STAT 6340 (Introduction to Statistical Learning, Spring 2021)
Solution: Mini Project #2
Instructor: Pankaj Choudhary TA: Sajith de Silva

1. (a) For exploratory data analysis, we looked at summary statistics of the variables and also examined their distributions and the relationships between them through appropriate graphs. Specifically, `Quality` ranges between 7.9 and 16.1, with an average of 12.4. From Figure 1, we see that `Quality` may be considered to have an approximately symmetric distribution. The variables `Aroma`, `Body`, and `Flavor` appear to have moderately strong, positive correlations among themselves and all also have moderately strong, positive linear relationships with `Quality`. The `Quality` also depends on `Region`, with 3 being the best, followed by 1, and 2. However, `Clarity` and `Oakiness` does not seem to have much effect on `Quality`. The data do not have seem to have any outliers with the possible exception of two observations seen in `Quality` versus `Clarity` plot: the left-most observation (# 12) and the lowest observation at `Clarity` = 0.9 (# 20).

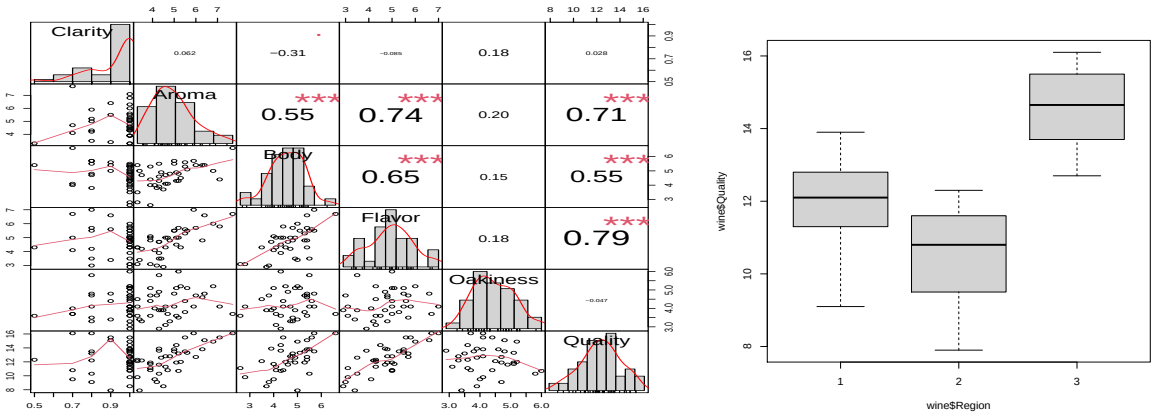

Figure 1: Graphical summaries of the variables and their relationships. (Left) Scatterplot matrix of the quantitative variables. (Right) Boxplots of `Quality` versus `Region`.

(b) Due to the approximate symmetry of the distribution of `Quality` and lack of serious outliers on this variable that we observed in (a), we may expect that a transformation of response will not be necessary. To confirm this, we fit a model containing all the predictor variables and examine a normal Q-Q plot of the residuals. The points in this plot (Figure 2) mostly cluster around a straight line, indicating that the normality assumption for response may be reasonable. We also explored Box-cox transformations and specifically, square-root and log transformations, but did not see any noticeable improvement. Therefore, we think a transformation of response is not necessary and will use `Quality` as the response for the rest of this problem.

(c) The table below presents p-values for each predictor from the single-variable models. All predictors except `Clarity` and `Oakiness` are significant at 5% level. These findings are consistent with what we found in part (a) on the basis of Figure 1.

Predictor	p**-value**	**Significant**
Clarity	8.65e-01	No
Aroma	6.87e-07	Yes
Body	3.61e-04	Yes
Flavor	3.68e-09	Yes
Oakiness	7.79e-01	No
Region	7.57e-03	Yes

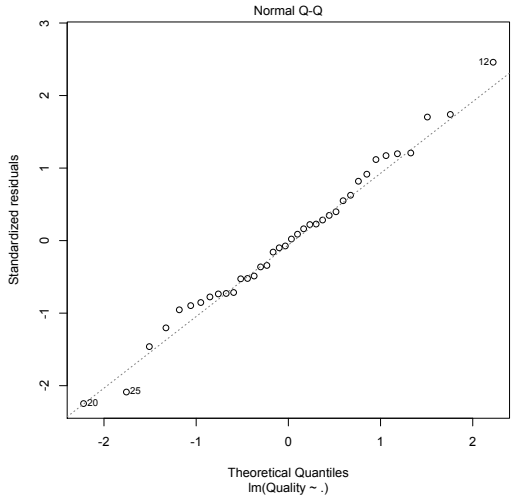

Figure 2: Normal Q-Q plot of residuals for wine data with all predictors in the model.

(d) The results from a multiple regression model with all predictors are summarized in the table below. The partial F-test p-value for `Region` is 0.0003. Only two variables—`Flavor` and `Region`—appear significant at 5% level. So the null hypothesis of no effect (in presence of all the other variables) is rejected only for them. The signs of the estimated coefficients for these variables are consistent with what we found in the exploratory analysis in part (a). Both R^2 (0.84) and adjusted R^2 (0.80) for this model are high.

Coefficient	Estimate	Std.Error	t value	p-value
(Intercept)	7.8144	1.9694	3.97	0.00042
Clarity	0.0171	1.4563	0.01	0.99074
Aroma	0.0890	0.2525	0.35	0.72691
Body	0.0797	0.2677	0.30	0.76806
Flavor	1.1172	0.2403	4.65	6.2e-05
Oakiness	-0.3464	0.2330	-1.49	0.14750
Region2	-1.5129	0.3923	-3.86	0.00057
Region3	0.9726	0.5102	1.91	0.06622

(e) From part (d), we got the impression that only `Flavor` and `Region` may be the important predictors. This was confirmed by the p-value of 0.653 for the partial F-test that compared the full model in part (d) with the reduced model containing only these two predictors. Next, we explored the interaction between the two variables, but it was not significant (p-value = 0.34). Thus, the model consisting of only the main effects of `Flavor` and `Region` is our candidate for the final model. Its R^2 and adjusted R^2 are 0.82 and 0.81, respectively. This model has higher adjusted R^2 than the full model in (d).

Next, we performed diagnostics on the candidate model by examining the relevant plots. The model assumptions appear reasonable with the exception that there is evidence of heteroscedasticity in the errors (see Figure 3). Since simple transformations do not appear to improve this aspect of the model, we will ignore this failing and take the candidate model as the final model.

(f) The final model has two predictors: `Flavor` and `Region`. It can be written as:

$$E(\texttt{Quality}) = 7.09 + 1.12 * \texttt{Flavor} - 1.53 * \text{I}\,(\texttt{Region} = 2) + 1.22 * \text{I}(\texttt{Region} = 3),$$

with `Region` = 1 serving as the baseline.

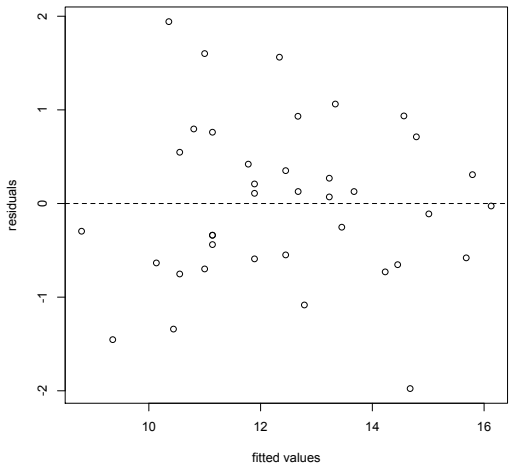

Figure 3: Residual plot for the final model for wine data.

(g) The sample mean for `Flavor` is 4.77. Plugging in this value in the fitted model equation gives 12.4 as the desired predicted response for `Region` 1. The corresponding 95% confidence and prediction interval are $[12.0, 12.9]$ and $[10.5, 14.3]$, respectively. The former implies that the mean `Quality` of a wine that comes from `Region` 1 and has average `Flavor` is estimated to lie between 12.0 and 12.9. The latter implies the `Quality` of an individual wine that comes from `Region` 1 and has average `Flavor` is predicted to lie between 10.5 and 14.3.

2. (a) The response is `Group` with three levels: 1 (`admit`), 2 (`do not admit`), and 3 (`borderline`). The two predictors are `GPA` and `GMAT`. The training set has 70 observations and the validation set has 15 observations. For exploratory analysis of the training set, we examined the class conditional distributions of the two predictors via boxplots in Figure 4. Since the class conditional distributions for each predictor differ across the three response classes, both predictors are useful. Specifically, we see that the `admit` group tends to have the highest `GPA` and `GMAT` values, followed by the `borderline` and `do not admit` groups, which makes intuitive sense.

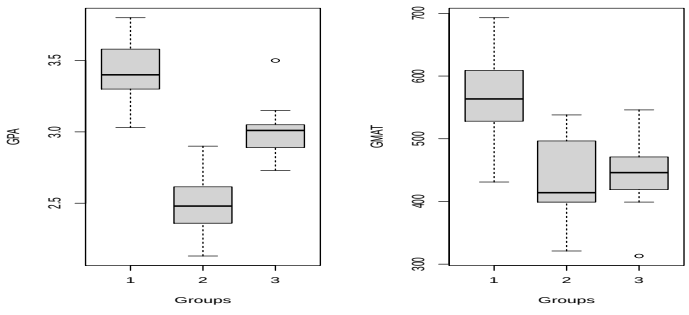

Figure 4: Boxplots `GPA` and `GMAT` for the three classes of `Group`.

(b) Figure 5 shows a plot of the training data superimposed with the LDA decision boundary. The boundary may be considered reasonable in that the misclassifications occur only near the class boundaries but the based on the distribution of points for `admit` and `borderline` classes, it seems that a nonlinear decision boundary may be better. The confusion matrices for LDA on training and validation data are given below. The misclassification rates on the training and

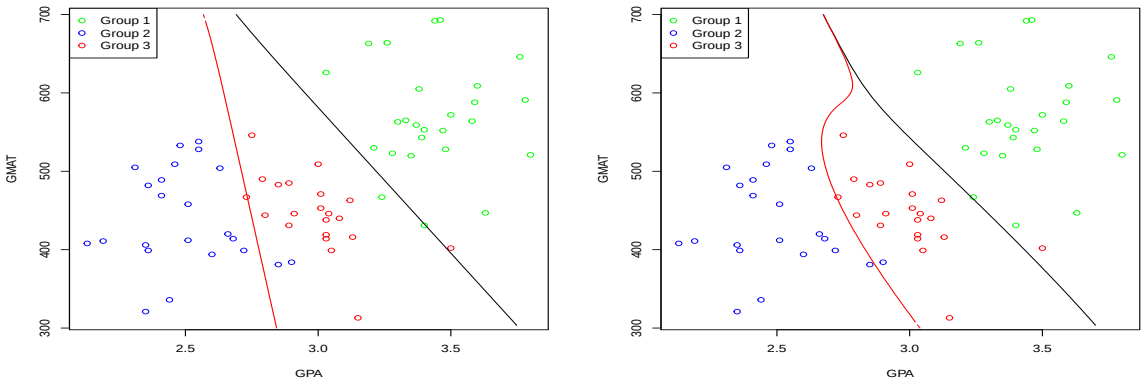

Figure 5: Training set for admission data superimposed with LDA (left) and QDA (right) decision boundaries.

validation sets are 0.086 (6/70) and 0.20 (3/15), respectively. In the training set, there are two misclassifications for each class. The model can perfectly separate the `admit` and `do not admit` groups, but naturally, it has some difficulty in separating them with the `borderline` group. The misclassifications in the validation set correspond to three `admit`s classified as `borderline`.

```
Confusion Matrix for LDA (rows: predicted class; columns: true class)
                        Train data                Test Data
                          1  2  3                   1 2 3
                      1 24  0  1                 1 2 0 0
                      2  0 21  1                 2 0 5 0
                      3  2  2 19                 3 3 0 5
```

(c) The QDA decision boundary is also shown in Figure 5. Its misclassification rates on training and validation sets are 0.029 (2/70) and 0.067 (1/15), respectively, both of which are lower than LDA. On the training set, the gain of QDA mainly comes from the reduction in misclassifying `admit` and `do not admit` as `borderline`, which are now 0 and 1, respectively, compared to 2 and 2 for LDA. On the validation set, QDA correctly classifies 2 of the `admit` as `borderline` misclassifications of LDA.

```
Confusion Matrix for QDA (rows: predicted class; columns: true class)
                        Train data                Test Data
                          1  2  3                   1 2 3
                      1 26  0  1                 1 4 0 0
                      2  0 22  0                 2 0 5 0
                      3  0  1 20                 3 1 0 5
```

(d) On the basis of QDA's smaller misclassification rate on the validation set and somewhat more sensible decision boundary than LDA between `admit` and `borderline` classes, we may prefer QDA over LDA.

3. (a) The diabetes data has 2000 observations on 9 variables. The response is a binary variable `Outcome` with classes 0 and 1. About 34% of the outcomes are 1 and 66% are 0. The remaining 8 variables are quantitative predictors. In the boxplots of their class conditional distributions (not presented here), none seems to have very similar distributions for the two response classes, indicating that all may be useful.

(b-c) The confusion matrices, sensitivity, specificity, and misclassification rates for LDA and QDA based on 0.5 cutoff are given below. Both models have higher specificity than sensitivity. This is because there is a much higher proportion of 0 response than 1 response (34% vs. 66%).

```
Confusion Matrix for LDA and QDA (row: predicted class; columns: true class)
                                LDA              QDA
                                  0   1            0   1
                              0 1174 298        0 1135 290
                              1  142 386        1  181 394

Misclassification Rate:     0.220            0.235
Sensitivity :               0.564            0.576
Specificity :               0.892            0.862
```

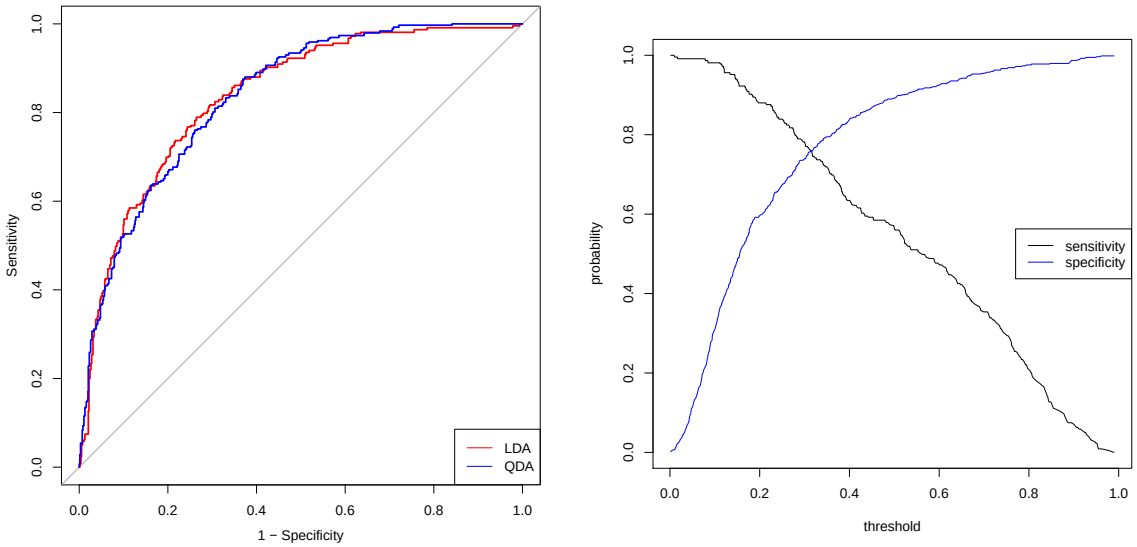

Figure 6: ROC curves for LDA and QDA (left) and sensitivity and specificity curves for LDA as a function of threshold (right).

The ROC curves are presented in Figure 6. They cross several times and there is little difference between them.

(b) With 0.5 cutoff, LDA is better than QDA on the basis of misclassification rate (1.5% lower) and specificity (1.3% higher) but has worse sensitivity (1.2% lower). The AUCs of both models are quite similar: 0.8369 for LDA and QDA for 0.8354. On the whole, there seems little gain with QDA over the simpler LDA. Therefore, we recommend LDA.

The plot of sensitivity and specificity of LDA as a function of threshold is also presented in Figure 6. The curves intersect at the point $(0.315, 0.759)$. Therefore, we may use 0.315 as the threshold. However, in order to have a slightly higher sensitivity of 0.80, we may set the threshold at 0.28, which gives 0.71 specificity.

R-Code

```
## Problem 1 ##
library(PerformanceAnalytics)
library(MASS)
library(lmtest)
library(car)

wine = read.table("wine.txt", header = T)

wine$Region = as.factor(wine$Region)

summary(wine)

# (a)
pdf("scatter_quality.pdf", width = 7, height= 7)
chart.Correlation(wine[ , -7]) #matrix of scatterplot
dev.off()


pdf("Region.pdf", width = 7, height= 7)
boxplot(wine$Quality ~ wine$Region)
dev.off()



# (b)
mod.1 = lm(Quality~., data = wine)
summary(mod.1)
plot(mod.1)

b = boxcox(mod.1) #evaluate possible Box-Cox transformations (MASS package must be installed)
plot(b)

mod.2 = lm(sqrt(Quality)~., data = wine)
summary(mod.2)
plot(mod.2)

mod.3 = lm(log(Quality)~., data = wine)
summary(mod.3)
plot(mod.3)

plot(mod.1$residuals,ylab="Quality_Residuals")
plot(mod.2$residuals,ylab="Sqrt_Quality_Residuals")
plot(mod.3$residuals,ylab="Log_Quality_Residuals")

# (c)
#For each predictor, fit a simple linear regression model to predict
#the response. Describe your results. In which of the models is
#there a statistically significant association between the predictor
#and the response? Create some plots to back up your assertions.
options(digits = 3)
pval=c()
# regress log transformed psa against each predictor separately
for (i in 1:ncol(wine))
{
  if(i!=6)
  {
    result=summary(lm(wine$Quality ~ wine[,i]))
    pval=c(pval,result$coefficients[2,4]) # record p-values of t-test
  }
}
M=NULL
M=data.frame(pval)
M=cbind(names(wine)[c(1:5,7)],M,ifelse(pval<0.05,'Yes','No'))
names(M)=c('Predictor','T-test p-value','Significant')
print(M) # print results


# (d)
#Fit a multiple regression model to predict the response using
```

```
#all of the predictors. Describe your results. For which predictors
#can we reject the null hypothesis
#regress length against all predictors
mod.1=lm(Quality ~., data = wine)
summary(mod.1)

mod.1b = lm(Quality ~Clarity + Aroma + Body + Flavor + Oakiness, data = wine)

anova(mod.1b, mod.1)

# (e)
#Build a "reasonably good" multiple regression model for these data. Be sure to explore interac-
#tions of Region with other predictors. Carefully justify all the choices you make in building
#the model and verify the model assumptions.
#Including interaction with qualitative variable
mod.2 = lm(Quality ~ (Flavor + Region + Flavor:Region), data = wine)
summary(mod.2)

mod.3 = lm(Quality ~ (Flavor + Region), data = wine)
summary(mod.3)

anova(mod.3, mod.1)

anova(mod.3, mod.2)


# Check model assumptions

plot(fitted(mod.3), resid(mod.3), ylab = "residuals",
xlab = "fitted values")
abline(h=0, lty = 2)

plot(mod.3)

shapiro.test(mod.3$residuals) # Shapiro-Wilk - normality
bptest(mod.3) # Breusch Pagan Test


# (g)
# predict Quality for wine whose quantitative predictors are at the sample means
# of the variables and qualitative predictors are at the most frequent category
7.09 + 1.12*mean(wine$Flavor)

#confidence and prediction interval
predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "1"), interval="confidence")
predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "1"), interval="prediction")

predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "2"), interval="confidence")
predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "2"), interval="prediction")

predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "3"), interval="confidence")
predict(mod.3, data.frame(Flavor = mean(wine$Flavor), Region = "3"), interval="prediction")



##################################################
## Problem 2 ##
library(MASS)

admission = read.csv("admission.csv", header = T)
attach(admission)

test_index = c()
for(i in 1:3)
{
  test_index = c(test_index,which(admission$Group==i)[1:5])
}

test.x = cbind(GPA,GMAT)[test_index,]
test.y = cbind(Group)[test_index]
train.x = cbind(GPA,GMAT)[-test_index,]
train.y = Group[-test_index]
```

```
as.factor(train.y)

# (a)
pdf("admission_explore.pdf", width = 7, height= 7)
par(mfrow=c(1,2))
boxplot(admission[-test_index,]$GPA ~ admission[-test_index,]$Group, ylab="GPA", xlab="Groups")
boxplot(admission[-test_index,]$GMAT ~ admission[-test_index,]$Group,  ylab="GMAT", xlab="Groups")
par(mfrow=c(1,1))
dev.off()

# (b)
# Performing LDA on training data
train = data.frame(cbind(train.x,train.y))
lda_fit = lda(train.y ~ GPA+GMAT, data=train)
coef(lda_fit)

lda.pred_train = predict(lda_fit, data.frame(train.x))
lda.pred_train$posterior
table(lda.pred_train$class, train.y)
mean(lda.pred_train$class != train.y)

lda.pred_test = predict(lda_fit, data.frame(test.x))
lda.pred_test$posterior
table(lda.pred_test$class, test.y)
mean(lda.pred_test$class!=test.y)

n.grid = 100
x1.grid = seq(f = 0, t = 4, l = n.grid)
x2.grid = seq(f = 300, t = 700, l = n.grid)
grid = expand.grid(x1.grid, x2.grid)
colnames(grid) = colnames(test.x)

pred.grid = predict(lda_fit, grid)

pdf("lda_decision.pdf", width = 7, height= 7)
prob1 = matrix(pred.grid$posterior[, "1"], nrow = n.grid, ncol = n.grid, byrow = F)
prob2 = matrix(pred.grid$posterior[, "2"], nrow = n.grid, ncol = n.grid, byrow = F)
plot(train.x, xlab = "GPA", ylab = "GMAT", col = ifelse(train.y == 1, "green",ifelse(train.y==2,"blue","red")))
contour(x1.grid, x2.grid, prob1, levels = 0.5, labels = "", xlab = "", ylab = "",
        main = "", add = T)
contour(x1.grid, x2.grid, prob2, levels = 0.5, labels = "", xlab = "", ylab = "",
        main = "", col="red",add = T)
legend("topleft", legend = c("Group 1", "Group 2", "Group 3"), col = c("green", "blue", "red"), pch=1)
dev.off()


# (c)
train = data.frame(cbind(train.x,train.y))
colnames(train) = c("GPA","GMAT","y")

qda_fit = qda(y~GPA+GMAT, data=train)


# Get predictions for test data

qda.pred_train = predict(qda_fit, data.frame(train.x))
table(qda.pred_train$class, train.y)
mean(qda.pred_train$class!=train.y)

qda.pred_test = predict(qda_fit, data.frame(test.x))
table(qda.pred_test$class, test.y)
mean(qda.pred_test$class!=test.y)

pdf("qda_decision.pdf", width = 7, height= 7)
prob3 = matrix(pred.grid$posterior[, 1], nrow = n.grid, ncol = n.grid, byrow = F)
prob4 = matrix(pred.grid$posterior[, 2], nrow = n.grid, ncol = n.grid, byrow = F)
plot(train.x, xlab = "GPA", ylab = "GMAT", col = ifelse(train.y == 1, "green",ifelse(train.y==2,"blue","red")))
contour(x1.grid, x2.grid, prob3, levels = 0.5, labels = "", xlab = "", ylab = "",
        main = "", add = T)
contour(x1.grid, x2.grid, prob4, levels = 0.5, labels = "", xlab = "", ylab = "",
        main = "", col="red",add = T)
```

```
legend("topleft", legend = c("Group 1", "Group 2", "Group 3"), col = c("green", "blue", "red"), pch=1)
dev.off()



########################################
## Problem 3 ##
library(MASS)
library(class)
library(pROC)

diabetes = read.csv("diabetes.csv", header = T)

train.y = as.factor(diabetes$Outcome)
train.x = diabetes[,-9]

# (a)
pdf("diabetes_explore.pdf", width = 14, height= 14)
par(mfrow=c(2,4))
boxplot(diabetes$Pregnancies.. ~ diabetes$Outcome, ylab="Pregnancies", xlab="Outcome")
boxplot(diabetes$Glucose.. ~ diabetes$Outcome,  ylab="Glucose", xlab="Outcome")
boxplot(diabetes$BloodPressure.. ~ diabetes$Outcome, ylab="BloodPressure", xlab="Outcome")
boxplot(diabetes$SkinThickness.. ~ diabetes$Outcome,  ylab="SkinThickness", xlab="Outcome")
boxplot(diabetes$Insulin.. ~ diabetes$Outcome, ylab="Insulin", xlab="Outcome")
boxplot(diabetes$BMI.. ~ diabetes$Outcome,  ylab="BMI", xlab="Outcome")
boxplot(diabetes$DiabetesPedigreeFunction.. ~ diabetes$Outcome, ylab="DiabetesPedigreeFunction", xlab="Outcome")
boxplot(diabetes$Age.. ~ diabetes$Outcome,  ylab="Age", xlab="Outcome")
par(mfrow=c(1,1))
dev.off()

# (b)
lda.fit = lda(as.factor(Outcome) ~ ., data = diabetes)
coef(lda.fit)

lda.pred = predict(lda.fit, diabetes)
table(lda.pred$class, diabetes$Outcome)
mean(lda.pred$class != diabetes$Outcome)

#  (c)
qda.fit = qda(as.factor(Outcome) ~ ., data = diabetes)
qda.pred = predict(qda.fit, diabetes)
table(qda.pred$class, diabetes$Outcome)
mean(qda.pred$class != diabetes$Outcome)

# (d)
#ROC Curve
roc.lda = roc(train.y, lda.pred$posterior[, "1"], levels = c("0", "1"))
roc.qda = roc(train.y, qda.pred$posterior[, "1"], levels = c("0", "1"))

pdf("ROC.pdf", width = 7, height= 7)
plot(roc.lda, col="RED", legacy.axes = T)
plot(roc.qda,add=T, col="BLUE", legacy.axes = T)
legend("bottomright", legend = c("LDA", "QDA"),
col = c("red", "blue"), lty =1)
dev.off()

lda.pred.adj = ifelse(lda.pred$posterior[, 2] > .30, "1", "0")

#Misclasification rate
table(lda.pred.adj, diabetes$Outcome)

pdf("threshold.pdf", width = 7, height= 7)

plot(roc.lda$thresholds, roc.lda$sensitivities, type = "l",
col="black", xlab = "threshold", ylab = "probability")
lines(roc.lda$thresholds, roc.lda$specificities,
col = "blue")
legend("right", legend = c("sensitivity", "specificity"),
col = c("black", "blue"), lty =1)
dev.off()
```