HW 3 101C JIASHU MIAO

Jiashu Miao

804786709

Question 1

a. On the test set, we expect LDA to perform better than QDA, because QDA could overfit the linearity on the Bayes decision boundary. If the Bayes decision boundary is linear, we expect QDA to perform better on the training set because its higher flexiblity may yield a closer fit.

b. We expect QDA to perform better both on the training and test sets, if the Bayes decision boundary is non-linear.

c. Usually when the sample size n increases, we think QDA (which is more flexible than LDA and so has higher variance) is recommended if the training set is very large, therefore the variance of the classifier is not a major issue.

d. False. When there are fewer sample points, the variance from using a more flexible method such as QDA, will lead to overfitting, which could cause an inferior test error rate.

Question 2

a.

```
require(ISLR)

## Loading required package: ISLR

require(MASS)

## Loading required package: MASS

require(class)

## Loading required package: class

attach(Weekly)
summary(Weekly)

##       Year           Lag1               Lag2               Lag3

##   Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.19
50
##   1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.15
```

```
80
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.24
10
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.14
72
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.40
90
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.02
60
##       Lag4               Lag5              Volume
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

```
head(Weekly)
```

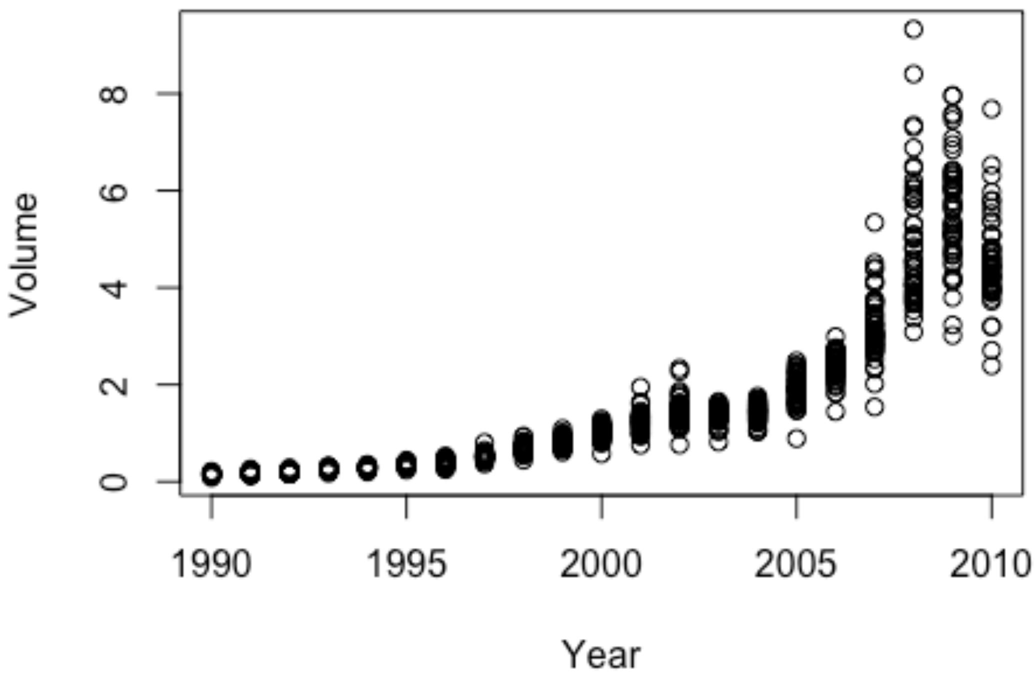
```
##   Year   Lag1   Lag2   Lag3   Lag4   Lag5    Volume  Today Direction
## 1 1990  0.816  1.572 -3.936 -0.229 -3.484 0.1549760 -0.270      Down
## 2 1990 -0.270  0.816  1.572 -3.936 -0.229 0.1485740 -2.576      Down
## 3 1990 -2.576 -0.270  0.816  1.572 -3.936 0.1598375  3.514        Up
## 4 1990  3.514 -2.576 -0.270  0.816  1.572 0.1616300  0.712        Up
## 5 1990  0.712  3.514 -2.576 -0.270  0.816 0.1537280  1.178        Up
## 6 1990  1.178  0.712  3.514 -2.576 -0.270 0.1544440 -1.372      Down
```

```
cor(Weekly[,-9])
```

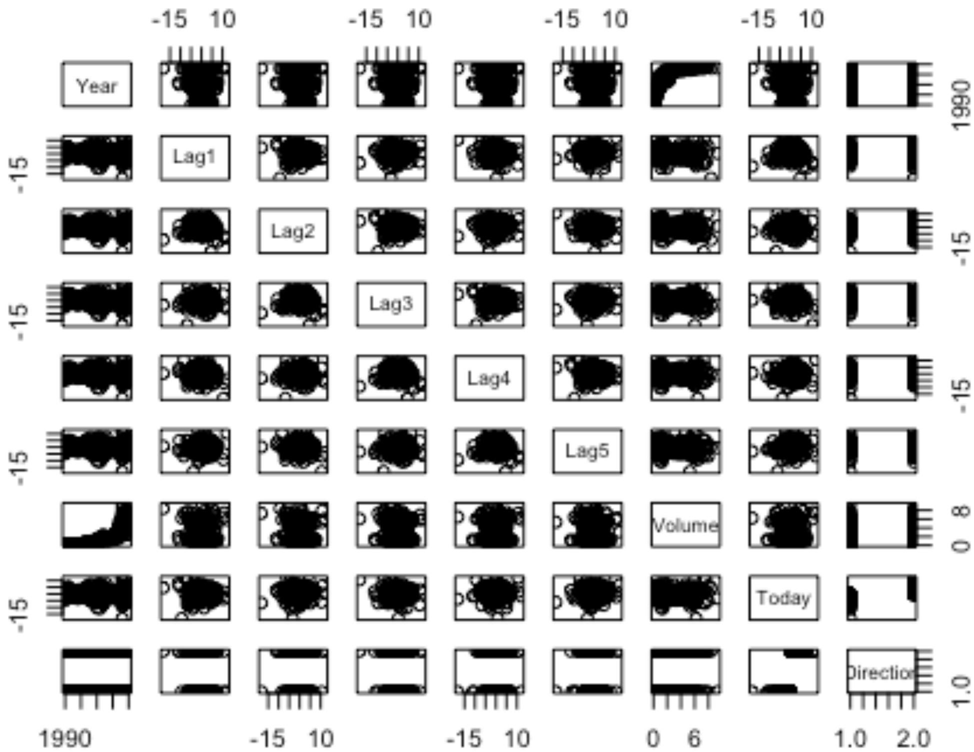
```
##               Year         Lag1        Lag2        Lag3         Lag4
## Year    1.00000000 -0.032289274 -0.03339001 -0.03000649 -0.031127923
## Lag1   -0.03228927  1.000000000 -0.07485305  0.05863568 -0.071273876
## Lag2   -0.03339001 -0.074853051  1.00000000 -0.07572091  0.058381535
## Lag3   -0.03000649  0.058635682 -0.07572091  1.00000000 -0.075395865
## Lag4   -0.03112792 -0.071273876  0.05838153 -0.07539587  1.000000000
## Lag5   -0.03051910 -0.008183096 -0.07249948  0.06065717 -0.075675027
## Volume  0.84194162 -0.064951313 -0.08551314 -0.06928771 -0.061074617
## Today  -0.03245989 -0.075031842  0.05916672 -0.07124364 -0.007825873
##                Lag5      Volume        Today
## Year   -0.030519101  0.84194162 -0.032459894
## Lag1   -0.008183096 -0.06495131 -0.075031842
## Lag2   -0.072499482 -0.08551314  0.059166717
## Lag3    0.060657175 -0.06928771 -0.071243639
## Lag4   -0.075675027 -0.06107462 -0.007825873
```

```
## Lag5    1.000000000 -0.05851741  0.011012698
## Volume -0.058517414  1.00000000 -0.033077783
## Today   0.011012698 -0.03307778  1.000000000
```

```
plot(Volume~Year)
```




```
pairs(Weekly)
```



- I find that volume and years are probably two factors that have correlation accordign to the corvariance table and summary, and when you plot, you find the volume factors increases as time goes by which confirm my conclusion from the summary.

b.

```
m1 <- glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume, family = binomial)
summary(m1)

##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
```

```
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

- According to the summary, only the facor Lag2 seems to be siginificant with p-value smaller than the siginificance level as 0.0296.

c.

```
m1predict <- predict(m1)
m1predict2 = exp(m1predict) / (1+exp(m1predict))
guess <- as.numeric(m1predict2 >= 0.5)

correct <- table(guess, Direction)
correct

##      Direction
## guess Down  Up
##     0   54  48
##     1  430 557

rate <- (correct[1,1]+correct[2,2])/sum(correct)
rate

## [1] 0.5610652
```

- The correct cases predicted by the logistic model is around 0.5610652. The logestic regression uses the training data, because usually it is better and more optimistic than simply using the testing data, because usually we are more interested in the future trend and movement from the model perfrom on the training data instead of using testing data to fit the model. We care more about the prediction and uknowness.

d.

```
training.data <- Year < 2009
testing.data <- Weekly[!training.data, ]
m2 <- glm(Direction~Lag2, family=binomial, data=Weekly, subset=training.
data)
glm.predict2 <- predict(m2, newdata = testing.data)
pred2 = exp(glm.predict2) / (1+exp(glm.predict2))
```

```
testpreds <- as.numeric(pred2 >= 0.5)
correct2 <- table(testpreds, Direction[!training.data])
correct2

##
## testpreds Down Up
##         0    9  5
##         1   34 56

rate2 <- (correct2[1,1]+correct2[2,2])/sum(correct2)
rate2

## [1] 0.625
```

- The correction rate of the testing data here is 0.625.

e.

```
ldafit = lda(Direction~Lag2, subset = training.data)
lda.pred = predict(ldafit, newdata=testing.data, type="response")
correct3 <- table(lda.pred$class, testing.data$Direction)
correct3

##
##        Down Up
##   Down    9  5
##   Up     34 56

rate3 <- (correct3[1,1]+correct3[2,2])/sum(correct3)
rate3

## [1] 0.625
```

- The correct rate for the testing data using LDA is 0.625 which is same as (d).

f.

```
qda.fit = qda(Direction~Lag2, subset = training.data)
qda.pred = predict(qda.fit, newdata=testing.data, type="response")
qda.class = qda.pred$class
correct4 <- table(qda.class, testing.data$Direction)
correct4

##
## qda.class Down Up
##      Down    0  0
##      Up     43 61

rate4 <- (correct4[1,1]+correct4[2,2])/sum(correct4)
rate4

## [1] 0.5865385
```

- This time the correction rate prediction goes down to 0.587 for the testnig test, which is lower than the rate from LDA and the logistic regression.

g.

```
train.X <- as.data.frame(Lag2[training.data])
test.X <- as.data.frame(Lag2[!training.data])
set.seed(1)
knn.pred = knn(train.X, test.X, Direction[training.data], k=1)
correct5 <- table(knn.pred, testing.data$Direction)
correct5

##
## knn.pred Down Up
##     Down   21 30
##     Up     22 31

rate5 <- (correct5[1,1]+correct5[2,2])/sum(correct5)
rate5

## [1] 0.5
```

- The overall correction rate with K=1 is 0.5.

i.

```
m3 <- glm(Direction~Lag2+Volume, family=binomial, data=Weekly, subset=t
raining.data)
glm.predict3 <- predict(m3, newdata = testing.data)
pred3 = exp(glm.predict3) / (1+exp(glm.predict3))
testpreds <- as.numeric(pred2 >= 0.5)
a <- table(testpreds, Direction[!training.data])
a

##
## testpreds Down Up
##         0    9  5
##         1   34 56

aa <- (a[1,1]+a[2,2])/sum(a)
aa

## [1] 0.625

lda.2 <- lda(Direction~Lag2+Lag1+Lag2+Lag3+Lag4+Lag5+Volume, subset=tra
ining.data)
lda.predict2 <- predict(lda.2, testing.data)
b <- table(lda.predict2$class, Direction[!training.data])
b

##
##         Down Up
##   Down    31 44
##   Up      12 17
```

```r
bb <- (b[1,1]+b[2,2])/sum(b)
bb

## [1] 0.4615385

lda.3 <- lda(Direction~Lag2+Volume, subset=training.data)
lda.predict3 <- predict(lda.3, testing.data)
c <- table(lda.predict3$class, Direction[!training.data])
c

##
##         Down Up
##    Down   20 25
##    Up     23 36

cc <- (c[1,1]+c[2,2])/sum(c)
cc

## [1] 0.5384615

qda.2 <- qda(Direction~Lag2+Lag1+Lag2+Lag3+Lag4+Lag5+Volume, subset=tra
ining.data)
qda.predict2 <- predict(qda.2, testing.data)
d <- table(qda.predict2$class, Direction[!training.data])
d

##
##         Down Up
##    Down   33 49
##    Up     10 12

dd <- (d[1,1]+d[2,2])/sum(d)
dd

## [1] 0.4326923

qda.3 <- qda(Direction~Lag2+Volume, subset=training.data)
qda.predict3 <- predict(qda.3, testing.data)
e <- table(qda.predict3$class, Direction[!training.data])
e

##
##         Down Up
##    Down   32 44
##    Up     11 17

ee <- (e[1,1]+e[2,2])/sum(e)
ee

## [1] 0.4711538

knn_result <- 0
for(i in 1:10) {
```

```
  result_i <- knn(train.X, test.X, Direction[training.data], k=i)
  knn_result[i] <- mean(result_i != Direction[!training.data])
}
knn_result

##  [1] 0.5000000 0.5384615 0.4519231 0.4615385 0.4519231 0.4230769 0.4
519231
##  [8] 0.4230769 0.4423077 0.4134615
```

- After comparing LDA,QDA, and linear regression, I find the original Lag2, K=1 is the best predictor to yield best result.

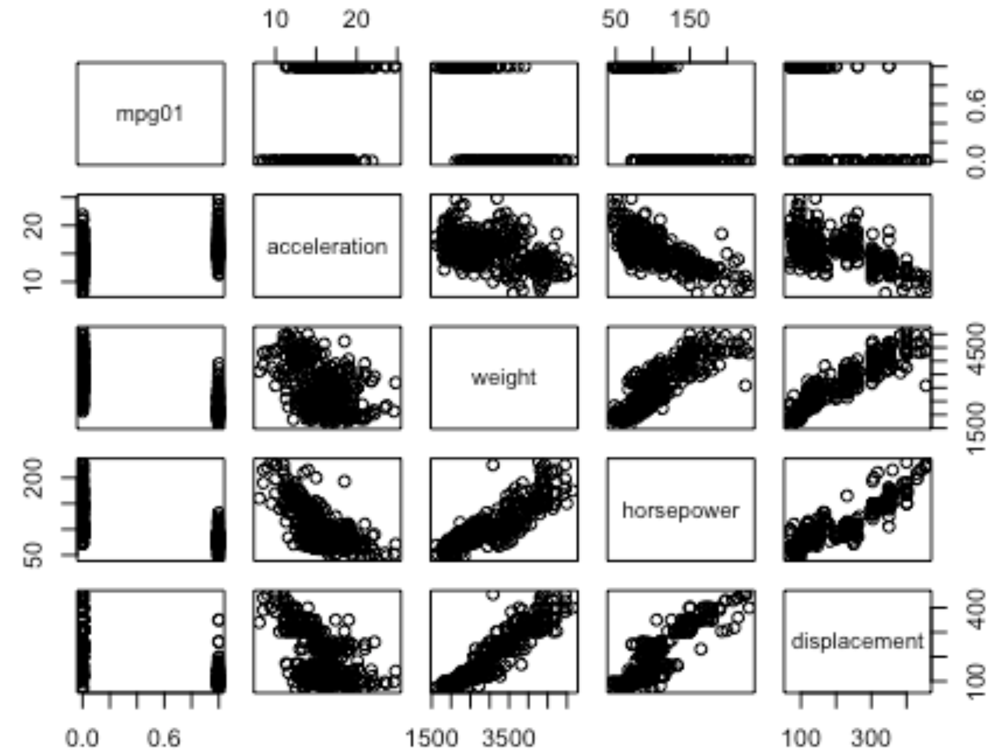
Question 3

a.

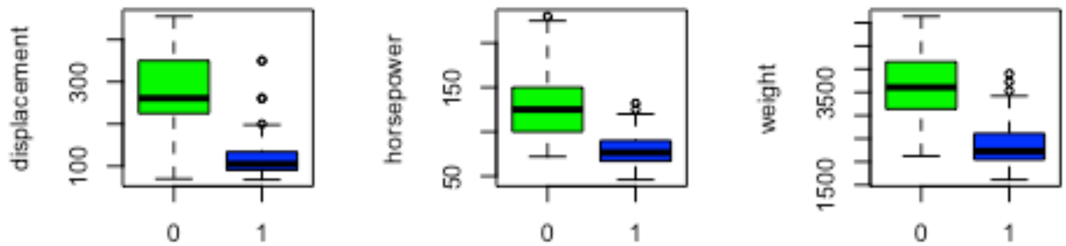
```
attach(Auto)
mpg01 <- rep(0,length(mpg))
mpg01[mpg > median(mpg)] <- 1
Auto <- data.frame(Auto, mpg01)
head(Auto)

##   mpg cylinders displacement horsepower weight acceleration year ori
gin
## 1  18         8          307        130   3504         12.0   70
  1
## 2  15         8          350        165   3693         11.5   70
  1
## 3  18         8          318        150   3436         11.0   70
  1
## 4  16         8          304        150   3433         12.0   70
  1
## 5  17         8          302        140   3449         10.5   70
  1
## 6  15         8          429        198   4341         10.0   70
  1
##                        name mpg01
## 1 chevrolet chevelle malibu     0
## 2         buick skylark 320     0
## 3        plymouth satellite     0
## 4             amc rebel sst     0
## 5               ford torino     0
## 6          ford galaxie 500     0
```

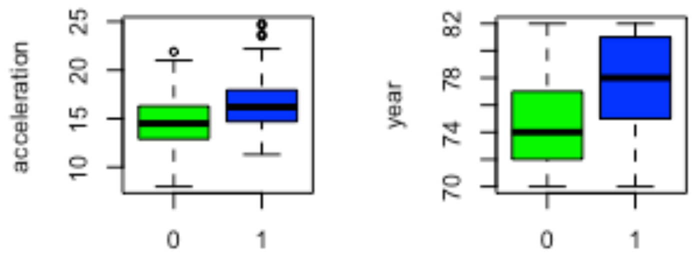
b.

```
pairs(mpg01~acceleration+weight+horsepower+displacement)
```




```
par(mfrow=c(2,3))
for(i in names(Auto)){
  # excluding the own mpgs variables and others categorical variables
  if( grepl(i, pattern="^mpg|cylinders|origin|name")){ next}
  boxplot(eval(parse(text=i)) ~ mpg01, ylab=i, col=c("green", "blue"))
}
```





- I feel that 4 factors which are accelerate, weight, displacement, horsepower have some association with mpg01.

c.

```
set.seed(76776889)
rows <- sample(x=nrow(Auto), size=.70*nrow(Auto))
trainset <- Auto[rows, ]
testset <- Auto[-rows, ]
```

- Split the data into 70% and 30%.

d.

```
auto.lda <- lda(mpg01~horsepower+weight+displacement+acceleration)
auto.lda.predict <- predict(auto.lda, testset)
table(auto.lda.predict$class, mpg01[-rows])

##
##      0  1
##   0 48  2
##   1 13 55

(2+13)/(48+2+13+55)

## [1] 0.1271186
```

- The testing error rate is 0.1271186 for lda.

e.

```
auto.qda <- qda(mpg01~horsepower+weight+displacement+acceleration)
auto.qda.predict <- predict(auto.qda, testset)
table(auto.qda.predict$class, mpg01[-rows])
```

```
##
##      0  1
##   0 51  3
##   1 10 54
```

```
(3+10)/(51+3+10+54)
```

```
## [1] 0.1101695
```

- The testing error rate is 0.1101695 for qda.

f.

```
auto.glm <- glm(mpg01~horsepower+weight+acceleration+displacement, fami
ly=binomial)
auto.glm.predict <- predict(auto.glm, newdata = testset)
p = exp(auto.glm.predict) / (1+exp(auto.glm.predict))
guess <- as.numeric(p >= 0.5)
table(guess, mpg01[-rows])
```

```
##
## guess  0  1
##     0 52  6
##     1  9 51
```

```
(6+9)/(51+6+9+51)
```

```
## [1] 0.1282051
```

- The testinmg error obtained through logistic regression is 0.1282051

g.

```
sel.variables <- which(names(trainset)%in%c("mpg01", "displacement", "h
orsepower", "weight", "acceleration"))

set.seed(76776889)
accuracies <- data.frame("k"=1:10, acc=NA)
for(k in 1:10){
  knn.pred <- knn(train=trainset[, sel.variables], test=testset[, sel.v
ariables], cl=trainset$mpg01, k=k)

  # test-error
  accuracies$acc[k]= round(sum(knn.pred!=testset$mpg01)/nrow(testset)*1
00,2)
}
```

```
accuracies

##     k   acc
## 1   1 11.86
## 2   2 11.86
## 3   3 11.02
## 4   4 11.86
## 5   5 11.86
## 6   6 11.02
## 7   7 11.86
## 8   8 11.86
## 9   9 10.17
## 10 10 13.56
```

- We can see that when K=9 has the lowest error rate, which is best choice.

Question 4

```
set.seed(76776889)
rowb <- sample(1:nrow(Boston),nrow(Boston)*0.7, replace = F)

train <- Boston[rowb, -1]
test <- Boston[-rowb, -1]
crime.rate <- rep(0, 506)
crime.rate[Boston[[1]] > median(Boston[[1]])] = 1
Y.train <- crime.rate[rowb]
Y.test <- crime.rate[-rowb]

attach(Boston)
summary(glm(crime.rate~zn+indus+chas+nox+rm+age+dis+rad+tax+ptratio+bla
ck+lstat+medv, family=binomial()))

##
## Call:
## glm(formula = crime.rate ~ zn + indus + chas + nox + rm + age +
##     dis + rad + tax + ptratio + black + lstat + medv, family = binom
ial())
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3946  -0.1585  -0.0004   0.0023   3.4239
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -34.103704   6.530014  -5.223 1.76e-07 ***
## zn           -0.079918   0.033731  -2.369  0.01782 *
## indus        -0.059389   0.043722  -1.358  0.17436
## chas          0.785327   0.728930   1.077  0.28132
## nox          48.523782   7.396497   6.560 5.37e-11 ***
## rm           -0.425596   0.701104  -0.607  0.54383
```

```
## age           0.022172   0.012221   1.814  0.06963 .
## dis           0.691400   0.218308   3.167  0.00154 **
## rad           0.656465   0.152452   4.306 1.66e-05 ***
## tax          -0.006412   0.002689  -2.385  0.01709 *
## ptratio       0.368716   0.122136   3.019  0.00254 **
## black        -0.013524   0.006536  -2.069  0.03853 *
## lstat         0.043862   0.048981   0.895  0.37052
## medv          0.167130   0.066940   2.497  0.01254 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 701.46  on 505  degrees of freedom
## Residual deviance: 211.93  on 492  degrees of freedom
## AIC: 239.93
##
## Number of Fisher Scoring iterations: 9
```

Logestic Regression

three variables: nox, rad, dis

```
lgglm3 <- glm(crime.rate~nox+rad+dis, family=binomial)
crglm.predict3 <- predict(lgglm3, newdata = test)
cr.glm.pred3 <- as.numeric(exp(crglm.predict3) / (1+exp(crglm.predict3))
 >= 0.5)
table(cr.glm.pred3, Y.test)

##              Y.test
## cr.glm.pred3  0  1
##             0 68 14
##             1  9 61

(68+61)/(68+14+9+61)

## [1] 0.8486842
```

four variables: nox, rad, dis, ptratio

```
lgglm4 <- glm(crime.rate~nox+rad+dis+ptratio, family=binomial)
crglm.predict4 <- predict(lgglm4, newdata = test)
cr.glm.pred4 <- as.numeric(exp(crglm.predict4) / (1+exp(crglm.predict4))
 >= 0.5)
table(cr.glm.pred4, Y.test)

##              Y.test
## cr.glm.pred4  0  1
##             0 71 14
##             1  6 61

(71+61)/(71+61+14+6)
```

```
## [1] 0.8684211
```

five variables: nox, rad, dis, ptratio, medv

```
lgglm5 <- glm(crime.rate~nox+rad+dis+ptratio+medv, family=binomial)
crglm.predict5 <- predict(lgglm5, newdata = test)
cr.glm.pred5 <- as.numeric(exp(crglm.predict5) / (1+exp(crglm.predict5))
 >= 0.5)
table(cr.glm.pred5, Y.test)

##               Y.test
## cr.glm.pred5  0  1
##              0 72 13
##              1  5 62

(71+61)/(71+61+14+6)

## [1] 0.8684211
```

- Logestic regression with 5 predictors yields best result.

LDA

```
lda(crime.rate~zn+indus+chas+nox+rm+age+dis+rad+tax+ptratio+black+lstat
+medv)

## Call:
## lda(crime.rate ~ zn + indus + chas + nox + rm + age + dis + rad +
##     tax + ptratio + black + lstat + medv)
##
## Prior probabilities of groups:
##   0   1
## 0.5 0.5
##
## Group means:
##          zn     indus       chas       nox       rm      age      di
s
## 0 21.525692  7.002292 0.05138340 0.4709711 6.394395 51.31028 5.09159
6
## 1  1.201581 15.271265 0.08695652 0.6384190 6.174874 85.83953 2.49848
9
##         rad      tax  ptratio    black     lstat     medv
## 0  4.158103 305.7431 17.90711 388.7061  9.419486 24.94941
## 1 14.940711 510.7312 19.00395 324.6420 15.886640 20.11621
##
## Coefficients of linear discriminants:
##                   LD1
## zn      -0.0054345920
## indus    0.0123186828
## chas    -0.0627520897
## nox      8.1340353679
## rm       0.0893872928
## age      0.0112885889
```

```
## dis       0.0407823970
## rad       0.0726344900
## tax      -0.0008619171
## ptratio  0.0501188217
## black   -0.0010241413
## lstat    0.0149784417
## medv     0.0377731894
```

three predictors: nox, rm, ptratio

```
crlda3 <- lda(crime.rate~nox+rm+ptratio)
crlda.predict3 <- predict(crlda3, test)
table(crlda.predict3$class, Y.test)

##    Y.test
##      0  1
##   0 68 12
##   1  9 63

(68+63)/152

## [1] 0.8618421
```

four predictors: nox, rm, ptratio,dis

```
crlda4 <- lda(crime.rate~nox+rm+ptratio+dis)
crlda.predict4 <- predict(crlda4, test)
table(crlda.predict4$class, Y.test)

##    Y.test
##      0  1
##   0 65 10
##   1 12 65

(65+65)/152

## [1] 0.8552632
```

five predictors: nox, rm, ptratio, dis, medv

```
crlda5 <- lda(crime.rate~nox+rm+ptratio+dis+medv)
crlda.predict5 <- predict(crlda5, test)
table(crlda.predict5$class, Y.test)

##    Y.test
##      0  1
##   0 65 14
##   1 12 61

(61+65)/152

## [1] 0.8289474
```

- For, LDA, when it is 3 predictors yields the best result.

QDA

three predictors: nox, rm, ptratio

```
crqda3 <- qda(crime.rate~nox+rm+ptratio)
crqda.predict3 <- predict(crqda3, test)
table(crqda.predict3$class, Y.test)

##    Y.test
##      0  1
##   0 67 17
##   1 10 58

(67+58)/152

## [1] 0.8223684
```

four predictors: nox, rm, ptratio,dis

```
crqda4 <- qda(crime.rate~nox+rm+ptratio+dis)
crqda.predict4 <- predict(crqda4, test)
table(crqda.predict4$class, Y.test)

##    Y.test
##      0  1
##   0 64 15
##   1 13 60

(64+60)/152

## [1] 0.8157895
```

five predictors: nox, rm, ptratio, dis, medv

```
crqda5 <- qda(crime.rate~nox+rm+ptratio+dis+medv)
crqda.predict5 <- predict(crqda5, test)
table(crqda.predict5$class, Y.test)

##    Y.test
##      0  1
##   0 67 16
##   1 10 59

(67+59)/152

## [1] 0.8289474
```

- For QDA, it yields best resuld at 5 predictors.

KNN

```
set.seed(1)
cr.knn.result1 <- rep(NA, 20)
cr.train3 <- train[, c("dis", "age", "medv")]
cr.test3 <- test[, c("dis", "age", "medv")]
```

```r
for(i in 1:20) {
  result_i <- knn(cr.train3, cr.test3, Y.train, k=i)
  cr.knn.result1[i] <- mean(result_i == Y.test)
}
max(cr.knn.result1)
```

```
## [1] 0.8289474
```

```r
set.seed(1)
cr.knn.result2 <- rep(NA, 20)
cr.train4 <- train[, c("dis", "age", "medv", "nox")]
cr.test4 <- test[, c("dis", "age", "medv", "nox")]
for(i in 1:20) {
  result_i <- knn(cr.train4, cr.test4, Y.train, k=i)
  cr.knn.result2[i] <- mean(result_i == Y.test)
}
max(cr.knn.result2)
```

```
## [1] 0.8289474
```

```r
set.seed(1)
cr.knn.result3 <- rep(NA, 20)
cr.train5 <- train[, c("dis", "age", "medv", "nox", "indus")]
cr.test5 <- test[, c("dis", "age", "medv", "nox", "indus")]
for(i in 1:20) {
  result_i <- knn(cr.train5, cr.test5, Y.train, k=i)
  cr.knn.result3[i] <- mean(result_i == Y.test)
}
max(cr.knn.result3)
```

```
## [1] 0.8618421
```

- The above are for 3, 4, 5 predictors and we could tell when it is 5 predictors, the result is best at correction rate 0.8618421
- So, the KNN with 5 predictors is best and we chose logestic regression method with 5 predictors which is best.