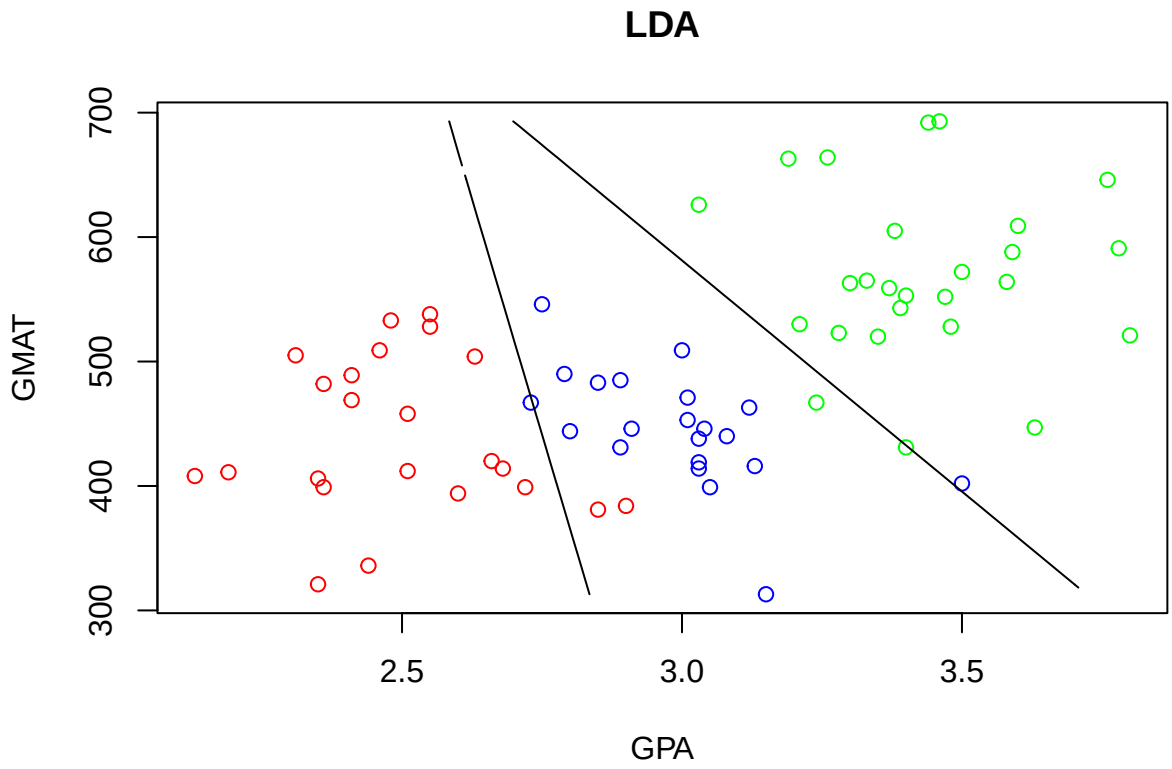
(a) We can see from the plots below that there is a great seperation for GPA, and that while for GMAT group 2 and 3 are semi close that the mean is different and not the same. Therefore, these two predictors are good use of predicting Group.





#LDA

Yes the decision boundary seems sensible for the using LDA as shown below.




```
## [1] "LDA confusionmatrix with top the true and side the pred of train"
```

```
##
##      1  2  3
##   1 24  0  1
##   2  0 21  1
##   3  2  2 19
```

```
## [1] "LDA confusionmatrix with top the true and side the pred of test"
```

```
##
##     1 2 3
##   1 2 0 0
##   2 0 5 0
##   3 3 0 5
```

```
## [1] "LDA overall misclassification rate test"
```

```
## [1] 0.2
```

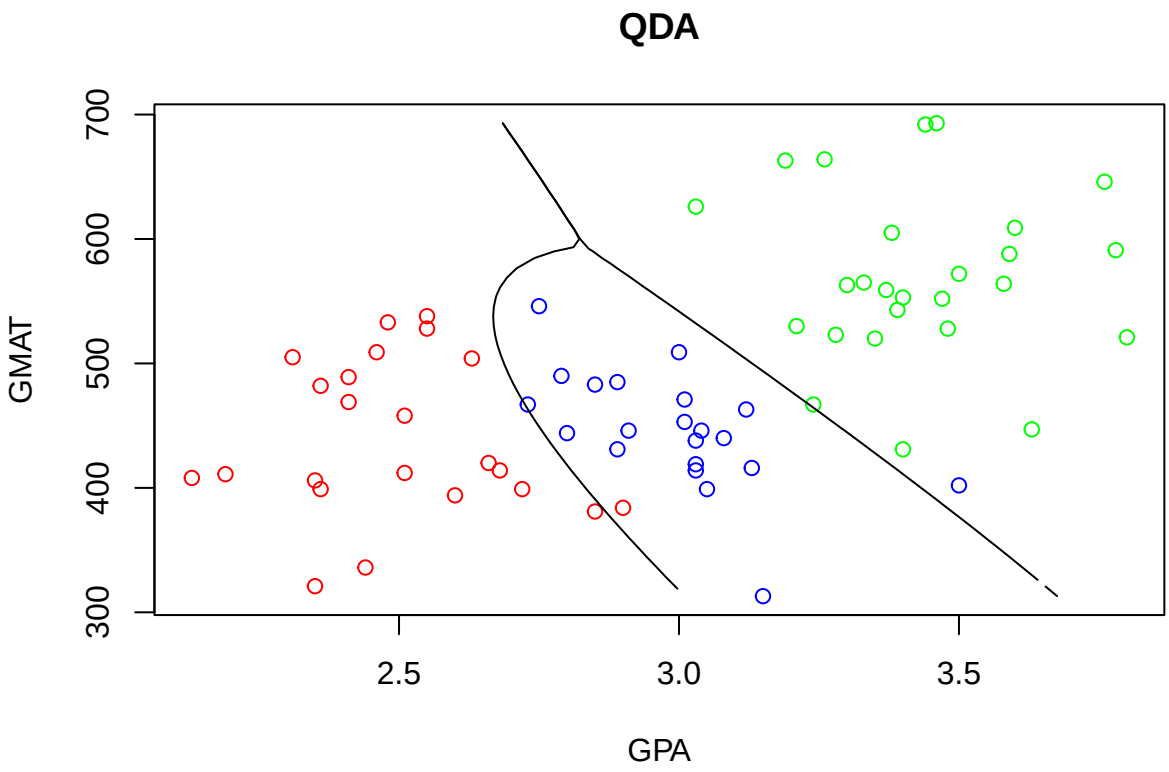
```
## [1] " LDA overall misclassification rate train"
```

```
## [1] 0.08571429
```

From looking at the Overall misclassification rate for both the training and test data we can see that we have a semi big difference between the the training error and test error rate. which could mean that this model is not the best for predicting group.

(c)

Yes the dicision boundary for QDA seems reasonable.




```
## [1] "QDA confusionmatrix with top the true and side the pred of train"

##
##      1  2  3
##   1 26  0  1
##   2  0 22  0
##   3  0  1 20

## [1] "QDA overall misclassification rate train"

## [1] 0.02857143

## [1] "QDA confusionmatrix with top the true and side the pred of test"

##
##    1 2 3
##   1 4 0 0
##   2 0 5 0
##   3 1 0 5
```

```
## [1] "QDA overall misclassification rate test"
```

```
## [1] 0.06666667
```

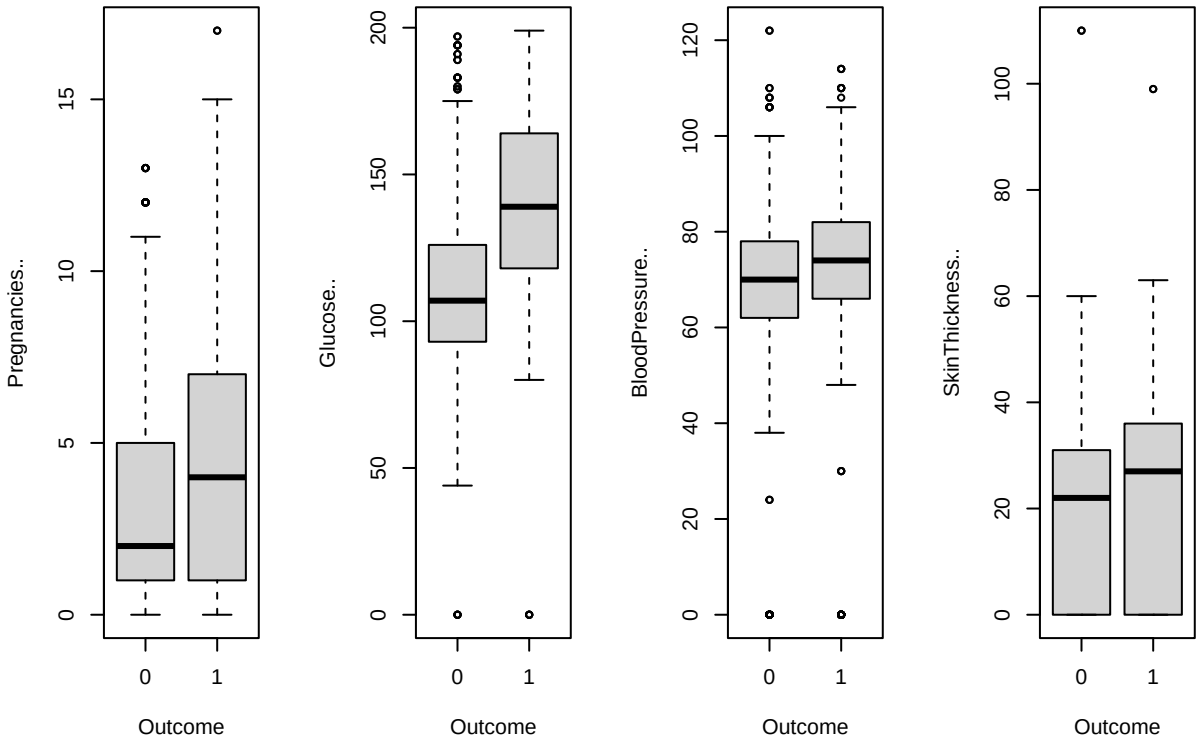
We can see from the Overall misclassification rate that QDA is working great bc the test and train error dont have a great difference.

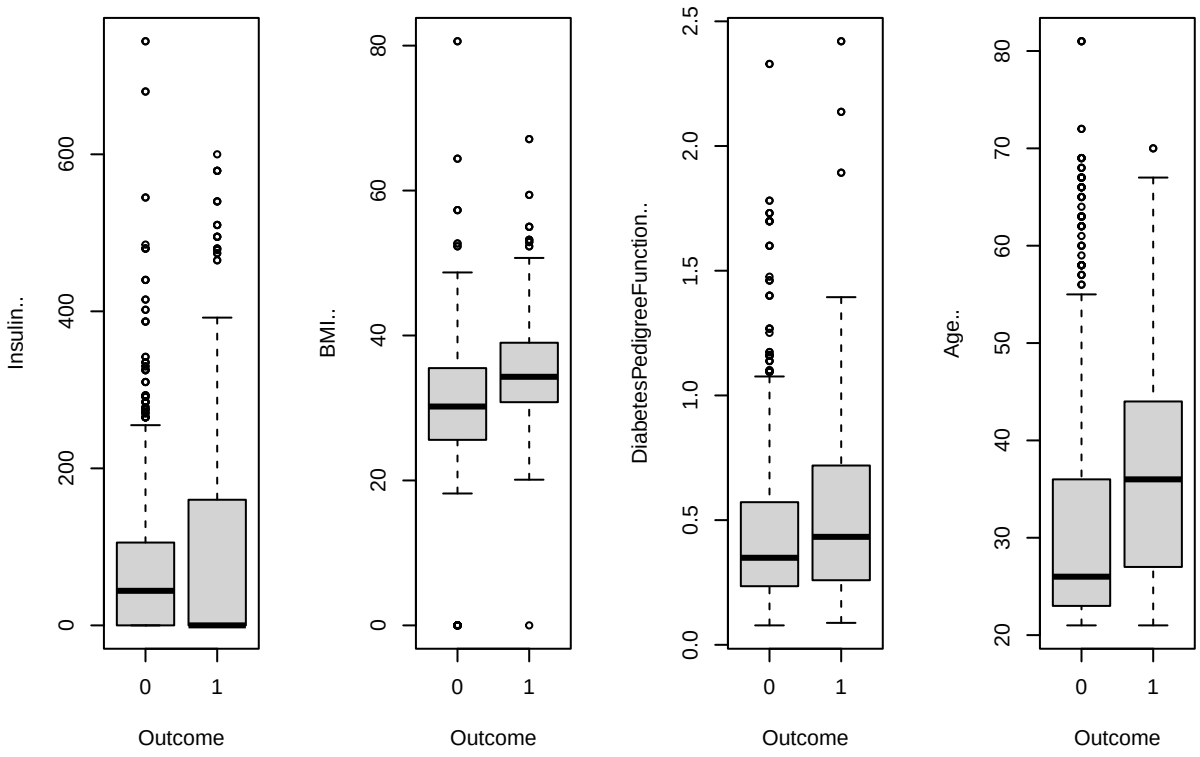
(d) We can see from the Overall misclassification rate of test and train that QDA is better, because the test error of LDA was .2 and the test error for QDA was 0.06667. Also the training error was 0.02857143 for QDA and 0.08571429 for the training error of LDA. From this we can confidently decide that QDA was a better model for this problem.

Question 3

(a)

From the box plots below we can see that that while some distributions look similar the case is not impossible because all the medians are clearly different.





(b)

We see from the information below that we have a semi high misclassification rate but that he diff between train and test is small. Also we see that we have a high specificity from our data using a 0.5 cutoff.

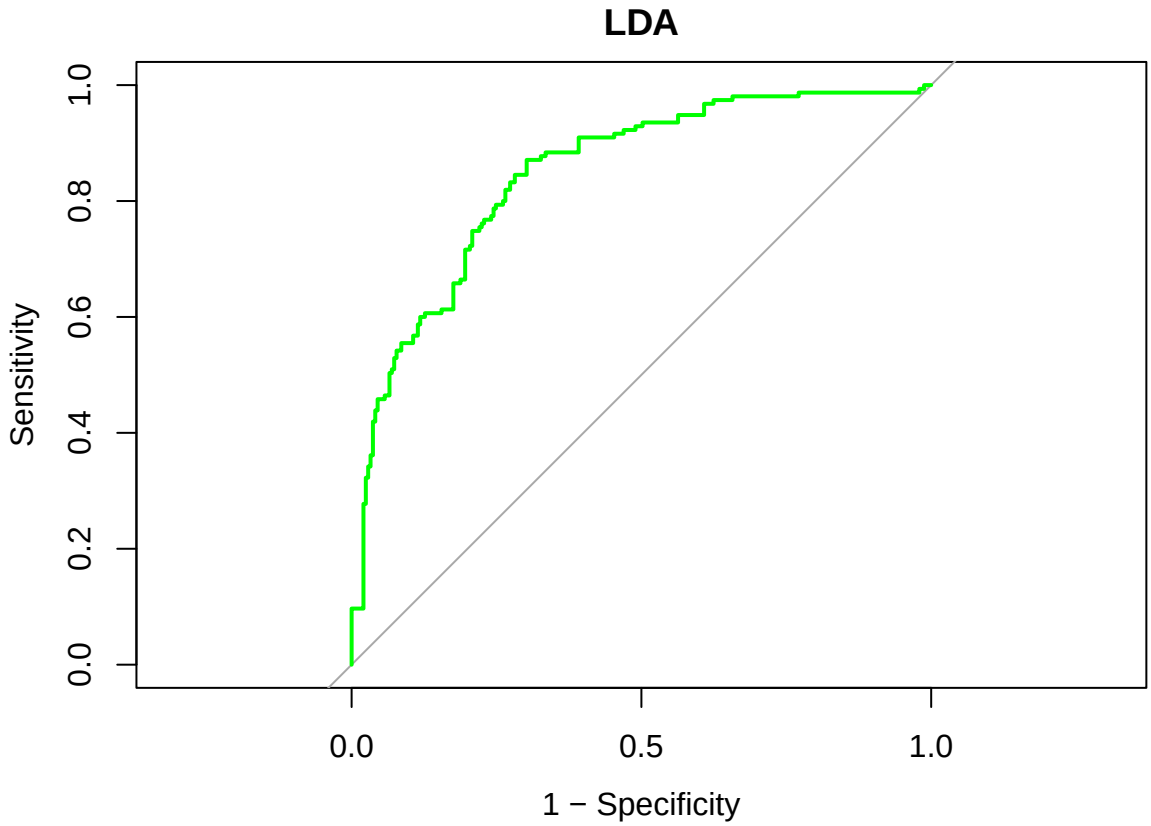
```
## [1] "LDA confusionmatrix with top the true and side the pred of train"

##
##       0   1
##   0 960 245
##   1 111 284

## [1] "LDA train sensitivity:  0.536862003780718"

## [1] "LDA train specificity:  0.896358543417367"

## [1] "LDA train Overall Missclassification rate 0.2225"

## [1] "LDA confusionmatrix with top the true and side the pred of test"

##
##       0   1
##   0 220  69
##   1  25  86
```

```
## [1] "LDA test sensitivity:  0.554838709677419"
```

```
## [1] "LDA test specificity:  0.897959183673469"
```

```
## [1] "LDA test Overall Missclassification rate 0.235"
```

From the ROC curve below we can see that our graph is better than the 45degree line and that it is in the shape we like indiacting a good model.



(c)

We see from the information below that we have that test error is actually smaller than train. We also see that we still have a high specificity using QDA with a .5 cutoff.

```
## [1] "QDA confusionmatrix with top the true and side the pred of train"
```

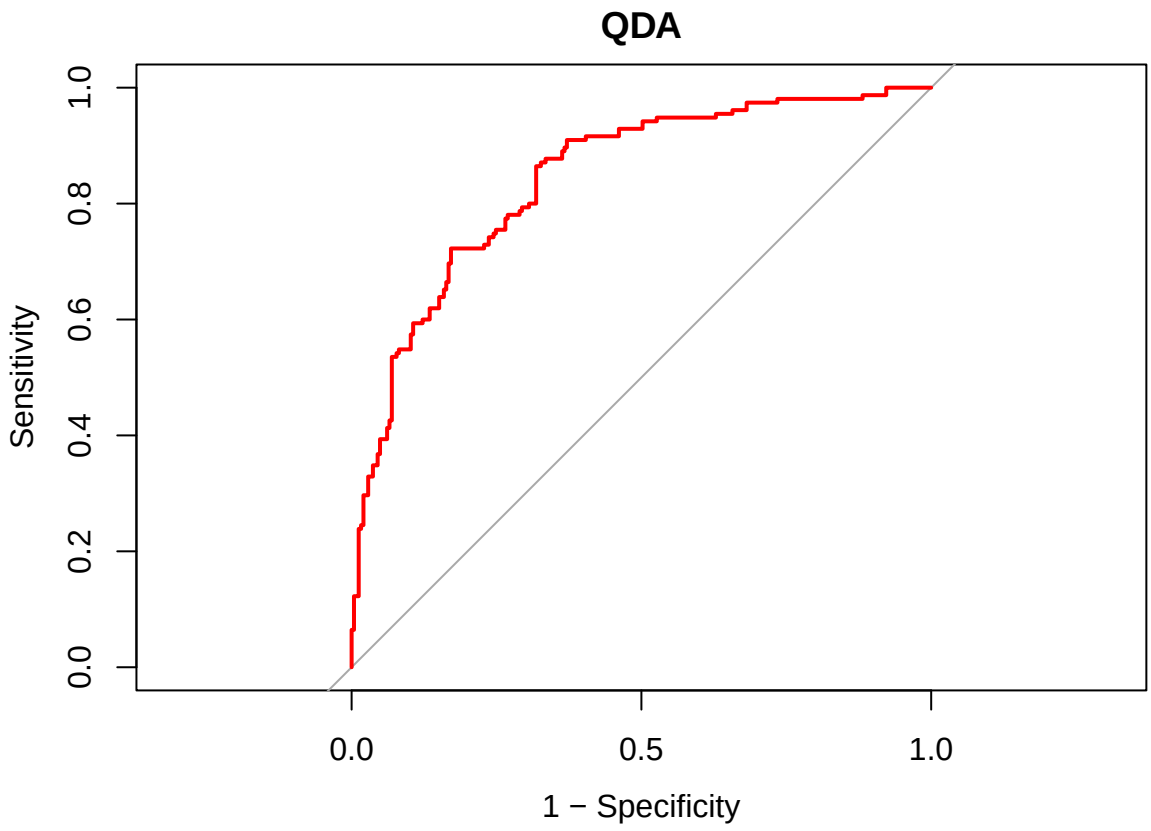
```
##
##       0   1
##   0 917 224
##   1 154 305
```

```
## [1] "QDA train sensitivity:  0.5765595463138"
```

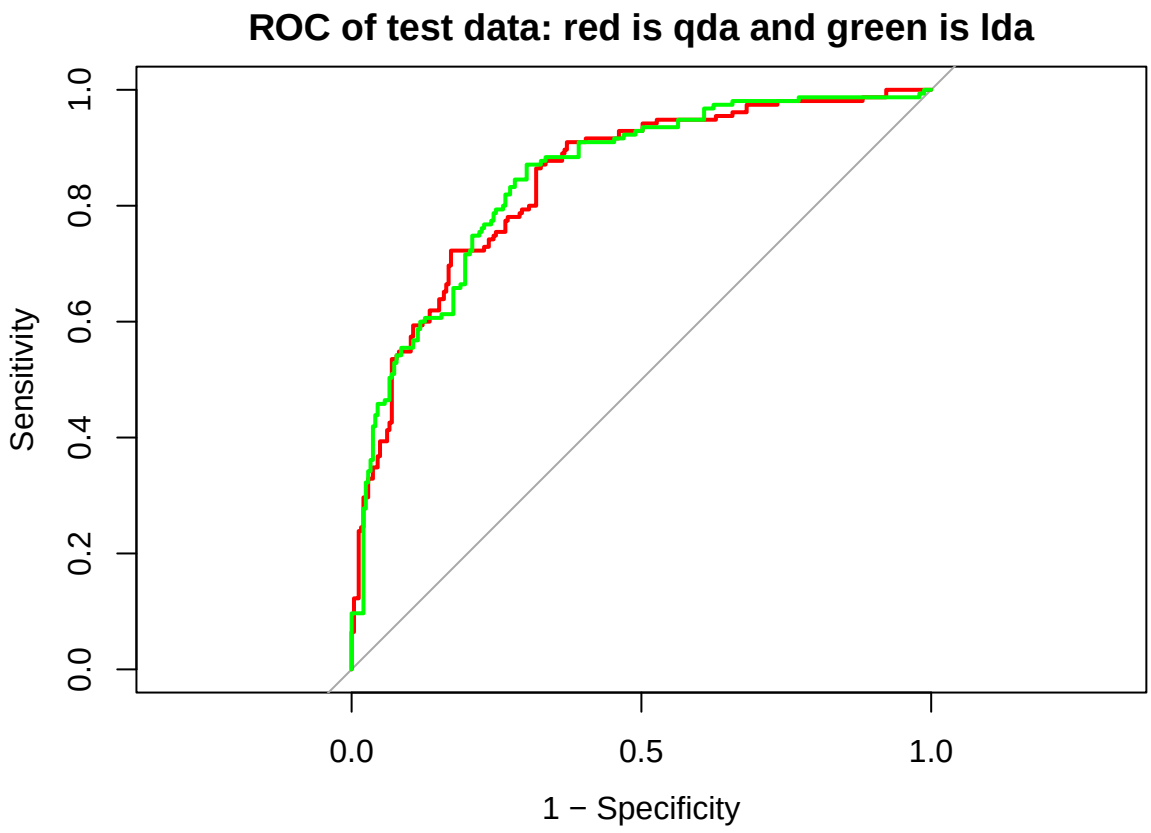
```
## [1] "QDA train specificity:  0.856209150326797"
```

```
## [1] "QDA train Overall Missclassification rate 0.23625"

## [1] "QDA confusionmatrix with top the true and side the pred of test"

##
##       0   1
##   0 212  59
##   1  33  96

## [1] "QDA test sensitivity:  0.619354838709677"

## [1] "QDA test specificity:  0.86530612244898"

## [1] "QDA test Overall Missclassification rate 0.23"
```

We also see that the the ROC curve looks good for QDA.



(d) We see that when plotting the ROC curves together that there is a lot of overlap which we can see that depending on a specific specificty or sensitivity the best model may be either one. But we can tell from the Areas under the curve for qda = 0.8429 and the Area under the curve for lda = 0.8463, and from this we can say overall that LDA is slightly better.




```
library(fastDummies)
library(reshape2)
library(ggplot2)
library(dplyr)
library(MASS)
library(car)
#setwd("C:/Users/jkenn/OneDrive/Documents/R/Stat4360/miniprojects/miniproject2/")
setwd("C:/Users/John/Documents/R/Programs/Stat4360/MiniProjects/Miniproject2/")
wine = read.delim("wine.txt")
# load in wine and getdummies also dropping region_2
wine = dummy_cols(wine, select_columns = 'Region',remove_selected_columns = TRUE)
wine = wine[, c(1:5,7,9,6)]
#View(wine)
df2 <- melt(wine, id.vars = "Quality")
ggplot(df2) +
  geom_jitter(aes(value,Quality, colour=variable),) +
  geom_smooth(aes(value,Quality, colour="purple"),
              method=lm, se=FALSE) +
  facet_wrap(~variable, scales="free_x") +
  theme(axis.text.x = element_text(angle = 30, hjust=1))
#exploratory analysis
ggplot(df2,aes(x = value)) +geom_histogram(bins = 5)
+ facet_wrap(~variable,scales = "free_x")


boxplot(wine[,c(1:7)])#histogram of response var
```

```
ggplot(wine,aes(x = wine[,ncol(wine)]))
+ geom_histogram(bins = 15) #hist of predictors

ggplot(df2) +#linear scatter plot
  geom_jitter(aes(value,Quality, colour=variable),) +
  geom_smooth(aes(value,Quality, colour="purple"), method=lm, se=FALSE) +
  facet_wrap(~variable, scales="free_x") +
  theme(axis.text.x = element_text(angle = 30, hjust=1))


#summaries of each simple linear model for each predictor
summary(lm(Quality ~ Clarity, data = wine))#not significant



summary(lm(Quality ~ Aroma, data = wine))#significant



summary(lm(Quality ~ Body, data = wine))#significant


summary(lm(Quality ~ Flavor, data = wine))#significant



summary(lm(Quality ~ Oakiness, data = wine))#not significant



summary(lm(Quality ~ Region_1, data = wine))#not significant



summary(lm(Quality ~ Region_3, data = wine))#significant
```

```
#model the data using all the predictors
fullmodel <- lm(Quality ~ ., data = wine)
summary(fullmodel)
```

```
#modeling using only significant predictors
reducedmodel <- lm(Quality ~Flavor + Region_1 + Region_3, data = wine)
summary(reducedmodel)
```

```
# confirming whether our model in good
anova(reducedmodel,fullmodel)
```

```
#QQ plot approach
par(mfrow=c(1,2))
hist(studres(reducedmodel), breaks=10, freq=F, col="cornflowerblue",
cex.axis=1, cex.lab=1, cex.main=1)
qqPlot(reducedmodel)
+ title("QQ plot of StuResiduals vs t-Quantiles", cex.main = 1)
```

```r
#plot residuals
residualPlot(reducedmodel, type="rstudent", quadratic=F, col = "dodgerblue",
pch=16, cex=1.5, cex.axis=1.5, cex.lab=1.5)
+ title("Rstudent Residuals vs Fitted values")
```

```r
#predicting PI and CI
newx <- data.frame(Clarity = mean(wine[,1]), Aroma = mean(wine[,2]), Body = mean(wine[,3]), Flavor = mea
                   Oakiness = mean(wine[,5])
                   , Region_1 = 1, Region_3 = 0, Quality = mean(wine[,8]) )
colnames(newx) <-
  c("Clarity",  "Aroma",    "Body",     "Flavor",   "Oakiness", "Region_1", "Region_3", "Quality" )
newx_reduced <- newx[,c(4,6,7)]
print("95% Confidence interval for the mean response")
predict(reducedmodel, newx_reduced, interval = "confidence",level = .95)
print("95% Prediction interval for the response")
predict(reducedmodel, newx_reduced, interval = "prediction",level = .95)
```

```r
#load in data for q2 and split the data into train and test
admission <- read.csv("admission.csv")
#View(admission)
split <- c(1:5,32:36,60:64)
train <- admission[-split,]
test <- admission[split,]
row.names(train) <- NULL
row.names(test) <- NULL
range(train[,"GPA"])
range(train[,"GMAT"])
#test
```

```r
#plots the boxplots of the predictors grouped by the observable variable
par(mfrow = c(1, 2))
boxplot(train[, "GPA"] ~ train[,"Group"],
        ylab = "GPA", xlab = "Group", ylim = c(0, 4))
boxplot(train[, "GMAT"] ~ train[,"Group"]
        , ylab = "GMAT", xlab = "Group", ylim = c(300, 700))
par(mfrow = c(1, 1))
```

```r
#fitting LDA
library(MASS)

lda.fit <- lda(Group ~ GPA + GMAT, data = train)

lda.fit
```

```r
# get the predictions
lda.pred.test <- predict(lda.fit, test[,c("GPA","GMAT")])
lda.pred.train <- predict(lda.fit, train[,c("GPA","GMAT")])
names(lda.pred.test)
#print(test)
```

```r
# Decision boundary
# Set up a dense grid and compute posterior prob on the grid

n.grid <- 50
x1.grid <- seq(f = min(train[, "GPA"]),
               t = max(train[, "GPA"]), l = n.grid)
x2.grid <- seq(f = min(train[, "GMAT"]),
               t = max(train[, "GMAT"]), l = n.grid)
grid <- expand.grid(x1.grid, x2.grid)
colnames(grid) <- colnames(train[,c("GPA","GMAT")])
#head(grid)
```

```r
pred.grid <- predict(lda.fit, grid)
```

```r
#calc the posterior curve for group one and 2 to add both contours to the plot
post1 <- (pred.grid$posterior[, "1"] -
            pmax(pred.grid$posterior[, "2"],pred.grid$posterior[, "3"]))
post2 <- (pred.grid$posterior[, "2"] -
            pmax(pred.grid$posterior[, "1"],pred.grid$posterior[, "3"]))
prob1 <- matrix(post1, nrow = n.grid, ncol = n.grid, byrow = F)
prob2 <- matrix(post2, nrow = n.grid, ncol = n.grid, byrow = F)

plot(train[,c("GPA","GMAT")],
     col = ifelse((train[,"Group"] == 1), "green",
                  ifelse((train[,"Group"] == 2), "red", "blue")),main ="LDA")

contour(x1.grid,x2.grid,z =  prob1, levels = 0,
        labels = "", xlab = "", ylab = "", main = "", add = T)
contour(x1.grid,x2.grid,z = prob2, levels = 0,
        labels = "", xlab = "", ylab = "", main = "", add = T)
#prob
```

```r
print("LDA confusionmatrix with top the true and side the pred of train")
table(lda.pred.train$class, train[,"Group"])
```

```r
print("LDA confusionmatrix with top the true and side the pred of test")
table(lda.pred.test$class, test[,"Group"])
```

```r
print("LDA overall misclassification rate test")
print(3/(nrow(test)))
lda_mr_test <- 3/(nrow(test))
```

```r
print(" LDA overall misclassification rate train")
print(6/(nrow(train)))
lda_mr_train <-6/(nrow(train))
```

```r
#fitting the QDA
qda.fit <- qda(Group ~ GPA + GMAT, data = train)
qda.fit
```

```
#predicting QDA
qda.pred.test <- predict(qda.fit, test[,c("GPA","GMAT")])
qda.pred.train <- predict(qda.fit, train[,c("GPA","GMAT")])
names(lda.pred.test)
```

```
# Decision boundary
# Set up a dense grid and compute posterior prob on the grid

n.grid <- 50
x1.grid <- seq(f = min(train[, "GPA"]), t = max(train[, "GPA"]), l = n.grid)
x2.grid <- seq(f = min(train[, "GMAT"]), t = max(train[, "GMAT"]), l = n.grid)
grid <- expand.grid(x1.grid, x2.grid)
colnames(grid) <- colnames(train[,c("GPA","GMAT")])
```

```
pred.grid <- predict(qda.fit, grid)
```

```
#calc the posterior curve for group one and 2 to add both contours to the plot
post1 <- (pred.grid$posterior[, "1"] -
            pmax(pred.grid$posterior[, "2"],pred.grid$posterior[, "3"]))
post2 <- (pred.grid$posterior[, "2"] -
            pmax(pred.grid$posterior[, "1"],pred.grid$posterior[, "3"]))
prob1 <- matrix(post1, nrow = n.grid, ncol = n.grid, byrow = F)
prob2 <- matrix(post2, nrow = n.grid, ncol = n.grid, byrow = F)

plot(train[,c("GPA","GMAT")], col =
       ifelse((train[,"Group"] == 1), "green",
              ifelse((train[,"Group"] == 2), "red", "blue")),main ="QDA")

contour(x1.grid,x2.grid,z =  prob1, levels = 0,
        labels = "", xlab = "", ylab = "", main = "", add = T)
contour(x1.grid,x2.grid,z = prob2, levels = 0,
        labels = "", xlab = "", ylab = "", main = "", add = T)
#prob
```

```
print("QDA confusionmatrix with top the true and side the pred of train")
table(qda.pred.train$class, train[,"Group"])
```

```
print("QDA overall misclassification rate train")
print(2/(nrow(train)))
qda_mr_train <- 2/(nrow(train))
```

```
print("QDA confusionmatrix with top the true and side the pred of test")
table(qda.pred.test$class, test[,"Group"])
```

```
print("QDA overall misclassification rate test")
print(1/(nrow(test)))
qda_mr_test <- 1/(nrow(test))
```

```
#loading in data for q3 and splitign data into train test
diabetes <- read.csv("diabetes.csv")
```

```r
set.seed(1)
split2 <- sample(seq_len(nrow(diabetes)),
                 size = floor(0.8 * nrow(diabetes)))

train.3 <- diabetes[split2, ]
test.3 <- diabetes[-split2, ]
row.names(train.3) <- NULL
row.names(test.3) <- NULL

for (i in 1:ncol(diabetes)) {print(colnames(diabetes)[i]); print(range(train.3[,i]))}
```

```r
# making box plot for each predictor grouped by outcome
par(mfrow = c(1, 4))
boxplot(train.3[, "Pregnancies.."] ~ train.3[,"Outcome"],
        ylab = "Pregnancies..", xlab = "Outcome", ylim = c( 0, 17))
boxplot(train.3[, "Glucose.."] ~ train.3[,"Outcome"],
        ylab = "Glucose..", xlab = "Outcome", ylim = c(0, 199))
boxplot(train.3[, "BloodPressure.."] ~ train.3[,"Outcome"],
        ylab = "BloodPressure..", xlab = "Outcome", ylim = c(0, 122))
boxplot(train.3[, "SkinThickness.."] ~ train.3[,"Outcome"],
        ylab = "SkinThickness..", xlab = "Outcome", ylim = c( 0, 110))

par(mfrow = c(1, 1))
```

```r
par(mfrow = c(1, 4))
boxplot(train.3[, "Insulin.."] ~ train.3[,"Outcome"],
        ylab = "Insulin..", xlab = "Outcome", ylim = c(0, 744 ))
boxplot(train.3[, "BMI.."] ~ train.3[,"Outcome"],
        ylab = "BMI..", xlab = "Outcome", ylim = c(0.0, 80.6))
boxplot(train.3[, "DiabetesPedigreeFunction.."] ~ train.3[,"Outcome"],
        ylab = "DiabetesPedigreeFunction..", xlab = "Outcome", ylim = c(0.078, 2.420))
boxplot(train.3[, "Age.."] ~ train.3[,"Outcome"],
        ylab = "Age..", xlab = "Outcome", ylim = c( 21, 81))

par(mfrow = c(1, 1))
```

```r
library(MASS)
#fitting LDA
lda.fit <- lda(Outcome ~ Pregnancies.. + Glucose.. +
                 BloodPressure.. + SkinThickness.. + Insulin..
               + BMI.. + DiabetesPedigreeFunction..
               + Age.., data = train.3)

lda.fit
```

```r
#predition from test and train data
lda.pred.test <- predict(lda.fit, test.3[,c(1:8)])
lda.pred.train <- predict(lda.fit, train.3[,c(1:8)])
names(lda.pred.test)
#print(test)
```

```r
print("LDA confusionmatrix with top the true and side the pred of train")
table(lda.pred.train$class, train.3[,"Outcome"])
```

```r
paste("LDA train sensitivity: ",284/(284+245))
paste("LDA train specificity: ", 960/(960+111))
paste("LDA train Overall Missclassification rate",
      mean(lda.pred.train$class != train.3[,"Outcome"]))
```

```r
print("LDA confusionmatrix with top the true and side the pred of test")
table(lda.pred.test$class, test.3[,"Outcome"])
```

```r
paste("LDA test sensitivity: ",86/(86+69))
paste("LDA test specificity: ", 220/(220+25))
paste("LDA test Overall Missclassification rate",
      mean(lda.pred.test$class != test.3[,"Outcome"]))
```

```r
#fitting ROC curve
library(pROC)
roc.lda.test <- roc(test.3[,"Outcome"],
                    lda.pred.test$posterior[, "1"], levels = c("0", "1"))
```

```r
#plotting ROC CURVE
plot(roc.lda.test, legacy.axes = T,col = "green",main = "LDA")
```

```r
#fitting the qda
library(MASS)

qda.fit <- qda(Outcome ~ Pregnancies.. + Glucose.. +
                 BloodPressure.. + SkinThickness.. + Insulin.. +
                 BMI.. + DiabetesPedigreeFunction.. + Age.., data = train.3)

qda.fit
```

```r
#predicting from test and train data
qda.pred.test <- predict(qda.fit, test.3[,c(1:8)])
qda.pred.train <- predict(qda.fit, train.3[,c(1:8)])
names(qda.pred.test)
```

```r
print("QDA confusionmatrix with top the true and side the pred of train")
table(qda.pred.train$class, train.3[,"Outcome"])
```

```r
paste("QDA train sensitivity: ",305/(305+224))
paste("QDA train specificity: ", 917/(917+154))
paste("QDA train Overall Missclassification rate",
      mean(qda.pred.train$class != train.3[,"Outcome"]))
```

```r
print("QDA confusionmatrix with top the true and side the pred of test")
table(qda.pred.test$class, test.3[,"Outcome"])
```

```
paste("QDA test sensitivity: ",96/(96+59))
paste("QDA test specificity: ", 212/(212+33))
paste("QDA test Overall Missclassification rate",
      mean(qda.pred.test$class != test.3[,"Outcome"]))
```

```
#fitting the ROC curve
library(pROC)
roc.qda.test <- roc(test.3[,"Outcome"],
                    qda.pred.test$posterior[, "1"], levels = c("0", "1"))
```

```
#ploting the Roc Curve
plot(roc.qda.test, legacy.axes = T,col = "red",main = "QDA")
```

```
library(pROC)
#code is redundant
roc.lda.test <- roc(test.3[,"Outcome"],
                    lda.pred.test$posterior[, "1"], levels = c("0", "1"))
roc.qda.test <- roc(test.3[,"Outcome"],
                    qda.pred.test$posterior[, "1"], levels = c("0", "1"))
```

```
#ploting the ROC curve for QDA and LDA together
plot(roc.qda.test, legacy.axes = T,col = "red",
     main = "ROC of test data: red is qda and green is lda")
plot(roc.lda.test, add = T, col = "green")
```