Homework 4 - STATS 415

Marian L. Schmidt

February 19, 2016

```
library(ISLR)
library(MASS)
```

Question 1

Run LDA and QDA on Training Data

```
x1 <- c(-3, -2, 0, 1, -1, 2, 3, 4, 5, -1.5, -1, 0, 1, 0.5, 1, 2.5, 5)
y1 <- c(-1, -1, -1, -1, 1, 1, 1, 1, 1, -1, -1, -1, -1, 1, 1, 1, 1)
length(x1); length(y1);
```

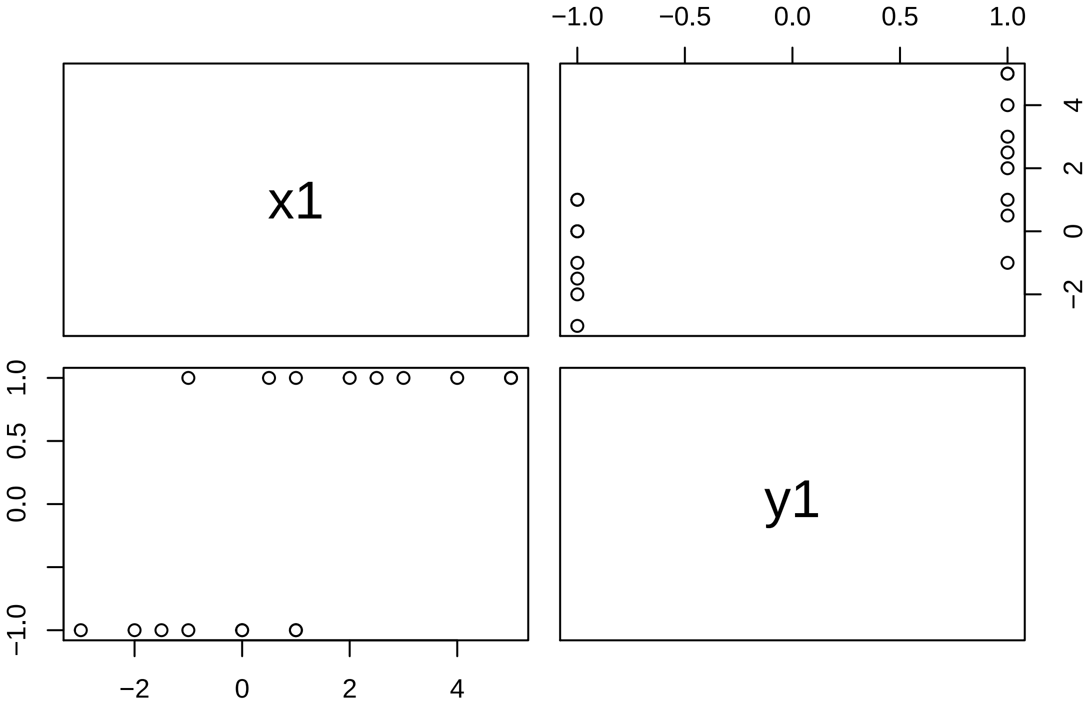
```
## [1] 17
```

```
## [1] 17
```

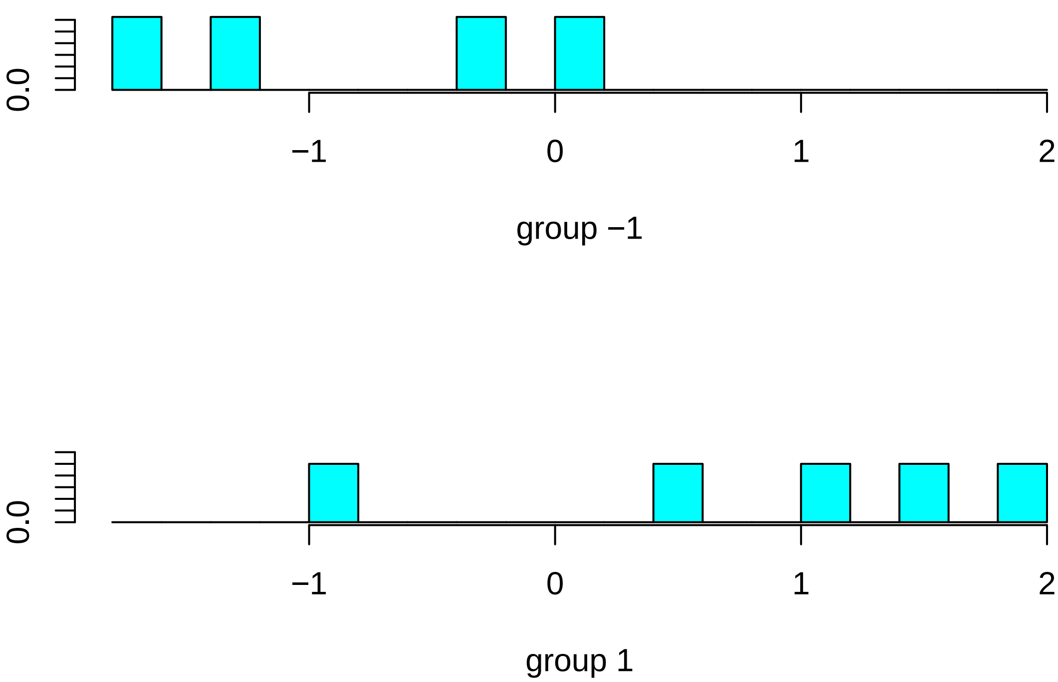
```
df <- data.frame(t(rbind(x1, y1))); df
```

```
##      x1 y1
## 1  -3.0 -1
## 2  -2.0 -1
## 3   0.0 -1
## 4   1.0 -1
## 5  -1.0  1
## 6   2.0  1
## 7   3.0  1
## 8   4.0  1
## 9   5.0  1
## 10 -1.5 -1
## 11 -1.0 -1
## 12  0.0 -1
## 13  1.0 -1
## 14  0.5  1
## 15  1.0  1
## 16  2.5  1
## 17  5.0  1
```

```
pairs(df)
```




```r
# Subset out Training Data
train <- df[1:9,]

# Run LDA on Training Data
lda.train <- lda(y1 ~ x1, data = train)
plot(lda.train)
```



```r
# Run QDA on Training Data
qda.train <- qda(y1 ~ x1, data = train)
```

Test quality of LDA and QDA with testing data

```r
testing <- df[10:17,]

# Run previous LDA on Testing Data
lda.class <- predict(lda.train, testing)$class
table(lda.class, y1[10:17])
```

```
##
## lda.class -1 1
##        -1  3 1
##        1   1 3
```

```r
mean(lda.class != y1[10:17])
```

```
## [1] 0.25
```

```r
# Run previous QDA on Testing Data
qda.class <- predict(qda.train, testing)$class
table(qda.class, y1[10:17])
```

```
##
## qda.class -1 1
##        -1  3 1
##        1   1 3
```

```r
mean(qda.class != y1[10:17])
```

```
## [1] 0.25
```

Above it appears that the test error rate of LDA is 0.25 and that the test error rate of QDA is 0.25.

LDA is a much less flexible classifier than QDA and therefore has a much lower variance. However, if the assumption of uniform variance is false, then LDA can suffer from high bias. In general, LDA tends to be better than QDA if there are relatively few training observations, so therefore reducing variance is crucial. Here, since we have a few (9) training observations then we would prefer LDA.

Question 2

In this problem, you will develop a model to predict whether a given car gets high or low gas mileage based on the Auto data set.

(a) Create a binary variable, `mpg01`, that contains a `1` if mpg contains a value above its median, and a `0` if mpg contains a value below its median. You can compute the median using the `median()` function. Note you may find it helpful to use the `data.frame()` function to create a single data set containing both `mpg01` and the other Auto variables.

```r
summary(Auto[, -9])
```

```
##       mpg          cylinders      displacement     horsepower
##  Min.   : 9.00   Min.   :3.000   Min.   : 68.0   Min.   : 46.0
##  1st Qu.:17.00   1st Qu.:4.000   1st Qu.:105.0   1st Qu.: 75.0
##  Median :22.75   Median :4.000   Median :151.0   Median : 93.5
##  Mean   :23.45   Mean   :5.472   Mean   :194.4   Mean   :104.5
##  3rd Qu.:29.00   3rd Qu.:8.000   3rd Qu.:275.8   3rd Qu.:126.0
##  Max.   :46.60   Max.   :8.000   Max.   :455.0   Max.   :230.0
##      weight      acceleration        year           origin
##  Min.   :1613   Min.   : 8.00   Min.   :70.00   Min.   :1.000
##  1st Qu.:2225   1st Qu.:13.78   1st Qu.:73.00   1st Qu.:1.000
##  Median :2804   Median :15.50   Median :76.00   Median :1.000
##  Mean   :2978   Mean   :15.54   Mean   :75.98   Mean   :1.577
##  3rd Qu.:3615   3rd Qu.:17.02   3rd Qu.:79.00   3rd Qu.:2.000
##  Max.   :5140   Max.   :24.80   Max.   :82.00   Max.   :3.000
```

```r
attach(Auto)

mpg01 <- rep(0, length(mpg)) # create mpg01
mpg01[mpg > median(mpg)] <- 1 # Assign 1 if mpg is above the median
Auto_mpg01 <- data.frame(Auto, mpg01) # combine Auto and mpg01
dim(Auto_mpg01)
```

```
## [1] 392  10
```

```r
head(Auto_mpg01[, -9])
```

```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 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
##   mpg01
## 1     0
## 2     0
## 3     0
## 4     0
## 5     0
## 6     0
```

```r
attach(Auto_mpg01)
```

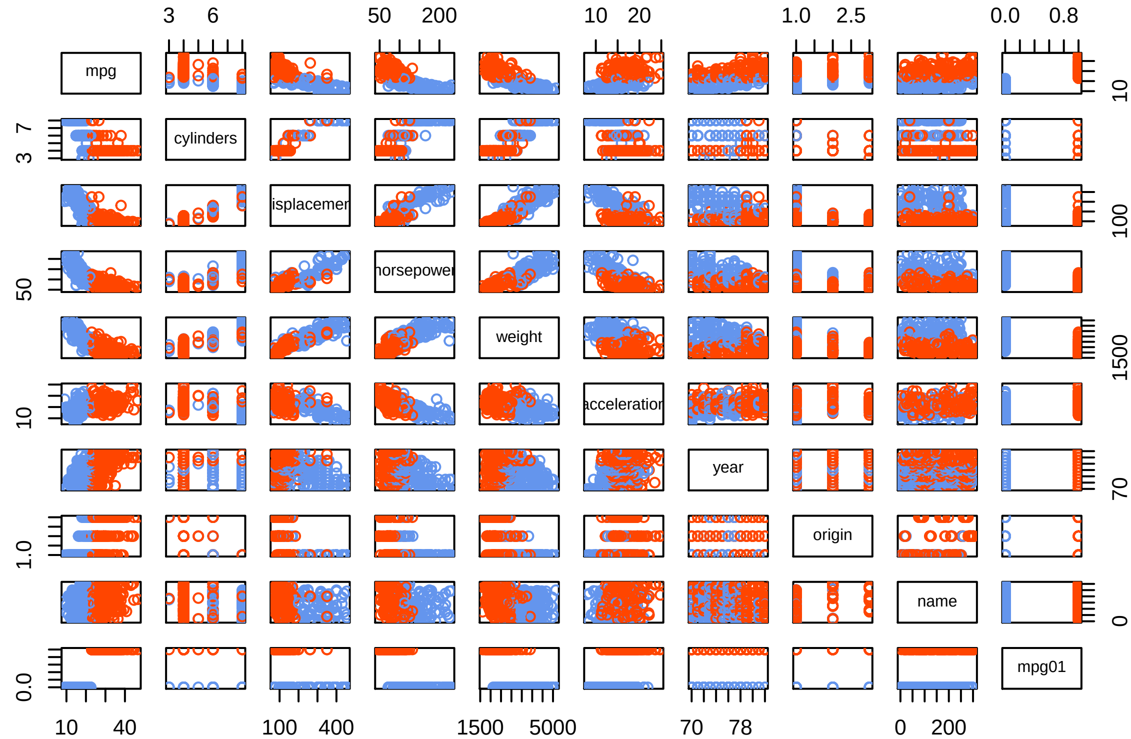
```
## The following object is masked _by_ .GlobalEnv:
##
##     mpg01
##
## The following objects are masked from Auto:
##
##     acceleration, cylinders, displacement, horsepower, mpg, name,
##     origin, weight, year
```

(b) Explore the data graphically in order to investigate the association between `mpg01` and the other features. Which of the other features seem most likely to be useful in predicting `mpg01`? Scatterplots and boxplots may be useful tools to answer this question. Describe your findings.

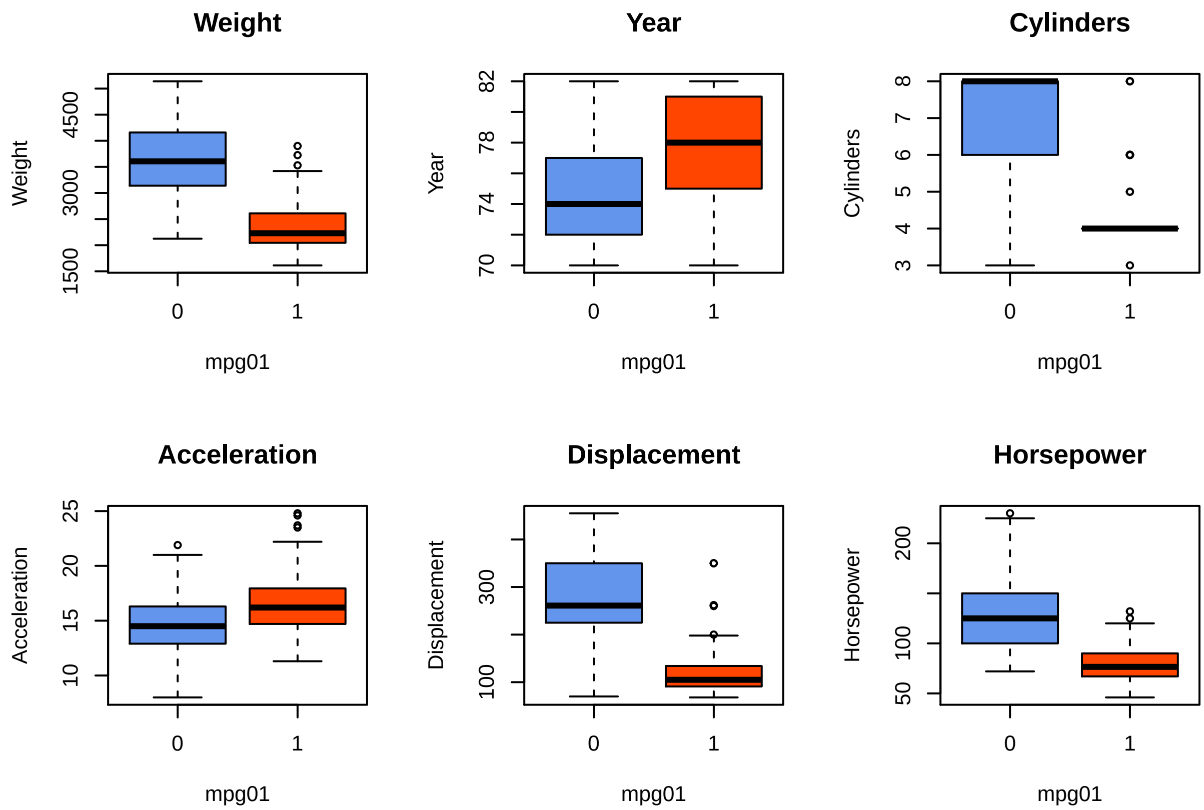
```
cor(Auto_mpg01[, -9]) # Show the correlations between the variables
```

```
##                     mpg  cylinders displacement horsepower     weight
## mpg           1.0000000 -0.7776175   -0.8051269 -0.7784268 -0.8322442
## cylinders    -0.7776175  1.0000000    0.9508233  0.8429834  0.8975273
## displacement -0.8051269  0.9508233    1.0000000  0.8972570  0.9329944
## horsepower   -0.7784268  0.8429834    0.8972570  1.0000000  0.8645377
## weight       -0.8322442  0.8975273    0.9329944  0.8645377  1.0000000
## acceleration  0.4233285 -0.5046834   -0.5438005 -0.6891955 -0.4168392
## year          0.5805410 -0.3456474   -0.3698552 -0.4163615 -0.3091199
## origin        0.5652088 -0.5689316   -0.6145351 -0.4551715 -0.5850054
## mpg01         0.8369392 -0.7591939   -0.7534766 -0.6670526 -0.7577566
##              acceleration       year     origin      mpg01
## mpg             0.4233285  0.5805410  0.5652088  0.8369392
## cylinders      -0.5046834 -0.3456474 -0.5689316 -0.7591939
## displacement   -0.5438005 -0.3698552 -0.6145351 -0.7534766
## horsepower     -0.6891955 -0.4163615 -0.4551715 -0.6670526
## weight         -0.4168392 -0.3091199 -0.5850054 -0.7577566
## acceleration    1.0000000  0.2903161  0.2127458  0.3468215
## year            0.2903161  1.0000000  0.1815277  0.4299042
## origin          0.2127458  0.1815277  1.0000000  0.5136984
## mpg01           0.3468215  0.4299042  0.5136984  1.0000000
```

```
cols <- character(nrow(Auto_mpg01))
cols[] <- "black"
cols[Auto_mpg01$mpg01 == 1] <- "orangered"
cols[Auto_mpg01$mpg01 == 0] <- "cornflowerblue"
pairs(Auto_mpg01, col=cols) # Plot the scatterplot matrix
```




```r
par(mfrow=c(2,3))
boxplot(weight ~ mpg01, data = Auto_mpg01, main = "Weight",
        xlab = "mpg01", ylab = "Weight",
        col = c("cornflowerblue", "orangered"))

boxplot(year ~ mpg01, data = Auto_mpg01, main = "Year",
        xlab = "mpg01", ylab = "Year",
        col = c("cornflowerblue", "orangered"))

boxplot(cylinders ~ mpg01, data = Auto_mpg01, main = "Cylinders",
        xlab = "mpg01", ylab = "Cylinders",
        col = c("cornflowerblue", "orangered"))

boxplot(acceleration ~ mpg01, data = Auto_mpg01, main = "Acceleration",
        xlab = "mpg01", ylab = "Acceleration",
        col = c("cornflowerblue", "orangered"))

boxplot(displacement ~ mpg01, data = Auto_mpg01, main = "Displacement",
        xlab = "mpg01", ylab = "Displacement",
        col = c("cornflowerblue", "orangered"))

boxplot(horsepower ~ mpg01, data = Auto_mpg01, main = "Horsepower",
        xlab = "mpg01", ylab = "Horsepower",
        col = c("cornflowerblue", "orangered"))
```



Based on the above plots, `mpg01` has a negative association with `Weight`, `Cylinders`, `Displacement`, and `Horsepower`.

(c) Split the data into a training set and a test set.

```
train <- (year %% 2 == 0) # Split by even years
Auto_mpg01.train <- Auto_mpg01[train, ]
Auto_mpg01.test <- Auto_mpg01[!train, ]
mpg01.test <- mpg01[!train]
```

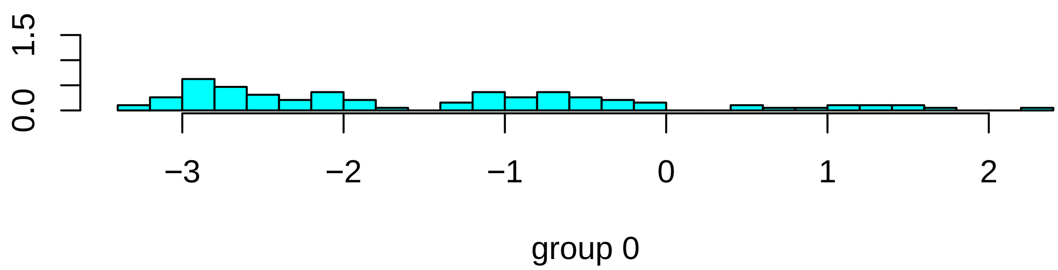
(d) Perform LDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

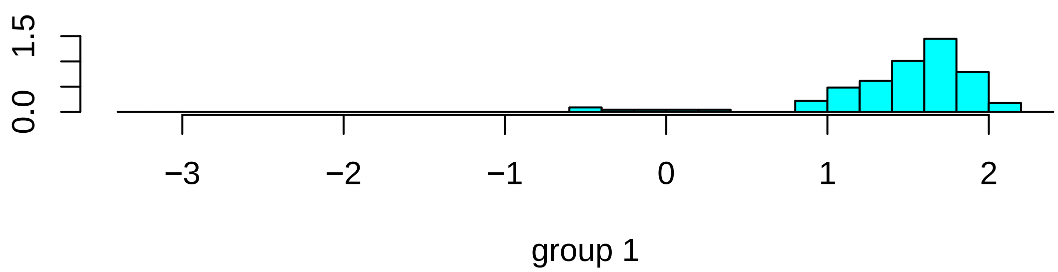
```
fit.lda <- lda(mpg01 ~ cylinders + weight + displacement + horsepower,
               data = Auto_mpg01, subset = train)
fit.lda
```

```
## Call:
## lda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto_mpg01,
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
```

```
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
##
## Coefficients of linear discriminants:
##                       LD1
## cylinders    -0.6741402638
## weight       -0.0011465750
## displacement  0.0004481325
## horsepower    0.0059035377
```

```r
plot(fit.lda)
```






```r
lda.class <- predict(fit.lda, Auto_mpg01.test)$class

table(lda.class, mpg01.test)
```

```
##          mpg01.test
## lda.class  0  1
##         0 86  9
##         1 14 73
```

```r
mean(lda.class != mpg01.test)
```

```
## [1] 0.1263736
```

Using all 4 predictors (`cylinders`, `weight`, `displacement`, and `horsepower`) with a Linear Discriminant Analysis the test error rate is 12.63736%.

(e) Perform QDA on the training data in order to predict mpg01 using the variables that seemed most associated with mpg01 in (b). What is the test error of the model obtained?

```r
qda.fit <- qda(mpg01 ~ cylinders + weight + displacement + horsepower,
               data = Auto_mpg01, subset = train)
qda.fit
```

```
## Call:
## qda(mpg01 ~ cylinders + weight + displacement + horsepower, data = Auto_mpg01,
##     subset = train)
##
## Prior probabilities of groups:
##         0         1
## 0.4571429 0.5428571
##
## Group means:
##   cylinders   weight displacement horsepower
## 0  6.812500 3604.823     271.7396  133.14583
## 1  4.070175 2314.763     111.6623   77.92105
```

```r
qda.class <- predict(qda.fit, Auto_mpg01.test)$class
table(qda.class, mpg01.test)
```

```
##          mpg01.test
## qda.class  0  1
##         0 89 13
##         1 11 69
```

```r
mean(qda.class != mpg01.test)
```

```
## [1] 0.1318681
```

Using all 4 predictors (`cylinders`, `weight`, `displacement`, and `horsepower`) with a Quadratic Discriminant Analysis the test error rate is 13.18681%.

(f) Perform logistic regression on the training data in order to predict `mpg01` using the variables that seemed most associated with `mpg01` in (b). What is the test error of the model obtained?

```r
glm.fit <- glm(mpg01 ~ cylinders + weight + displacement + horsepower,
               data = Auto_mpg01, family = binomial, subset = train)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = mpg01 ~ cylinders + weight + displacement + horsepower,
##     family = binomial, data = Auto_mpg01, subset = train)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.48027  -0.03413   0.10583   0.29634   2.57584
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  17.658730   3.409012   5.180 2.22e-07 ***
```

```
## cylinders    -1.028032   0.653607  -1.573   0.1158
## weight       -0.002922   0.001137  -2.569   0.0102 *
## displacement  0.002462   0.015030   0.164   0.8699
## horsepower   -0.050611   0.025209  -2.008   0.0447 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 289.58  on 209  degrees of freedom
## Residual deviance:  83.24  on 205  degrees of freedom
## AIC: 93.24
##
## Number of Fisher Scoring iterations: 7
```

```r
probs <- predict(glm.fit, Auto_mpg01.test, type = "response")
glm.pred <- rep(0, length(probs))
glm.pred[probs > 0.5] <- 1
table(glm.pred, mpg01.test)
```

```
##         mpg01.test
## glm.pred  0  1
##        0 89 11
##        1 11 71
```

```r
mean(glm.pred != mpg01.test)
```

```
## [1] 0.1208791
```

Using all 4 predictors (`cylinders`, `weight`, `displacement`, and `horsepower`) with a Logistic Regression Analysis the test error rate is 12.08791%.

(g) Perform KNN on the training data, with several values of `K`, in order to predict `mpg01`. Use only the variables that seemed most associated with `mpg01` in (b). What test errors do you obtain? Which value of `K` seems to perform the best on this data set?

```r
library(class)
train.X <- cbind(cylinders, weight, displacement, horsepower)[train, ]
test.X <- cbind(cylinders, weight, displacement, horsepower)[!train, ]
train.mpg01 <- mpg01[train]

###  K = 1
set.seed(12345)
knn.pred.1 <- knn(train.X, test.X, train.mpg01, k = 1)
table(knn.pred.1, mpg01.test)
```

```
##           mpg01.test
## knn.pred.1  0  1
##          0 83 11
##          1 17 71
```

```r
mean(knn.pred.1 != mpg01.test)
```

```
## [1] 0.1538462
```

With KNN Analysis (K = 1), the test error rate is 15.3846154%.

```r
###  K = 5
set.seed(12345)
knn.pred.5 <- knn(train.X, test.X, train.mpg01, k = 5)
table(knn.pred.5, mpg01.test)
```

```
##           mpg01.test
## knn.pred.5  0  1
##          0 82  9
##          1 18 73
```

```r
mean(knn.pred.5 != mpg01.test)
```

```
## [1] 0.1483516
```

With KNN Analysis (K = 5), the test error rate is 14.8351648%.

```r
###  K = 50
set.seed(12345)
knn.pred.50 <- knn(train.X, test.X, train.mpg01, k = 50)
table(knn.pred.50, mpg01.test)
```

```
##            mpg01.test
## knn.pred.50  0  1
##           0 81  7
##           1 19 75
```

```r
mean(knn.pred.50 != mpg01.test)
```

```
## [1] 0.1428571
```

With KNN Analysis (K = 50), the test error rate is 14.2857143%.

```r
###  K = 500
set.seed(12345)
knn.pred.100 <- knn(train.X, test.X, train.mpg01, k = 100)
table(knn.pred.100, mpg01.test)
```

```
##             mpg01.test
## knn.pred.100  0  1
##            0 81  7
##            1 19 75
```

```r
mean(knn.pred.100 != mpg01.test)
```

```
## [1] 0.1428571
```

With KNN Analysis (K = 100), the test error rate is 14.2857143%.

With the above KNN Analyses K = 100 and K = 50 perform the same with a the test error rate of 14.2857143%.