Homework 4

Alec Rospierski

Octoboer 21st, 2022

Interpret Logistic Regression Coefficients

E1. Build a logistic regression model to predict the probability that a student will be in the honors class, based on information we know about the student: male, math = 65, reading = 70. What is the probability?

Probability of a male with a math score of 65 and reading score of 70 being in the honors class is 41.18%.

E2. There are two students, A and B. A's math score is 10 points higher than B's. Build an appropriate model to answer: The odds of A getting in the honors class is ________ times the odds of B getting in the honors class.

Answer: C. 4.775

E3. There are two students, Mary and John (note: gender is implied by name).

(a) If they have the same math score and we have no information about their reading and writing scores, who do you think has a higher chance of being in the honors class?

I think that Mary would have a higher chance to be an honor student. Looking at the table discussed earlier, it can be seen that in general, females have a higher chance of being in honors class.

(b) Suppose John's math score is 7 points higher than Mary's, which statement is right about their odds of being in the honors class?

Answer: A. The odds of John is about 16% higher than the odds of Mary.

ISLR Section 4.7 Exercise 11

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.

```
library(ISLR)
head(Auto)
```

```
##   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
##                        name
## 1 chevrolet chevelle malibu
## 2         buick skylark 320
## 3        plymouth satellite
## 4             amc rebel sst
```

```
## 5                ford torino
## 6           ford galaxie 500
```

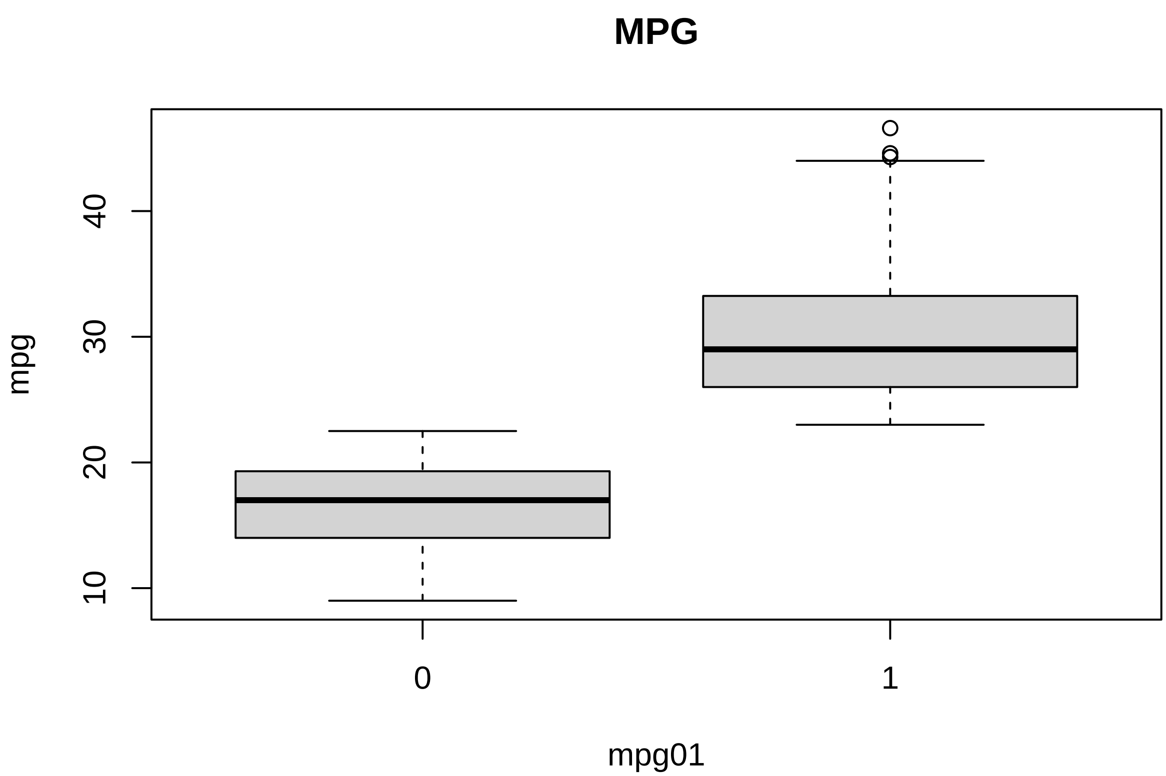
(a) Create a binary variable, mpg01, that contains a 1 if mpg contains a value above its median, and a 0 if mpg contains a value below its median. You can compute the median using the median() function. Note you may find it helpful to use the data.frame() function to create a single data set containing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

```
df <- Auto
df$mpg01 <- ifelse(df$mpg > median(df$mpg), 1, 0)
head(df)
```

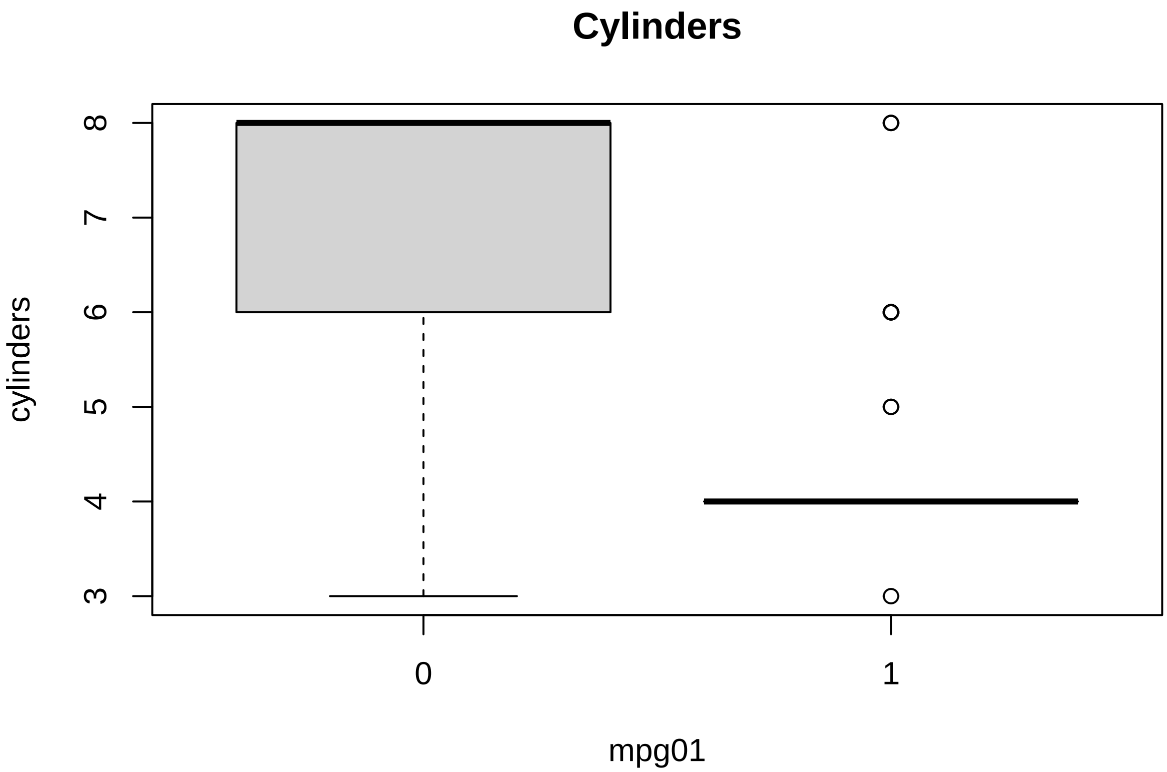
```
##   mpg cylinders displacement horsepower weight acceleration year origin
## 1  18         8          307        130   3504         12.0   70      1
## 2  15         8          350        165   3693         11.5   70      1
## 3  18         8          318        150   3436         11.0   70      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
##                        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) 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.

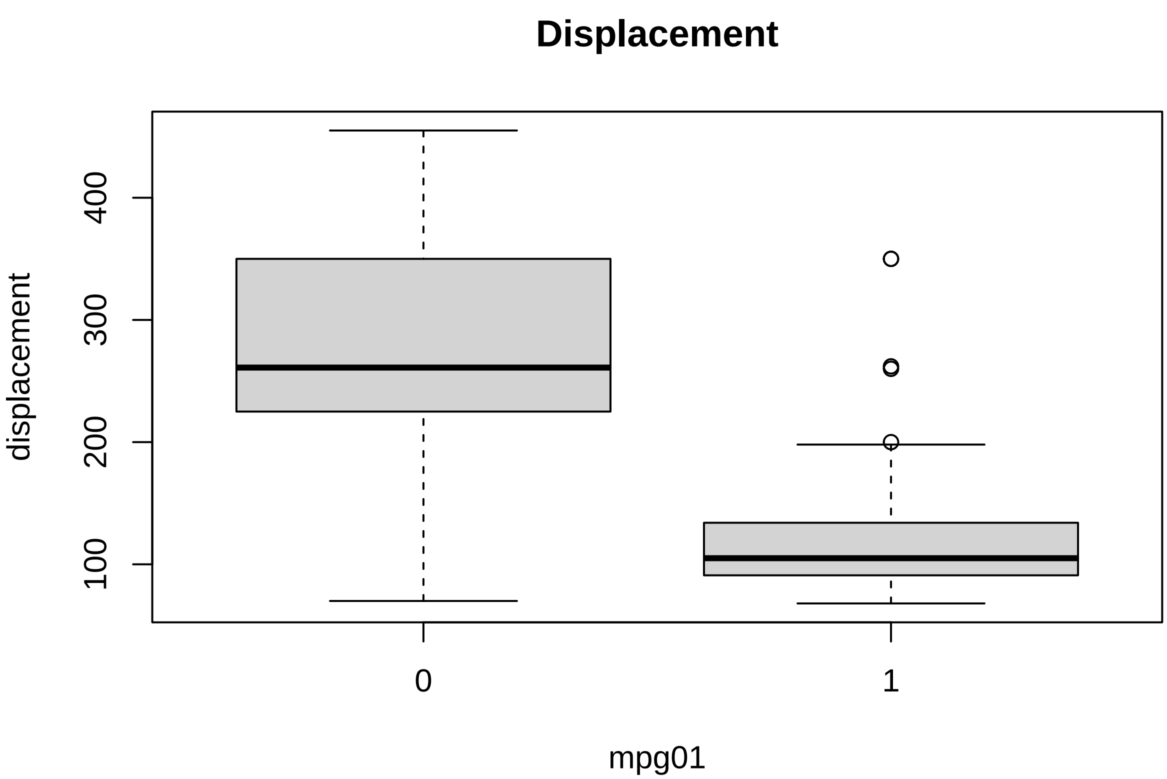
```
boxplot(mpg ~ mpg01, data = df, xlab = 'mpg01', ylab = 'mpg', main = 'MPG')
```



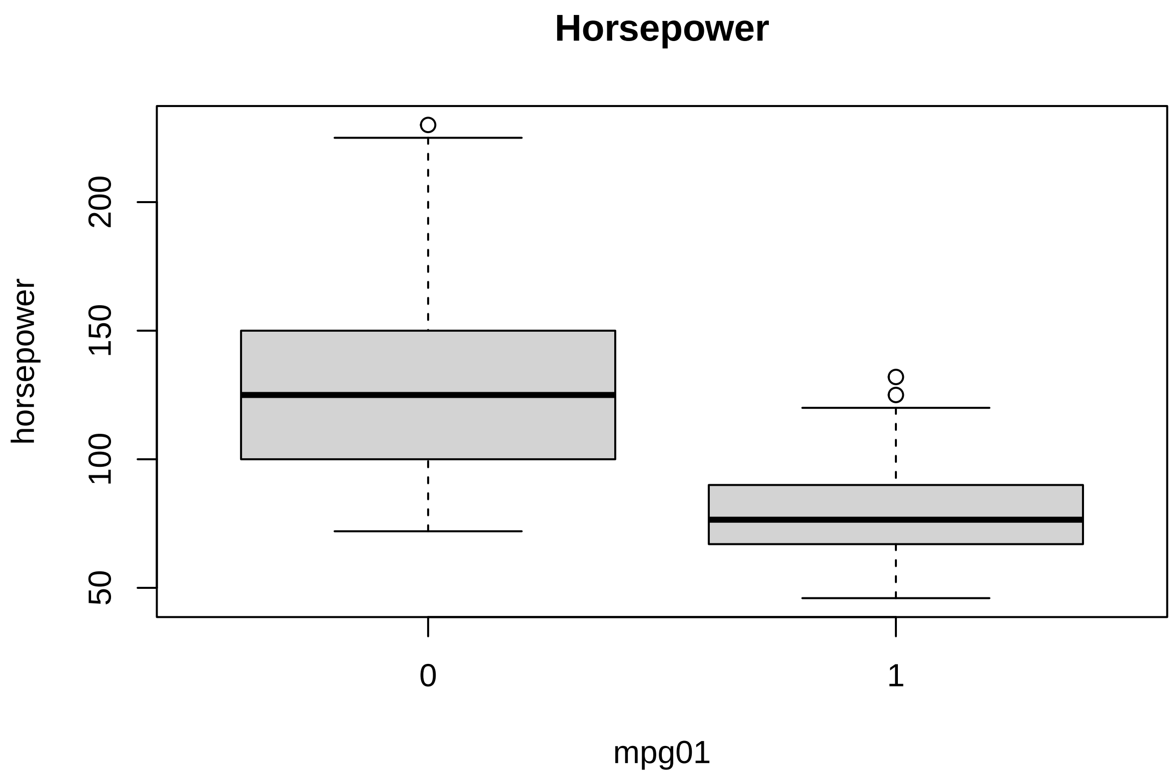
```
boxplot(cylinders ~ mpg01, data = df, xlab = 'mpg01', ylab = 'cylinders', main = 'Cylinders')
```



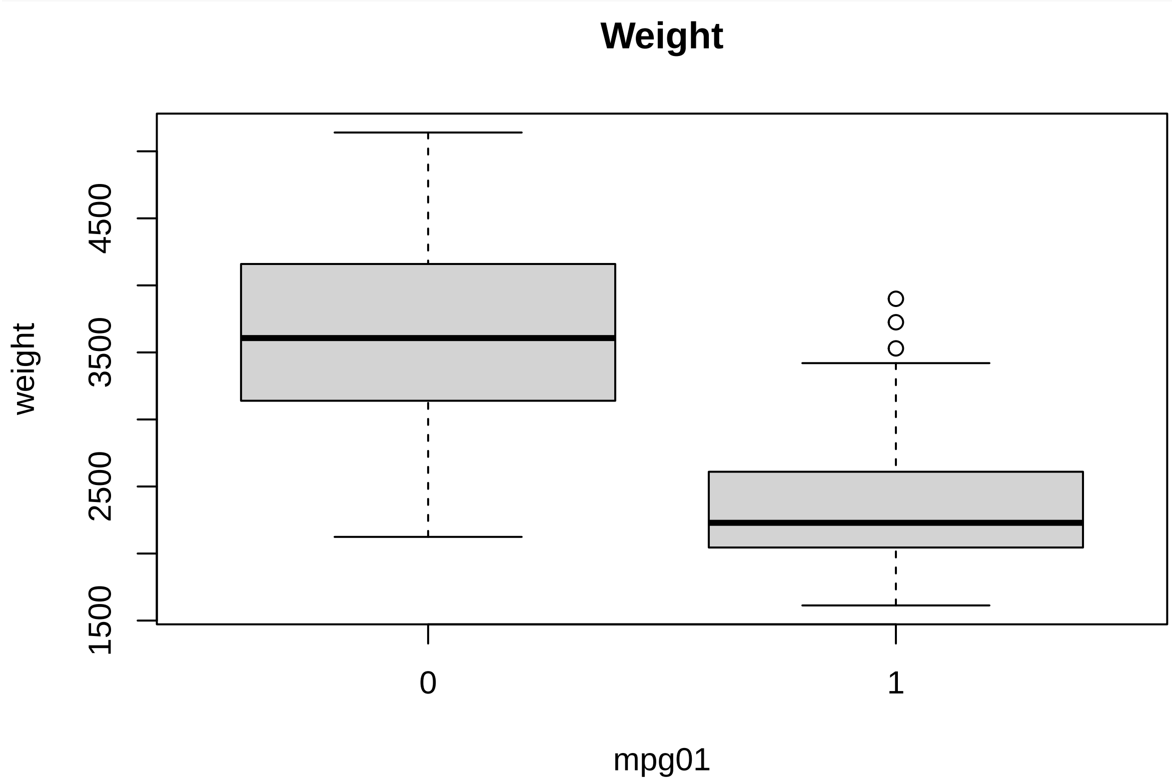

```
boxplot(displacement ~ mpg01, data = df, xlab = 'mpg01', ylab = 'displacement', main = 'Displacement')
```



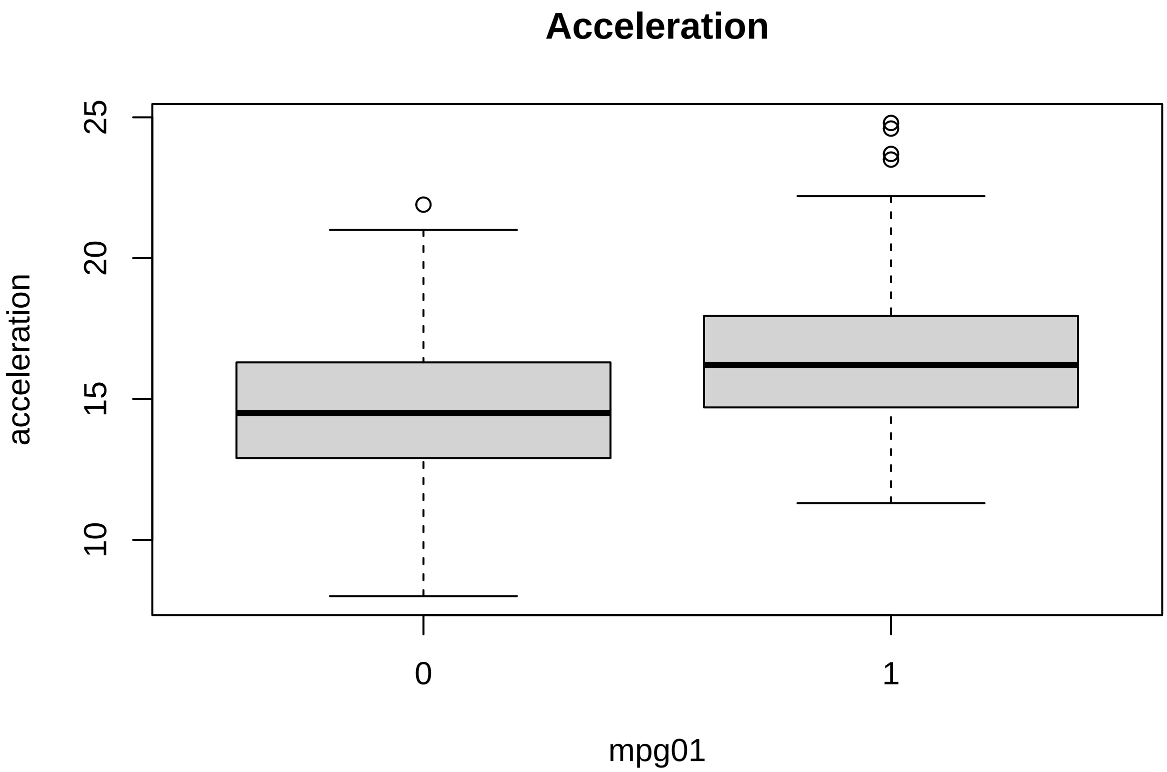
```
boxplot(horsepower ~ mpg01, data = df, xlab = 'mpg01', ylab = 'horsepower', main = 'Horsepower')
```



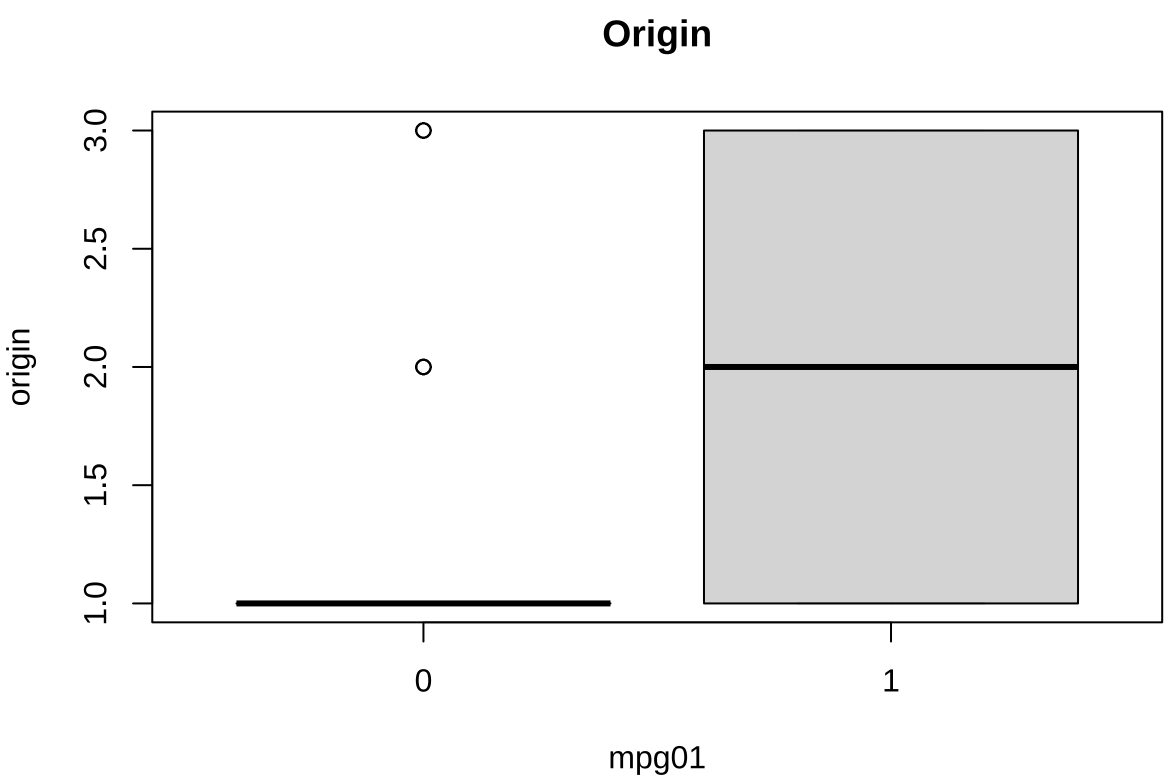

```
boxplot(weight ~ mpg01, data = df, xlab = 'mpg01', ylab = 'weight', main = 'Weight')
```



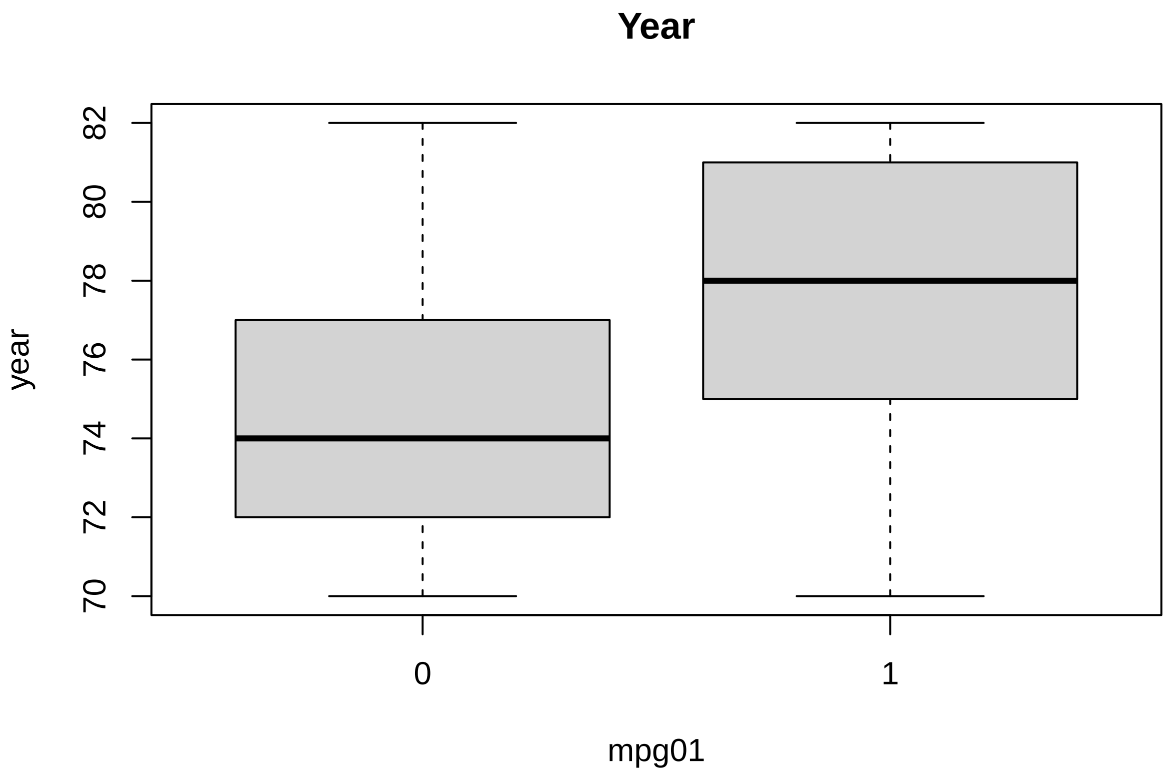
```
boxplot(acceleration ~ mpg01, data = df, xlab = 'mpg01', ylab = 'acceleration', main = 'Acceleration')
```




```
boxplot(origin ~ mpg01, data = df, xlab = 'mpg01', ylab = 'origin', main = 'Origin')
```



```
boxplot(year ~ mpg01, data = df, xlab = 'mpg01', ylab = 'year', main = 'Year')
```



After reviewing these plots, I chose to use the following columns: - MPG - Cylinders - displacement - horsepower - weight - origin - year These were all chosen because there was a significant distribution difference between those with a mpg01 of 0 and 1.

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

```
smp_size <- floor(.75 * nrow(df))

set.seed(0)
train_idx <- sample(seq_len(nrow(df)), size = smp_size)

train <- df[train_idx, ]
test <- df[-train_idx, ]
print(nrow(df))
```

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

```
print(nrow(train))
```

```
## [1] 294
```

```
print(nrow(test))
```

```
## [1] 98
```

(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.

```
library(MASS)
lda_md <- lda(mpg01 ~ mpg + displacement + horsepower + weight + origin + year, data = train)
summary(lda_md)
```

```
##         Length Class  Mode
## prior    2     -none- numeric
## counts   2     -none- numeric
## means   12     -none- numeric
## scaling  6     -none- numeric
## lev      2     -none- character
## svd      1     -none- numeric
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list
```

```r
test_preds_lda <- predict(lda_md, newdata = test)
lda_test_acc <- sum(test_preds_lda$class == test$mpg01) / nrow(test)
sprintf('Test Accuracy: %f', lda_test_acc)
```

```
## [1] "Test Accuracy: 0.969388"
```

```r
table(test_preds_lda$class, test$mpg01)
```

```
##
##       0  1
##   0  50  0
##   1   3 45
```

Test Accuracy of LDA Model = 96.94%

(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_md <- qda(mpg01 ~ mpg + displacement + horsepower + weight + origin + year, data = train)
summary(qda_md)
```

```
##         Length Class  Mode
## prior    2     -none- numeric
## counts   2     -none- numeric
## means   12     -none- numeric
## scaling 72     -none- numeric
## ldet     2     -none- numeric
## lev      2     -none- character
## N        1     -none- numeric
## call     3     -none- call
## terms    3     terms  call
## xlevels  0     -none- list
```

```r
test_preds_qda <- predict(qda_md, newdata = test)
qda_test_acc <- sum(test_preds_qda$class == test$mpg01) / nrow(test)
sprintf('Test Accuracy: %f', qda_test_acc)
```

```
## [1] "Test Accuracy: 0.918367"
```

```r
table(test_preds_qda$class, test$mpg01)
```

```
##
##       0  1
##   0  47  2
##   1   6 43
```

Test Accuracy of QDA Model = 91.84%

(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.

```
log_md <- glm(mpg01 ~ mpg + displacement + horsepower + weight + origin + year, data = train, family = b
```

```
## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(log_md)
```

```
##
## Call:
## glm(formula = mpg01 ~ mpg + displacement + horsepower + weight +
##     origin + year, family = binomial, data = train)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -2.094e-04  -2.100e-08   2.100e-08   2.100e-08   2.517e-04
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.132e+03  1.369e+05  -0.008    0.993
## mpg           5.753e+01  7.193e+03   0.008    0.994
## displacement -1.493e-01  4.520e+02   0.000    1.000
## horsepower    1.023e+00  9.911e+02   0.001    0.999
## weight       -3.826e-03  1.157e+02   0.000    1.000
## origin       -1.396e+01  1.064e+04  -0.001    0.999
## year         -2.790e+00  3.875e+03  -0.001    0.999
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4.0735e+02  on 293  degrees of freedom
## Residual deviance: 1.9865e-07  on 287  degrees of freedom
## AIC: 14
##
## Number of Fisher Scoring iterations: 25
```

```
test_preds_log <- predict(log_md, newdata = test, type = 'response')
test_preds_log <- ifelse(test_preds_log > 0.5, 1, 0)
log_test_acc <- sum(test_preds_log == test$mpg01) / nrow(test)
sprintf('Test Accuracy: %f', log_test_acc)
```

```
## [1] "Test Accuracy: 0.989796"
```

```
table(test_preds_log, test$mpg01)
```

```
##
## test_preds_log  0  1
##              0 53  1
##              1  0 44
```

Test Accuracy for Logistic Regression Model = 98.98%

(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_knn <- train
train_knn$name <- NULL
train_knn$acceleration <- NULL
test_knn <- test
test_knn$name <- NULL
test_knn$acceleration <- NULL

best_model_k <- 0
high_acc <- 0
for (i in seq(1, 12, 2)) {
  knn_md <- knn(train_knn, test_knn, train_knn$mpg01, k = i)
  print(paste('KNN Model K =', i))
  cm <- table(test_knn$mpg01, knn_md)
  print(cm)
  test_knn_error <- mean(knn_md != test_knn$mpg01)
  if (1 - test_knn_error > high_acc) {
    best_model_k <- i
    high_acc <- 1 - test_knn_error
  }
  print(paste('Test Accuracy =', 1 - test_knn_error))
}
```

```
## [1] "KNN Model K = 1"
##    knn_md
##      0  1
##   0 43 10
##   1  4 41
## [1] "Test Accuracy = 0.857142857142857"
## [1] "KNN Model K = 3"
##    knn_md
##      0  1
##   0 44  9
##   1  3 42
## [1] "Test Accuracy = 0.877551020408163"
## [1] "KNN Model K = 5"
##    knn_md
##      0  1
##   0 45  8
##   1  5 40
## [1] "Test Accuracy = 0.86734693877551"
## [1] "KNN Model K = 7"
##    knn_md
##      0  1
##   0 43 10
##   1  3 42
## [1] "Test Accuracy = 0.86734693877551"
## [1] "KNN Model K = 9"
##    knn_md
##      0  1
##   0 42 11
##   1  4 41
## [1] "Test Accuracy = 0.846938775510204"
## [1] "KNN Model K = 11"
```

```
##    knn_md
##      0  1
##   0 42 11
##   1  4 41
## [1] "Test Accuracy = 0.846938775510204"
```

```r
print(paste('Best K for KNN = ', best_model_k, 'with Test Accuracy of', round(high_acc, digits = 4) * 10
```

```
## [1] "Best K for KNN =  3 with Test Accuracy of 87.76"
```